Addressing complex design problems through inductive learning

Sean Hanna

December, 2011

Submitted in partial fulfilment of the requirements for the degree of
Engineering Doctorate in Virtual Environments, Imaging and Visualisation

Bartlett Faculty of the Built Environment
University College London

I, Sean Hanna confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Optimisation and related techniques are well suited to clearly defined problems involving systems that can be accurately simulated, but not to tasks in which the phenomena in question are highly complex or the problem ill-defined. These latter are typical of architecture and particularly creative design tasks, which therefore currently lack viable computational tools. It is argued that as design teams and construction projects of unprecedented scale are increasingly frequent, this is just where such optimisation and communication tools are most needed. This research develops a method by which to address complex design problems, by using inductive machine learning from example precedents either to approximate the behaviour of a complex system or to define objectives for its optimisation.

Two design domains are explored. A structural problem of the optimisation of stiffness and mass of fine scale, modular space frames has relatively clearly defined goals, but a highly complex geometry of many interconnected members. A spatial problem of the layout of desks in the workplace addresses the social relationships supported by the pattern of their arrangement, and presents a design situation in which even the problem objectives are not known. These problems are chosen to represent a range of scales, types and sources of complexity against which the methods can be tested.

The research tests two hypotheses in the context of these domains, relating to the simulation of a system and to communication between the designer and the machine. The first hypothesis is that the underlying structure and causes of a system's behaviour must be understood to effectively predict or simulate its behaviour. This hypothesis is typical of modelling approaches in engineering. It is falsified by demonstrating that a function can be learned that models the system in question—either optimising of structural stiffness or determining desirable spatial patterns—without recourse to a bottom up simulation of that system. The second hypothesis is that communication of the behaviour of these systems to the machine requires explicit, a priori definitions and agreed upon conventions of meaning. This is typical of classical, symbolic approaches in artificial intelligence and still implicitly underlies computer aided design tools. It is falsified by a test equivalent to a test of linguistic competence, showing that the computer can form a concept of, and satisfy, a particular requirement that is implied only by ostensive communication by examples.

Complex, ill-defined problems are handled in practice by hermeneutic, reflective processes, criticism and discussion. Both hypotheses involve discerning patterns caused by the complex structure from the higher level behaviour only, forming a predictive approximation of this, and using it to produce new designs. It is argued that as these abilities are the input and output requirements for a human designer to engage in the reflective design process, the machine can thus be provided with the appropriate interface to do so, resulting in a novel means of interaction with the computer in a design context. It is demonstrated that the designs output by the computer display both novelty and utility, and are therefore a potentially valuable contribution to collective creativity.

Acknowledgements

The research presented here has been supported by a grant from the UK Engineering and Physical Sciences Research Council, via UCL's EngD Centre in Virtual Environments, Imaging and Visualisation. I wish to express my gratitude to the many people I have met through the academic network this allowed, including the editors and organisers of the journals and conferences that provided feedback on my early publications, the community surrounding Space Syntax, the Design Computing and Cognition and related conferences, and my colleagues at the Bartlett. Only a few can be mentioned here.

My supervisor, Alan Penn, has provided pointed advice while remaining open to exploration, and for this I thank him, particularly in pushing what at one point may have been a purely technical thesis to take on deeper problems. To the extent that I have been able to investigate the creative process and express the importance of induction, and to realise that these have always been the important issues at the core of this work, it is due to his invaluable guidance.

For innumerable informal discussions of nearly all matters tangential to the interests explored here, and as a model of generous intellectual curiosity, I wish to thank Alasdair Turner. As a colleague, teacher and friend, he has given more in the way of inspiration and advocacy than he ever could have known.

None of this work would have been possible without the love and support of my family, for which I will always be grateful. I wish to thank my father, for his unconditional support for all my decisions along the way, and my mother, for her patient encouragement finally to complete the work. For her constant confidence, assurance and entomological analogies I thank Winnie Wong, who has supported me daily through these years.

Disclaimer

Much of the structural optimisation work described in chapters 4 and 5 was undertaken in collaboration with Siavash Haroun Mahdavi, also investigating microstructures for use in his EngD research. For an overview, see Hanna and Haroun Mahdavi (2004). Shared projects include the bulk of the genetic algorithm and gradient descent optimisation presented in chapter 4, and the initial experiment with a support vector machine to learn the objective function of model A in chapter 5, although not its theoretical context.

Haroun Mahdavi's research is in embodied evolutionary robotics, and his motivation has been the design of space frames with a high degree of flexibility to form the physical structure of robots that learn to adapt to an environment (Haroun Mahdavi and Hanna 2004a). My own focus has been on how optimisation and machine learning are applied to this complex design problem itself, as an example of stiffness optimisation and then by extrapolation to the other problems that will be raised.

Prior publications

The following published papers are directly related to the work presented in this thesis, and contain material presented here.

Hanna S. (2009) Spectral comparison of large urban graphs, In Koch D, Marcus L and Steen J (eds.) *SSS7: Proceedings of the 7th International Space Syntax Symposium.* Royal Institute of Technology (KTH), Stockholm, Sweden.

Hanna S. (2008) Numerical approaches in modelling design, in Gero, JS (ed), *Proceedings of the NSF International Workshop on Studying Design Creativity.*

Hanna S. (2007) Inductive machine learning of optimal modular structures: Estimating solutions using support vector machines, *Artificial Intelligence for Engineering Design, Analysis and Manufacturing,* 21, pp 1-16. Cambridge University Press.

Hanna S. (2007) Automated representation of style by feature space archetypes: distinguishing spatial styles from generative rules, *International Journal of Architectural Computing 5(1).* ISSN: 1478-0771

Hanna S. (2007) Defining Implicit Objective Functions for Design Problems, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2007.* ACM Press.

Hanna S. (2007) Representation and generation of plans using graph spectra, *Proceedings of the 6th Space Syntax Symposium, Istanbul.*

Hanna S. (2006) Representing Style by Feature Space Archetypes: Description and Emulation of Spatial Styles in an Architectural Context, In Gero JS (ed) *Design Computing and Cognition '06*, Springer.

Hanna S and Haroun Mahdavi S. (2006) Inductive Machine Learning of Structures: Estimating a Finite Element Optimisation Using Support Vector Machines, In Gero JS (ed) *Design Computing and Cognition '06*, Springer.

Hanna S. (2005) Where Creativity Comes From: the social spaces of embodied minds, In Gero JS and Maher ML (eds) *Computational and Cognitive Models of Creative Design VI*.

Haroun Mahdavi S and Hanna S. (2004) Blurring the Boundaries between Actuator and Structure: Investigating the use of Stereolithography to build Adaptive Robots, *Proceedings of ICARCV 2004*.

Hanna S and Haroun Mahdavi S. (2004) Modularity and Flexibility at the Small Scale: Evolving Continuous Material Variation with Stereolithography. In Beesley P. et. al. (Eds.) *Fabrication: examining the digital practice of architecture*.

Haroun Mahdavi S and Hanna S. (2004) Optimising Continuous Microstructures: A Comparison of Gradient-Based and Stochastic Methods, *Proceedings of SCIS & ISIS 2004, the joint International Conference on Soft Computing and Intelligent Systems and International Symposium on Advanced Intelligent Systems*.

Haroun Mahdavi S and Hanna S. (2003) An evolutionary approach to microstructure optimisation of stereolithographic models, *Proceedings of CEC2003, the Congress on Evolutionary Computation*.

Contents

Chapter 1: Introduction

1.1 Research objectives

1.1.1 Coping with complexity—importance of the interface

Complex systems are characterised by the organisation and interaction of their many parts (Weaver 1947; Kauffman 1995; Simon 1996; Arthur et al. 1997; Auyang 1998; Cilliers 1998; Johnson 2006) so they are highly resistant to *reductionism*—the attempt to analyse these parts individually. Design problems involving complex systems are difficult because of the lack of knowledge we can gain of the various parts and their interaction at all levels of resolution, but thankfully, many systems exhibit a high level robustness indicated by two long established principles in complexity, cybernetics and dynamical systems research (Pask 1961; Ashby 1956). The first is that the system's high level, emergent behaviour may be understood as qualitatively distinct from the parts—individual cells, animals and ecosystems, for example, can have very different kinds of explanation. Much of science is based on this, when it deals with complex systems by making an approximate model of the behaviour of the system based on high-level observations. The second is that while a system's parts cannot be understood in isolation, an approximation of a part's behaviour can yield the same result as long as the interaction structure between parts remains (Arthur et al. 1997; Cilliers 1998; Colander 2000).

Simon (1996) follows these principles to argue for a hierarchy of parts in social, biological and physical cases in which complex systems are 'nearly decomposable'—they have relatively simple interfaces between highly complex subsystems. He proposes a 'partly formalisable' theory of design that deals with complexity by allowing the possibility of accurate top-down approximations of a system at various levels, and gives the example of a watchmaker making a more easily constructed and robust machine by grouping elements into subassemblies (p.188). If a system is nearly decomposable it becomes a straightforward matter both to specify a clear external design goal for a subsystem and to realise a specific internal behaviour for it. This view has been criticised (Haugeland, 1998) as oversimplifying the interface between 'inner' and 'outer environments'. In another illustrative example, Simon considers the complex path of an ant on a beach to be a result almost entirely of the external complexity of the environment, which can be removed from consideration of the ant itself. This is meant to argue the possibility of designing something like the simple ant, but the flaw in decomposability becomes more acute as the same logic is used then to separate a *thinking* person from a [bodily] person (Simon 1996, p.53), thought processes from memory (p.99), and finally argue that thought itself is simple—all else is just environment.

The tradition of classical artificial intelligence (AI) and cognitive science from which this view originates (Haugeland (1998) has called this Good Old Fashioned AI, or GOFAI) assumes that as humans, our process of acting in the world involves making models (mentally or otherwise) that correspond to observed phenomena, and we use these to reason. An opposing view is advocated by ‘embodied’, ‘Heideggerian’ (Brooks 1991; Dreyfus 1992, 2007; Wheeler 1996, 2005) and dynamical systems approaches (Van Gelder and Port 1995) in which direct action with the environment eliminates the extra step of representation. Skilful coping in the world here becomes the goal rather than accurate modelling. Where the opposing views agree is on the utility of top-down estimations based on high-level observations, as these are frequently necessary when fine level data are unavailable. Whether estimations take the form of an explicit model or a more direct adaptation capable of anticipating the behaviour of an observed system, in both cases the underlying reality of the world is often more complex than we are able to fully appreciate. An important difference between the views is in the *interface* with the environment—it is simplified for classical AI. Simon’s aim in formalising the design process is the clarity of the physical symbol systems hypothesis (Newell and Simon 1976), which reduced intelligence to the formal manipulation of atomic tokens. Once set, they are rigid[*], and this ultimately reduces too much. In considering intelligence in an environment or the interactions between subsystems in a design, the embodied approaches indicate the importance of allowing for complexity at the interface

1.1.2 Specifying design process—instructive, elective and reflective

That design problems are ill-defined and not easily specified is evident in Rittel and Webber’s (1984) description of planning problems as “wicked”—they characteristically resist a definitive formulation and are essentially unique. Rather than consisting of clear subsystems, wicked problems have no natural hierarchy as each is “considered to be a symptom of another problem” (Rittel and Webber 1984). Moreover, many aspects, including emergent consequences, lack of conventional criteria and the cost of implementation, make the clear testing of their solutions impossible.

Human designers succeed nevertheless. One possibility is that our most complex artefacts, such as cities and vernacular architecture, have evolved over extended time periods in a process of

[*] The same has been said about other formal design methods with an atomic structure, such as Alexander’s ‘pattern language’ (Alexander et al., 1977). The background of Alexander’s method is discussed in the next section.

‘unselfconscious design’ (Alexander 1964) of gradually changing tradition, but this limits the scope of what the ‘self-conscious’ designer can achieve in limited time. Alexander proposes a method by which one can proceed by first analysing the problem to decompose a set of interconnected situations, then synthesising a solution by applying rules to each. It is more sensitive to the highly interlinked nature of ‘wicked’ problems designers face in the real world, but this analysis-synthesis method has been criticised (Hillier 2006) both for reductionism and for its resemblance to untenable theories of knowledge based on pure induction. While Alexander intends the designer’s mind to be unbiased with respect to the design situation as it presents itself, Hillier (1972, 1996) points out that prejudice is impossible to avoid, and the rules themselves require prior experience to be applied to any real design. The design system is missing its most crucial element. Hillier suggests instead that the main strategy in design instead follows Popper’s (1959) theory of scientific knowledge as ‘hypothetico-deductive’, in that individual designs are put forward as hypotheses and are then tested against the complex conditions of the real world. A crucial contrast between the two theories is that where inductive inference and Alexander’s synthesis can be said (after Lederberg 1958; Medawar 1960) to be *instructive*, the hypothetico-deductive method is *elective*. Both in design (Steadman 2008) and scientific discovery (Popper 1979 p.144), the two approaches have been likened to opposing views of evolution—those of Lamarck and Darwin—in that Lamarck considered an animal’s acquired characteristics to directly inform the features of its offspring while Darwin’s theory is based on testing and selection of arbitrary variety. The accepted evolutionary theory and scientific method thus reinforce what appears to be a greater descriptive value of the elective theory.

But while elective processes unquestionably operate in knowledge and design at a certain level, elective theories also typically lack a crucial element. They unfortunately avoid the question of how hypotheses are generated in the first place—a question that is highly contentious as it seems to address the root of human creativity itself. Popper (1959, §2) explicitly rules it out of the discussion of scientific discovery up front[*], and Kuhn (1962) considers it potentially insoluble because it is the result of an individual mind, therefore inscrutable and private (Kuhn 1962, p90)[†]. Perhaps due to the analogy with Darwinian selection, the most popular explanation, when one is given, is that this too is a selective process on potential hypotheses generated exhaustively or at random. Poincaré (see Koestler 1964) and Popper (1979 Chapter 6) considered creative insight as a selection from innumerable trials in the unconscious mind,

[*] Schön (1963, p. 92) also notes this: that Popper (1959) and N. R. Hanson (in *Patterns of Discovery*) misuse the word ‘discovery’, and consider only “validation after the fact”.
[†] It is also often ignored or dismissed in works dealing specifically with induction (e.g. Kemeny 1953)

approaches to generative grammar propose a similar ‘random generator’ to form syntactically correct sentences (Seuren 2004), and stochastic ‘generate and test’ processes (Liu 2000; Saunders and Gero 2001) are the basis of numerous design algorithms, including genetic algorithms (Holland 1975). In Hillier’s (1972) description, a number of constraints serve to reduce the infinite variety of possible conjectures to a finite set of hypotheses to be tested. Some external constraints (cost, structural efficiency, etc.) are quite explicit and straightforward—they are the objectives already used in engineering optimisation—but others, such as accepted norms of appearance, and the designer’s cognitive map, again appear contingent, creative, wicked.[*] As Schön (1963 p.15) notes, explanations that rely on the screening of generated ideas neglect the emergence of new screening concepts. If random generation is the case, then the contentious question simply moves up a level to where these constraints come from and how they are implemented.

Of course this question of the origin of theories on one hand or the application of design rules on the other is only contentious at the level of systematic explanation—we know human beings themselves can learn from their experience. Without denying the importance of either analysis or hypothesis testing in science, Hacking (1983) points out the importance of experiment in ‘creating the phenomenon’ which is then interpreted to form a new theory. This potential cycle is readily acknowledged in the activity of designers—Schön’s (1983) ‘reflection-in-action’ is an iterative process of alternating proposal and reflection, and the hermeneutical circle identified by Snodgrass and Coyne (1997) is a cycle of interpretation that involves (after Gadamer, 1975) constant projecting and revising. The hermeneutical approach identifies reason not just with logic, but with norms of community, accepted practices, style of presentation and analogy (Coyne 1999)—the constraints identified above as problematic—and asserts that tacit, social, ‘human practices’ precede the narrow rationality implied by ‘wickedness’ (Coyne 2004). It recognises the ‘one-off’ nature of design problems in that their solutions may be contingent on the situation. This does not mean they are subjective, but grounded in hermeneutical community of “those who share tacit understanding” (Snodgrass & Coyne 2006 p 122).

These explanations are consistent with the elective methods of Popper (in science) and Hillier (in design). Both admit the impossibility of the absolute certainty of knowledge. Popper’s insistence on ‘intersubjective tests’ (1959) and ‘critical discussion’ (1979) in corroborating theories can both be identified with the testing of projections in practical application (Gadamer

[*] This is not to say that norms of practice are *guaranteed* to provide the best solution to wicked problems any more than explicit objectives guarantee the best solution in optimisation, just that the processes used by unselfconscious designers are similarly ill-defined and complex.

1975)[*] by a hermeneutical community and the assessment of scientific models by consensus (Snodgrass and Coyne 1992) or with accounts of our knowledge of truth as based on consensus and 'intersubjective communication' (Habermas 1974). While these apply to scientific knowledge, they are particularly relevant in the arts and social sciences (Wilson 2010, p. 10), where definitive experiment is impossible and 'wicked' problems dominate. Popper (1979 p.162) explicitly acknowledges the discovery of knowledge as a hermeneutical process in both science and the humanities, but casts this as only in terms of what he terms the 'third world' of abstract knowledge (theories, reported facts, statements) and its affordance of new problems for our consideration. However, the reflective explanations also go further in contributing a possible instructive element. In hypothetico-deduction, observed phenomena in the world (Popper's 'first world') are valuable only in falsifying a theory by contradicting prior predictions, and there their use ends. Hacking's phenomena, along with the reflective and hermeneutical approaches, differ from this view of 'generate and test' in that rather than answering only a simple true or false, experience of the world can alter the content of the hypothesis (Snodgrass and Coyne 1997).

These approaches suggest that not only the 'third world' of abstract ideas but also the 'first world' of real, physical phenomena serve as the source material for creative insight. Here again, embodied views of cognition are relevant, in that they accept the two can never be considered in isolation. The third world is typically described in terms of abstract, often formal and logical concepts in the manner of classical AI, but an embodied mind can only understand these in terms of metaphors of physical experiences of the first world (e.g. Lakoff and Johnson 1999)[†]. In the case of design this is even more pertinent, as the results of design are physical objects—buildings, drawings, etc. Similarly, we understand events of the first world always in context of the third—there is always a cultural context, an evaluation with respect to a 'theoretical framework' (Popper 1979) or an appreciation of something as 'ready-to-hand' (Heidegger 1962). Whatever we reflect on in the process of design has an element simultaneously in both worlds.

From the above we may simply say that humans are intuitive, tacit, learning machines, but this still falls short of a satisfactory explanation. This research aims to provide one by demonstrating

[*] Gadamer (1975, p. 269) and Popper (1979, p.345) both also share the term 'horizon' for the *limit* of our observations in the world, and for the frame of reference through which we interpret them. In both cases the 'horizon' gives events their meaning—for Popper only events in the 'first world'. Popper's earlier (1959) emphasis is on logic but shifts in (1979) to 'critical discussion'.
[†] Non-Euclidean spaces, for example, are typically described as curved surfaces in more familiar Euclidean space. Lakoff and Núñez (2000) describe an extensive list of plausible metaphors for other mathematical abstractions, including infinite and complex numbers.

how this tacit understanding might be replicated by a machine. From a technical point of view, the simultaneous observation of real phenomena with abstract context occurs in supervised machine learning. From a practical point of view, to the extent that design is hermeneutical and dialogical, the computer can contribute far more if it can participate in this process. Whether the design task at hand is accomplished by elective or instructive means, the crucial step lacking in algorithmic explanations and emphasised in reflective ones is that of interpretation. It is this ability to interpret observations that will be the focus of the work.

1.1.3 RESEARCH HYPOTHESES

Design as described above demands addressing both the complex system itself, and the communication required for participation in the reflective process. As this work is motivated by the prospect of using computational algorithms in complex design situations, two corresponding hypotheses, frequently assumed in contemporary engineering and computation, will be investigated. The attempt will be made here to falsify both[*]:

- *Hypothesis A*: The first is that *to effectively predict or design for the behaviour of a complex system we must understand and simulate its underlying structure and causes of its behaviour*. This may be thought of as the modern engineering approach; unfortunately such models run into problems when for many complex systems not all details are known. Falsifying this hypothesis will involve building a higher-level approximation of a system's behaviour based only on observations, without knowing underlying causes, and using this in design optimisation. Ideally this should perform nearly as well as bottom up simulation in full detail, but have advantages of speed and (more importantly) the capacity to deal with unknown details. Inductive machine learning will be used to form the approximation based on prior observations or precedents.
- *Hypothesis B*: The second hypothesis is that *communication with the machine that runs this approximation must be explicit, using a priori definitions and agreed upon conventions of meaning*. This is implied in classic artificial intelligence approaches based on symbolic logic, and typical readings of the 'Physical Symbol System' hypothesis (Newell and Simon 1976) and classical AI[†]. An important variant also

[*] Cases will be shown in which the hypotheses do not hold, demonstrating that they are not always true, but not that they are always false. The argument is thus not against full understanding of systems or explicit communication in all cases, but proposes that there are other important factors in design, which should be acknowledged and supported.

[†] Major AI projects such as naïve physics (Hayes 1979) and CYC (Lenat and Guha 1990) attempted to build a priori ontologies by which the computer could understand our world. It was assumed "there must

> appears in many apparently opposed, embodied, ‘Heideggerian’ arguments for or against artificial intelligence (Lakoff and Johnson 1999; Dreyfus 1992) that suggests we human beings understand one another only to the extent we share similar bodies and senses—here the body is a priori, but the hypothesis is similar. It will be falsified by demonstrating systems that produce output satisfying a particular requirement that is communicated only by ostensive communication by examples. If alternative means of communication are possible, they may be particularly useful in creative design by providing a method by which to incorporate intuition and experience into optimisation scenarios. If the human mind designs effectively based on mental responses to the behaviour of complex systems which by their nature are resistant to full description, these mental responses may be communicated to a computer implicitly.

Central to both hypothesis will be the machine’s capacity for learning from examples—its capacity for *induction*. This learning will be a matter not only of selecting between alternatives, but an interpretation of structure that can be considered inherent in the data. In the first hypothesis this is the structure of a particular physical system under consideration, which can then be used within a design optimisation scenario when dealing with systems too complex to state explicitly. The intended practical result will be a learned approximation of the behaviour of a system that is accurate and robust enough to determine viable complex objectives, and algorithms to produce a well suited design via a computationally tractable method. In the second hypothesis, concepts in the mind of the designer are to be learned by induction, addressing the communication of ideas among groups of designers and the mechanisms of creativity itself. Because of the importance of induction in both cases, and the variety of approaches to it, it is outlined briefly in the following section.

1.2 On induction

Induction, or *inference from prior observations*, has been questionable as a method of acquiring knowledge since Hume (1740), and especially so since Popper (1959). It nevertheless describes the research presented here (for two reasons, each related to one of the two Hypotheses A and B), so some clarification is needed. Hume’s ‘problem of induction’ is that observations appearing to validate inductive reasoning do not imply it is always valid, yet it appears

be a common *representational* framework within which the meaning-content of any piece of representation can be related to any other” (Hayes 1979). This was considered essential for communication: “It takes common sense to understand each other” and “you need to have quite a bit of ‘common sense’ before you can learn to talk” (Lenat 1996). The CYC project is still ongoing, now the basis of Cycorp Inc.

necessary for practical reasoning about the world. Popper resolved this by showing that validation is impossible and replacing induction with the falsification of hypotheses, refuting induction as a valid route to knowledge. But the use of the term here differs from what Popper calls “genuine induction by repetition” (Popper 1979, p.98), and two reasons for its present use are described in this section.

1.2.1 Use of the term: why ‘induction’?

The distinction may be made (Wilson 2010, p.12) between mathematical methods as used in aid of hypothetico-deduction, and statictical methods that make an inference from data, which are therefore inductive. Machine learning as applied in this research essentially does the latter. It is typically connectionist, in which neural networks are joined together by weighted connections that determine their function. These weights are initially random, but modified by a learning rule when exposed to example inputs. Neurons themselves are simple: when presented with an input value they compute a corresponding output based on connection weights. The Perceptron update rule provides an illustration of learning for a single neuron on a classification problem: if the neuron’s output matches the class of the given example it does nothing, otherwise it modifies the weights to better classify the example. Learning is thus based entirely on *errors* between predicted observation and the actual data

Induction, as used here, is not quite what Popper refers to as the “bucket theory of mind” (Popper 1979, p.61), in which the mind is considered a theory-less receptacle for knowledge received directly via the senses. The starting weights through which information from the world is interpreted might be considered a kind of ‘theory’, but although they are required, their particular values are of less importance and may be arbitrary (Chapter 8 will demonstrate that this doesn’t matter for *individuals*, and Chapter 10 *among different communicating agents*). The ‘bucket’ theory admits the mind may not be a *tabula rasa*, however, and a more important difference is in learning itself. In induction Popper describes this as simply a restructuring of knowledge taken in directly, whereas it should be a modification of an organism’s dispositions to react, specifically a correction of expectations due to a disappointed expectation (pp.343–4). The fact that neurons update their theory based on errors between prediction and observed data in this sense actually resembles hypothetico-deduction rather than induction.

But this is not exactly hypothetico-deduction either, both because of the type of process and because of the nature of the data. As Popper describes it, the reconstruction of the frame of reference required of a false expectation appears to be either an active process without a specific mechanism (p.345) or an explicit selection between competing theories. In the case of the learning algorithm, on the other hand, for each single observation (and corresponding hypothesis), the neural update itself follows a simple gradient descent rule determined entirely by the data, so it is an instructive rather than a selective method. Furthermore, the data, in most cases, is already selected—as defined in induction, it is a set of prior observations. The hypothetico-deductive method spans past and future by formulating hypotheses first and then designing future tests specifically to falsify them. Popper (p.346) calls this the 'searchlight theory' rather than the 'bucket theory' because of the role hypotheses play in telling us where to look. In training the learning algorithm, by contrast, because the set of observations is given before the first learning trials it cannot be designed to refute a theory or to select between competing theories. Given this reliance on prior observations for learning, *induction* is a wholly appropriate term.

The second reason for referring to *induction* is due to the 'basic statements' that are meant to correspond to the observations by which hypotheses are tested in the hypothetico-deductive method. The concepts comprising these are to be defined explicitly using universal names (Popper 1959, §17) by which it is thereby assumed that their meaning is guaranteed to be certain and invariable. This atomistic approach to language that denies polysemy is difficult to justify after Wittgenstein (1958) and denied by a hermeneutical perspective in which interpretation is crucial (Snodgrass and Coyne 1997)[*]. Popper (1959, §20) does explicitly require 'intersubjective verification' of these statements, but this is to avoid errors of the senses, and does not address the possibility that meaning of statements may not be identical for every subject. This is not to say that in the case of theories stated with the explicit precision that Popper considers in science there is enough variation in interpretation to cast doubt on the objective knowledge produced by hypothetico-deduction, but when dealing with more complex, 'wicked' problems that aren't so clearly defined there is a potential problem of miscommunication. In addressing the second hypothesis of this research, and particularly in Chapter 10, it will be shown that communication of a complex concept is still possible, but this will be done by sharing example *observations*. Agreed meaning will be shown as due not to shared symbolic convention or perceptual structure but to shared data. The state learned by an agent is in this sense determined almost entirely by this data, so again *induction* seems appropriate.

[*] When Popper discusses hermeneutics (e.g. Popper 1979, Chapter 4) interpretation of the third world is in terms of which theories we take as relevant to frame new problems, not the meaning of third world statements themselves.

In the consideration of the design process in this work, inductive learning is in any case relevant not to the verification of proposals but to their formation. It deals with the point of inspiration, and interpretation. In its broader sense, induction forms the basis of Bayesian probability (Yates and Estin 1998; Duda et al. 2001), and inductive learning much of machine learning (Russell and Norvig 1995, p 529), including embodied approaches (Brooks 1991; Beer 1995; Wheeler 1996, 2005) that negotiate directly with an observed environment. In spite of the differences from hypothetico-deduction, the use of induction in this sense to guide action can be entirely consistent with Popper's views.

1.2.2 Why acting on induction is consistent with Popper

Popper's main concern is the logical testing of theories to build objective knowledge; design is concerned with two other things. The first, mentioned above, is how ideas come to be generated in the first place. The second is practical action. While a theory can never be logically validated, design requires we must propose solutions, and actually build on the basis of one or the other. Hume was forced to concede that we rely on induction out of pragmatic necessity. Popper refined this in stating from a rational point of view, we have no justification to rely on any theory, but as a basis for action, we should *prefer* the best-tested theory (1979, p.22), and in this he is in agreement with inductive methods.

Hume's logical problem of induction can be interpreted in two different ways. Popper states it as "reasoning from [repeated] instances of which we have experience to other instances [conclusions] of which we have no experience?" (1979 p.4). The first interpretation refers to the potential for a radical difference between past and future: i.e. the theory which might have been true for all events before time *t* might be false after time *t*. This is the problem referred to by Kripke's (1982) 'quus' (meaning 'plus' only for values less than 57) and Goodman's (1955) 'grue' (green before time *t* and blue after), a problem that cannot be avoided even if we have complete knowledge of the past. Popper acknowledges this as possible (our world may disintegrate in the next second) but takes a pragmatic stance surprisingly like Hume's reluctant acceptance, in that such possibilities are not worth consideration because we can't do anything about them (Popper 1979 p.22). This pragmatic dismissal is unnecessary however, as they also suppose a singular, improbable arbitrary event to coincide exactly with time *t*, so can never be the 'best tested' theory.[*] As our knowledge is always incomplete, the second (and more general)

[*] In addition to the fact that, by definition, 'best-tested' is based only on prior tests, there are two separate reasons for this, pragmatic and probabilistic: the futility of predicting an arbitrary future event without

interpretation of problem of induction is: given any finite set of observations, what is the best tested theory for reasoning to unobserved events, whether past or future? Theories contradicting observations are clearly false, but if several competing theories remain unfalsified, one must be selected.

In this case the better tested theory is the better *testable*—the one that requires fewer singular statements (observations) to falsify it. This solution is not only pragmatic; Popper (1979 pp.47–53) also considers the better testable theory to have greater empirical truth content, or *verisimilitude*, and therefore nearer to objective truth even prior to any tests. This is a point on which Popper repeatedly stresses a strong opposition to probabilistic theories of induction—induction claims such theories are *more* probable while Popper claims they are *less*—but this is a difference in definition that vanishes in practice. The more highly falsifiable is defined as less probable because it requires the truth of a larger number of basic statements (observations) and it is assumed that each of these is a priori of equal probability. This is illustrated with a 'pie-slice' model (Popper 1959 pp.112) in which basic statements are points on the circumference and the probability is proportional to the area of the segment containing them. This is purely the *a priori* probability of theories. In induction, however, some observations always precede the theory, and even under this model, the highly falsifiable theory quickly becomes the most probable given subsequent observations.[*] In agreement with theories of induction (Kemeny 1953; Jeffreys 1957), Popper (1959) defines these preferred theories by their simplicity[†]. Based on past observations, both methods require that we should prefer as the basis for action the most highly falsifiable unfalsified theory.

---

grounds is also related to Goodman's (1955) pragmatic idea of projectibility; the extreme unlikeliness of its occurrence is related to probability and Popper's (1959) discussion of simplicity.

[*] This can be illustrated by two competing hypotheses equally corroborated by existing observations: e.g. a line (H_{line}) and a higher order curve (H_{curve}) through two existing points. Both may be tested by deriving a prediction of a third point, at a particular location on the horizontal axis. In the case of the line only one value is possible, but many possibilities exist for the curve. Given bounds and a degree of precision of measurement this number is large but finite $\boldsymbol{c}$, so if the prior probability of each theory is estimated by the number of basic statements it will accept the curve is $\boldsymbol{c}$ times more probable $P(H_{curve})/P(H_{line}) = \boldsymbol{c}$. A specific third point (E) exists that does not falsify either theory, but the probability of predicting it for each differs: $P(E|H_{line}) = 1$ and $P(E|H_{curve}) = 1/\boldsymbol{c}$. By Bayes' rule the relative probabilities of the theories after this point is observed become equal $P(H_{line}|E)/P(H_{curve}|E) = 1$. Another observation falsifying neither makes $P(H_{line})$ $\boldsymbol{c}$ times greater, and so on.

This says nothing about the absolute probability of the theory being true (Popper 1979 p.101 explains the reasons for this), but only its truth relative to another theory.

[†] Accounts of induction prior to Popper also rely on simplicity or uniformity. E.g. Mill justifies induction by assuming nature is uniform, and Occam's razor has been justified (Sober 2000) by the fact that nature is simple, but these still beg the question of why uniformity or simplicity should be preferred. The probabilistic explanation in Popper (1959) shows why the simple is the most likely hypothesis given the data, so that simplicity is not a prior assumption about nature, but one is just highly unlikely to encounter complicated data that looks simple just by chance.

In design practice, therefore, a method consistent with Popper must meet two criteria: it must test theories rigorously and select the simplest among the corroborated. In scientific testing, Popper demands that hypotheses come before observations, but the use of prior data may be necessary in design because of the nature of wicked problems. Several features of these (their solutions are not true-or-false; hypotheses can be refuted in more ways than in science; experiments are unrepeatable; see Rittel and Webber, 1984) rule out the opportunity to seek the most appropriate future observations in design, but if we cannot literally meet this demand we can meet its intended content—that all tests must make unexpected predictions and be sufficiently severe (Popper1979 p.353–4). Even though many predictions in the data will be expected, the algorithm is trained only on its (unexpected) errors, so the requirement for these becomes a matter of preselecting data that will contain the necessary unexpected observations. Kemeny (1953) showed that this is a matter of the size of the set—even with an allowed level of deviation in measurements, we have an increasing chance of selecting the true theory as the number of observations increases. The demand for severity is related to a theory's degree of precision, which can actually work against the second criterion of simplicity. Because of inaccuracies in observation, most highly falsifiable theories (even if true) will be judged falsified by inaccurate measurements (Kemeny 1953), but this is overcome in practice by allowing an acceptable level of tolerance. The appropriate balance between precision and simplicity has been quantified by Akeike (1973) to measure the a priori predictive accuracy of set of curves, but in practice this may be handled by a number of error estimation methods (Reich and Barai 1999). The purpose of these is to balance accuracy with simplicity in avoiding ovefitting of the hypothesis to the data; even though all data is given in advance, they take a Popperian approach in withholding a set purely as a test of a prior hypothesis.

1.3 Two design domains: overview

Complexity in design can arise in two distinct ways. In many cases, the system itself that one is designing may consist of a vast number of interacting parts or subsystems, such that its behaviour is too complex for a designer to anticipate, even if the desired outcome is known. So long as the complexity of the system does not exceed what can be realistically simulated, this is the typical profile of an optimisation problem. The other possibility is that it is the context in which the design is to function that is complex, such that even the desired behaviour of the design is difficult to state. Such problems are common in design, are characterised by not being fully specified by a brief, and are considered to require creativity.

The work will investigate two different design domains, selected for this major difference in the source of their complexity. The first is a structural problem, in which the physics and desired objective is well understood and easily quantified, but the size of the system renders it complex. The second is a spatial problem, in which the layout of office spaces operates within a social context. This social behaviour is not well understood but methods have been developed to produce realistic models based on abstractions. Both of these problems are problems of complexity, not easily addressed by traditional engineering approaches, and they have been chosen as representative of the kinds of problems architects must solve.

In the case of the structural problem, that of space frame optimisation, there are well established methods. A structure under load can be approximated using a simple model and evaluated by the Finite Element Method (FEM) (Turner et al. 1956; Argyris and Kelsey 1960) to yield a prediction of its behaviour. The design is then improved over successive iterations to approach a desired goal that is typically simple to define explicitly: a stiffness to weight ratio, for example. Two aspects of the particular problem chosen in this work make it both complex and not suited to this established technique. First, as the number of members in the structure increases, so does the number of interactions between them. The problem becomes more complex and computation time for optimisation increases exponentially. Second, the model provided by the FEM is an abstraction that by definition differs slightly from the behaviour of the real structure: a strut is represented in the model as a single linear member with constant material properties and section, whereas real structures can only be built to a set degree of tolerance. If the structure is to be built at a very small scale, the effects of the manufacturing process on the material properties and geometry become more pronounced, and control the behaviour of the structure to a greater degree. Learning algorithms will be used in the first case to improve the tractability of the structural optimisation problem with many members and in the second to develop a method in which these manufacturing imperfections do not have to be modelled explicitly, but are learned.

The spatial problem is also a social problem, and so is less amenable to explicit explanation because the units are people with their own personal complexities and idiosyncrasies. The space of the workplace consists of the arrangement of desks and office spaces to reflect a given set of social relationships that exist in an organisation. While these cannot be directly quantified in the same manner as the FEM, a number of methods for quantifying spatial measures based primarily on visual relationships have been developed in space syntax research (Hillier and Hanson 1984; Hillier 1996), and these have been shown to correlate highly with social aspects of space use, such as movement (Desyllas and Duxbury 2001; Turner et al., 2001; Peponis et al.

1989; Hillier et al. 1993), crime (Hillier and Shu 2001), and interaction within office environments (Spiliopoulou and Penn 1999). With such a design problem however, even the objective cannot be explicitly stated. In interviews with practicing designers conducted in the course of this research (chapter 6), most stated goals were broad compositional devices such as minimum clearances and increased density, which have little to do with the details of interaction and use. Individuals specifying their own preferences for seating speak easily of simple material and environmental conditions such as proximity to windows, but the specific interaction they want with colleagues is far less clear.[*] Furthermore, even the comparatively efficient models provided by existing spatial analysis tools (e.g. Depthmap, Turner 2001) take significant time for computation when run for the number of iterations required for optimisation. Machine learning will be used in this domain both to contend with a model of spaces that runs faster and to induce an objective function implicitly from provided examples.

The problems provide a test in two domains of a general method, in which machine learning will be used to derive solutions based on existing precedents. Because of this similarity, the more readily quantified structural problem serves as a test case for many of the techniques and algorithms used again in the spatial domain.

They differ in the source of their complexity however—in the first, the *model* of the structural system is too complex to simulate explicitly in its entirety, whereas the second, the optimisation *goals* are unknown. They therefore exemplify contrasting strategies for the use of machine learning, in that the learning step may be applied at two distinctly different stages in the design process. Figure 1 illustrates both alternatives as a progression from top to bottom, with the learning phase represented by a diagrammatic neural network.

- Strategy A begins with optimisation (Figure 1.1, a1). Because the entire system is too complex to model whole, a limited series of small structural modules are optimised individually to a pre-determined fitness criterion (e.g. stiffness). A range of local loads are considered, ignoring context, although complexities of the material and fabrication process are incorporated into the simulation or measured directly from real samples.
- Training of the learning algorithm follows (Figure 1.1, a2). The local loads and corresponding structural modules found by optimisation are used as input and output

[*] This is not to say that people are not unconsciously or intuitively aware of the qualities that contribute to a good workplace. One of the uses of machine learning will be to allow these intuitions to be quantified via examples that embody them.

sets for a supervised learning algorithm. The resulting function of this algorithm thus maps any local force to its ideal structure automatically.

- The final step uses this function as an *instructive* method to generate new designs (Figure 1.1, a3). Global analysis of a loaded, complex form reveals a distribution of stresses varying over each point in its volume. This vector field is sampled at a fine resolution, each local load now becoming the context for a single module of structure and input to the trained learning algorithm. The combined output for each point is a high resolution, optimised microstructure.

- Strategy B, applied to workplace interiors, inverts the above order of operations. Because the goals of optimisation are initially unknown, learning is applied before the optimisation step to derive these from an initial set of examples, ideally real-world precedents. The first step here is simply the selection of these examples and their evaluation based on their known suitability, ideally over an extended time period (Figure 1.1, b1).
- The learning algorithm is trained next, to define the objective implicitly (Figure 1.1, b2). The samples and corresponding evaluations are used to define a subspace of the samples' full feature space in which any new plan may also be evaluated. This should capture aspects of the initial evaluation (b1) that may not be easily explained, but may include the complexity of varied social factors.
- In this strategy, optimisation is the final step—designs are produced by an *elective* method. Traditional optimization (by genetic algorithm, gradient descent, etc.) may now be performed by using the trained learning algorithm to evaluate fitness. Constraints of the project (floor plate shape, etc.) are input as context and finished plans are produced by the optimisation loop.

The two strategies are not mutually exclusive, as the learning phase could conceivably be used at both stages of the design process. The two problems chosen illustrate the strategies more clearly however, thereby providing a firmer basis for evaluation of the methods.

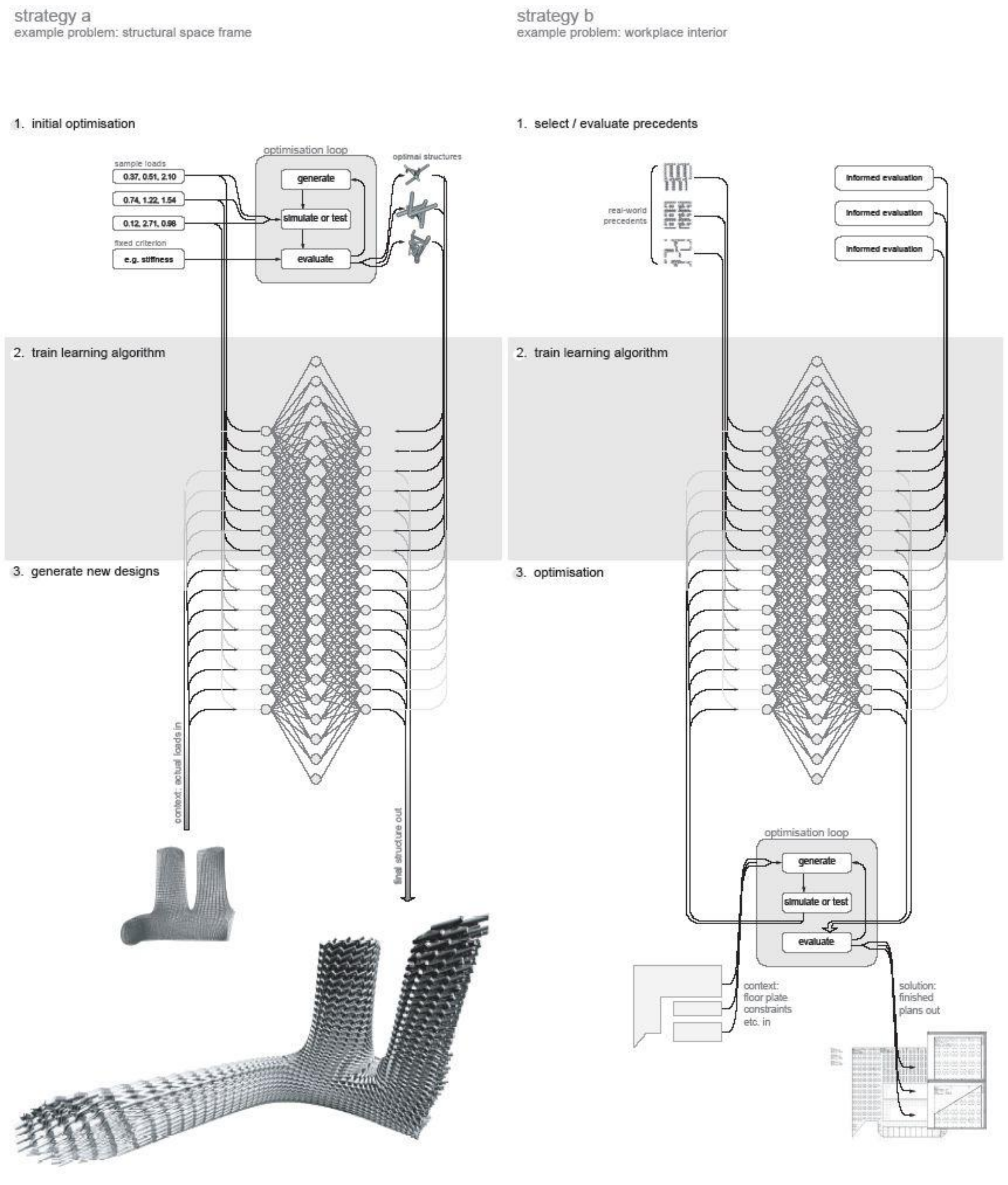


Figure 1.1. Diagrammatic representation of a learning algorithm incorporated into the process of the two design problems.

1.4 Outline of chapters

The main body of the research will develop and test the methods for making design decisions based on inductive models, in the context of the two design domains—first the structural and then the spatial. This is followed in the final chapters by a discussion that places these methods in the context of collaborative design and creativity, and shows how they might be used. The following is a summary of each chapter.

2. *Design in practice*
   This chapter gives background reasons for the research—why is it being done? Science progresses as empirical, tacit knowledge becomes codified and explicit. Engineering deals with the latter, but most real problems involve a significant amount of the former as well. Suitable tools of computation exist for easily quantifiable search and optimisation, but not to aid our intuition on problems with too many variables or to aid our communication of tacit understanding among teams with many members. As design problems and design teams are growing ever more complex, there is a need for such tools to enable better solutions. There is also a normative, social drive—particularly in the case of strong 'signature' architects, there is a need to reinforce established identity and to understand types appropriate to a given problem. Methods to aid in quantifying and communicating typological traits are therefore desirable.

3. *Literature review*
   This chapter gives the technical and methodological background required to address the hypotheses—how will it be done? Optimisation methods typically require high accuracy and precision in modelling the problem. Alternative means are introduced from cybernetics, complexity and dynamical systems theories to show how high level, emergent behaviour may be approximated, then the embodied approach to cognition is also reviewed, in which internal representations are replaced with more direct adaptations to make sense of the world. In both cases it is more appropriate to speak of an 'approximation' rather than a 'model'. Methods of machine learning are presented by which these approximations may be derived as interpretations of existing examples, and thereby offer an opportunity for the machine to engage in the processes of reflective design and phenomena creation that are central to this work. In order to evaluate the research hypotheses, linguistic competence is proposed as a test.

4. *Structural optimisation for stiffness*
   To prepare for the first evidence against Hypothesis A, optimisation is introduced on a readily quantified structural problem of space frame design, an example of a complex structural system in which the interacting elements are interconnected physical members. To replace part of a complex system with an approximation, it is vital to ensure the interface between these is adequate. The structures used here are complex and not 'nearly decomposable' as Simon (1996) assumed, but they can be broken down

into simpler modules by standardising the interface between them, while acknowledging this interface is just as complex as anywhere else in the structure. In doing so, optimisation for high stiffness to weight is made computationally tractable for any size structure, as are other design goals such as anisotropic deformation and variable Poisson's ratio. With a standard interface, it becomes possible to replace a modular volume of an object with an optimised module different in structural detail, preparing for these to be learned from examples in the next chapter.

5. *Inductive machine learning of optimal structures*
   Machine learning is introduced into the structural optimisation and the results of the combined method presented as evidence against Hypotheses A and B. Against Hypothesis A the full replacement is made not only of structural cubes with approximations, but the entire model/simulation/optimisation process is replaced by output from a learning algorithm. It is shown that this can operate without knowledge of the system's causal details—levels of resolution, material properties, etc. that are not modelled. The production of appropriate designs does not involve explicit simulation, but resembles much more an embodied direct coping with environmental requirements. This is as effective as and many times faster than standard modelling and optimisation. As evidence against Hypothesis B, this output is based on high level observations of prior example structures. While in this case the clear goals of stiffness and mass would be easily specified, the algorithm in fact has no knowledge of them, a fact that will be more relevant in the spatial problems where objectives are less clear.

6. *Initial planning tool*
   The office space planning problem is introduced, and the space planning tool developed for Foster + Partners in year one is presented as an initial approach to the problem. This incorporates basic optimisation and no learning algorithms, but outlines the expectations and working method of the design team, and provides a graphical interface for a top-down parametric design tool. A discussion of its limitations in actual design outlines requirements for the next three chapters—the lack of flexibility due to predetermined patterns and the simplification of goals to dialogue box statistics make the same sort of reduction as Alexander's 'patterns' or Simon's 'ant'. The interface therefore also becomes important for Hypothesis B, in allowing designers, or agents, to communicate adequately. Chapters 7, 8 and 9 will apply the same machine learning methods to this space planning problem to represent and communicate goals that are

difficult to define explicitly.

7. *Flexible representation of plans by graph spectra*
Falsifying Hypothesis A will require the approximation of high level system features with a learned approximation; as a prerequisite for learning, this chapter looks at quantifiably representing the structure of a plan so this can be done. The geometrical placement of desks in a plan will be replaced with an approximation created via the spectra of topological graphs. These quantify the plan without the simplification of the tool in Chapter 6, as is demonstrated by the use of the representation first in a classification exercise, then in generating close matching plans in optimisation. It is found that, rather than simple resolution, adding several diverse types of graphs improves the method considerably. The result is a single feature vector for a plan that may be used as its approximation in high-bandwidth communication with the computer.

8. *Design through learning and mimicking archetypes*
Falsifying Hypothesis B requires the ostensive communication of a complex concept—in this case a design objective—through examples of existing designs. The general notion of a simplified approximation of a system in representing design examples in Chapters 4 and 7 is here specifically identified with dimensionality reduction. A method is proposed to derive the objective from selected precedents by using machine learning to automatically determine which features are relevant. In so doing, a feature space (Φ) of reduced dimensionality is extracted in which to measure plans such that only these features appear. The point in this space that best defines this set of plans is also found by the method and together with the feature space is termed the *archetype*. This is used as the objective in generating plans in a scenario of building by aggregation. A detailed internal model is not used to do so, but rather an adaptation allowing the designing system to function: the algorithm is an elective design process that selects affordances offered up by the evolving design. This method makes communication possible in a manner not dependent either on shared conventions or on a predetermined perceptual or bodily structure, but because of induction on the shared data.

9. *Optimisation of plans*
This offers a full demonstration of the above methods in a genetic algorithm (GA) optimisation to generate office desk plans. As a GA, it is an elective process analogous to ‘generate and test’ or ‘hypothetico-deduction’, except design objectives are learned as archetypes from real example plans rather than set explicitly or via a simulation. The

algorithm produces plans with a particular form (e.g. desk cluster size or shape) without being given geometric information (only topology is encoded) or a model plan (the final form is not present in the example set), and thus falsifies Hypothesis A. The fact that all communication of the archetype is ostensive (through the examples) falsifies Hypothesis B.

10. *Discussion*

    This chapter suggests that not only does Hypothesis B not hold, but we *should* not predefine communication in creative dialogue. It discusses how technical work of the previous chapters fit into the context of the larger creative process as a whole, suggesting ostensive communication with the computer resembles tacit communication within design teams. As before, it is crucial to ensure the appropriate interface, here between communicating parties. An outline is given of a software tool to aid real design teams in their communication. The algorithm's ability to make sense of and produce new drawings and plans allows it to participate to some degree in the creative process, as communication between user and machine can take a more natural form resembling human communication by intersubjective interpretation of designs. Creativity is discussed as a collaborative process, and the essential creative act of reinterpretation is explained with reference to Kuhn's 'paradigm shift'. The most important requirement for a creative agent is argued to be this ability to interpret examples in the world, which machine learning enables. An agent model implementing the creative feedback loop in a social context is presented to demonstrate this.

11. *Conclusions*

    Approximations will have been made of the behaviour of complex systems; these can be based on existing precedents, and used to guide a design search. These design methods can be learned by a machine just as by a human designer, and in doing so the computer can give form to complex design problems. Design combines elements of instructive (e.g. traditional, habitual) and elective (e.g. critical, hypothetico-deductive) methods, but each of these explanations leaves out the essential point of creative insight. But this is not completely inscrutable, and it is likely that a reflective (e.g. the hermeneutical, Hacking's 'phenomena creation' or Schön's 'reflection-in-action') interpretation provides the necessary framework. The manner in which the computer can take in and make sense of high level observations of the world such as structures and plans (Hypothesis A) and communicate naturally with human designers (Hypothesis B) offers both a potential explanation and aid to design of complexity.

Chapter 2: Design in practice

Summary: This chapter gives background reasons for the motivation behind this research. The distinction is made between explicit scientific knowledge and the tacit knowledge useful in design, the case is made that the use of the latter in the contemporary design firm, which deals with unprecedented complexity, requires new working methods and tools.

This work was undertaken with the industrial sponsorship of architects Foster + Partners, with whom the general design problems to be addressed were determined. The practical motivations behind the research parallel the dual hypotheses made in the introduction. On one hand the issues arising from complexity in a design problem must be dealt with, and steps to understand these are to the benefit of both designer and client. On the other hand, design is a collaborative process, and a means to improve the necessary communication and collaboration among designers themselves is equally desirable.

This chapter outlines the importance to the contemporary designer both of adequate methods for engineering in complex situations (related to Hypothesis A) and of alternate means of communication in the context of practice (related to Hypothesis B). The current state of engineering and practical collaboration are first described, followed by a discussion on how the research will be applied to aid option generation for problems with many variables or to aid our communication of tacit understanding among teams with many members.

2.1 Task issues: complexity in engineering

In advocating reflective practice, Schön (1983) opposes the limited model of engineering as one of “Technical Rationality”, an essentially Positivist view in which problems are well defined and ends are agreed a priori. His clearest example of the limitations of this view is Simon’s (1996) proposal of a science of design—where Simon criticises the apparently intuitive and “cookbooky” nature of design teaching and practice, he places too much faith in principles of optimisation. Simon casts even traditionally elective systems such as diet as optimisation problems by listing as variables the possible foods prepared, their costs, the required minimum needs for vitamins and nutrients, etc., and proposing then that if well formed, the problem of optimal diet can be solved. As Schön and others[*] are quick to point out, such interventions are susceptible to failure because the situation is often not clearly understood. This explicit formulation of problems and methods in engineering practice has historically brought both benefits and disadvantages. On one hand it is a powerful analytical approach, and on the other it runs contrary to many design tasks in seeking to reduce a problem that in fact is irreducibly complex.

Reduction in the practice of engineering has evolved with the technical and institutional frameworks of modern practice. It occurs in two forms: in generalising a situation that is particular and unique to one that is universal and abstract; and in decomposing the situation and isolating component parts from one another.

2.1.1 Abstraction

Technical Rationality is the result of a development over the past several centuries, in which scientific knowledge derived from pure research has come to be seen as a distinct and prior basis on which decisions of practice can be made (Schön 1983). In the field of design, the increasingly effective methods of abstraction yielded by science are reflected in the simultaneous development of engineering. In a pre-scientific, craft-based, traditional or 'unselfconscious' (Alexander 1964) context, innovation is predominantly a result of small changes in the copying of existing precedents. These changes may be conscious or unconscious, but are modifications within a type. The benefit of the abstraction that science and engineering provide is that it allows innovative leaps by theory. The type may be irrelevant, as design may be explicitly directed by a specific problem.

Late medieval cathedrals represent some of the most complex examples of the former process. Structurally, each is a highly complex and elegant system, but one evolved incrementally from previous precedents rather than understood theoretically. By retaining tested features of established practice, each new building could be considered relatively immune to disastrous failure, and where modelling was performed to confirm the viability of innovations, common practice was to scale test the actual structural form as a complete system rather than an abstract concept (Addis 2007). This was possible due to the scale invariance of masonry behaviour, and was in fact a true physical test of the particular design. Innovation could be quite rapid, as seen

[*] Pollan (2008) gives numerous examples of the complexity of diet. The history of baby formula (pp. 20–22) as a series of incremental additions of newly discovered nutrients shows our clearly incomplete understanding at any given time, and nutritionists themselves admit (p. 62) the failure of "nutritionism" in isolating individual components for study.

in the almost monotonic growth of French cathedrals in a short period from 1175 (Laon, 23m in height) to 1225 (Beauvais, 50m in height), but the overall form remained relatively constant. These cathedrals represent the limit of what is possible with compressive masonry, but reached their most advanced form immediately prior to the emergence of engineering as a science in the Renaissance[*].

The birth of 'design' (in the sense of the word's 16th century etymology: 'to mark out') and engineering as a distinct task in construction required the use of the drawing and mathematics as a means to calculate the structural behaviour of a projected project, rather than rely on physical testing. Descartes introduced both the idea of reductionism (Descartes 1637) and began the geometric analysis of algebra that has formed the basis for modern reasoning and representing of space and engineering (Booker 1963).

The subsequent shift in innovation from the particular and concrete to the general and abstract is evident in the changes to engineering representation through the 17th and 18th centuries, in which the use of construction drawing can be seen to gradually change from a tool to aid the craft of building to a means for prior planning. Stereotomy, the representation of a three dimensional object by several orthographic projections, was initially grounded in the specifics of craftsmen's experiences of building. Early descriptions such as by L'Orme (1567) were collections of rules developed throughout the middle ages and Renaissance for calculating the angles of complex intersecting masonry, but specifically aimed at a variety of problems at hand. Durand's (1643) *L'Architecture des Voutes* furthered this with a much more comprehensive set of examples dedicated mainly to the complex curves formed at the intersections of vaults, and Desargues' (1639) *Brouillon-Project* attempted to systematise projective geometry by describing the process of stonecutting as an exact, step-by-step procedure. At this point, an overall theory within which to understand all geometries began to supersede the specifics of practice. The leap into abstraction, and thus to a more universal system that underlies modern engineering drawing has been credited to Monge's (1795) *Géometrie Descriptive* (Booker 1963; Addis 2007). With this systematic, encompassing set of rules, the use of two orthogonal projections describes any point or set of points in three-dimensional space, and is thus able to represent any object that can be defined exactly by precise vertices and edges. This is the system

[*] The Duomo in Florence provides a counterexample of this kind of incremental change, marking the transition from medieval to modern forms of innovation. Brunelleschi's method of brick construction is celebrated as a quite radical departure from existing practice, which, after over 100 years from the start of the cathedral, at last made it possible to span its unbuttressed diameter. It is this point in its history that the cathedral is frequently described as an example of Renaissance architecture, with its more theoretical understanding of building.

of our modern architectural and engineering geometry represented by plan and section. Compared with earlier methods, it bears no relation to material or process, consisting not of solid volumes but only of lines and transparency.

2.1.2 Hierarchy and decomposability

The adoption of statics in explicit calculation entailed a simplification in structure in the manner of Simon's (1996) 'near decomposability'. This change in practice is particularly evident in the development of industrial mill construction contemporary with Monge. Bage's Meadow Lane mill (1802) was the first in which the web of the beam was ignored to simplify calculations of strength, and in which beams were simply supported as effective pin joints rather than continuous across multiple columns. As Addis (2007) notes, this was a design that mirrored the mathematical model, rather than the other way around, and that model was relatively simple. It was revolutionary because element interactions (a single span, a beam flange) could be considered in isolation.

In decomposing the structure into parts, traditional engineering practice also employs a hierarchy of elements. As inherited from 19th century mills, a primary structural system such as a column grid, for instance, supports a secondary system of beams, which in turn support floor plates, curtain walls, etc. This allows each system to be designed in relative isolation as all levels below it in the hierarchy can be eliminated from consideration, and all levels above may simply be summed as a combined load. The disadvantage is that in adding additional load to those supporting them, the hierarchical arrangement of these subsequent systems increases the overall load of the structure.

2.1.3 Engineering complex structure

In the twentieth century a shift has occurred to allow more complex forms and dynamic structures to be modelled. A geodesic dome employs what Fuller (1975) terms *synergy*—it is a far more efficient structure because all members act together to reinforce one another in tension or compression as needed. But such systems are more difficult to design precisely because of these beneficial interactions. Elements cannot be considered in isolation.

The first general methods for designing these were in some sense a return to the physical testing that predated explicit calculation. The well known hanging chain models by Antoni Gaudi and the soap film models by Frei Otto were physical systems used to find a complex form that was both beyond the available calculation technology and particular to a given design. The change to digital analysis since the 1970s was made possible with the growth of computing speed, but effectively implemented a virtual version of the same complex models. Otto collaborated with John Argyris, one of several key figures in the development of Finite Element Analysis (FEA) in the 1950s, to implement early versions of this technology in the same design process. FEA effectively models fluid dynamics, stress in unusually shaped or loaded bodies and fabric structures by breaking down a continuous, dynamic system into an approximation built of many simple elements, but not in the manner of the hierarchical decomposition of traditional engineering—instead, the behaviour of each is intimately linked to every other.

Methods like finite element analysis mark both a technological and conceptual change. Systems are understood as many individual elements modelled together, the most important properties of elements being their *relationships* to other elements. There is a shift from basing the design on what can be simply modelled (as in Bage's modular mills) to the use of a complex model based on a possibly unique design, and thus a move toward addressing problems that are particular and 'wicked' again.

Some design problems remain beyond the capacity of these more complex models. The use of poorly understood composite materials, for example, often requires physical testing and experiment for several reasons. First, a component of the design might have multiple functions, satisfying several objectives simultaneously. In Michael Maltzan's Leona Drive Residence (Beesley and Hanna 2004), carbon fibre composites are used sometimes for their structural, and sometimes for their weatherproofing and aesthetic properties, as individual elements of the building perform dual roles as cladding and structure.[*] Second, like individual elements, entire subsystems within composites are also difficult to consider in isolation. Rigidisable, inflatable structures developed by ILC Dover (Ibid.) are based on a combination of textile-based reinforcement in a matrix material such as epoxy resin that can be activated after inflation. This matrix would appear to act in compression, taking the responsibilities of the initial air pressure while the textile acts in tension, but the reality is more complex: in fact the load passes through the fibres, and the matrix, which actually has a low compressive capacity, holds the fibres in place. This relationship is more complex and beneficial than simply substituting air with resin. Third, as with all complex systems, the interface between subsystems is crucial. In architecture,

[*] Masonry construction, developed long before the science of engineering, has no problem combining cladding and structure, but this is rare for structures incorporating tensile elements. It also makes this combination in a different way. At its simplest, masonry also follows a very strict hierarchy from the ground up. At its most complex, e.g. in a gothic cathedral, it evolved through gradual physical testing and elective processes as described in §2.1.1.

this is particularly evident in connections between materials with different properties—expansion and contraction, or flexibility under load. Steel and glass cannot join. Expansion joints must be left in a steel-framed brick façade. In Peter Testa's speculative project for a tower built of carbon fibre and composite materials a strategy is to eliminate joints and eliminate such abrupt changes in material, thereby allowing each element to act together with its neighbours (Testa, personal communication). Although the structure is quite flexible, the largely compressive, helical shell of the building is woven into the tensile floors in such a way that each holds the other up when subjected to vertical load under gravity.

These examples share a common emphasis on physical testing and experiment. Because of their complex nature this is essential in considering how subsystems will perform in their particular context. As design problems they are relatively unique so pre-existing, general models do not always apply. Foster + Partners' projects are frequently complex and of a much larger scale. The West Kowloon Cultural District covers an entire city neighbourhood; Beijing airport is one of the world's largest airports; and the planned project for Moscow's Crystal Island, at approximately 2.5 million m^2, is the largest floor plate of any building in the world. Many of these large projects employ structural space frames similar to those addressed in this research, and many have geometrical, environmental or site constraints that demand non-uniform solutions. While the structures are many times larger than those investigated here, the same issues raised by irregular loading, geometry and number of members apply at this larger scale.

2.1.4 Engineering complex interaction in space (the workplace)

While structural behaviour of buildings may be complex, the social behaviour of people within them is far more so. As a design problem for the architect, this can be affected only indirectly through the configuration of space. The role of space as an active participant in the shaping of social behaviour has been shown often in space syntax research (e.g. Hillier 1999; Hillier and Hanson 1984; Penn 1998). The similarity with the structural examples above is that both are complex systems based on interaction between parts—structural members or people. The space syntax approach makes the link between social and spatial via the visual and permeable connection between points within a space, and the computational methods of axial line or agent based analysis implement these in a manner that parallels FEA[*]. Technically these also compute

[*] Within Space Syntax and FEM analyses, each also has at least two different ways of defining the basic elements involved. Visibility graph analysis (VGA) and mesh-based finite elements both break up their domain of analysis into a set of linked elements that are essentially arbitrary in terms of shape, size and placement, except that their resolution must be fine enough to describe the smallest relevant features. Axial lines, isovists and units of convex spaces, by contrast, are maximal subdivisions of the domain, are

a network of connected, simple elements to approximate an overall behaviour for the complex system, and in practice they are used as a simulation not of structure but of anticipated behaviour of people. The role of Space Syntax Ltd. with respect to architects/designers is thus similar to that of consulting engineers.

Many of the large objectives in engineering of better space are understood, both in the urban context and in the space of work. In *the design implications of social interaction in a workplace setting*, (Backhouse and Drew, 1991) the argument is made that patterns of human interaction within the space of an office are derived from that space. Evidence is given in quantitative micro-analysis in which the complexity of this is revealed. The model of diversity and interaction proposed by Jane Jacobs in her *Death and Life of Great American Cities* (Jacobs, 1961) has also been popularly accepted as exemplifying principles of a healthy workplace (Gladwell 2000a). The design of modern offices has changed due to an acknowledgement of the need for interaction and the role of spatial proximity (Allen 1977), evident in the more frequent adoption of the open plan and a growing preference for deep floor plates of low-rise and refitted industrial space over the traditional office tower.

The change is partially due to a cultural shift in organisational structure, with dedicated spaces for social contact and similar principles cemented during the rise of creative industries and technology boom of the late 1990s (Florida 2002). It is primarily the creative needs that are acknowledged here, as earlier predictions of the effects of digital networks from a decade previously often assumed the reverse—that individuals would increasingly work from distant locations, and in extreme cases that cities themselves were no longer necessary. As Simon (1996) did with his analogy of the ant, this view again underestimated the complexity of the interface between people. It is this complex interaction that the creative workplace seeks to encourage.

But beyond the major design moves, engineering space for human interaction is highly problematic; little is certain about the full nature of the problem. In just the same way as a structural model is an abstraction of a complex reality, there is also a sense in which the problem as represented by the data provided by Space Syntax measures is a simplification. The measures themselves are explicit and scalar, and just as goals (e.g. deflection, strength) are given prior to the solving of a clear engineering problem, these assume that what is being

defined by the geometry itself, and are therefore intrinsically meaningful in functional terms. When the finite element method uses single elements that represent real construction elements (beams, columns, etc.) or maximal divisions of geometry (as will be the case in chapters 4 and 5), it is analogous to these latter ways of working.

looked for in a space (e.g. integration, intelligibility) is already clearly determined. New measures continue to be proposed as new design situations and motivations for analysis arise, but in the most complex of design problems involving the most complex of spaces, we may not know enough about what specific measures are most appropriate at the outset. Something equivalent to the sorts of physical testing required for novel structures, which examines the whole system in its particular context, may be desirable.

Design goals are also difficult to set. The desire for ‘more creative’ spaces is broad and unspecific, as the finer details may change over time and between particular organisations. Looking at the finer scale of how desks themselves are arranged in an open plan, different typologies may be identified as characteristic to specific occupations for reasons specific to their job requirements. The desk arrangement at Foster + Partners, which also occurs in a number of other architectural practices, is of open rows with partial division of desks below eye level, but with some flexibility in the row and just sufficient space at each station for a computer and an A0 size drawing. IT professionals may sit in small groups facing outward toward their computer and a wall or partition, allowing for required periods of intense concentration followed by optional discussion with colleagues across a shared convex space. Accountants may sit with no barrier between desks facing one another, allowing documents to be easily passed. Where such types occur they may have evolved to suit the particular needs of the organisation very well, but identifying and relying on them as models limits their use as needs or context changes. Here again they would be simplifications, as the assumption that the entire organisation is a homogeneous group is usually not valid, different job descriptions are not necessarily covered, and the effects of the context of a different overall space are unknown.

In practice, design of such spaces involves both analysis and intuition. The hermeneutic process of critical reflection allows for the generation and subsequent testing of numerous design proposals. Each of these may be based partly on formal hypotheses derived from previously established, explicit theories such as Space Syntax, and partly on intuition. Testing may likewise involve a quantitative evaluation with respect to clear objectives, and where these are absent, evaluation by other people with independent intuitions.

2.2 Professional issues: design environment

In addition to the complexities of the design problem, those of the design environment itself are relevant. Foster + Partners comprises over 1,000 employees, so issues of collaboration and communication are of particular concern. Large projects require large groups focused on a single task, and there is a further, higher level need to exchange knowledge between different projects and to maintain continuity as a single organisation. Communication is required both in the form of explicit discussion and the transfer of tacit norms.

2.2.1 Working environment at Foster and Partners

The working method at Foster + Partners, as with most architectural firms, consists of a collaboration among a number of individual designers over an extended period of time. It is acknowledged that much of the creativity may be an emergent property of their interaction, and thus accessible to explanation, as opposed to 'black-box' and 'individual genius' myths of creativity. The structure of this collaboration is carefully managed, with rigorous design review and quality control processes, but individual designers are not tightly constrained as to their role in projects. Teams are based on projects, rather than disciplines, providing each member with at least a view of the overall design task.

Design is carried out as a reflective process, primarily via architectural drawings and models. As a general rule, the efforts of individuals are often focused on the generation of a large number of architectural options, usually reviewed and discussed in groups, from which the guidelines for the next set of design possibilities are derived. These options may consist of complete buildings, individual details, urban impact studies, rough massing models, plans showing brief required areas or floor treatments, etc., in short, at all scales and at all times in the duration of the project. There is a hierarchical structure within the design team and office, so the ultimate decision as to the direction of the design rests more with some of these individuals than with others, but the generative process is distributed. It is this method of working which allows both for flexibility during the design process, and the ability to simultaneously address issues at extreme ends of the scale. The task of designing a building is one of the most complex that can be undertaken precisely because it must navigate and coordinate small scale variables like expansion joint widths and placement of furniture or fittings, and large scale variables like gross floor area and neighbourhood traffic patterns. In some cases these are mutually independent and can be dealt with completely at a relevant stage in the design, but more often they form complex dependencies, to be solved only by the iterative collaborative method described above. Distributing the process among many designers allows diverse problems to be tackled strategically and in detail, only because each has overall knowledge of the context.

As a design firm, there is a complex and constantly changing set of relationships to external consultants, as the demands of new projects call for different collaborators. Within the architectural office and among the larger design team that includes external consultants, each member sees the project from the point of view of their own discipline and each works with domain specific concepts, language and symbolic representations. The multidisciplinarity of teams is also facilitated by communication through drawings of options. The rich, high bandwidth communication provided by drawn options allows collaboration when different domains interact.

This process of individuals generating design options for review in a group context is nearly ubiquitous in architectural education and practice, and is the basis for the various reflective (Schön 1963), hermeneutical and systems (Czikszentmihalyi 1988) models of creativity as will be examined in more detail in chapter 10.

2.2.2 Guiding the search: norms and types as repertoire

The description of professional practice (in all disciplines) that Schön (1983) gives for how designers actually deal with such problems as mentioned in §2.1 draws on past experience by building up a repertoire of prior solutions and situations to be reused. These are usually referred to as types.

A number of approaches to typology exist in architecture, many of which implicitly assume or explicitly state that a type does not change, just as the set symbolic systems of classical AI are static and a priori. The use of types in this sense may be essentially conservative, to replicate structures that are known to have worked in the past (Krier 1988)[*], or it may acknowledge the changing reinterpretation of different generations (Rossi 1982), but the type itself is permanent. Rossi (1982) uses type to explain the universal and permanent character of cities. He describes the actual form of architecture as variations on a theme, but the type that informs this is permanent, logical and prior to this form (pp. 40–1). The notion that types are unchanging lends itself to attempts to codify a system of types, as Alexander et al. (1977) explicitly do in their Pattern Language.

Some typological approaches do stress change however, and allow a more flexible idea of type. For Colquhoun (1967), ‘types’ are solutions to related past problems, but it is not stated that

[*] Krier’s (1988) argument for limiting urban density uncritically assumes e.g. the Manhattan density regulations of 1935 were inherently bad, primarily because they are unlike previous cities. This leaves little option to adapt to a radically different modern environment. Similarly, a conservative approach such as new urbanism is limited to recreating American small towns. Even if these work well, there is no indication of how to scale up.

these problems are universal. Thus it would appear that types themselves may be constantly recreated by "adapting forms derived either from past needs or from past aesthetic ideologies to the needs of the present" (Colquhoun 1967, p.47) and that different needs may be identified each time. Colquhoun's description of the use of types highlights two crucial factors in contemporary design. The first is the proposal that modernism divides design into two types of task—the rational, clearly understood problem, and the remaining intuitive tasks. It is the latter, ill-defined and 'wicked', for which recourse to types is necessary. The second factor is that our age is characterised by change, and so type solutions from the past must be modified to fit problems "without precedent in any received tradition" (p.49). The first observation suggests that types are not easily reduced, and the second refutes the notion of an unchanging type. As these factors make the setting out of an explicit method for creating and applying types more difficult, Colquhoun does not do this.

The multidisciplinary, distributed nature of the contemporary design environment reinforces this latter, more flexible notion of type. Discipline and background affect the interpretation of names, or understanding of components, and it is in the nature of working together that design teams must simultaneously build their own ontology (Johnson, 2005). This understanding may develop radically even over the course of a single project.

Schön's account of design practice does describe the possibility of change, accounting for both novelty and utility[*]. First, the application of a scheme from the repertoire does not subsume the new situation under the old category. It is the means by which the designer sees the new situation as something familiar, but this 'seeing as' produces a new concept—even if the designer should act in the new situation as in the previous one the new context produces a different outcome. This is the fundamental mechanism for the origin of novel concepts in Schön's (1963) 'displacement of concepts', and related theories such as Koestler's (1964) 'bissociation of matrices' and Akin's (1996) 'frames of reference'. The second difference has to do with the scheme's utility, in that the designer selects from a number of possibilities the one most suited to the particular problem at hand. For complex problems this is not always clearly defined, but explanations such as Czikszentmihalyi's (1988) systems model and hermeneutical approaches identify utility with a design's relevance with respect to a changing culture—with an established cultural norm.

[*] Novelty and utility are widely acknowledged as the key characteristics of creative ideas (Boden 1990; Gardner 1993; Cropley 1999; Sosa and Gero 2003).

In the above accounts, type may be considered somewhere between a logical abstraction and a real form, but its structure is not clearly defined, and this must be done if it is to be represented to a computer. Chapter 8 of this thesis will attempt to do this (§2.2.3 describes several current methods). After Schön's concepts, which can be both conservative and innovative, a mechanism for typological change over time will also be proposed in Chapters 8 and 10.

2.2.3 Current techniques for specifying type

For a collaborative practice, building up a repertoire of types for application or norms for evaluation raises the issue of how these can be communicated, both between colleagues and to the computer. A number of algorithmic approaches have been proposed for the representation of architectural types; these can be roughly divided into those for the synthesis of new designs and for analysis of existing ones.

- *Generative approaches:* Rule systems have been developed to generate new designs of a particular type, such as Hersey and Freedman's (1992) computer implementation to create possible Palladian villas. Shape grammars (Stiny 1976, 1980) are possibly the most widely studied, and have been successful in providing generative descriptions of many building types. A descendant of linguistic generative grammars (Chomsky 1957; Langendoen 1998), they provide an explicit rule-based method for producing final designs, and have yielded examples in the apparent styles of Palladian villas (Stiny and Mitchell 1978), Frank Lloyd Wright's prairie houses (Koning and Eizenberg 1981) and Mughul garden designs (Stiny & Mitchell, 1980). As an approach to typology, a type or style is often (e.g. the examples above) encoded with a specific grammar, unlike linguistic grammars that generate a particular language with any number of possible styles. A creative human then works with the shape grammar to make a specific design of the predefined type. As a tool for analysis, the grammar or rule set is constructed by a human designer, a fully automatic process seen as undesirable or impossible (Knight 1998). In its generative capacity it is then followed by a user choosing which rule to apply at each stage in the process to create designs comparable to originals.

- *Analytical approaches:* Analytical methods have been proposed that model similarity or group designs based on a count of pre-defined features. Chan (1994) uses a list of material and compositional features to evaluate buildings as to their style, and finds a correlation with the judgements of test subjects. Experiential qualities of architecture have also been structured or mapped to rule sets to guide architects (Alexander et al.

1977; Koile 2004, 1997), and this approach has been implemented by Koile (1997) in an expert system that is also able to recognise building type as a count of defined design characteristics. Another approach to typology proposes that it is not defined by clear and predetermined features or rules, but can be quantified by various measurements taken from examples of the works. More general analytical techniques using information theoretic measures have been used to measure distance between individual plans (Jupp and Gero 2003), and to derive values for measurements such as entropy (Gero and Kazakov 2001, 2002), morphology and topology (Jupp and Gero 2004) that can be used to evaluate examples of a style. These have the advantage of quantifying designs of any style as real values on the same scales, so that variations within or between styles can be measured uniformly.

In practice, if not in principle, the above examples in both the generative and analytical categories strive to set the terms of the definition prior to the specific examples. Generative approaches set the rules first, and then work within them; and analysis is done with predetermined measures as is the case with space syntax, mentioned above. These are in line with Popper's view that scientific hypotheses precede observations. Similarly, in considering norms in art, Gombrich (1960) suggests that art provides categories by which to sort our impressions: "without some starting point, some initial schema, we could never get hold of the flux of experience" (p.88). This echoes Whitehead's (1941) more general statement that "we cannot understand the flux which constitutes our human experience unless we realise that it is raised above the futility of infinitude by various successive types…" (as quoted in Hillier et al. 1976, p.147). In these views a type may be predetermined, usually symbolically, and a general approach to representing this is rooted in identifying the equivalent of these symbols, either as generative rules of the work or features to be analysed.

But in fixing either the generative rules or evaluation measures in advance, most implementations fall short of the creation of new concepts essential to Schön's 'seeing as', and simply add a further example to an old and unchanging type[*]. Such use has been criticised in thereby remaining limited to domains of stylistic uniformity both in detail and overall composition (van Leusen, 1993), and are often a simplification that will either produce some designs that would be considered outside the style or fail to produce all possibilities within it (Hersey and Freedman 1992, chapter 2).

[*] Schön (1963) notes that the process of creative innovation is already finished by the time it can be spelled out.

To enable the interpretation of examples required by reflective design processes, the goal of naming a particular class of measures that we can specify in advance to contain the description of all styles would seem to be misguided as it then allows only one interpretation. A process of defining type retrospectively is needed, in which relevant features can only be determined in relation to the works, not beforehand. This is the ultimate aim of many of the computational methods, if it has not yet been realised. In the case of shape grammars, for example, although often implemented purely at the level of symbolic representations, the grammar is meant to operate on emergent shapes embedded in an evolving design; this difference from linguistic grammars was a primary motivation in their creation (Stiny, personal communication). Recent approaches have begun to address this by allowing decompositions on finished or existing designs to generate new rules for design exploration (Prats et al. 2006) and by implementing recognition of emergent shapes. These aims are instances of induction from examples, to be addressed in this work.

2.2.4 Current uses of automation in the iterative design process

For simple design tasks, automation can have a positive effect on efficiency by improving the ability to create real options and thereby enabling faster and better reflection on them. A CAD system that allows the drawing process to be more efficient allows more options to be explored in the generation phase of the process and a potentially more fruitful discussion to result. There are many problems that don't warrant a full review and discussion process, but must nevertheless be drawn and explored. Determining the number of desks in an office, parking spaces in a lot, or (sometimes) panels on an elevation are examples which depend on simple rules, and yield simple results but require some effort to calculate, and for which the next level of automation is obvious. This level has the effect of producing options automatically for presentation and review and could in theory tackle much more complex problems. Where goals can be known and simulation is possible, an optimisation algorithm can use a predetermined objective function to make some decision on behalf of the designer, thereby encompassing both halves of the iterative process and optimise certain problems with autonomy, presenting only those options for discussion which truly require a full review.

The first and most obvious benefit is in terms of speed, where simple and repetitive tasks are automated and the advantages in efficiency are immediate. A possible improvement of design quality may follow from the greater number of options in that ideas are generated in the course of exploration, in sketching or modelling, etc., which would not have been foreseen at the outset. These benefits are realised at present by Foster + Partners, in implementations of the above methods in the office or by external consultants.

Computational automation at more advanced levels—such as optimisation—becomes problematic. It occurs in other design domains, such as the automotive or aeronautical industries, but these have a different organisational model. In general, the process from design through fabrication is more tightly contained in a single organisation, with the resulting communication protocols and standards. Much more can be controlled and anticipated in advance about the process of design to fabrication in the case of a car than a building, and so optimisation of clearly defined parameters may be more readily applied. More expenditure can be made of time and effort up front in the setting of a parametric model, as the schema of relationships in that model can be anticipated and the production methods ensure an economy of scale.

The nature of the architectural design task and the collaborative environment set limits to these types of automation. Because of the ill-defined nature of the brief and complexity of each situation, the problem itself is always radically changing from one building to the next. There is a similar change in the composition of the team, relationship to external consultants and management from design through construction. Optimisation is used, but rarely and within a clearly defined domain such as a decomposable structural system.

2.2.5 Parametric modelling and its limits in practice

The use of parametric modelling with packages such as Bentley Generative Components is currently fashionable in architectural practice, and Foster + Partners is among the growing group of firms using and developing the forefront of these new technologies. The benefits of this sort of computation are great, but as their principle use is in structuring the logic of a design model, usually in a symbolic, hierarchical manner, these also have corresponding limitations. Changes in detail are accommodated with ease (often cited as the main benefit of the method) but unforeseen changes to the symbolic structure of the model are not. If radical design changes are required, as they often are in the early stages when such models are used, the model must usually be reconstructed from the beginning.

These limitations are similar to those associated with the a priori setting of norms and types above (§2.2.2), and so intimately related to the processes of successive reinterpretation and novelty generation in creative design. It is quite possible that this relatively fixed symbolic structure is the main contributing factor (rather than expense of time or lack of technological knowledge) in the relatively low frequency of some advanced computational techniques in practice (from shape grammars to optimisation) and any rift that exists between the computational technologists and the manual designers. Certainly, the proliferation of recent parametric packages has contributed to closing this gap, but their use is often similarly constrained to specific sub-tasks, and often sub-groups of designer-specialists, because of the need to define relationships in advance.

Alternative approaches are possible, in line with the nature of reflective practice as exploration rather than optimisation to preset objectives. Rosenman (1997) uses the lack of predetermination as a definition of creative design, suggesting "the lesser the knowledge about existing relationships between the requirements and the form to satisfy those requirements, the more a design problem tends toward creative design" (p.69). Gero (1994) goes further to suggest "exploration in design can be characterised as a process which creates new design state spaces" (p.318), changing the framework in which optimisation occurs, and such approaches have been incorporated in optimisation in the simultaneous evolution of 'problem' and 'solution' spaces (Maher and Poon 1996), for example. The actual working methods representing the forefront of advanced computation in architectural practice are also likely less parameterised and more flexible. While some formal parametric modelling is employed at Foster + Partners, their specialist modelling group has developed a less structured toolkit of algorithms and small applications that can be selected from at the appropriate stage in design. These function as a kind of repertoire—more flexible and amenable to unforeseen changes. To the extent that these are implemented not in the symbolic structure of a parametric model, but on standard 'dumb' CAD geometry and other raw data, they are more in line with principles of computational induction.

2.3 Application of the research

The description of contemporary practice above—both in the complexity of the design task and collaborative environment—highlights the problem with Technical Rationality and its resulting computational approaches. In unambiguously stating and codifying objective knowledge, one of the effects of science and engineering is to gradually transform tacitly understood knowledge to explicit knowledge. Empirical, tacit knowledge is handled informally as in Alexander's (1964) 'non-selfconscious' community of traditional designers, but when this community becomes 'selfconscious', its scientific, explicit knowledge is handled formally. Most design involving complex and unique situations (wicked problems) involves a substantial amount of the former, but computational and engineering tools deal only with the latter, quantifiable and codifiable. Unfortunately there are problems with both.

- *There are dangers in self-consciously reinventing entirely new forms based on the aspects of the brief that are clearly understood.* The more complex the task becomes, the more variables the designer is required to understand and the greater is the capacity for error, and optimisation and similar explicit methods solve less of the problem. The dangers are particularly heightened when variables and systems interact with one another. Large housing projects of the middle 20th century are frequently cited examples, in which many surface features such as unit volume and green space are apparently well provided but deeper structural issues were missed. The size of these determines the impact of any error—in small projects and in the slow process of unselfconscious evolution of standard types, 'mistakes' are small, local and are either removed without great cost or mediated by other small corrections in subsequent buildings. Today, with the unprecedented size of projects (§2.1.3) and design teams (§2.2.1) this is no longer the case.

- *On the other hand, it is not possible to un-selfconsciously base designs on existing forms.* Contempory technological and cultural change is too rapid, demanding novel solutions faster than can be provided by un-selfconscious copying. Again, the size of projects is also greater than can be accommodated. Reliance on previously tested designs is insufficient because in these cases there is often no clear precedent or tradition on which to build.

If reflective design methods offer a means to address these problems, they are frustrated by this dilemma. Large creative teams require communication and collaboration among more members, often distributed among different disciplines, and advanced computation and information technology. These apparently require discussion and theory and the making of tacit knowledge explicit. On the other hand, many of the problems are so complex that these become a detrimental simplification. The resolution suggested here is in the form of computational and engineering tools that are capable of dealing with the tacit elements of design. These would provide a means for allowing a discussion of sorts—i.e. the social sharing of concepts—without requiring that they ever be made explicit or inappropriately simplified.

2.3.1 Engineering complexity

The first class of practical applications for which the research is proposed is roughly that of optimisation, in allowing an optimisation algorithm to function without explicit simulations or goals, and more like a tacit, human designer. Polanyi (1967, p. 4) points out that in tacit knowing we are able to recognise something—a face, a mood or emotion—without being able to tell how we recognise it. Alexander (1964, p. 53) similarly describes how we can often recognise a ‘good’ or ‘bad fit’ of form to context without being able to describe the rules to create it—for example in the un-selfconscious design of Slovakian shawl patterns simply by not copying the bad ones. Non-selfconscious processes of craft and trial modifications of precedent thus pass along knowledge embodied directly in the artefacts themselves, without the need for decomposition, abstraction or symbolic representation. The computational tools proposed here are a way of incorporating this type of knowledge transfer into design optimisation algorithms. In effect, this is the kind of complex knowledge transferred by the sharing of drawings, models and actual buildings, but while drawn proposals are hypotheses to be generated and criticised in a hypothetico-deductive or reflective process entirely by trained designers, this research seeks to provide the machine with a similar capacity.

Behaviour too difficult to simulate may be approximated from examples. For design problems in which the goals might be difficult or impossible to state explicitly, examples of previous designs may be provided instead. The construction of a repertoire of design patterns from precedent depends upon a large selection of samples, which can be found in previous projects of a given type. For instance, the interior layout of an open office space is a generic and common enough occurrence to provide a wealth of prior successful examples from previous Foster + Partners projects. Several computational tools of this sort are possible:

- The modular space frame designs investigated are implemented on a very small scale to take advantage of rapid prototyping technology and the resulting ease of fabrication. The algorithm developed allows for the optimisation of any number of units within a reasonable amount of time, and thus will be capable of crafting materials with bespoke properties when the manufacturing process is developed further. A structure of specific Poisson’s ratio or structural stiffness can be emulated (chapters 4 and 5). Uses such as this have been recently reviewed by the press (The Economist 10.3.2005; Architects’ Journal 8.12.2005) as serving such diverse applications as aerospace and medical implants, but the ability to craft a material with gradually changing properties in architecture would be of benefit at any junction between differing materials: the delamination of carbon or glass fibre composites at their mechanical fasteners for example.

- More immediately relevant in an architectural context is the ability to address large scale structural problems. While space frames of several thousand members are handled with relative ease by a standard laptop, optimisation times typically increase exponentially with the size of the structure. With the scale of such projects as mentioned above (section 2.1), Foster + Partners is one of the few firms worldwide that is required to deal with structures on the edge of our computational abilities. The structural approaches developed here will be applicable in such projects.

- In space planning, the goals may be improved beyond simplistic requirements such as occupant density by incorporating knowledge of social interaction based on space syntax analysis, and allowing the user to specify objectives by selecting examples of previous designs. A machine-learning algorithm is applied to the spatial problem so the objectives to which designs are generated can be learned from existing examples that are known to work well. The characteristics of a particular graph derived from the configuration of an existing space can be applied to a new space of different size or geometry (chapter 7). Machine learning allows the goal to be derived from not one, but a series of known example designs (Chapters 8 and 9). If the assumptions that space determines social interaction via its patterns of visibility are correct, then manipulating and replicating properties of these graphs in generating designs amounts to the replication of social properties of those spaces. For an existing organisation moving to a new space, a tool could, for instance, transplant the layout of an existing office to an entirely different space while maintaining desired properties of its socio-spatial graphs. Particular measures need not be pre-determined to do so.

2.3.2 Communicating styles and solutions

The second set of applications are in aid of communication and the reflective working methods of selfconscious, reflective engineers and architects. They will endeavour to make personal intuition into transferrable and sharable knowledge, not by the usual explicit method of theory but by the communication of norms implicit in the designs themselves.

A modern distinction (drawn from Durand 1880) is made between *style* as a subjective matter of judgement, and *type* as a matter of utility (Westfall 1991), but in practice the fact that this utility is difficult to define ensures that many useful features may be passed on via what appears to be a style. Certainly this is the case for unselfconscious traditions that replicate precedents. Style is often considered (as in Gombrich 1960) the method of expression as opposed to the content, but Goodman (1975) argues for a broader definition of style to include aspects of both *what* is expressed in addition to *how*—the style ‘consists of those features of the symbolic functioning of a work that are characteristic of author, period, place or school’. Both type and style are of interest here for the same reasons. Both are norms resulting from social conditions that are not fixed but mutable, and both can not be adequately communicated explicitly but are held tacitly. The distinction between style and type does highlight the two distinct reasons for understanding and working with design norms.

The first, associated with what would normally be called style, is to reinforce the established identity. Foster + Partners represents one of the strongest ‘signature’ design practices in architecture, and there is very powerful commercial need to remain this way. In an office of this size (with many people, diverse backgrounds, changing staff, several locations) the unity of image appears somewhat paradoxical, and grows ever more so as the number of people increases. As the company has grown, it has been innovative in maintaining this by developing new methods and technologies (extensive materials library, intranet knowledge management) to ensure this. The use of machine learning researched here is seen as an additional means to this end. Algorithms will be presented to learn stylistic solution spaces for analysis, aid in communication and guide optimisation algorithms by refining the search criteria. This research is still in an early stage, but may offer a natural addition to the other technologies that help to maintain a collective creative unity even throughout growth and dispersal of individual staff.

The second reason for understanding norms is the functional connotation of what is normally called type: to provide design solutions that are appropriate to a particular problem at hand. While every new design situation is unique, most also overlap with a history of precedents, both historically and within a single, large design group. With some simpler problems in the course of a design, the architect knows intuitively the effect a particular move will have on the design because of experience gained in situations of a similar type. It is proposed that tools that also have a similar knowledge can help to further speed up, even automate some of the process, even for more complex problems.

Tacit design norms are not easily made explicit—again the danger is that they are simplified. Alternatively, machine learning and analytical techniques developed here (chapters 7, 8) offer a means of visualising some of the relationships between previous projects, and thereby helping to understand their commonalities. The methods are aimed specifically at handling very large amounts of data, and so are suited to the extensive catalogue of Foster + Partners' projects, and to aiding communication between the many designers that make up the firm.

1. In tools for analysis, machine learning will be applied to generalise and find patterns in example designs. If there is an implicit evolution of norms within the collaborative environment, it is a method to quantify, to record and communicate these. In cases where a human designer is overwhelmed by the amount of data and precedents available, or influenced by prior assumptions, a learning algorithm actually performs better as the data grows, and provides an alternate means to understand these.

2. Tools for communication aid collaboration between designers, as a supplement to the normal transmission of design intent via sketches and discussion. In the context of Schön's (1983) description of the process as 'framing', 'moving' and 'reflecting', 'moves' are provided by the sketches and the other steps are discussed. While these sketches are rich with implicit information, studies of problem framing (Kvan and Gao 2006) indicate that collaborative design activities may become more effectively interlinked when mediated by lower bandwidth means of communication. In quantifying relations between past designs, the numerical analytical methods proposed provide an alternate means of framing for discussion.

These same methods overlap with those of the optimisation tools above in providing a potentially more natural means of communication with the computer. All are based on the natural, tacit evaluation of design examples. In many design problems where designers are reluctant to apply an algorithmic search, it is anticipated that the more flexible approach to setting these objectives can overcome a designer's aversion to computation based on the perceived necessity to be explicit about goals. Generation and optimisation of designs by the machine are intended to save time and enable designers to work more effectively. The computer is naturally able to process data very quickly, but by changing the mode of interaction these potentially allow the machine to function more autonomously. Optimisation provides solutions strictly in a relevant context (chapter 10), and may therefore act on one level as another specialised designer within a group.

Chapter 3: Literature review

Summary: This review covers the technical and methodological background required to address the hypotheses. The simplifications required in optimisation are contrasted with alternatives to representations in complexity and dynamical systems theories, and machine learning is presented as a means for approximating observed system behaviour. Testing of linguistic competence is reviewed as a means to test the main hypotheses.

The two hypotheses address the problem of complexity at the level of the design task and at the level of the interaction of real designers who might reflect on their work, collaborate and share knowledge. This chapter will provide a technical overview of the current approaches to these issues, and suggest which are potentially applicable and where research is required.

Hypothesis A assumes explicit simulation and clear problem statements of the sort that make optimisation possible. This is at odds with the loosely formulated brief of a wicked or creative problem and the apparent unpredictability of a complex system, and so immediately raises a number of methodological difficulties. Optimisation methods will be used in this research, but one of the main purposes of inductive learning will be to allow them to be used for these complex tasks. §§3.1 – 3.3 describe the current background necessary to do so, in optimisation, complexity and machine learning.

Hypothesis B adds the requirement of communication with the computer but raises the issue of how this might be done without predefining meaning. This immediately creates a problem of how the hypothesis might be tested. §3.4 will highlight various stances on communication and cognitive theories, show their relevance to design, and outline the theory of linguistic competence from the literature to test the hypothesis. The relationship between theories of representation and action in the world will be discussed to clarify how the computer will be considered to interact with the user and the design throughout the research to follow.

3.1 Optimisaton

Optimisation methods attempt to find the best solution to a clearly defined engineering problem. Ravindran et al. (2006) describe four necessary tasks in setting up a problem for optimisation: define the system boundaries, select a performance criterion, define the independent variables that will determine candidate solutions and construct a model of the system. Two of these are the focus of this research. The performance criterion, also called the objective function or fitness function, is used to evaluate potential solutions. The visualisation of this on a vertical axis against the independent variables on horizontal axes forms a surface often called a fitness landscape[*]. The system model is a ‘simplified mathematical representation of the real system’ (Ravindran et al. 2006, p. 5). To perform optimisation, each of these must precisely defined in advance.

A range of mathematical programming techniques exist for finding the minimum (or maximum) of a sampled function (Rao 1996). These are iterative processes, in which each step consists of an evaluation of the objective function based on the behaviour of the model of the system. Based on this, the independent variables are changed, and the evaluation repeated for a different candidate solution. Through many such evaluation steps, progressively better solutions are expected to be found.

3.1.1 EXAMPLE TECHNIQUES

Techniques for optimisation may be deterministic given initial conditions, or stochastic, employing random or pseudo-random values to guide the search. The following two common techniques are both used in this work. (For fuller descriptions see, e.g. Duda et al. pp. 225–7; Mitchell 1996.)

Gradient descent is one of the most straightforward techniques and is deterministic. It is a calculus based method (Rao 1996) that employs a search that samples the gradient of the objective function in the immediate vicinity of a given candidate solution. The independent variables are then altered in the direction of this gradient. For approximately parabolic functions, this has the effect of sequentially stepping toward the minimal point or optimum solution. For these it is highly efficient, but for more complex functions, particularly those with multiple optima, it is unsuited.

Genetic algorithms (GA) are stochastic methods based principles of biological evolution, which distribute a population of candidate solutions throughout the fitness landscape. The values of

[*] In the case of multi-objective optimisation there may in fact be a number of vertical axes as well. As all cases of optimisation in this work are single-objective, this simpler description of the fitness landscape is sufficient.

their independent variables, referred to as the *genotype*, are combined probabilistically based on the fitness of the solutions, to breed the next population.

3.1.2 OPTIMISATION OF STRUCTURES

Several techniques have been devised for generating the topology of continuous solids. Both GA and non-stochastic iterative methods have been used. Many deterministic approaches stem from an iterative process of material distribution first proposed by Bendsøe and Kikuchi (1988), including the shifting of nodal points in the FEM representation toward high stress zones (Chen 2002). Alternatively, stochastic approaches such as GAs have been used to specify a configuration of holes and solid using Voronoï diagrams or a list of hole shapes (Schoenhauer 1996). These methods can determine the number and position of holes in a cantilevered plate, for instance, but do not deal with truss-like structures.

Discrete element structures (e.g. trusses, space-frames) of the kind considered here involve both the design of the topology of connections, as well as their position and size. Much early research in this area (e.g. Adeli and Cheng 1993) has been in refining only the shape or member sizes, rather than the topology (in terms of members connecting the nodal points of the structure) to optimise the weight of space trusses by determining the width of each member in a given structure. The shape and load points are fixed in advanced, and the cross sectional areas of groups of members are encoded in the genome, then selected to minimize the total weight.

More recent research has concentrated on topological optimization, or both topology and shape together. Steel frame bracing topologies for tall buildings have been designed by GA, either by encoding the possible member connections within each structural bay in the genome (Kicinger et al. 2005, Murawski et al. 2000), or evolving a set of generative design rules (Kicinger et al. 2005), or by beginning with an acceptable unoptimised solution and refine the topology by removing connections (Ping 1996). These typically begin with set node positions, but generative, rule based systems including shape annealing (Shea and Cagan, 1998) in which transformations are selected depending on design criteria, and those evolved by GA (Kicinger et al. 2005), provide more open ended approaches.

The problems of topology and geometry can also be separated and solved by nesting one within the other. In von Buelow (2002), a two stage algorithm was used in which an outer GA evolved a topology for the structure expressed as a matrix representing the structural connections, while another GA found the geometry for each member of the population, expressed as real valued node positions. Both GAs and gradient descent are used in work presented here.

Versions of the above geometry and topology optimisations have been applied to modular structures, particularly to realise desired properties in material microstructures (Bendsøe and Sigmund, 1999). These are typically arrayed as identical modules to form an isotropic structure. Material distribution methods have also been extended to find solutions requiring materials of free density and elasticity parameters (Bendsøe and Sigmund, 2003, pp. 190-204), usually approximated by composites of discrete materials in fabrication. It is for such problems that the present method is proposed, to allow parameters to change continuously between modules.

Foster + Partners, like many architects, have employed iterative geometry optimisation at a larger scale. The British Museum Great Court roof serves as one example that also reflects the ability of a construction process to fabricate a structure in which each member has been structurally optimised to a different shape and size. The triangulated steel grid shell that covers the courtyard of the existing building reconciles the contrasting rectangular and cylindrical geometry of the main building and reading room, and as such the shape and division of its surface are also complex. The 4,878 steel members and 3,312 glass panels that comprise it are each different in dimension and shape. (Sischka et al. 2004) Their specific configuration was derived by an iterative algorithm by Chris Williams at the University of Bath which tuned the geometry from an initial start position using an iterative relaxation process to derive node points in the frame (Williams 2002). To manufacture these unique elements, manufacturing contractor Waagner Biro used a modified automobile industry robot to cut the steel members to shape directly from the digital model.

3.1.3 ENGINEERING MODELS: FINITE ELEMENT ANALYSIS

In all cases of optimisation, a model or simulation of the system provides the evaluation. To perform structural and similar types of analysis, finite element analysis (FEA) is an analytical method that uses an approximate model of a real, physical system or object to predict its behaviour. The numerical basis of the analysis is known as the finite element method (FEM). FEA substitutes the usually continuous object with a set of linked, discrete and much simpler finite elements. Each of these, in itself, is far simpler to solve. As a general method for numerically solving partial differential equations, this was developed in work by Courant (1943) using triangular regions, and by Hrennikoff (1941) with rectangular lattices, and was later generalised in the mid-century work of Argyris (Argyris and Kelsey, 1960), Turner et al. (1956), etc. on the analysis of deflection and stress within complex plate structures. Although originally developed for the analysis of stress in loaded objects, the subsequent growth of computing technology has made this the standard method for such problems as fluid dynamics, heat transfer, electromagnetism and earth sciences.

The basic principal behind FEA is that a continuous and complex, non-linear function of the actual system is approximated with a set of discrete, linear functions—in the case of structural modelling these are represented by a mesh. Although the geometry of a real object is not reproduced exactly, in all its detail, the principle is that the approximation to the actual behaviour becomes more accurate for finer meshes.

3.1.4 Spatial models: Visibility graph / axial line analysis

Space has been shown to have a significant effect on social behaviour (Peponis et al. 1989; Backhouse and Drew 1991; Hillier et al. 1993; Spiliopoulou and Penn 1999; Desyllas 2000; Turner et al., 2001; Desyllas and Duxbury 2001; Hillier and Shu 2001; etc.). To represent relevant features for space planning optimisation problems, the computer requires an appropriate general method for modelling space. This spatial domain is similar to the structural in FEA in that both can be modelled as a relational system of discrete interconnected elements. Two related space syntax techniques both provide an approximation of how people actually move through or use a space, using an analysis of the plan based connections between discrete lines of sight. *Visibility graph* analysis quantifies the connectivity of a set grid of points within a space by the unobstructed sightlines that exist between them. *Axial graph* analysis quantifies the connectivity of sightlines themselves (Hillier et al. 1983; Hillier and Hanson 1984), derived from plan vertices by a standard algorithm (see Turner 2005). From each of these models, various measures such as integration, connectivity or mean depth of points are typically used to derive a statistical analysis of the space based on the plan (Turner et al. 2001).

Properties of visibility and axial graphs have been shown to be strongly related to both spatial perception and resulting behaviour of people within spaces. Strong correlations have been found with measures of visibility graphs and observed way-finding, movement and use in buildings (Turner et al., 2001), and urban pedestrian movement (Desyllas and Duxbury 2001). Axial graphs have likewise been shown to be closely related to directly observed movement (Peponis et al. 1989; Hillier et al. 1993), building use and social interaction (Spiliopoulou and Penn 1999), and indirect behaviour such as land values and crime (Desyllas 2000; Hillier and Shu 2001).

A relevant difference between the two methods is that visibility graphs are based on arbitrary grid representations, while axial lines are tailored to the morphology of the object of interest. In many cases both will yield similar values for measurements taken from them, but axial lines will be used most often in this work because of their fit to specific geometry. The methods developed in chapters 7–9 will often not use predetermined scalar measures but will use the graph itself as the machine's input. In this context, the axial graph provides better initial data for the machine learning algorithm because sightline endpoints are invariant to plan scaling or rotation.

In addition to measures of these two graph types, agent simulation algorithms (Turner and Penn 2002; Turner 2006; Yan and Kalay 2006) provide another approach to the analysis of the same kinds of spaces and will be discussed occasionally. Except for the manual calculation of measures for some small graphs, Depthmap software (Turner 2001) is used for all three types of analysis.

3.1.5 Theoretical problems of a model

The use of a 'simplified mathematical representation of the real system' (Ravindran et al. 2006, p. 5) presents problems of accuracy that are particularly relevant when stochastic or complex systems are involved, and problems for creativity and ill-defined design problems because it must be predetermined in advance.

Accuracy in modelling a complex system is crucially important, and therefore problematic for several reasons. Simon (1996 p. 147) outlines the first in that prediction requires both reliable data about initial conditions, and a theoretical understanding of the phenomena to be predicted. Many models are known to fail because no achievable level of accuracy in the data is sufficient. In a weather model, for instance, the atmosphere follows well known laws of physics, but to make reliable long range predictions the state of every molecule would be required. Such models are generally approximated statistically by assuming they are stochastic at fine levels of resolution, but it is the fine scale differences that really matter, and over time the real system diverges from the prediction. For some systems, patterns of higher order stability may be possible to find, and these will be essential in this research.

In other models, the problem is that the system itself is not fully understood. Simon makes a possible mistake of assuming many systems are "nearly decomposable", with orders of magnitude more interaction within the subsystems than at their interface (p. 197), but for complex “wicked” problems, this interface may be more complex. Complexity will be addressed in §3.2.

The other problem is the requirement that the more difficult design tasks are not understood fully in advance. Model theory (Coyne 1999 p. 86) involves: the notion of a thing to be represented; a language for representing it; a modeling language; and functions for mapping the object language to the representation language, all of which must be predetermined before optimisation or any use of the model can take place. Unfortunately it is commonly noted (Simon 1996 p. 14) that the computer can only do what it is programmed and the results of this model are no better than the assumptions built into it, and in design these are incomplete. Simon (p. 16) admits that we nevertheless do work with poorly understood systems when top down approximations are possible of the functional description of parts at their interfaces (and that science is actually built on these), but when these interfaces are not clear, regularities in the behaviour are seldom found. This is a statement of the position of Hypothesis A. It will be proposed that these regularities may exist, only more subtly (§3.2.6), and machine learning (§3.3) may be used to find them.

3.2 Complexity

Complex systems contain a large number of parts, but differ from the merely complicated, or complex *disorganised* systems (Weaver 1948) by virtue of their interaction. One of the most familiar examples of a design problem in which this organized complexity is most immediately grasped is the example of the city. In 1961 Jacobs (1961) attributed the problem of understanding a city not to the global statistical analyses then used in urban studies but to small scale interactions between people in the very immediate terms of firsthand accounts.

Jacobs’ reference is specifically to Weaver’s (1948) account of the increasing complexity of scientific thought from problems of simplicity, to problems of disorganized complexity, and finally to the then emerging science of organized complexity. The first phase, of seventeenth to nineteenth century science, dealt with simple problems of one or two variables only. The second, disorganised complexity, with problems of potentially billions of variables such as the pressure of a gas from the combined effect of many particles. Thermodynamics, heredity and early communication and information theory were developed in this class, which was viewed from a statistical point of view as probabilities. It is the third class of organised complexity, arising first in the mid-twentieth century life sciences, that encompasses the problems of urban planning that Jacobs (1961) documented. Structural systems such as space frames and textiles (Fuller 1975) and flexibly structured organisations (Gladwell 2000) are typically efficient, robust and non-heirarchical for the same reasons. All are situations that "involve dealing simultaneously with a sizable number of factors which are interrelated into an organic whole" (Weaver 1948, p539). As problems, these are described as 'wicked' by Rittel and Webber (1984) and what are normally meant by 'complex'. Whereas a complex disorganised system may have no relevant interaction between parts, in these situations the structure of these interrelationships is crucial.

This kind of complexity arises at several levels in this work. It aims to aid in designing complex structures, and in designing spaces that might contain creative social systems like the city as described by Jacobs, or a healthy workplace in which a large number of individuals are interacting effectively. These designed products have a complex organisation. At another level there is complexity in the process of design itself. Tools created to aid designers must address the fact that all creativity occurs as part of a larger social system, as maintained by systems models of creativity (Czikszentmihalyi 1988), and their historical context (Boden, 1990). Acknowledging complexity at this level helps in understanding how designs will be created and communicated.

3.2.1 A DEFINITION

The study of complex systems ranges from the particle interactions of condensed matter physics, through biology to the modelling of the economy, and complexity itself therefore has specific definitions associated with each domain. To summarise (e.g. Nicolis and Prigogine 1989; Serra and Zanarini 1990; Cilliers 1998; Auyang 1998; Johnson 2006), a general definition of complex systems would include the following:

1) They are composed of a large number of individual parts, each of which interacts with many others.
2) The interactions are non-linear and typically local. Each part can only receive and send information or physical effects to a subset of the parts of the whole system. As these may in turn interact with others, the interaction may have a global effect or feed back on the initial parts.
3) The system as a whole interacts with its environment. It is an open system that adapts.
4) At the global level, the system's behaviour cannot be predicted or described by the individual parts, but by their interaction structure. This is referred to as synergy (Fuller 1975, p.3) or more frequently, emergence (Ashby 1956, 6/18).

The first three points refer mainly to the *structure* of the system, whereas the fourth is most relevant in distinguishing the *behaviour* of a complex system.

3.2.2 Complex models, approximation and accuracy

The term *emergent* (Lewes 1891) refers to phenomena that cannot be reduced to the sum of their components, and is typically associated with the high level behaviour that arises in complex systems (Cilliers 1998; Holland 1998). The complexity of emergent behaviours has been measured in terms of information content, which has lead to two apparently contradictory implications on the predictability of an emergent system. On one hand the analysis of a system's output as with data compression algorithms (Wolfram 2002) implicitly links it to *high* levels of Kolmogorov complexity or Shannon entropy (Cover and Thomas 1991). This is expressed also by Cilliers (1998) as the condition that 'to describe a complex system you have […] to repeat the system' (p. 10), or require one of even greater complexity (p. 69). On the other hand, descriptions of emergent self-organisation in emergent behaviour link it to high levels of order, or *low* entropy (Auyang 1998, p. 321; Ashby 1956).

The two opposing assertions can be reconciled when one considers the level of detail at which the system is viewed. Entropy is counted over the number of possible states (Ashby 1956; Wiener 1948). While at the finest level of detail these are unique, at a lower level of resolution many of them become indistinguishable from one another. This has implications with respect to the modelling of a complex system and what is meant by the 'accuracy' of such a model. Cilliers' claim that it is necessary to have a still more complex system at an even higher level of detail to simulate it exactly may be true from the first point of view, but only in a very particular sense. My brain, for example, is an instance of a highly complex system, and to simulate its actual behaviour to the extent that the simulation is not only indistinguishable from a plausible human (as in the Turing test) but actually behaves *identically to me* would require at least a complete simulation of the exact relationships between every neuron in my brain. At the next level down, the activity of each neuron is itself determined by its precise physical and chemical composition. These factors are normally replaced by stochastic variables in neural network simulations, but to fully describe *my* brain, they would have to be modelled precisely. The minimal system thus required to fully describe my own brain exactly then would be a physical brain that is identical down to the subatomic level, and a computer simulation of the precise position and trajectory of every physical particle would naturally require far greater complexity.

But it is not necessary in a design context to simulate the system exactly and uniquely—the goal in modelling is not a simulation not of *my particular* brain, only of *a plausible* brain, thus accuracy is still possible. In prototypical mathematical models of complexity like the Lorenz attractor (Lorenz 1963) the overall behaviour over time is described precisely independently of any particular trajectory. Likewise, a simple visual agent (in Turner 2006) has a highly unusual and unrealistic path for a real person, but a group of these over time trace aggregate paths that simulate real groups accurately. As an architect designs for a large number of people over time, it is this emergent high level behaviour rather than the specific actions of a single individual that is most relevant.

Although this is an approximation in terms of resolution, it acknowledges the emergence resulting from the structure of the system. The kind of reductive models employed by the urban scientists Jacobs argued against are inappropriate to capture this, but the following sections will look at how this can be modelled. As with the resolution of view, approaches and accounts of emergence can either take the low-level or the high-level perspective. In low-level accounts, each of the components of a model can be replaced with another as long as the appropriate interaction between them is maintained to yield the same higher level, emergent behaviour. Fuller's (1975) description of synergy focuses on the benefit of the individual components of the system acting together to reinforce one another, and one member of a geodesic dome is essentially interchangeable with any other. The finite element, visibility and agent analyses (section 3.1) are of this nature. High-level accounts acknowledge similar behaviour between whole systems that differ in kind. Wilson (2006) demonstrates an analogy between urban and ecological systems, which allows them to be described by the same equations. In Haken's (2006) high-level description, a 'slaving principle' suggests the high-level behaviour of the system as a whole appears to supervene on its parts, guiding them into a certain behaviour. The soap films used in form finding prior to the availability of digital FEA and optimisation may be considered in this way, as individual molecules flow from one region to the next while the overall shape finds a minimal energy solution (Ball 1999). High-level modelling strategies may replace a larger system with one that is structurally different, providing its behaviour is similar.

Such approximations are fundamental to our ability to conceptualise, plan and react to complex systems. At the high level, our reasoning about the behaviour of an unknown system may be aided by a known one, by way of analogies or 'superconcepts' (Wilson 2006, 2010) that translate between systems of different kinds[*]. At the low level, we can often ignore everything but one particular scale; living systems, for example, display levels of order unique to various scales—the cell, the organism, the ecosystem—each of which is modelled at that scale by largely interchangeable parts. Both high and low levels of approximation are directly related to Hypothesis A, in that they potentially allow the prediction of the emergent behaviour of a system without fully understanding the underlying causes.

3.2.3 Simplification of low-level parts in a model

Haugeland (1978) makes an explicit distinction between three different types of reduction based on three corresponding types of explanation. *Derivational-nomological* explanations, the type employed in nineteenth century statistical physics and criticised by Jacobs (1961) in understanding cities, express relationships between quantitative variables in the form of equations. In contrast to these, he defines *morphological* explanations as those dependant on a specific form or structure, like the double-helix model of DNA, and *systematic* explanations as those describing the organised interaction between distinct parts, as required to explain a television or automobile engine. It is a defining characteristic of a *system* that it is composed of interacting *functional components* (Haugeland 1978), and these components are in principle replaceable by others that function equivalently (Haugeland 1995), such as a transistor in the television or (arguably) a neuron in the brain. The points at which the components interact are their *interfaces* (e.g. connecting wires, synapses), where interaction is relatively simple and well-defined, and it is at these points that the replacement of a component may be made. Systematic explanations define the behaviour of each component in terms of interaction at the relevant *interfaces*, regardless of the particularities internal to each. By describing the specific manner in which the set of components interact, systematic reduction eliminates only the detail of these particularities. While *derivational-nomological* and *morphological* approaches violate point 4 of the above definition of complexity by seeking to understand the system via the parts themselves, the *systematic* approach deals precisely with the interaction structure mentioned.

Systematic explanations thus allow a model of the system to be made by replacing components of the system with simpler ones that are functionally equivalent at their interfaces. At this collective level of interaction the low level simplification is of predictive use, both in our own mental models and in computer simulations of a more complex world. At this level the *specific* motivations of each agent are irrelevant in comparison to how they are connected, a feature of

[*] Complexity (organised and disorganised), systems and computer models are all described as superconcepts in themselves in Wilson (2010).

complex systems generally. Arthur states in regard to the economy: *"How* individual agents decide what to do may not matter very much. What happens as a result of their actions may depend much more on the interaction structure through which they act – who interacts with whom, according to which rules." (Arthur et al. 1997, p.9) The simplifications possible in these approximations are therefore not the same as Weaver's (1948) *disorganised complexity*, in which a simple statistical assumption of independent parts (e.g. gas molecules) is sufficient. Instead, Ashby (1962) makes the point that a theory of the *organised* system is concerned not with intrinsic properties of the parts, but the relations between them. This connectedness of the parts and their mutual influence are therefore the most important aspect of the simulation of a complex system.

It is the knowledge of this interaction structure that makes prediction possible, by making the assumption in the model that this structure remains constant, and mimicking the behaviour of elements within it. Johnson (2006) explicitly distinguishes system dynamics within an existing, stable structure (or "backcloth relations", such as a network) from changes to the structure itself, to make the point that all (currently known) simulation methods only allow prediction of the former[*]. The latter are often assumed, ideally as constants. The persistent and known street network in a Space Syntax model and the structure of agent communication in the El Ferrol bar problem both provide the means by which phenomena as complex as group social dynamics can be predicted. These predictions are also thereby coherent with Popper's account of inductive inference to the extent that greater structural change (i.e. of these backcloth relations) may be considered the kind of singular, improbable, arbitrary event setting it apart from the 'best tested' hypotheses (§1.2.2).

To make prediction not only possible, but feasible in practice, differences within the individual components can be to some extent ignored so long as they don't affect the overall structure of interaction. The microscopic interactions of specific people involve distinct personalities that, at the ultimate level of accuracy, are well beyond the reach of a computationally tractable predictive model (after Cilliers 1998, p.10, et al., above), but in constructing a bottom up simulation of a large group these specific differences are generally omitted. Such study of complex systems involving the interaction of *many* people has been useful, particularly in economic and sociological modelling (Ball 2004), and this does not rely on an accurate portrayal of individuals. "While knowing the rules that govern the behaviour of individuals does

[*] Johnson (2006) defines these as type-1, for changes in traffic within an existing set of backcloth relations, and type-2 for the changes to the relations themselves. This should not be confused with Clark and Thornton's (1997) definition of type-1 and type-2 learning problems in §3.2.6.

not necessarily help us to predict the behaviour of the mob, we *may* be able to predict the very same mob behaviour without knowing very much at all about the unique personalities of the individuals that make it up." (Watts 2003, p.27) Space syntax analytical methods using axial lines (Hillier et al. 1983; Hillier and Hanson 1984) and agents in Arthur's (1994) "El-Ferrol Bar" simulation of group equilibrium also rely on such approximations of parts. Real people have many factors influencing their decisions—histories of past experience and goals for the future—yet we share the property of navigating by sight and we share the ability to notice when a bar is full to capacity. The factors that differ between us may be assumed to do so randomly, so they average out and may be removed from the model. What remains are the lines of sight, whether incorporated into an agent model or measured from an axial graph. Models in other domains parallel these social models—the design of a geodesic dome is simplified by assuming all the members are roughly equal, therefore only one needs to be worked out in detail.

But care must be taken to ensure that the reduction does not obscure the relevant structural details, particularly in the case of the interface between parts. Simon's (1996) approach to engineering design is based on the low-level simplification of the systematic model described here. In his view, scientific approximations of relatively poorly understood systems are possible because of the modularity and organisation of parts and their functional description at these interfaces (p. 16), and these are easily made because most natural systems are "nearly decomposable"—there are orders of magnitude more interaction within subsystems than at their interface (p. 197). Many systems are not so easily decomposable however—particularly in the biological sciences, problems of organised complexity and "wicked" problems of design—as the interfaces are often highly complex in themselves. In describing the nature of the interface between real agents and their environment, Clark (2008, p. 33) states that interaction is well defined, but not a 'narrow bandwidth' bottleneck. In the structural modelling to follow (chapter 4) the interface between units will be just as complex as anywhere else. In the learning of objectives for the spatial problem (Chapters 8 and 9), a high bandwidth interface with the design problem will be used instead of simplified symbolic communication.

The structure of connections is also crucial. These have been studied extensively in graph theory and its application to sociology. Original studies of random graphs (Erdos and Renyi 1959) have been found not to model much of the behaviour of real social networks as different personality types pass information in different ways (Gladwell 2000) and there may be distinct types of connections between them or structure within the graph (Granovetter 1973; Watts 2003). In space syntax analyses it is the space itself that determines this structure. Realistic models of emergent global behaviour have been produced by concentrating on the connection structure of the society, for example in modelling the spread of ideas in design (Saunders and Gero 2001; Sosa and Gero 2002, 2005). The finite element method used for analysis of structures also utilises the topology of connections in simulating the properties of crystalline or non-homogeneous organic materials such as wood (Astley et al. 1997). The connection structure might also change dynamically, as is the case in particle simulations mimicking the motion of fluids, cloth or astronomical bodies (Dubinski et al. 2003) that use arbitrarily detailed parts for accurate calculations of viscosity or gravity, and flock behaviour simulations that rely on local influence (Reynolds 1987).

In addition to simplifying the internal behaviour of each part, a further simplification of these models can be made by reducing the overall number of parts, to approximate the system at a lower resolution. The reduced model is an approximate *homomorphism* (Ashby 1956, 6/12) of the first, in that a single one of its parts replaces several parts of the original with little discernable change in behaviour. Both structural and spatial analyses are alike in the fact that they reproduce the behaviour of a continuous or very high resolution discrete system (a field of socio-spatial interaction or a physically loaded object) as a collection of discrete, interconnected elements. In the case of the finite element method, it is by definition a continuous function that is replaced by a finite number of piecewise linear functions. The new model is only approximately homomorphic however, because as the number of these is decreased it will approximate the original function less. The value in simplification at this level is that this generally degrades gradually as long as the resolution is high enough to model relevant details. One must take care, however, to remain above this threshold: in the case of FEA, to ensure the mesh size is no larger than minimal structural elements, in the case of VGA, to ensure that the grid size is no larger than minimal corridors, openings, etc.

3.2.4 Complexity or chaos

An often noted property of the global behaviour of many complex systems is their robustness or stability over time. This is evident in comparison with a particular type of system, the chaotic system, which is also formed by the interaction structure of many parts. In terms of the four points definition above, the interactions in both systems are typically non-linear (points 1 and 2), and both display a higher level, global behaviour that is emergent (point 4). Cilliers (1998, ix.) proposes that the number of relevant parts may be less in chaotic systems (point 1), but the important difference between the two is in how each reacts to changes in its environment (point 3). Both systems are open, but external conditions have very different effects. Chaotic systems are characterised by an extreme sensitivity to initial conditions, and therefore to any external perturbation. The classic example of such a system, and source of the so-called ‘butterfly effect’, is that of a weather prediction model (Lorenz 1963) that displays a vastly different state after a given time if its initial state is changed only slightly. Systems described as complex do not have this extreme sensitivity, but tend toward a relatively simple, or stable behavioural pattern at the level of the system (Colander 2000) in that the particular interaction structure of the parts allows them to adapt to such changes. They are self-organising (Ashby 1956).

The behaviour of such stable systems can be said to evolve toward a particular region of state space known as an attractor. This may only exist when the state space maps the high level behaviour of the system as a whole, such as the convergence of behaviour over time in agent-based models (Turner 2007; Arthur 1994). It is this relative robustness over time that contributes to our ability to usefully model a complex system (section 3.4) in a way that is impossible with a chaotic system, and will be a factor in the application of machine learning algorithms (section 3.5).

3.2.5 Approximation of higher level behaviour

Hypothesis A requires that to model high level behaviour, the lower level behaviour must be known. The existence of stability and attractors in systems suggests the possibility that this may not always hold. Falsifying this hypothesis for a particular system will require that regularities can be found in its high level behaviour that allow this to be predicted. With certain systems this is possible in practice.

The hierarchy often described in the structure of systems (Simon 1969; Haugeland 1978) is due to the fact that they can be described at multiple levels, and in addition to approximating the individual parts or components of a system, it is sometimes possible to approximate the behaviour of the higher level sub-system as a whole. In finite element analysis, for example, Rudnyi et al. (2005) employ model order reduction, which considers the behaviour of the system at a high level, and computes a subspace of the possible states of a system of finite elements in which the model is likely to move. In two models of bone of 130 000 and 900 000 degrees of freedom, approximations of only 10 and 25 degrees of freedom were able to accurately provide harmonic simulations up to a threshold frequency. In encapsulating the behaviour of a detailed model in a low-dimensional approximation, this analytically replaces the model’s differential equations with a reduced set that don’t relate to any individual elements in the mesh but produce a similar result overall.

Such approximations have been made with an underlying knowledge of the model's structure, but this is not strictly necessary. Statistical approximations are sometimes used, as in the higher level emergent behaviour evident in Ashby's example of high level approximation necessary in engineering design.

"Were the engineer to treat bridgebuilding by a consideration of every atom he would find the task impossible by its very size. He therefore ignores the fact that his girders and blocks are really composite, made of atoms, and treats them as his units. As it happens, the nature of girders permits this simplification, and the engineer's work becomes a practical possibility." (Ashby 1956, 6/14)

This seems like a simplification of parts as in the previous section, except girders have a very different interaction structure from atoms, and here a whole system of the latter are simplified to a single approximation—the girder. For practical purposes, even the interaction structure between the atoms (so important for low-level systematic models) does not need to be known. Rather, the model is derived from physical tests of similar material samples (Addis 2007). Much of engineering is founded on such statistical data.

Our treatment of a system in which the inner structure is unknown is termed in cybernetics a *black box*, as in the case of a child opening a door without needing to know the inner working of the latch (Ashby 1956, 6/1). One takes advantage of the concept of *isomorphism* (Ashby 1956, 6/8; Hofstadter 1979, p.49), in which two systems are *isomorphic* if they have a one-to-one correspondence between their states and transition between them in the same way. Door handles are isomorphic even if their internal mechanisms are different, so we can function with a mental model that is isomorphic without knowing these details. In more complex systems with a much greater number of states, we may be unable to distinguish between certain high-resolution differences, and in this case may employ a *homomorphism* (Ashby 1956, 6/12): a system in which a fewer number of states can be transformed one-to-many to the states of the first. The transition functions of such machines are assumed to be deterministic at the lowest level, but a problem arises for the observer when one is unable to distinguish between two initial states that lead to different consequence states, for the homomorphism is no longer applicable. Pask (1961) suggests two possibilities, either examination of the system at a higher level of detail or statistical approximation. The first is practically feasible only up to a point, and certainly with continuous or high resolution systems the model will ultimately rely on the statistical approximation.

This suggests a homomorphism of some systems can thus be constructed without knowing their internal details, but models are susceptible to being pushed beyond the limits of their validity—

a bridge may fail when a girder is sized larger than the context of prior tests. With respect to homomorphic modelling of biological and other complex systems, Ashby makes the point that they should be complete within themselves. The following homomorphism (Ashby 1956) is a very small portion of the whole system of integer multiplication:

$$\begin{aligned} \text{Even} \times \text{Even} &= \text{Even} \\ \text{Even} \times \text{Odd} &= \text{Even} \\ \text{Odd} \times \text{Even} &= \text{Even} \\ \text{Odd} \times \text{Odd} &= \text{Odd} \end{aligned}$$

but it is internally complete, while this second set of statements (Ashby 1956) is not:

$$\begin{aligned} 2 \times 2 &= 4 \\ 2 \times 4 &= 8 \\ 4 \times 2 &= 8 \\ 4 \times 4 &= 16. \end{aligned}$$

In practice, incompleteness in the model is minimised by testing and sampling many states of the system. The fact that error estimation and model accuracy improves with the size of data sets (Reich and Barai 1999, Duda et al. 2001) is an essential principle in machine learning.

3.2.6 Approximation of complex organisation

An even more pressing problem is that the statistical approach may be dangerous when applied to organised complex systems (e.g. Jacobs 1961) as it completely misrepresents their structure. The organisation of parts yields regularities—if they exist—that are more subtle. The fact that we live in and deal with an environment of complexity is some evidence of identifiable patterns, however, as we rely to some extent on top down simplifications to cope. Classical economic theories in the tradition of Adam Smith (1776) and many modern counterparts influenced by game theory (Nash 1950) are based on an equilibrium established by rational agents. By contrast, more recent theories relevant to complexity admit that we do not behave as purely rational beings with an omniscient view of the world, as we must make do with only partial information, and instead rely on *‘approximate’* or *bounded* rationality (Simon 1955), just as do the agents in the El Ferol simulation (Arthur 1994). The human brain in this view has evolved not to perform logic but as an excellent recogniser of patterns. Thus in dealing with complexity in the world we naturally construct simplified internal hypotheses or schemata to model the world through a process of inductive reasoning (Arthur 1994; Holland et al. 1986; Rumelhart 1980).

The task of approximation differs from a standard statistical approach in that the pattern to be found in the data is not obvious. Clark and Thornton (1997) make a distinction between two types of learning problems: type-1, in which relevant regularities in data are immediately statistically apparent; and the more difficult and complex type-2, in which the properties to be learned involve relations between inputs, so the data must be recoded before regularities are visible. They note that the second is far more prevalent in real world data, and that human learners are clearly able to deal with these. Certainly most interesting and creative design problems are of type-2, thus potential solutions to these require a similar recoding.

Clark and Thornton assert that what appears on the surface to be type-2 learning is really the reformulation of a type-2 problem in "terms that reduce it to type-1" (Clark and Thornton 1997, p.64). But in raising the question of where such recodings originate they reject a nativist explanation. They suggest instead (as does e.g. Wheeler 2005, p. 80) that language and culture have evolved as methods for transferring or preserving relevant structures, thereby reducing type-2 data to type-1 problems. The ordering of training data in language learning (in Elman 1993) demonstrates one example of how "[…] a problem whose mature expression poses an intractable type-2 learning problem can be reduced to a developmental sequence of tractable type-1 mappings. Moreover, this can be achieved without going so far as to build in the required recoding bias at the very outset. The full nativist solution favoured by Chomsky is, in such cases, not compulsory." (Clark and Thornton 1997, p. 62). In whatever form they are shared between communicating agents, these structures or recodings are an essential part of communication. Their transmission may be considered either as an additional part of the data set itself or as an unspoken social context in which the data are normally seen. Shared tools of analysis such as space syntax methods may also provide such recodings. Clark and Thornton suggest we may have a repertoire of previously learned recodings that can be searched through in response to new data (p.64) and many theories of creative insight (e.g. Schön 1963, 1983; Koestler 1964; Akin and Akin 1996) propose similar phenomena as the mechanism for analogical reasoning.

The learning of recodings is achieved via 'external scaffolding'—external structures like language that are used in thought (Clark 1998), which may be interpreted in two different ways. In the first interpretation, these may involve structured *processes* of learning, like the requirement that simple sentences must be mastered before more complex, nested statements (e.g. Elman 1993), or *higher level properties* inherent in the data that is used to learn, like Plunkett and Sinha's (1992) observation that the relative frequency of irregular verbs in English is the same as that required in a data set for stable learning by a neural network (Clark (1993, p. 160). These may be structures that are intrinsically stable, sets of shared social practices over time, or structures which (Clark and Thornton 1997 suggest) may have evolved to fit the human biological profile, but in these cases the structures are far from arbitrary. The second interpretation is that the scaffolding itself can be learned, as in the case of the tags and labels that constitute language (Clark 2008). These are assigned to a complex variety of real objects in the world, determine our perception of them as equivalence classes (ibid. p. 46) and thus serve to structure our thought. In this case the structures are to some degree arbitrary (in the sense that class membership may differ between languages and multiple class schemes may coexist with a single language) and can be communicated by increasing the data set to include meta-data of relevant tags or labels. This second interpretation thus permits the inductive learning of such recodings by an arbitrarily structured agent and is the interpretation taken here. Class labels will be crucial to supervised learning in later chapters.

To refute Hypothesis A it is a ‘recoded’ statistical approach that will be used, providing a homomorphism or isomorphism of the system without being provided its internal details. As the patterns to be found are more difficult and subtle due to system complexity, the test of hypothesis A for each domain will be whether these can be found. It will be the task of the machine learning algorithm, as described in §3.3, to do so. To allow this, a set of recodings must be transferred along with the more apparent data in communication with the machine. In the case of design much of this is certainly unspoken, and takes the form of observed performance of finished designs on one hand, and social cues, norms and fashions on the other. Communication with the machine will be discussed in §3.4.

3.3 Learning

The strategy to be employed in approximating the behaviour of systems will involve machine learning from observations of those systems. Induction can be either hypothetical or enumerative (Harman 1992), in which the first derives an explanatory hypothesis to account for observed events, and the second makes a generalisation from instances. It is enumerative induction that is being used when a homomorphism is used to predict the behaviour of a complex system in the previous section. As explained in the introduction, machine learning will be applied to unknowns at one of two points in the design/optimisation process, to generalise either:

- the behaviour of the system, when one is designing with a system not known accurately enough to simulate, or

- the desired performance or objective, when this is unknown.

In both cases the algorithm can learn from precedents via induction. Initial data to be used in training any learning algorithm can typically come from several sources, including experts, previously published historical or experimental data, and simulation (Reich 1997). Machine learning techniques themselves can broadly be classified as either supervised, which attempts to find a function to map observed inputs to observed outputs of a system, and unsupervised, which attempts to find structures in a single data set. This section will give several examples of each that are relevant to design, and discuss the kinds of algorithms to be used in this work.

3.3.1 UNSUPERVISED LEARNING AND STATISTICAL ANALYSES

Unsupervised approaches to observed high dimensional data commonly involve a combination of dimensionality reduction and classification by cluster analysis (Duda et al. 2001). The automatic generalisation of a descriptive concept from examples has been explored in domains such as musical style (Cope 2001; Tillmann et al. 2004), but is more firmly established in techniques of machine classification and learning used in fields like machine vision (O'Reilly and Munakata 2000) and linguistic analysis (Burgess and Lund 1997; Reiger 1983). Dimensionality reduction is often used in applications such as face recognition (Turk and Pentland 1991) to infer distinguishing features from a given set of high-dimensional data. Principal component analysis (PCA) provides a new set of axes aligned to the characteristic vectors (eigenvectors) of the covariance matrix of the data set. The dimensions in which the data varies least are discarded to yield a lower dimensional subspace of the data that can then be used to make classifications. The principal components of face images, for example, referred to as 'eigenfaces' (Figure 2.1) (Turk and Pentland 1991), are used by face recognition software to effectively quantify a new face by how it measures against each, and its best match found from existing data.

More closely related to our experience of architecture and space is the problem of a robot visually navigating through a real, unstructured environment (Brooks 1991). Dimensionality reduction has been used on the image data recorded by the moving camera of a robot's visual system to allow the computer to generalise its own concepts for objects. The thousands of single-frame images of an environment may be plotted against the set's eigenvectors, wherein they fall into distinct clusters which generally correspond to objects such as 'tree' or 'rock' (Durrant-White 2004; Kumar et al. 2006). It makes the same kinds of distinctions and classifications that we would, thereby allowing it to navigate.

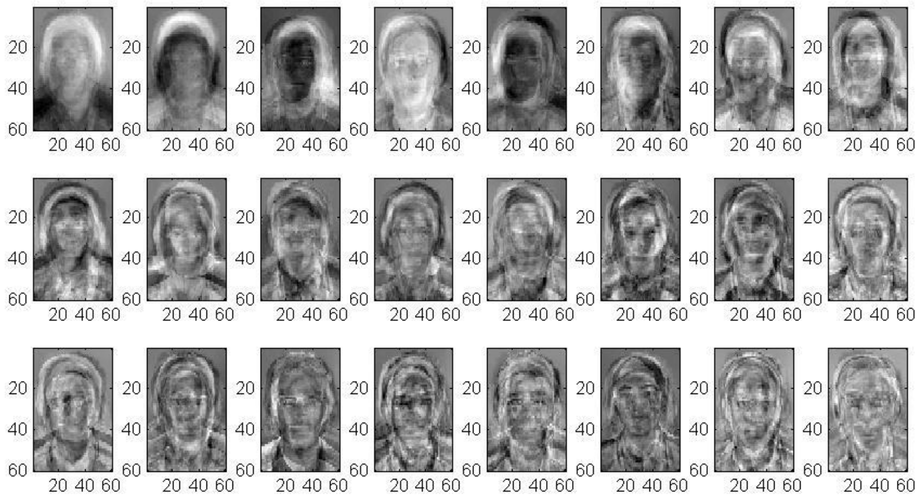


Figure 2.1. ‘Eigenfaces’, the first 24 principal components of a set of images.

3.3.2 SUPERVISED LEARNING

While the data in the above examples is all of a single type, such as images, many situations provide data naturally in two matched forms—a multidimensional input associated with a typically one dimensional label. In such situations supervised learning is used to derive a function that maps one to the other. Often this takes the form of classification, in which the labels define discrete sets, but the same methods are used in regression in which the function is real valued. Supervised learning algorithms are used in many of the same domains as unsupervised, including linguistics (Elman 1991, 1993) and models of visual perception (Kumar et al. 2006), and supervised approaches have long been applied to structures and in the domain of civil engineering, most commonly as an enhancement of the optimisation process.

In the design of complex systems, the most common use of supervised learning is in the prediction of system behaviour, usually by means of regression. An example is structural prediction in the field of bioinformatics, in which the molecular composition of proteins can be too computationally expensive to simulate fully. One stream of research is in the prediction of the secondary and tertiary structure of proteins by machine learning, where the inputs are the actual DNA string and the outputs are the predicted three-dimensional structure of the protein. Various learning algorithms have been used, including artificial neural networks (Meiler and Baker, 2003) and support vector machines (Wang et al. 2004). Various machine learning algorithms have also been used to find a function to predict movement in time of a dynamic system. The simulation of a physics-based model is possible to an arbitrarily high degree of accuracy, but computationally demanding, and the emulation of this behaviour by a trained learning algorithm is more efficient. The NeuroAnimator uses a neural network trained on physics-based models to produce realistic animation of systems ranging from a pendulum to the swimming of a dolphin. (Grzeszczuk et al. 1998) The method also serves as a control mechanism given a goal (such as balancing the pendulum or swimming toward a target) in the environment, and in this case is similar to the problem of optimization.

In optimisation problems, a recurring bottleneck is the simulation of a design's behaviour, which can either be time consuming due to the complexity of the model, or simply incorrect due to incomplete knowledge. This can be addressed by 'shallow modelling' a system's observed behaviour with inductive learning (Arciszewski and Ziarko, 1990). Discrete, symbolic[*] learning methods have been used to construct rule-based systems, which draw relationships between design parameters that predict the performance of systems from individual beams (Arciszewski and Ziarko, 1990) to the steel skeletons of entire buildings (Szczepanik et al, 1996). Sub-symbolic inductive methods such as artificial neural networks have been used also to predict structural and material performance (Reich and Barai, 1999) and the behaviour of mechanical systems such as propeller blades (Reich and Barai, 1999; Neocleous and Schizas 1995).

Another use is in guiding the search, to get around the other major hurdle in optimisation—the repeated iteration of generating and evaluating solutions. Inductive learning has been found useful to improve the speed and quality of this loop by reusing knowledge of previous designs or iterations. Murdoch and Ball (1996) have used a Kohonen feature map to cluster bridge designs in an evaluation space, and Schwabacher et al. (1998) have used a symbolic learning algorithm, C4.5 (Quinlan, 1993), to select appropriate starting prototypes and search space formulations for a parametric optimisation of yacht hull and aircraft designs. Both allow a rapid re-evaluation of previous work which improves the optimisation when run again to new specifications or fitness criteria.

While much previous research has concentrated on inferring rules to guide a design in optimisation (Arciszewski and Ziarko, 1990; Szczepanik et al, 1996; Reich and Barai, 1999; Neocleous and Schizas 1995), or on suggesting a starting point on which to improve (Murdoch and Ball 1996; Schwabacher et al. 1998), induction will be used in chapter 5 to derive a function

[*] Representations used in learning may be symbolic, in which the finest constituents of the system are the representational tokens themselves, or sub-symbolic, in which representations are composed of still finer elements that derive their meaning only in context of one another. The connection weights of a neural network or the parameters of a support vector machine kernel function, both used in this work, are sub-symbolic in that it is their combination that is required to produce a representation, and any individual weight or parameter in isolation is meaningless.

that directly maps a given load condition to an optimal solution. It is the aim of the present work to use a learning algorithm to replace the optimisation process entirely – both the simulation and evaluation loops.

3.3.3 Algorithm selection: numeric versus symbolic

The goal of the algorithm, in this case, is induction: to derive a generalisation based on previous evidence of optimal structural solutions or spatial arrangements. The choice of algorithm is frequently dependent on the context of the learning problem at hand, including the form and availability of data, and the goal of learning (Reich 1997).

A crucial distinction in the form of the data is between a *symbolic* and a *numeric* (or quantitative) representation. The symbolic approach uses a series of tokens to represent the data, but these are arbitrary, so the system of tokens and their mappings must be predetermined completely. The numeric, by contrast, can take advantage of natural relationships that are inherent in any comparison of number (Wheeler 2005, Van Gelder and Port 1995), and can thus represent any new data previously unseen. Communication of new data between parties using a numeric system is immediately possible as it requires only a judgement of similarity (or difference) between data to assign a relative position within the scale. Theories of metaphor as the basis of communication (Lakoff and Johnson 1999) frequently use these inherent relationships to explain the possibility of mapping across domains such as space to time[*] (not to mention the process of 'mapping' itself). Communication via a symbolic system is not immediately possible without predetermining the meaning of every symbol. If these are not already known, they may be discovered only by the extent to which the symbol system coincides with its semantic content. In the case of a formal system like arithmetic, for example, the meaning of symbols in theorems (e.g. "2 + 3 = 5"; "1 + 0 = 1") may be presumed given that the whole set of theorems corresponds to reality (i.e. 2 + 3 = 5; 1 + 0 = 1), but as the symbols are arbitrary the system is brittle, and this mapping can only be found by a process of conjecture and test (Popper's (1959) hypothetico-deduction) rather than by enumerative induction. For any non-obvious mappings this is unrealistic due to the potentially infinite number of tests required.

A further distinction may be made within numeric data between the analogue and the digital. Goodman's (1976; see also Haugeland 1981) definition of the digital consists mainly in the

[*] The degree to which such quantitative domain concepts themselves are inherent or a priori (as space and time are supposed e.g. by Kant 1797, A77) may be disputed, but if they are learned it is clear this occurs early and unproblematically. They would appear to be a 'type-1' (Clark and Thornton 1997) problem (see §3.4.4 below).

possibility of falsification, i.e. for any pair of types it is possible to determine that a particular token does not represent one of them. With respect to Popper's (1959) notion of falsification, which is possible only of *statements*, it is evident that in addition to symbolic statements, digital statements may be used, but not analogue. This presents a potential problem with respect to the real world, in which events are usually found to be placed in a continuum without clearly marked boundaries—the world is analogue even if its representation is digital. The solution to structural shape and the characteristics of a space that will be addressed in this work are best treated as continuous functions, and it has been noted that their discretisation is detrimental to optimisation performance, or can lead to large learning error rates (Reich, 1997). The boundaries between digital tokens are given in practice by non-overlapping, allowable variances in a signal which prohibit ambiguous statements near the boundary. It can be seen that the same occurs in the case of clustering algorithms that group continuous, analogue data into discrete categories, and in the learning of perceptual representations in the brain (e.g. phoneme distinction in speech, Oudeyer 2006). It will be required of the chosen learning algorithm that they be capable of making this distinction from analogue data (particularly chapters 8–10).

An algorithm operating on a numerical representation is therefore essential. Duffy (1997) lists six major machine learning techniques used in design related research. Two of these (agent-based learning and knowledge compilation) are primarily concerned with the learning of action or procedure, rather than solutions themselves, and a third (genetic algorithms) is an optimisation rather than a learning technique in the strict sense. The remaining three potentially apply to learn from pre-existing design precedents.

- What Duffy terms *induction* – specifically symbolic induction – allows a general rule or pattern to be generated to fit the data. Symbolic algorithms with discrete output such as rough sets (Arciszewski and Ziarko, 1990) and C4.5 (Quinlan, 1993), yield explicit classification or parameter ranges, and have therefore been used to estimate behaviour or recommend design decisions in symbolic or labelled form.
- *Analogical* or case-based reasoning techniques explicitly represent past examples in such a way that they can be retrieved and adapted to suit new problems.
- *Artificial neural networks* are part of a class of sub-symbolic algorithms (including, more recently, support vector machines (Vapnik, 1995)) that can result in a continuous output, and therefore interpolate exact output values to a finer degree than is specified by the input set. These also perform induction in the form of a continuous function.

There is some overlap between these, the division being based largely on conventions of practice, and some aspects of all three are therefore relevant in this work. If the first two techniques are implemented as numerical rather than the symbolic methods described, they become the methods by which prior precedents are generalised and represented respectively. The sub-symbolic third category represents the actual algorithms by which this is accomplished. As the problem is real valued overall and output is of higher dimensionality than input, it is this sub-symbolic class of algorithms that is appropriate. The relationship of this category to the symbolic tradition of classical AI is described in §3.4 below.

3.4 Computational approaches to communication, cognition and action:

Hypothesis B will be concerned with visual and spatial aspects of design and therefore with communication in a broader sense than language alone. As an explanation of how communication and emergent order might occur in the context of a community of designers, Hillier and Hanson (1984) introduce the concept of *morphic languages*, in which the expressions of the language may be the various forms of designed artefacts. This is explained via the notion of the genotype as a formal description of the underlying rules that generate the system; however, as there is no coded equivalent to the DNA of biological genetic transmission, an 'inverted genotype' is proposed in which underlying descriptions are repeatedly retrieved from instances of the language. This allows a distinct mechanism for change that does not exist in natural genotypes, as the system has no separate genetic memory (Hillier and Hanson 1984 p.44) and can thereby undergo revolutionary change both by deliberate action (i.e. design) or by external perturbation (e.g. catastrophe). As an insight into design the theory reinforces those treating design as a conversation with the artefact or drawing (Schön 1983; Lawson 2006) and larger systems views (Czikszentmihalyi 1988) in that each stage of transmission is not directly at the level of genotype, but via a phenotype in the world. It also reinforces those stressing interpretation (Snodgrass & Coyne 2006) in that this phenotype may be reinterpreted at each stage.

It is as a theory of languages, however, that it helps clarify the nature of communication with respect to Hypothesis B, in that it stresses the distinction between an underlying 'genotype', or abstract rule, and the instances of that rule. Transmission in design is unlike biology but like language in this sense—instances of design themselves are made and interpreted much like utterances. Because of this similarity, the phenomenon of Hypothesis B (communication of design with the computer) is analogous to competence in a language, and the testing of this hypothesis will be based on testing linguistic competence. The means for doing so is given by linguistic theory.

3.4.1 LEVELS OF COMPETENCE: PROCESSING AND STRUCTURE

Theories of linguistic competence differ in their assumptions of psychological reality of a language. In discussing competence more generally, Devitt (2006) makes an essential distinction between *structure rules* and *processing rules*. Structure rules, when they exist, govern a system's actual outputs (e.g. chess players are bound by legal moves for each piece), while processing rules govern the production of those outputs (e.g. the grandmaster's psychological process likely involves broader strategy and memory of configurations). Grammars are structure rules, about a purely linguistic reality—a *symbolic* system, not a psychological reality. To display competence, it is necessary only that structure rules be respected, and so for a language it is just these that need to be shared.

The relationship between structure rules may be one of two alternatives. The classical cognitive science approach assumes structure and processing rules are identical. Chomsky doesn't make the distinction so his view on this point can't be stated with certainty, but his work on syntactic *structure* is generally intended as describing also a psychological *process* (Chomsky 1957, 1965; Clark 1990; Seuren 2004; Devitt 2006). Devitt (2006) notes that this assumption is unquestioned among many classical linguists, and Chomsky takes it as "[obvious that] every speaker of a language has mastered and internalized a generative grammar that expresses his knowledge of his language" (Chomsky 1965, p.8). This assumption does allow for differing degrees of competence among different speakers ("*his* knowledge"), but to the degree that effective communication exists it assumes shared processing rules.

An alternative approach is that while structure rules are shared, processing rules may be very different for each speaker. Competence requires only that processing rules 'respect' the structure rules by producing that structure (Devitt 2006). Only their outputs must be isomorphic.

The same distinction holds for theories of competence in cognitive science more generally (Clark 1990). Here the distinction is between Marr's (1977) level-1 theory of what is being computed and the level-2 algorithm for how*, which are roughly equivalent to structure and processing rules respectively. Classical cognitive science—'GOFAI' (Haugeland 1998) or 'cognitivism' (Varela et al. 1991)—is based on the premise that intelligence is a result of

* Marr (1977) actually uses the term 'type' rather than 'level', referring to the highest level (1 being higher than 2) at which a given theory of competence is valid. Classical AI seeks type-1 theories. Other sources, following Marr (1982), use the term 'level' and refer also to a level-3 corresponding to the actual hardware implementation.

symbol systems of the right kind (Newell and Simon 1976), which are manipulated by formal and a priori rules. Clark (1990) therefore defines the classical approach in part by the fact that at level-2 it directly represents these symbols and carries out these processing rules, both as described at level-1. Clark's alternative approach is connectionism, in which there is necessarily a 'dam' between levels 1 and 2, introduced by a dimension shift between any symbols used and the processing of the neural network. Although not named as such, statistical and dynamical systems (Van Gelder & Port 1995) approaches are also non-classical alternatives.

The display of competence is concerned only with the level of structure rules (or level-1). In evaluating the success of communication between individuals or with a computer it is only this that matters. Such evaluation is intersubjective in Popper's (1959) sense.

This distinction provides more precision to the notion of *meaning* referred to in Hypothesis B, which states that communication requires prior, shared definitions of meaning—roughly the computational equivalent to nativist explanations of human communication (e.g. Chomsky 1957; Fodor 1983). As individual psychological processes are given by processing (not structural) rules, Hypothesis B requires that these processing rules be shared. There are three possible cases in which this is true:

i) In a classical model this is necessarily the case. These operate formally on pure syntax, not semantics, and take for granted an equivalence between the two levels.

The alternative (e.g. connectionist) approach, on the other hand, admits the same structural rules may be produced by different underlying processing rules. If they are the same, either:

ii) They are innate. If a Chomskian universal grammar exists in the mind it is at some level connectionist, would likely be embodied rather than represented (Devitt 2006, p.244). One interpretation of the connectionist method of investigation[*] (Clark 1990) is that networks share properties (size, connection configuration) with the human brain if they make the same kinds of generalisations as we do (Dreyfus 1992, p. xxxviii). The 'language of thought' hypothesis (Fodor 1975) could also support this.
iii) They are learned early on before communication begins. This appears to raise the issue of an infinite regress that can only end in nativism (Devitt 2006), but it may be that prior learning structures subsequent understanding (Clark and Thornton 1997) and that

[*] Where level-1 is an explanation of what is to be computed, level-0.5 is a still higher level 'task analysis' and level-1.5 is a more refined theory that falls short of a full algorithmic description (Clark 1990). The connectionist begins with this level-0.5 description and working level-3 implementation (the machine itself), then works backward and upward to understand something like level-1.5 (e.g. interpreted cluster analysis of weights) (Clark 1990). To the extent that this explains a cognitive process, is this level that should be assumed shared, but it sits somewhere between the processing and structure rules so is open to some interpretation.

complex linguistic processing can only be learned on a basis of prior, simpler structures (Elman 1993). Hypothesis B supposes the processing rules about the domain are learned prior to effective communication in the domain.

A fourth case would refute Hypothesis B:

iv) Processing rules are not shared, but effective communication can occur such that individuals still produce the same structure rules.

Morphic languages at first appear to support Hypothesis B. The genotype is described as a generative rule for producing the form, and therefore as a processing rule, and it seems to be shared. All of the first three possibilities above are possible. Hillier and Hanson first refer to the prevailing *classical* (i) theories of artificial intelligence. To explain how rules might be retrievable, however, they appear to endorse a non-classical (*by Clark's definition, above) version of the nativist stance (ii). They quote (Hillier and Hanson 1984 p.46) Von Neuman's (1958) description of an alternative logic and mathematics as the 'primary language truly used by the central nervous system' as distinct from the 'secondary language' we use in outward communication of 'our' logic, and imply that there is a unique, common syntax for this primary language. It is because we each share knowledge of this syntax for a morphic language that descriptions of the genotype inherent in the phenotype are retrievable. Elsewhere, they reject the nativist assumption, in line with Clark and Thornton's (1997; also Elman 1993) position on recodings, and propose (iii) that our most abstract and general concepts (e.g. mathematical) are "learned through our elementary transactions in the world" (Hillier and Hanson 1984 p.47).

The refutation of Hypothesis B will be based on the alternative (iv): that while a grammar posits structure rules for a language, there might be many underlying process rules to produce it. This is still consistent with the notion of a morphic language, in that the inverted genotype cannot be communicated directly but requires retrieval by interpreting from instances. If the processing rules can be interpreted differently while still remaining competent at outputting instances of form that conform to the structure rules, Hypothesis B does not hold.

An important distinction must be made between the classical and broadly connectionist (including statistical, dynamical systems, etc.) approach, as only the second applies to (iv)[*].

[*] There are also purely practical reasons for adopting the sub-symbolic, distributed or connectionist approach in this work. First, as mentioned above, the design problems dealing with statistical approximation of complex systems tend to be numeric and continuous rather than discrete and symbolic. Second, the flexibility and robustness of the connectionist approach are particularly desirable when dealing with real designers (see chapter 6). Finally, communication with these design algorithms needs to be richer and more subtle than the symbolic approach allows (see chapter 10). One of the central

Idealist approaches typical of GOFAI in which meanings exist a priori are not compatible with (iv), therefore the work will take a realist approach, relying on the external world. Coyne (1999) more precisely differentiates *representational realism*, in which “different minds *share* the same reality by having the same representation of it” (p. 82), from *pragmatic realism* (p.87-88), which relies on embodied engagement with the world instead of assuming the objective rules (pp. 87–8; also chapter 5 on phenomenology). Case (iv) clearly allows only the latter.

In practice, a number of different positions exist as to the essential qualities distinguishing this approach from classical and representational alternatives (Haugeland 1985; Brooks 1991; Dreyfus 1992; Cilliers 1998; Wheeler 2005), but these hinge on two major issues:

- Representation: Cilliers (1998), for example, argues that a distributed representation is most important, and essential for dealing with complexity. If a symbolic representation exists, it is not fundamental but only a higher level description of this.
- Cognition as embodied/embedded: Wheeler (2005), for example, states that many distributed representations don’t go far enough as they attempt to create modular, context free representations of the world. For an agent embodied within the world, context is not represented but all perception and action is dependent on context, and is ‘richly temporal’. Cognition is seen as a dynamical system interacting with the dynamical system of the world.

These are discussed in §§3.4.2 and 3.4.3 respectively.

3.4.2 REPRESENTATIONAL LEVELS OF RESOLUTION: ISOMORPHISM AND HOMOMORPHISM

Communication, at least in language, does appear largely symbolic at the level of public transmission. The classical view is symbolic also at the level of processing rules, and holds meaning as formal and established a priori. Operating on ‘meaningless tokens’ from which ‘all meaning had been purged’ (Newell and Simon 1976), the Physical-Symbol System Hypothesis defines a limited form of interpretation and designation only in terms of a system’s ability to act upon and manipulate a symbol. Because meaning remains formal and syntactic, communication is possible just because everyone manipulates symbols in the same way. If this formal meaning must not be pre-given (against Hypothesis B), it raises the question of how communication itself may be established.

proposals of this work is that an algorithm may be used even for problems the designer is not able to fully express explicitly, which renders symbolic communication extremely difficult.

This is a semantic issue rather than one of pure syntax (e.g. Chomsky). Saussure's structuralist model of language focuses on the relation between signifier (the word itself) and signified (the concept it represents). For Saussure, this meaning is not inherent in the signifier/signified relationship or the concept itself, but arises from the way that one sign may differ from other possible signs. The concept of a train, the '8:25 Geneva-to-Paris' for example (Saussure 1974, p.108), is distinguished from other trains in the schedule, and maintains a constant identity in relation to these even though it may be a different physical train each day. Saussure's model of language is aligned with the view of language as a complex system (Section 3.2), in that meaning is given by the structural relations between elements, rather than anything inherent in the elements themselves. Classical approaches to analogy (Gentner 1983; Holyoak and Thagard 1997; Lakoff and Johnson 1999) similarly define analogy and metaphor as sets of attributes that can be mapped with the same structure between a source and target domain.

As in a Physical-Symbol System, communication is possible to the extent that two parties will share the same structure of relationships between tokens[*], but such structures need not be represented symbolically. Classical approaches attempt "to establish an isomorphism—a set of consistent, one-to-one correspondences—between the elements of the source and target" (Holyoak and Thagard 1997, p.36), but connectionist approaches may instead establish a *homomorphism* by allowing the token to map one-to-many to a representation at a much finer resolution. Smolensky (1991) mentions several possible relationships between the 'hard', symbolic and the 'soft' connectionist approaches. These range from total incompatibility and separation of the two systems to a system which is 'soft' at the lower level of processing, but complex enough that "hardness emerges" (Smolensky 1991, p. 284). In this final stance, the symbolic is considered to be emergent at the higher level.

This allows the symbolic structure to be learned by lower level processing rules operating at a much finer resolution. A number of implementations exist, each of which has pre-determined inputs (sometimes corresponding to sense data) at the sub-symbolic level that must be mapped to a higher level symbolic structure via a homomorphism. The processing rules to do so are learned based on the structures found in the data, and so meaning is established by the system itself. Models of semantic space originated as a parallel to the classic semantic net as a means to generate linguistic relationships automatically from natural language texts (Reiger 1983), but have come to be appreciated as a representation in themselves in line with Saussure's model and used with connectionist neural networks (Ping et. al, 1999). The Hyperspace Analogue to

[*] The semantic network of classical, symbolic AI is also clearly structural; its concepts are fixed and still locally represented.

Language (HAL) model (Lund and Burgess 1996, Burgess and Lund 1997) is constructed by the global co-occurrence of words (i.e. within a certain distance from one another) in large natural language texts. The resulting model is a high dimensional hyperspace in which individual words label data points in such a way that words with similar meanings lie near one another. These models are truer to the interaction inherent in a complex system; a local change to a semantic network is purely local, but a change to the input matrix used to place signifiers in semantic space potentially changes place of words across the whole system. Rummelhart and McClelland (1986, Vol 2: 216-271) and Elman (1991) show network learning of the higher level structure of verb tenses in a manner similar to that observed in children. In an approach adopted by Steels (2000) and formalised by DeJong (2000), concepts are based on selective distinctions that are successively refined when the need for disambiguation arises. These are used (Steels 2000) specifically in agents that learn to establish a system of higher level, symbolic communication.

By allowing the system to establish meaning in symbolic communication, a crucial result of the one-to-many homomorphism (as opposed to the one-to-one isomorphism) is the fact that it permits different interpretations of the same symbols. This is acknowledged in post-structuralist accounts. Although the structuralist view permits one linguistic token to correspond to a distributed representation of meaning, this meaning itself remains constant, given stable differences between signs. Post-structuralism destabilises even this. For Derrida, meaning arises in the sum of other signs as they are used within the dynamics of the system, so there is no longer a one-to-one correspondence between a word and its meaning (Cilliers 1998, p. 80). Elman's (1991; 1993; 1995) models illustrate this dynamic, in which simple feedforward multilayer perceptrons with a recurrent hidden layer have been trained to predict words in the context of grammatically correct sentences. The result develops a distributed model like those above, as examination of the hidden layer reveals the development of something like a semantic space, with particular clustering into grammatical types such as nouns and verbs. An analysis of how this state space is navigated while producing utterances over time, however, shows that occurrences of the same word in different grammatical contexts correspond to slightly different points in the space.

Chomsky's (1957) argument for the need for a generative grammar is based on the impossibility of a simple finite state machine to produce the recursive complexity found in natural language. This is demonstrated by a first-order Markov model in which states correspond to words via an isomorphism (Chomsky 1957, pp. 18–21). But this simple mapping is questionable. If instead the words correspond to many possible states via a homomorphism, so that a single word can have many slightly different shades of meaning, the lower level processing can in fact operate as a first-order machine[*]. In Elman (1993), the structure of embedded clauses, seen as recursive and hierarchical in Chomsky (1957) is shown to result from a low level connectionist approach that does exactly this. Each processing state contains a hidden history of previous words, which is lost when mapped to the symbolic level.

The problem of communication has not been solved absolutely. If structure rules and processing rules are related by this sort of homomorphism, and there is no reason to suppose that different users of the same language—formal, natural or morphic—use similar processing rules, it would appear impossible to assure a complete isomorphism between two individuals' structure rules. Learning demonstrably improves this isomorphism, but does not guarantee marginal cases. Clark (1990) notes this is the case in scientific theories, in that Newtonian physics is roughly isomorphic with Quantum physics and general relativity, but this breaks down at the very small and very large scales. The purpose of learning is to improve the isomorphism between communicating individuals. Chapters 9 and 10 will demonstrate the possibility of social convergence of structure rules to establish a common meaning in a morphic language.

An advantage of the homomorphism in marginal cases is that while different individuals may respect the same structure rules in general, their different underlying processing rules operate differently at the sub-symbolic level, and may produce different valid outputs. Dennett (1991) suggests this may be a stable possibility, even for unlimited example sets. Chapter 10 will discuss the effect of this novelty in a social context and argue that it is an important aspect of creativity.

3.4.3 ACTION IN THE WORLD OVER TIME

The problem of communication above was cast largely an issue of the nature of representation, but design is also a matter of *action*, and interaction with a developing design. Our relationship to objects with which we interact in the world has been considered (Heidegger 1926; Dreyfus 1991; Wheeler 2005) not to be one of abstract, non-contingent representation, but highly contextual and time dependent. This enables the criticism (Cilliers 1998; Wheeler 2005) of many implementations of even the flexible, connectionist representations above, on the basis

[*] A higher order Markov model with memory of previous states can capture the recursive structure of language. An *x*-order model with *y* states can be recast as a first order model with max. x^y states by means of a homomorphism by recursively dividing the states of the original model into states representing the prior state. The fact that the number of states increases exponentially explains why we have a practical limit on how deeply we nest phrases in actual speech. Distributed, real-valued representations (Elman 1995) turn this into an issue of resolution, which accounts for human abilities such as understanding right-branching sentences more easily than centre-embedded ones.

that many neural networks operate to compute a simple function, mapping an input set to an output, and are removed from context and temporal action.

The distinction is often made (Ryle 1949; Dreyfus 1992; Devitt 2006) between *declarative* ("knowing-that") and *procedural* knowledge ("knowing-how"). Action involves the latter. The rationalist assumption for intelligent behaviour in a domain, from Descartes to early Wittgenstein, is that one "must abstract a theory of a domain [by] finding invariant features in terms of mapping specific situations to appropriate responses" (Dreyfus 1992, p. xxxiii). These are the features that are mapped directly to symbols in classical AI. After Heidegger, later Wittgenstein and connectionists such as Rosenblatt, however, Dreyfus and Dreyfus (1988, p. 328) suggest such theories of the world may be unnecessary.

The alternative to representing problem spaces is to eliminate representations entirely, and accept the space of the environment as its own representation. The act of being 'in the world' described as *embodiment*[*] (Quick et al. 1999, Dourish 2001, Wheeler 2005) or *structural coupling* (Maturana and Varela, 1987) requires that there is two way perturbation between an individual and the environment. For the individual acting in the world, their experience in time also affects the dynamic state of their mind, such that "each time a new significance is encountered, the whole perceptual world […] changes" (Dreyfus 2007, p.1154). Embodied approaches to cognition do not attempt to accurately model the world as in both classical and connectionist approaches; instead the individual's smooth coping is emphasised (Brooks 1991; Dreyfus 2007).

Affordances (Gibson 1979) are opportunities for action provided by the environment. Perception of an affordance in an object in the world makes that object immediately meaningful, without requiring meaning to be imposed through the structure of a symbolic system. In Gibson's description, affordances are perceived directly, so that a pattern of light falling on the retina 'looks' to an animal, for example, like something to eat. Evidence exists for this as a plausible neurological model. Freeman (1991, in Dreyfus 2007) has documented the phenomenon in responses of neuronal cell assemblies in rabbits to stimuli in the environment—affordances are perceived directly as 'basins of attraction' into which a stimulus falls, and these basins are

[*] There are at least two distinct uses of the term *embodied* in relevant literature, both of which contrast it with mere representation. Devitt (2006, p. 58) uses "embodied" to mean "built in to a system", in the sense that rules may be implicit in the structure of the system itself rather than *represented* explicitly as data. The term as used by the authors referenced here means somewhat more, referring to a system with respect to an external world, i.e. "coupled to an environment". Some authors (e.g. Wheeler 2005) draw a further important distinction between "embodiment", referring to the physical aspects of an individual's being, and "embeddedness", referring to the situational aspects including social and temporal.

dependent on the rabbit's state of hunger, thirst or sexual arousal. Computational approaches include a form of ('deictic') representation, in which only the possibility for action is represented, and this triggers an agent's response directly as the agent interacts with its environment over time (Agre 1997); embodied robotics implementations "use the world as its own model" (Brooks 1991, p.139), instead of a representation; some agent models (Turner 2006) do the same with a virtual environment. Seen as a larger coupled system of organism-environment, these are not "nearly decomposable" in Simon's (1996) sense, as they require high bandwidth between agent and the world (Haugeland 1995 p. 235)

Discussion of affordances in design is typically concerned with a consideration of the affordances a finished design will offer to the user (Norman, 1988), but the concept is equally valid in considering the possible design moves afforded to the designer at a given point in the development of a design. At any stage of the design of a building there are only certain possibilities open to the architect, provided first by the site and brief, later by a structural grid, later by a cladding system, etc. The act of guiding the evolving building in one direction rather than another can be seen in this sense as a selection from the afforded alternatives. The direct perception of affordances in this context is supported by Gestalt theory; Lawson (2006, p. 132–3) suggests designers use schemata (Bartlett 1958) to organise past experiences, by which new stimuli could be recognised as such. Tang and Gero (2001) suggest the act of sketching, with constant drawing and re-evaluation is such a process, choice of rules is the inherent design activity of shape grammars (Stiny 1976, 1980) and other generative processes, and the explicit representation of choices as design spaces has been implemented in CAD environments (Brockman and Director 1991). In these cases, the form of a finished design can be considered to be a choice (as Gombrich 1960) between afforded alternatives.

After recognition, there must be a mechanism in place to act on the affordance. In Freeman's (1991) neurological description this is straightforward, as the basins of attraction into which a stimulus can fall correspond directly to paths of action in the animal's nervous system—paths that lead to eating, drinking or mating. Dreyfus (2007) considers this a resolution of the binding and frame problems, by turning them into a simpler problem of *selection* of a relevant stimulus from the environment*. In the case of design, this selection could be an initial choice between generative rule systems, or more realistically as a repeated choice of design moves within a

* The binding problem concerns the brain's mechanism for selecting and associating the relevant disparate stimuli together to form a single impression of a perceived perception. The frame problem is roughly the same for AI. Both problems exist if it is assumed the data exist context-free: one needs to know what is significant to choose the binding/frame, but one needs to know the binding/frame to know what data is significant.

system—the ongoing selection of affordances as the design evolves over time. While it may not be conscious (Goodman 1975) the act of creation implies this sort of repeated choice of one design decision over another.

Freeman’s research is a purely dynamical systems account of perception and action. The practical implementation of such models falls somewhat short of this, as the best understood techniques for learning declarative knowledge (e.g. backpropagation) do not apply to the more complex, time-based systems that are thought to be required—a criticism of connectionism levelled by both classical and non-classical (Dreyfus 1992; Wheeler 2005) AI. Such models have been implemented in practice (Elman 1995 is a good example), but to date actions involving sequences are simply more efficiently modelled as an algorithmic process, with symbolic rules.

Given the emergent higher level of symbolic representation (§3.4.2), this symbolic approach to action warrants some consideration, so long as it doesn’t affect the communication essential to Hypothesis B. Clark (1998) proposes two ways in which this might be examined. The first (pp. 146–7) proposes a continuum of representation from the most fully embodied ‘adaptive hook-up’, through increasing degrees of inner complexity, to a fully disembodied system operating only on inner codes. The situations deemed appropriate for the latter are termed ‘representation hungry’, and characterised by reasoning about a) “non-existent or abstract states of affairs” or b) “complex and unruly” phenomena (p. 167). Design arguably involves both. The second—a ‘rogue’ connectionist approach (Clark 1990), or ‘external scaffolding’ (Clark 1998)—more specifically addresses embodied interaction. Embodied approaches suggest that a certain amount of intelligence is embedded in tools and cultural practices that are external to the individual (Haugeland 1995 p. 235). The ‘rogue’ approach again supposes sub-symbolic cognition that appears symbolic at a higher level (as §3.4.2), but symbols may be externalised, and are then acted upon as such. Multiple digit arithmetic (Clark 1990) and to some extent natural language (Clark and Thornton 1997) are examples. Seen as affordances, these external, symbolic resources ‘represent’ some of the opportunities for action.

In the ‘rogue’ approach it is the more difficult to implement procedural knowledge that is implemented algorithmically and symbolically. In design, just as in long division, this higher level procedural structure appears important for clarity and communication of processes. It has the properties of a digital*, as opposed to analogue, system: ‘copyability’, the resistance to

* Often literally as well—we count on our fingers.

deterioration when copied repeatedly, the fact that interesting cases have ‘complexity’ of many parts, and ‘medium independence’ in that the equivalent structure can exist in different media (Haugeland 1998, p.76). Dawkins’ account of meme transmission describes the teaching of a carpentry skill as essentially a digital process, just as gene transmission is digital. “Driv[ing] in the nail until the head is flush, […] stitches in knitting, knots in ropes or fishing nets, origami folding patterns, useful tricks in carpentry or pottery: all can be reduced to discrete elements” (Dawkins 2006, p.193) in the process of making that can be repeated much as a digital, symbolic algorithm. It is particularly the copyability of symbolic and rule based approaches that make them good models for the communication of this sort of procedural knowledge.

Such an algorithm doesn’t address the maker’s judgement about how to hold the hammer in driving in the nail, a more flexible process and complex process that should more closely resemble the state paths described by Freeman (1991). In some cases this may not matter—differences in holding a hammer produce slightly different paths that are convergent to the same result. As external procedures, many of these are shared publically, and if the methods for manipulation are well agreed interpretation matters little. In other cases, one needs to make a relevant choice in a complex situation, and a symbolic algorithm ignores the “pre-eminent role of pattern discrimination” in real world thinking (Dreyfus and Dreyfus 1988, p.318). These choices can be considered as the selection of affordances, which, as an instantaneous process, can be readily implemented in a sub-symbolic manner.

This suggests a hybrid approach in which *procedural* knowledge is implemented as a classical algorithm, and affordances are selected from the world by a sub-symbolic process on the basis of *declarative* knowledge. In terms of communication in Hypothesis B, the strategy will be to implement a standard procedure for *how* action occurs, and needs only to communicate *what* sort of choices need to be selected[*]. Depending on how specific the procedure is, this presupposes some degree of domain-related nativism on the implementation, and a reduction of a procedure that in nature might be far more complex, but so long as the full range of action in the domain in question can be expressed via affordances/choices, all communication of the user with the computer can be declarative. The procedure in question must therefore be as generic as possible. Optimisation fits easily into this model. The design itself is external to the agent in the optimisation scenario, rather than represented abstractly as part of a procedure, while design decisions are made not by symbolic rules but by internal connectionist evaluation for the judgement of affordances.

[*] With respect to the way design processes were specified in Chapter 1, an ‘instructive’ process is captured in procedural knowledge, and an ‘elective’ decision can be defined by declarative.

The structural problem (Chapter 4) will produce design responses directly. The space planning problem (Chapters 7–9) will communicate the goals for selection of affordances within several standard procedures: an aggregation process mimicking urban growth (Hillier and Hanson 1984), and a genetic algorithm. These goals take the place of the genotype of a morphic language (Hillier and Hanson 1984), but are declarative rather than procedural and will be described as an archetype (Chapter 8).

3.4.4 ACTION-ORIENTED REPRESENTATIONS AND CONFIRMING COMPETENCE

At the processing level, the means the machine will have for understanding its input of design precedents will not be a representation in the classical or representational realist sense. Summarising its requirements from the two sections above, it will:

- Produce structure that is *homomorphic* to actual states of system it models (§3.4.2), so that it stands as a simulation (Hypothesis A).
- Produce structure that is *isomorphic* to nameable and testable states of system it models (§3.4.2), so that communication is possible (Hypothesis B) and competence can be demonstrated.
- Be declarative (§3.4.3), so that communication is possible (Hypothesis B).
- Be a means for selecting between affordances (§3.4.3), thereby allowing embodied action.

The requirements are related in that testing of competence can only be via its embodied action, as is the case for any non-nativist account. Wittgenstein's (1958) 'language games' or their computational implementations (e.g. Steels 2000) likewise confirm that communication has occurred based on whether the listener acts in a way consistent with the expectations of the speaker, even when the internal states of the mind are different. Arguing against Frege's notion that a concept must be clearly defined, Wittgenstein stresses this action: "One gives examples and intends them to be taken in a particular way.—I do not, however, mean by this that he is supposed to see in those examples that common thing which I—for some reason—was unable to express; but that he is now to *employ* those examples in a particular way. Here giving examples is not an *indirect* means of explaining— […] *this* is how we play the game." (Wittgenstein 1958 §71). Hofstadter (2001, p. 527), likewise describes minds as "far more unlike than are the sports of soccer and basketball", yet they are generally able to visualise an equivalent set of plays ("the same thought") in either of the two games.

Although what the machine learns is not a classical representation, it will be difficult to avoid the word. What will be meant by representation in this sense is what Clark (1998, p. 149) refers to as an ‘action oriented representation’. He suggests two axes in which to place any representation, the first ranging from symbolic to distributed (the classical versus connectionist question, §3.4.2) and the second indicating the degree to which inner states stand alone or are coupled to the environment (embodied action, §3.4.3), and holds that emergent properties of a system that coincide with states of the world are not proper ‘internal representations’ (p. 168). Whereas representations in a symbolic system are independent of their medium (Haugeland 1998, p.76), the declarative representation relying on affordances is strongly coupled to the environment and ‘action oriented’ in that it has meaning only in the context of that act of selection. In terms of testing competence, Clark (p. 150) also emphasises the fact that the action oriented representation cannot be interpersonal, in that it can not be shared, but is interpersonally valid, in that the actions it produces will be.

This testing of multiple interpretations is the basis of objective knowledge, and consistent with Popper’s (1959) requirements of the scientific method. *Intersubjective testability* far outweighs in importance the need for a *prior hypothesis*. It is the method by which falsifiability can be established and therefore the basis of his original principle of demarcation, which determines the domain of possible objective or scientific knowledge. Refuting hypothesis B will therefore involve the demonstration that very different algorithms—with different internal structures—can arrive at the same result or output given the same data.

Chapter 4: Structural optimisation for stiffness*

Summary: Optimisation is performed on a complex structural system with many members, by approximating it as a number of interchangeable low-level modular elements. This prepares for the first evidence against Hypothesis A, in that this cellular arrangement provides a clear definition of sub-system and interface, which will allow for the behaviour of the cell to be abstracted from its structure in chapter 5.

Hypothesis A refers to a complex system. Before addressing the spatial layout problem in chapters 6-9, optimisation and machine learning methods are applied to the more readily quantified structural problem in this chapter and the next. This is a visible example of a complex structural system in which the interacting elements are interconnected physical members. It may be viewed as an analogy with the living self-organising systems of natural wood or bone. Thompson (1917) describes the form of these as a 'diagram of forces' due to the clearly visible effect of applied loads in the external environment on cell growth patterns (figure 4.1); the individual cells adapt to these loads and perform a natural optimisation.

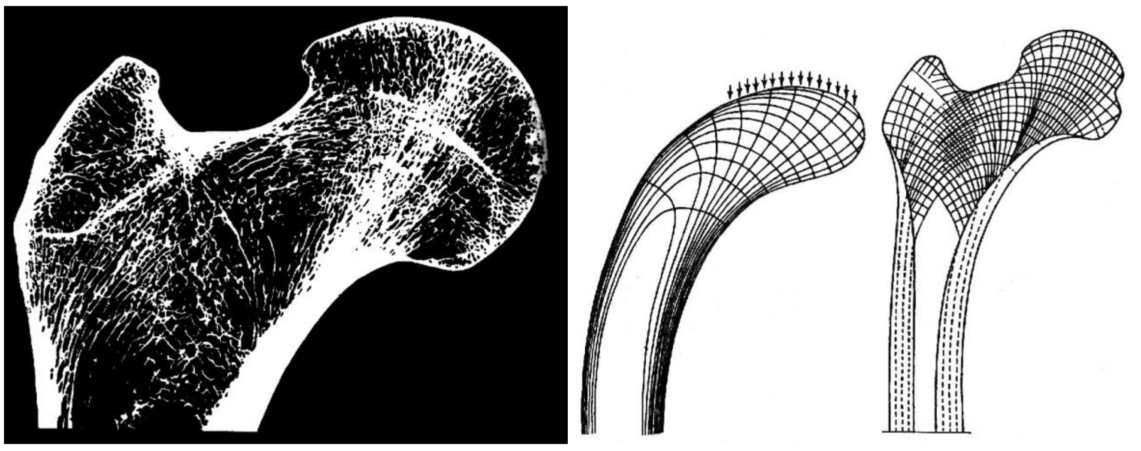


Figure 4.1. The cells of a bone (left) clearly show the 'diagram of forces' under which they grow. Both images Thompson (1917).

The main aim of this chapter is to show that an *isomorphic* simple structure can be used to replace a complex one without detrimental effect. This reflects a homomorphism on which finite element analysis (FEA) itself is based. Although analysis is based on discrete elements, these are theoretically standing in for regions of a continuous solid, with which they are *isomorphic*.

* Much of this chapter has been published (in a different context) in Hanna and Haroun Mahdavi (2004).

At a much finer scale—cellular or molecular—any continuous solid is in reality a complex material structure that is only approximated by measurements of elastic modulus, density, etc. There are a potentially infinite number of possible internal states of this internal structure which are *homomorphic* with a single state of the continuous model. A single finite element, then, is a simple system designed to simulate the same structural behaviour of a highly complex structure under the same loads and boundary conditions at its interface—a homomorphism for a region of the material. The further isomorphism used here, will be a new structural unit optimised to replace the simple finite element.

The interface of the structural unit is therefore a crucial issue. In the case of a piece of wood, for example, assume a given volume of cells must to be replaced with a new structure such that the overall structural performance is the same. It is difficult to determine where to draw this boundary at the large scale as the wood is roughly homogeneous. At the cellular scale the cell boundaries themselves would appear the obvious choice, but connections between cells are structurally just as important as the cells themselves. Any cut is roughly as complex as any another, so the system is not nearly decomposable (Simon 1969). A modular approach will be taken here, partly in anticipation of the learning applications to follow (Chapter 5), in which the complexity of the boundary interface is acknowledged. What makes it modular is that the interface remains constant because each unit is similar to the next.

The individual modules will first be optimised in isolation (section 4.1–4.4), then a more complex object of many units will be designed (section 4.5). This will use an approximation of the whole under the assumption that it is continuous, but show that all elements of the volume can be replaced by the structural modules. Because of the standardisation of the interface, these can still be optimised in isolation.

By way of the living cell analogy one can discern two kinds of optimisation: a genetic optimisation provided by evolution, identical to each cell, and an adaptability that allows each cell to change its shape in response to specific forces. This chapter applies standard optimisation methods to these two problems, before learning is introduced in chapter 5 to improve unit adaptation.

4.1 Modular space frames: a sample problem

As in nature, the principle of material distribution methods for optimisation (Bendsøe and Kikuchi 1988) is the placement of material where it is most needed and removal where it is not.

Approaches to topology optimisation employing a raster grid of finite elements, such as the Solid Isotropic Material with Penalisation (SIMP) model (Bendsøe and Sigmund, 2003), begin with a uniform medium density, and then iteratively steer each element to a solution with discrete densities of zero or one to represent solid or void. Similar methods have been employed with a free parameterisation of materials (Ringertz 1993; Bendsøe and Sigmund, 2003), which result in continuous variation of intermediate values for density and local anisotropy. The result is a vector field of principle strains that varies continuously at each point in the volume (Figure 4.2). These are intended primarily as an exercise for attaining upper bounds on performance with real materials or as a framework for composite design, where the continuous field may be approximated with assemblies or laminates of existing materials. Because these do not exactly satisfy the continuous solution and because boundaries between joined materials introduce abrupt changes in properties, the solutions must still be over-engineered.

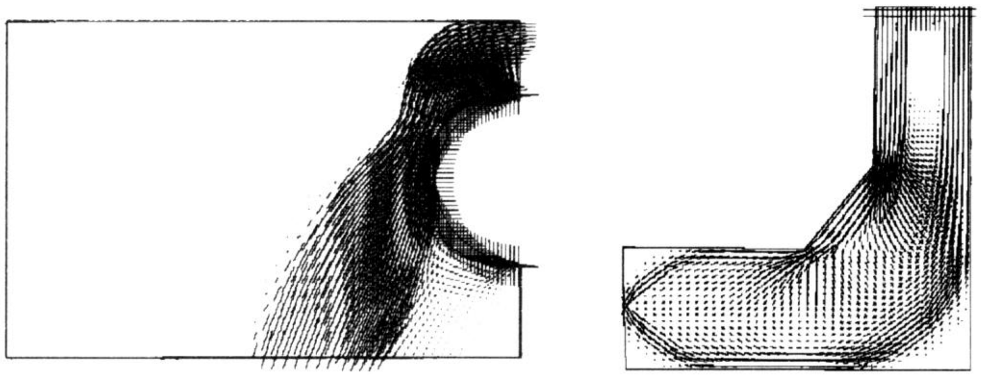


Figure 4.2. Fields of principal strains for two optimisations using theoretically optimal materials. Images from Bendsøe and Guedes (1994).

The design problem explored here is that of a structure that produces the continuous optimal material of varying density and anisotropy for any complex load condition to an arbitrary degree of accuracy by means of a space frame structure with varying local strut geometry. Space frame structures consist of a set of linear members oriented in any direction in 3-dimensional space, and connected at node points either by rigid or flexible connections. The smaller these struts can be made, the more they approach the scale of the microstructure of the natural materials above and the closer they may approximate the continuous field of optimal material properties. The specific problem addressed is that of small scale microstructures such as that shown in Figure 4.3. The overall dimensions of this object as fabricated are 1cm × 1cm × 2cm, and the individual struts within it are less than 1mm in length. These are intended to be manufactured at such scale by stereolithography (a rapid prototyping process), and so material and fabrication properties are necessarily incorporated into the simulation and design algorithm.

The structures consist of a modular unit that can adapt to various imposed load conditions as do living cells in the biological analogy. The problem of design involves determining both the strut connections between node points in space (the topology) and their locations. Bendsøe and Sigmund (2003) split the optimisation problem into the two coupled sub-problems of local anisotropy, and material distribution, which can be addressed by altering the direction and thickness of the struts respectively.

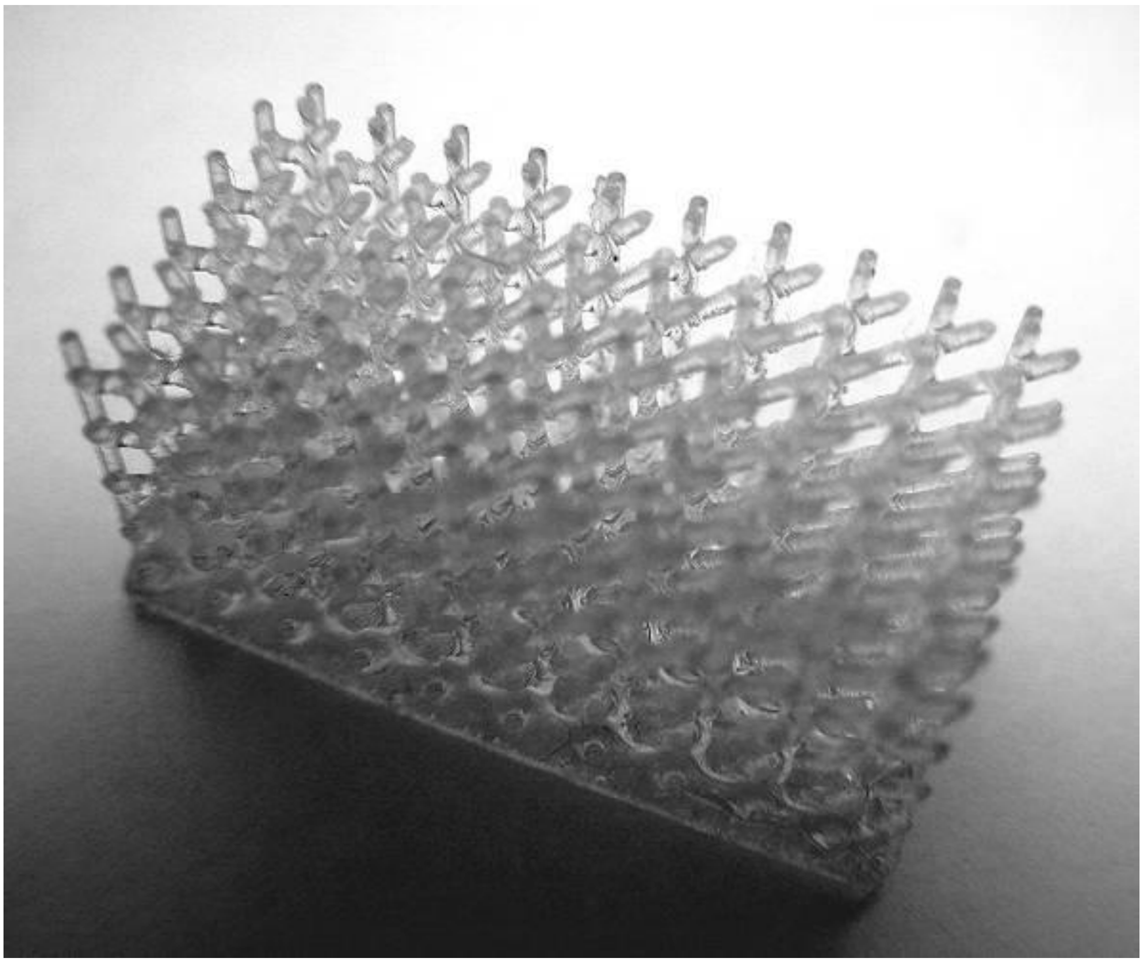


Figure 4.3. A modular structure fabricated by stereolithography.

4.1.1 Problems of scale in optimisation complexity

The major problem associated with increased complexity is that of the high dimensionality (and thus high computation time) posed by structures of many members. Iterative optimisation techniques based on FEA are excellent for dealing with several hundred or even several thousand members (Rudnyi et al. 2005) but for structures of the many thousands (or millions) of members this small scale allows, standard methods that consider each member individually become computationally intractable. Methods have developed to improve computation time, often by sequential mesh refinement (Bey 1995), but also by projecting the model onto a lower dimensional subspace that approximates the original system (Del Tin et al. 2006; Rudnyi et al. 2005). It is typical for FEA to employ banded matrices and therefore computation time scales approximately linearly with the problem size (Badea 2004), but each step of the node optimisation problem alone must search a $3n$-dimensional space, and so the computational complexity is at least $O(c^n)$. Computation time increases exponentially and limits the size of problem.

The strategy adopted in this chapter to handle this problem is the simplification of the low level parts of the model. As mentioned in section 3.4, the internal behaviour of the parts of a complex system may often be approximated, and the resolution of these parts may be reduced, with little discernable effect on the overall system, so long as the interaction structure is preserved. This is the property that allows a material with a complex microstructure to be approximated by a continuous solid of given material properties, and this to be modelled by a finite element mesh of far lower density. The method proposed here takes this in reverse, beginning with a global approximation of stresses given by a finite element approximation, then designing the complex structural details to fit this at a finer level of detail.

Rather than performing this optimisation on a complex global model however, only the internal behaviour of a single part is considered. The large volume of a complete structure can be subdivided into a grid of cubes, which will be referred to as ‘unit cubes’, each containing a small portion of structure such that each is connected to its neighbours to form a continuous structure. The overall vector field of principle stresses (figure 4.2) for the whole object under a complex loading condition changes from point to point, and if these stresses are sampled at the location of one of the unit cubes, they can be used to optimize the module of structure within that cube. The local vector of stresses in the three (x, y and z) axes represents a loading condition for the structure in that cube, and for each stress vector there is an optimal set of node point positions and strut thicknesses to best resist that load. This simplified local optimisation problem consists of only a few struts, so is easily optimised.

As a complex system, it is possible to isolate individual parts of the model and make this simplified approximation of the continuous field within each part because the interaction structure between units is maintained (section 3.4). An identical topology is used in all units, ensuring that each part connects with its locally adjacent neighbours in the same way.

A second, related problem associated with increased complexity is that physical structures manufactured from real materials may contain geometrical or material properties that differ from the idealised computer model. A finite element model of a space frame will typically employ a single member for each strut, but these are far from uniform in practice, and irregularities become proportionally greater as member size decreases. Many more elements could be added to the FEA model to increase its accuracy, but again this is an opportunity for simplification within the unit. The application of learning in the next chapter is particularly relevant to this.

4.1.2 Manufacturing

The structures considered are designed to be fabricated by a digitally controlled process, the main advantage of which is the low cost of complexity. Such techniques are increasingly used in such large scale manufacturing as automobiles and architecture (Sischka et al. 2004), but the development of smaller scale rapid prototyping technology allows manufacture at scales less than a millimetre.

Rapid prototyping techniques are now beginning to be investigated as an alternative method of construction for objects of high complexity, particularly with intricate internal structures. This has not yet become commercially viable for mass production, but several researchers are preparing for the increasing accuracy and decreasing cost of the technology in the future. Molecular Geodesisics, Inc. (1999), for example, is investigating structures based on a regular tensegrity space frame which would, at a microscopic size, be useful as biological or industrial filters.

Stereolithography, specifically, is the method considered here. This begins with a tank of liquid photopolymer which is sensitive to ultraviolet light. An ultraviolet laser ‘paints’ the object as a series of horizontal layers, exposing the liquid in the tank and hardening it. Once completed, the object is rinsed with a solvent and then baked in an ultraviolet oven that thoroughly cures the result. The machines used to fabricate the example in Figure 4.3 are capable of creating very fine structures, and build to a resolution of 0.05mm.

The horizontal stratification inherent in the process adds a degree of complexity to the problem of optimization, as members built at different angles to this horizontal plane have varying strengths. These were measured and factored in to the model used for optimisation.

4.2 Method[*]

The volume of the overall object to be designed is divided into a three dimensional grid of cubes, each forming a modular unit, or ‘unit cube’. The objective of optimisation is maximum stiffness for a set quantity of material, and real-world units of force and size are not crucial. When actually manufactured the structures have a real scale of one to several millimetres for each unit cube, but this module is used as the unit of volume throughout. Each unit cube is immediately adjacent to 26 neighbours in three-dimensional space, including edge and corner adjacencies, and the structural module in each cube may be connected to any or all of these adjacent modules.

This unit is a uniform cubic volume in 3-d space, but at no point are the structural elements fixed within it. Struts are not constrained to the edges, corners or any place within the unit cube. The interface between any two modules, as mentioned above, is no less complex than anywhere else, and the cube boundary could be translated to any other point in space without effect. Modularity simply assures the boundary is similar between adjacent cubes.

The structural module itself is defined as a series of linear struts that connect at node points in three-dimensional space. Two distinct features determine the form of the structure: its topology, indicating which nodes are connected by structural members, and its geometry, or the positions of the node points in space. In terms of the analogy of the bone cell, every cell consists of a set of parts identical to every other—a nucleus, stiff cell walls, etc.—determined genetically by evolution. In the same way, each structural module consists of an identical strut topology, which is found by the computationally analogous process of a genetic algorithm. Each bone cell has a unique shape and density however, determined in large part by individual cells adapting locally to the stresses imposed by their environment (figure 4.2). This corresponds to the geometry of each structural module, which is optimised locally to its unique load.

The geometry of the node points in 3d space varies from unit to unit, but their connections to one another do not. This provides two benefits in terms of scalability. First, breaks in the continuity of the structure are minimised by allowing a difference between one unit cube of structure and the next which can be scaled down to arbitrary units of precision. These discontinuities would be expected to result in corresponding weak points if the topology were to change, as adjacent units of incompatible structures would be zones of weakness in the overall object. Second, it allows one single graph of connections to be evolved which can be applicable

[*] The outline of this method has been presented in Haroun Mahdavi and Hanna (2003).

to all points. This allows an object of any size and any number of units to be evolved once, rather than running the GA many times to generate structures for large objects or complex load conditions.

4.2.1 Representation of solutions: topology and geometry

Topology refers to the structural connections between the node points. A change in the topology of a structure is a change in the number, or way in which the members are connected (Figure 4.4). The two structures shown can not be made equivalent simply by moving the positions of the nodes. Given *n* points in space, this is a discrete property that is represented by a graph of connections (the ‘connection graph’) that is independent of the locations of any of the nodes. It is represented by a series of binary upper triangular matrices, valued 1 for a connection and 0 for none, that describe which nodes are connected by structural members. The first matrix describes the internal connections of the *n* nodes within the cube. The other 26 matrices describe connections between the nodes within the original centre cube and the corresponding nodes in the surrounding cubes on each adjacent face, edge or corner. For *n* node points therefore, the size of the combined matrix is *n* by 27*n* (Figure 4.5).

A genetic algorithm is used to optimise the topology of structures, in which the genome is the above representation. The number of connections (1s) in the matrix is initially set low, and tends to decrease further as more efficient structures are evolved.

Geometry refers to the positions in 3-d space of the node points joining the structural members, as shown in the above diagrams of two structures with the same topology but different geometries. The connections and number of members are the same, but the coordinates and orientations of these members differ. The geometry is represented simply as the Cartesian coordinates of each of the node points in the structure, so *n* node points are defined by 3*n* real numbers. The centre of the unit cube is taken as the origin and so the range of the module is from (-0.5, -0.5, -0.5) to (0.5, 0.5, 0.5), but these limits are not imposed on the nodes themselves, which are free to be placed anywhere in space, even outside this boundary.

As determining these coordinates is a continuous, real-valued problem, it is amenable to a large number of search techniques including genetic algorithms and deterministic, gradient-based methods (section 3.1).

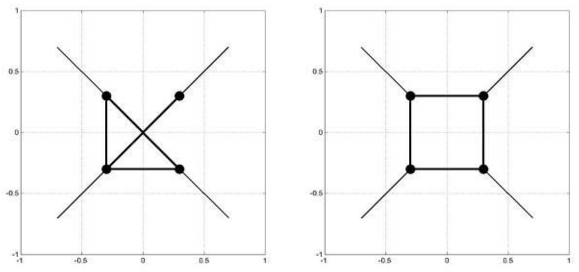


Figure 4.4. Changes in topology

	1	2	3	4	1a	2a	3a	4a	1b	2b	3b	4b		1z	2z	3z	4z
1	0	1	1	0	0	0	1	1	0	0	0	0	...	0	0	0	0
2		0	0	1		0	0	0		0	1	0	...		0	0	1
3			0	1			0	0			0	0	...			0	0
4				0				0				0	...				0

Figure 4.5. The genotype

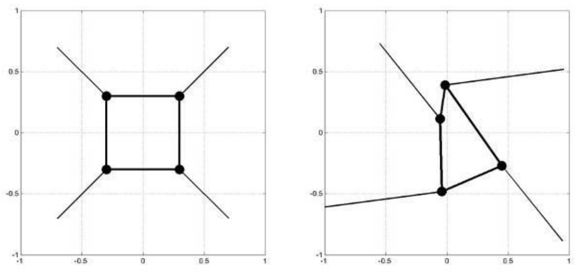


Figure 4.6. Changes in geometry

4.2.2 Evaluation of solutions

The goal of optimisation is to maximise stiffness for a given mass of material, under a given load. In all cases, overall mass is kept constant throughout optimisation, and stiffness is measured by the maximum deflection of any node in the structure when load is applied. The model is evaluated by finite element analysis (FEA). Given a structure defined as a series of linear members within a unit cube and a load applied to the members which connect to the adjacent unit cubes, the FEA yields the stresses in each member and the deflections of each of the node points in the structure. The total of these deflections gives an indication of the overall performance of the structure under that load condition, and the individual stresses indicate the relative performance of each member. These are then used by the algorithm to optimise the geometry and member thicknesses for that specific load. Each strut is modelled by a 12 degree of freedom beam element with one node at each end. All joints between struts are fixed in both bending and rotation.

4.2.3 Material considerations

In construction by stereolithography, the horizontal stratification in every part of the model results in an inability to construct the underside of any portion at an angle of less than 30º from horizontal. Also, members constructed at differing angles to this ‘grain’ therefore have differing strengths and their calculated deflections were modified accordingly in assessing the fitness of the solution. This was approximated in the model by a function determining strength from angle, estimated based on physical tests (figure 4.7). All angles below 30º were set at a stiffness of zero. To determine deflection in the evaluation of solutions, the calculated deflection of all individual members was divided by the value of this function to derive a more accurate model of the actual behaviour of the system.

Physical tests were also used to determine the minimum size of structure. The stereolithography machine used manufactures geometry to a resolution of 0.05mm; however, the minimum size hole through resin was observed to drain was 0.6mm, so the eventual fabrication scale of the unit cubes was kept to between 1 and 2 mm.

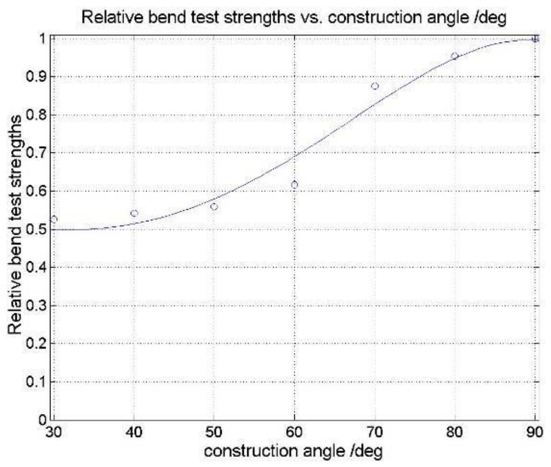


Figure 4.7. Plot of the angle vs. relative strength.

4.3 Optimisation of unit topology by genetic algorithm

The optimisation of a shared topology for all structural modules, just as the 'genetic intelligence' of the bone cells, is found by evolution. The use of a genetic algorithm (GA) to do so is described in this section*.

4.3.1 Determining the geometry for each topology

Topology alone is the design task of the GA, but geometry is also needed for the FEA to calculate the performance of a given structure. In sections 4.4 and 4.5, this is found by a separate optimisation loop, but as the evaluation of phenotypes is typically the most costly step in a GA, several rough approximations were used to estimate node positions while evolving the topology alone.

Node positions were found by a deterministic process based on each applied load—an iterative relaxation of points to achieve structural equilibrium, i.e.:

$$\sum F_x = 0 \qquad \sum F_y = 0 \qquad \sum F_z = 0 \tag{4.1}$$

This standard analysis of structural equations made some assumptions on the structure which would not be entirely accurate in context of an assembled structure, namely that the entire assembly would be in static equilibrium, but it was found that the solutions provided a sufficient approximation to derive a set of geometries to be evaluated. Elastic moduli and cross sectional areas are considered equal in all members so are not taken into consideration. More sophisticated methods were used in the next stage of research (chapter 5).

Estimation of a range of external conditions was also required. The optimisation goal was for a general solution that would be viable (with some translation of node positions) under a range of tensile or compressive forces in any direction. Weighting the equilibrium calculation [4.1] in the direction of the forces to be applied was used as an approximation of different loading conditions for each unit cube. To do this, applied forces were broken into their respective (x,y,z) component vectors as are each of the members in the calculation, and the node point with its connecting members treated as an isolated structural unit under those forces. The different

* More detailed explanation of GAs in general can be found in Mitchell (1996), and of this implementation in particular in Mahdavi and Hanna (2003).

component vectors of the applied tensions were divided between the unit vectors of the members, and the resulting tension solved by the element analysis equations used to weight the averaging calculation. Thus under different loading conditions the shape of the structure is seen to shift to accommodate the change in forces (figure 4.8).

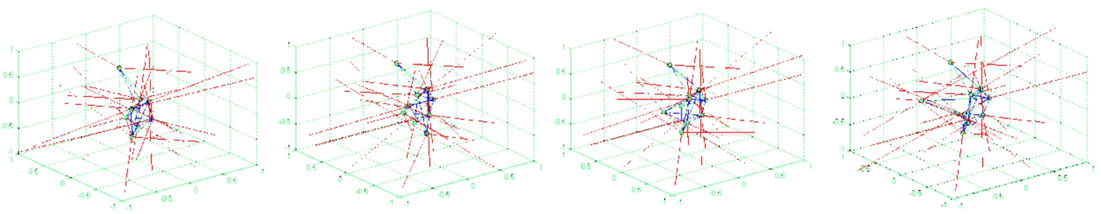


Figure 4.8. A unit cube that has been orientated to remain in equilibrium when exposed to forces coming from different directions. (Equal tension in all axes, then 5:1 in the x, y and z axes respectively.)

Fitness was calculated by summing the maximum node deflection under four different loading conditions: first a load of equal tensile forces of one unit in each of the three (x,y,z) axes, then a 5:1 tensile force oriented in each of the axes respectively. The efficiency of the calculation to determine the node geometry was improved by taking advantage of the fact that there is a greater difference in the position of the points found under each load condition and their starting point at the origin than between either of the different solutions found. The point locations were first found for the equally tensioned condition, and this result is then used as the starting position for each of the other conditions.

The resulting solution was then evaluated for deflection using the direct stiffness method (McGuire 2000). The set of all nodes in the unit, with their internal connections and those to adjacent unit cubes, are considered as an isolated structure placed under the applied external forces. The deflection of each member is calculated, and these used in the fitness function to assess the solution.

4.3.2 The Genetic Algorithm

A genetic algorithm was used to evolve the connection graph that determines topology (section 4.2.1). A total of 12 nodes were used, requiring a 12×324 genome of upper triangular matrices. The total number of independent alleles was $12 \times 11 \times 27$, or 3564, for a search space of 2^{3564} or 7.43×10^{1072} possible graphs. Although the genome size remained constant, the 12 possible nodes were in practice only a maximum bound, as any point with a lack of connections is effectively and implicitly eliminated from the structure. The genotype allowed this to occur as an automatic method of simplifying the overall structure, which, if found to increase the fitness by reduce weight while still maintaining a stable structure, would be likely reduce the effective number of points over successive generations.

To initialise the population, each bit was created randomly with a higher probability of connections to nodes within the unit cube than to adjacent units due to the greater number of possible connections (26:1) to surrounding cubes in the chromosome. The probability of a connection between two internal nodes was 0.5, whereas to other cubes it was set to around 0.03.
The nature of the genetic algorithm would reduce the overall number of connections to achieve a saving of redundant mass, but the setting this latter probability low in the initial population was done to give this process a head start*.

The unusual specification of the genotype required the following considerations on the crossover and mutation operations:

- *Crossover:* Ordering of specific points in the graph is arbitrary, so there is no natural conversion to a standard 1-dimensional gene string. The crossover function simply swaps the values of x randomly chosen points from all the overall matrix of each parent. A crossover rate of 30% was used.
- *Mutation:* The difference in density between connections within the cube and to adjacent cubes was taken into account for the mutation operator, with a goal of maintaining the mean density of that portion of the chromosome. For the central cube, mutation points are chosen depending on the mutation rate, and these bits are simply flipped (0 replaced with 1 or vice versa) in the usual manner of binary mutation. The sparseness of the connection graph for the surrounding cubes requires a slightly different mechanism to avoid mutation biasing the adding of connections. The algorithm first selects whether to flip a 0 or 1 with equal probability, and then randomly chooses a bit of that value to mutate. Over time this would vary the proportion of 0s and 1s based on the fitness rather than a pre-existing bias.

The overall fitness of a member of the population was based on the following four factors:

- The number of angles below 30^o
- The overall mass of the individual, as calculated by total member length
- The maximum deflection within the system

* This was shown to be justified by the results: the number of connections to other cubes further decreased from 0.03 to 0.0016 during evolution in the longest run of the algorithm.

- Whether the iterative node placement had found a solution that settled down. (Occasionally the method used oscillated around a solution for too long and was halted after a set number of iterations. In this case the penalty was introduced to save computation time.)

As minimisation of each of these was sought in optimisation, their values were incorporated into a weighted sum in the objective function:

$$f = \frac{1}{1 + \mathrm{A} \times \mathrm{lowAngles} + \mathrm{W} \times \mathrm{weight} + \mathrm{D} \times \mathrm{maxDeflection} + \mathrm{S} \times \mathrm{penalty}} \tag{4.2}$$

The algorithm employed fitness proportional selection, in which members of the population are selected for gene crossover with a probability in proportion to their fitness. The fittest two members of the population were retained in each generation.

4.3.3 Results

Figure 4.9 displays the maximum and mean fitness over time for the longest of several runs of the combined algorithm. A population size 50 was used and the program was run for 10,000 generations, searching a maximum of only $1 / 1.49 \times 10^{1067}$ of the total search space. Over the course of this run the maximum fitness rises from 0.0575 to 0.3766 and the mean increases from 0.0322 to 0.1580.

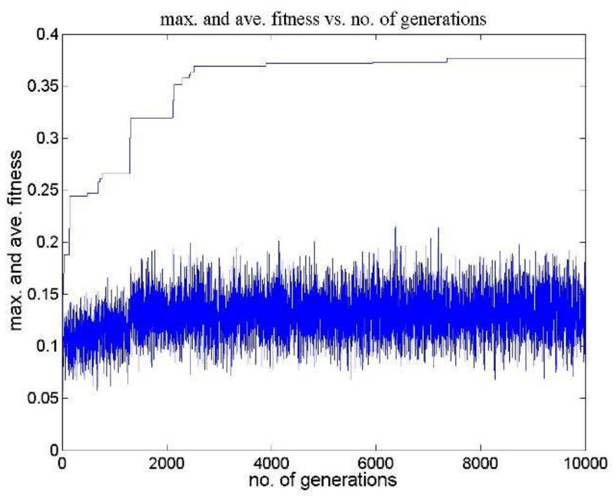


Figure 4.9. A plot of the maximum and mean fitness at each generation.

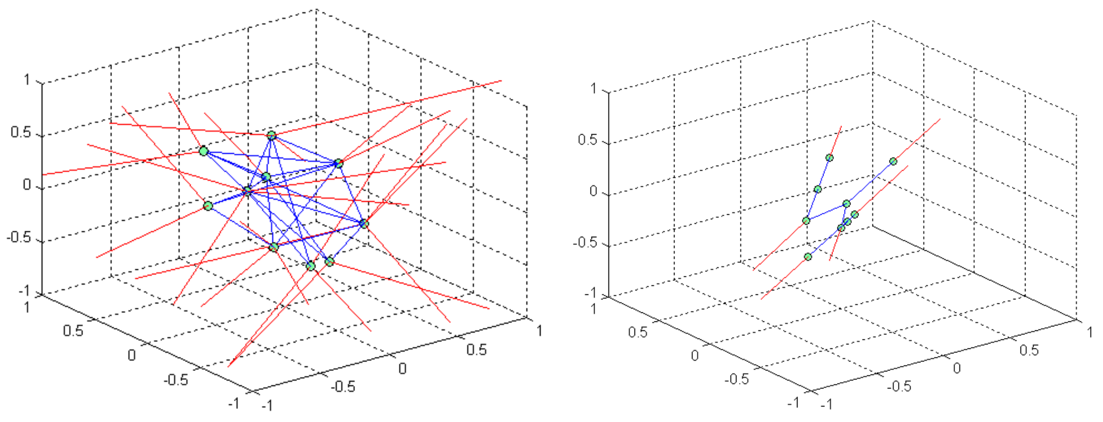


Figure 4.10. A unit cube of a typical initial topology (left) and the best individual evolved (right).

Figure 4.10 shows a typical topology in the initial population compared to the best individual in the final generation. They show several trends that were observed in each of the test runs:

- The number of connections to adjacent cubes always decreased.
- The number of internal connections always decreased.
- Node reduction over time was observed.
- The angles of the members always increased beyond 30° with a preference for steeper angles.

Both removal of connections (trends 1&2) and of nodes (trend 3) resulted in a reduction of the relative mass of a solution, and by removing redundancies, the algorithm increased the unit's stiffness to mass ratio. It was intended that simplification of structure by node elimination would be possible over time, and this was confirmed. Of the twelve initial points in the connection graph, three had been entirely eliminated from the genome by removal all connections to other nodes. A further five were effectively removed from the solution by having a connection to only one or two other nodes, and thereby aligning multiple members into a common linear member.

The overall solution, when arrayed as an interconnected set of unit cubes, reveals characteristics based on the material considerations built into the algorithm. Viewed from the vertical axis the structural members are aligned to the edges of a tiled pattern of almost regular hexagons with approximately equal angles. When viewed from the side however, near vertical elements predominate due to the fact that the hardened photopolymer has a greater structural strength at steeper angles (trend 4). The structure appears elongated in this direction[*]. Also, no members at an angle of less than 30^o are found.

The module can be repeated as a unit cube to manufacture the final object of any size (figure 4.11). The resulting structure is self-supporting and optimized for the material properties of the liquid photopolymer, the stereolithography process and an identical stress condition throughout.

While topology optimisation of structures typically seeks a single solution for a particular load condition, the above process was intended to apply to a range of solutions presented as a set of differing force vectors (section 4.3.2). Although the initial evolutionary search requires an initial investment of computation time, the node positions of a fit solution can be recalculated quickly to changes in load. A single topology generated for the particular properties of the material and a given range of forces can therefore easily modified to vary its geometry to fit changes in stresses over the object. The following section investigates this.

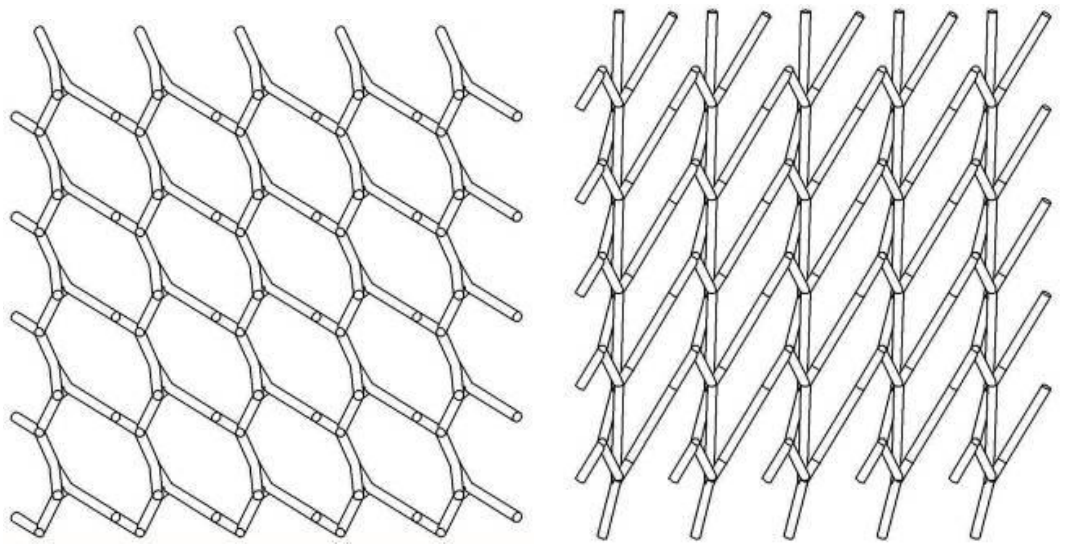


Figure 4.11. The top and side views of the final structure evolved.

[*] The vertical elongation is also accompanied by an obvious diagonal directionality. This specific angle is not the direct result of selection pressure, as optimisation used a symmetrical estimation of the load. As horizontal members are ruled out by the construction process, this is simply the result of a minimal structure breaking the symmetry in one direction or the other. There was also no penalty for asymmetry.

4.4 Optimisation of geometry by gradient-based methods

The second stage of optimisation is analogous to the adaptation of the individual bone cell in response to changes in local stress. In this case an algorithm is required for determining the optimal geometry for a given topology.

4.4.1 Algorithm options

Both stochastic methods such as GAs, and gradient-based deterministic methods were considered. The major difference between these is in the way they search the space of possible solutions in relation to their fitness, sometimes referred to as the fitness landscape. Gradient descent begins at an arbitrary starting solution, then makes changes with respect to local measurements of fitness to trace a single path through the solution space. As such, it is most suited to problems with a smoothly and consistently varying fitness, but if this is not the case and many local optima exist, gradient descent may become trapped rather than achieving a global optimum. GAs and other stochastic methods solve this problem by simultaneously testing a range of divergent solutions and making large leaps across the solution space, but at increased computational cost.

The solution space to be searched consists of the node positions in 3-dimensional space for n nodes, or a 3n-dimensional space. In samples formed by 6 node points we are therefore searching in 18 dimensions. Figure 4.12 roughly illustrates the fitness landscape with 2-dimensional slice through this 3n-dimensional space determined by two random orthogonal 3n-dimensional vectors and centred on an optimal solution found by gradient descent. In the plot on the left, all members are considered to be of equal stiffness regardless of their angle, and at right, irregularities are caused by the material considerations relating to the manufacturing process (section 4.2.3). The changes in stiffness of differing angles of strut against the horizontal stratification of the resin contributes to a more complex fitness landscape than would otherwise be the case. The characteristic valleys correspond to solutions with members at shallower angles and therefore greater deflection. Rather than a single, clear optima, several local optima occur in each of the 3n dimensions. These local optima in the more complex fitness landscape are potential problems for the local gradient-based methods.

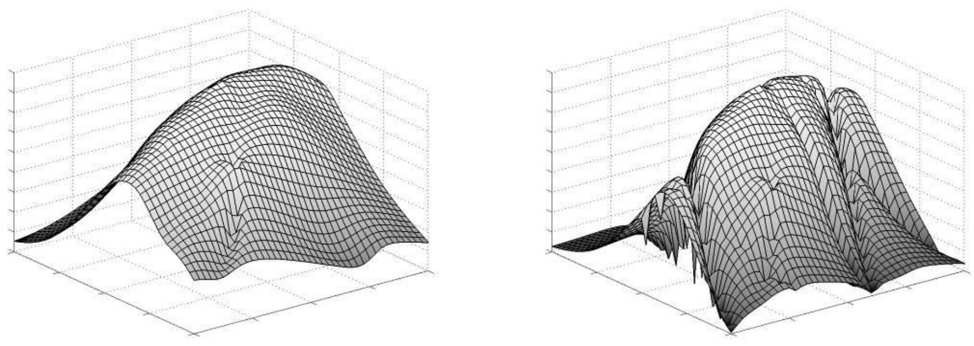


Figure 4.12. Two-dimensional slices through 3n-dimensional solution space.

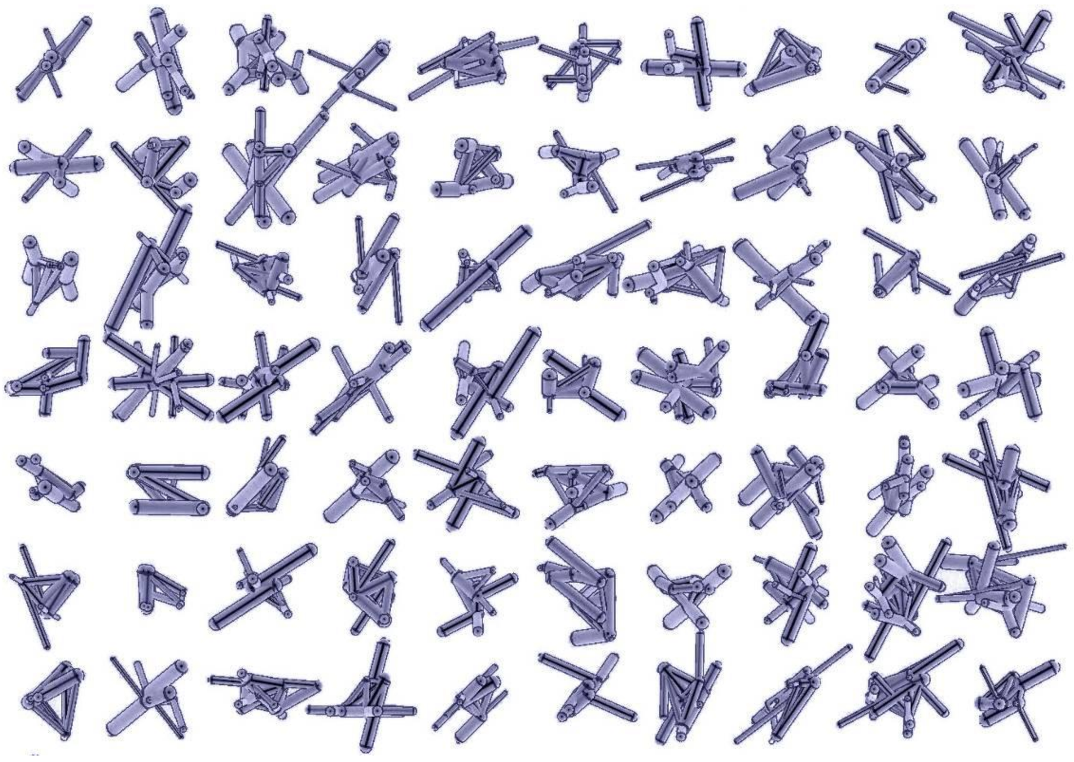


Figure 4.13. 70 sample topologies used to test optimisation methods.

4.4.2 Comparison of GA and gradient descent

An experiment was conducted to compare both a GA and gradient descent as optimisation algorithms on the above spaces. This tested a) the fitness of solutions, b) the speed of optimisation, c) the confidence in finding an optimal solution, and d) the degree to which topology affects this confidence. A set of 70 sample topologies were generated randomly (figure 4.13), and both optimisation algorithms were employed from random initial start solutions.

a. *Fitness of solutions:* Fitnesses of solutions found were approximately equal for the two methods, both when member angles are ignored and when they are factored in to yield the more complex fitness landscape (figure 4.14).

b. *Speed of optimisation*: The gradient descent algorithm was found to be faster than the GA, which required an average of 4.2 times the number of fitness calculations compared to gradient descent for the examples tested. It was found that gradient descent reached an optimal solution (i.e. showed no further improvement) after an average of 718 fitness calculations (39.9×18 dimensions) when strength due to member angles is not considered, and 703 fitness calculations (39.1×18 dimensions) when it is. The GA found similarly fit solutions after approximately 3000 fitness calculations (300×10 members in the population). The plots in figure 4.15 show examples taken from the same randomly generated topology over five runs of the algorithm.

c. *Confidence in finding an optimal solution:* The variance in fitness over several runs of the algorithm is taken as a measure of confidence in the accuracy of the result (i.e. a small variance in the final fitnesses would indicate a high level of confidence that these approximate the global optimum). The decrease in variance also shown in the graphs (decreasing from 0.0351 to 0.0020 for the GA, and from 0.0288 to 0.0015 for gradient descent) would indicate that both the GA and the gradient decent algorithm are not sensitive to initial starting conditions, but find similarly fit solutions each time.

   Although the solution fitness was seen to be approximately equal for both the GA and gradient descent, it was found that the two methods behave differently in searching the space. The images below show the mean and variance of node point positions for five runs of the GA and gradient descent on the same topology. Over all sample topologies, the average variance for node points in solutions evolved by GA was 0.55 units, compared to an average of 0.37 units for gradient descent indicating that the GA searched the space more broadly but gradient descent was able to find similar solutions from different initial start points (figure 4.16).

d. *Topology and confidence:* Both algorithms display some variance in both point position and in fitness, but the effectiveness of the search does not appear to be biased toward topologies with smoother fitness landscapes. Of the 70 randomly generated samples, if only topologies with an easily found optimum were suited to optimisation, one would expect a correlation between the variance in point position and the variance in final fitness, as only topologies with consistent solutions should have consistently high fitness. This is not the case (figure 4.17).

Overall results of the comparison indicate that the GA searches the space more broadly but gradient descent achieves similar fitness with equal confidence. Either method can be used to derive fit solutions. The gradient descent algorithm is advantageous however, as it is deterministic and results in a lower computational cost (the GA required 4.2 times as many fitness calculations). Although the material properties of the fabrication method produce some irregularities in an otherwise smooth fitness landscape, the results indicate that these are not substantial enough to greatly hinder a gradient descent optimisation. As this can be searched effectively and more efficiently by gradient-based methods, gradient descent would be used in subsequent work.

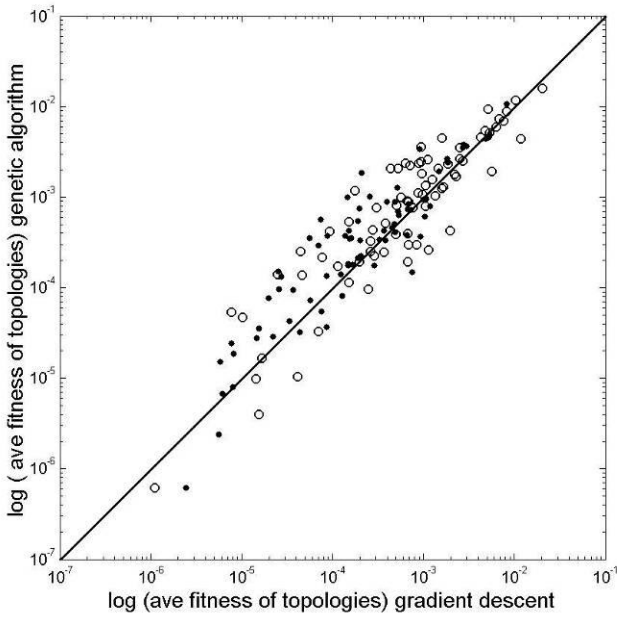


Figure 4.14. Average fitness found by gradient descent plotted against the average fitness of the GA for each topology. Fitness when member angles are ignored (•) and used to adjust the stiffness calculation (○) are shown.

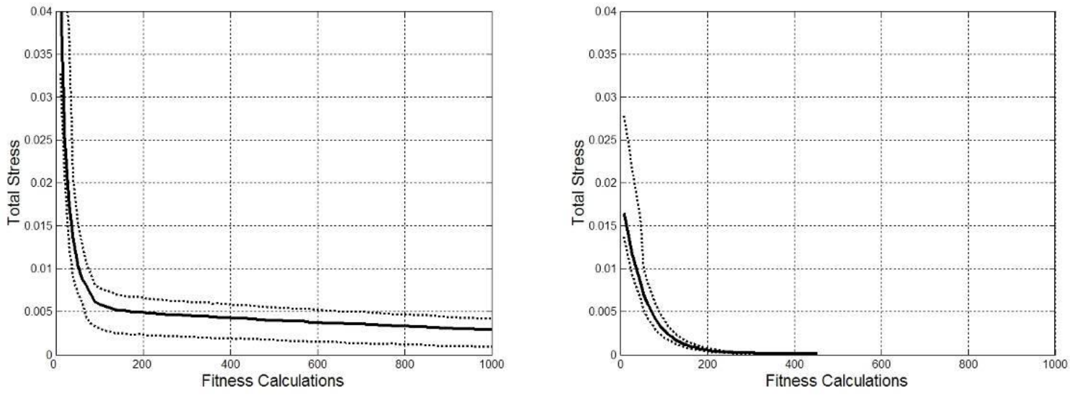


Figure 4.15. The mean and variance of the GA (left) and gradient descent (right)

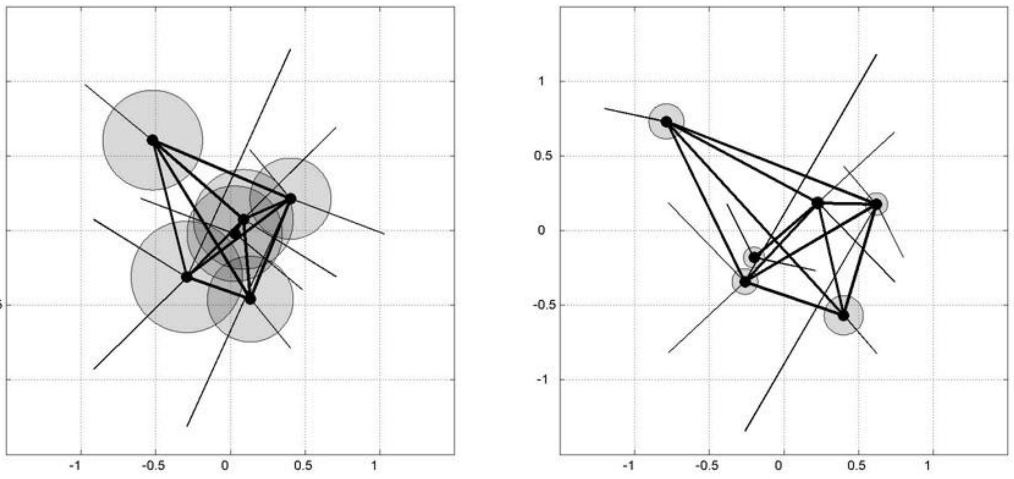


Figure 4.16. Mean node points found by the GA (left) and gradient descent (right). Variance is indicated by the grey circles.

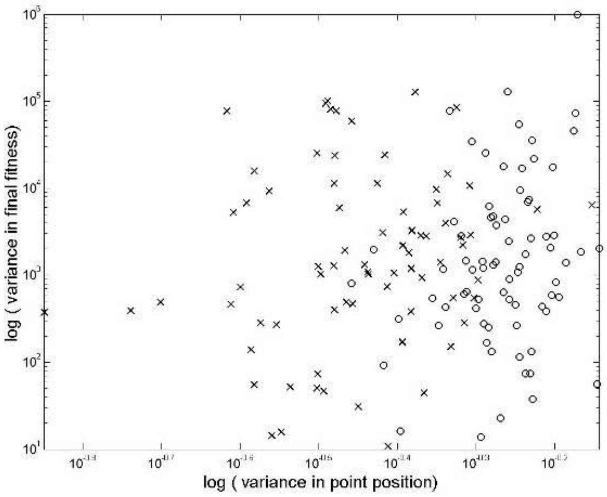


Figure 4.17. A graph of the variance in point position vs. variance in final fitness (GA depicted by o; gradient descent depicted by ×).

4.4.3 Extensions: Emulation of specific material properties via optimisation methods

In addition to stiffness, other dynamic properties of structures are amenable to design by the same optimisation and learning method. Properties such as Poisson's ratio (the expansion in the plane normal to the applied force) are generally considered material properties, but are dependant on the dynamics of an internal structure, and have been investigated as such (Bendsoe and Sigmund 2003; Lakes 1987). Such properties can be calculated by the same techniques as used to determine stiffness in this chapter, namely the deflection found by FEA. Examples of simple structures indicated that such dynamic behaviour is a result primarily of the geometry of a structure rather than its topology, as illustrated in figure 4.18. In this case the configuration on the left has a negative Poisson's ratio, while the same topology on the right is positive.

Achieving these properties in materials has been the focus of much design effort (Bendsøe and Sigmund 1999; Lakes 1987); however, in the same way that stiffness was optimised in the above section, these other properties appear to be achievable by optimisation simply by changing the objective. The fitness landscape for Poisson's ratio of the same topology in section 4.3 is shown in figure 4.19. This is not as smooth as that of stiffness (figure 4.12) but is still navigable by gradient descent. Dynamic properties such as anisotropic deflection in any desired direction and Poisson's ratio were explored in a number of randomly generated topologies and it was found that specific values of these could be approached in each. Figure 4.20, for example, shows the same topology with both a positive (left) and negative (right) Poisson's ratio. By using the difference between measured values and a specified target as an objective function, specific properties of a material may be achieved.

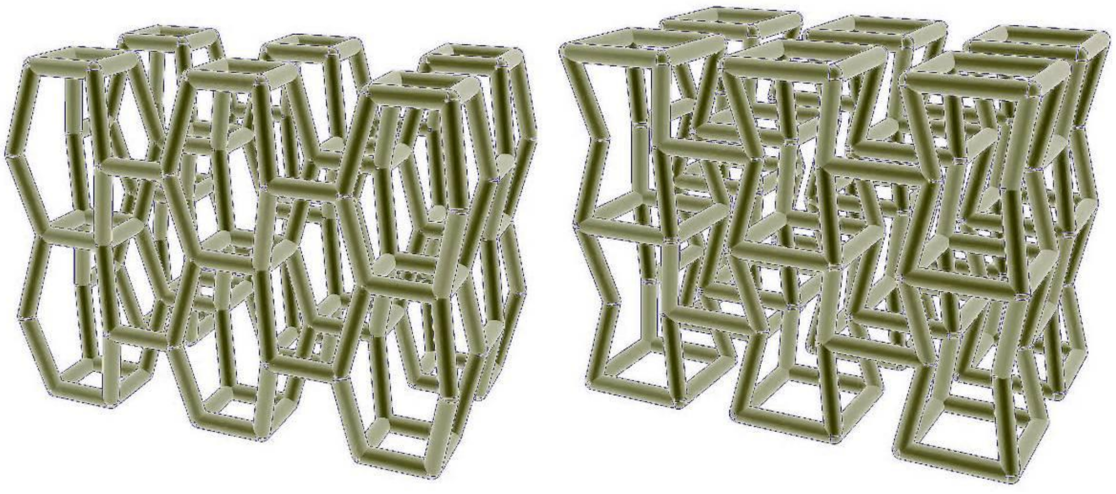


Figure 4.18. Positive and negative Poisson's ratio structures of the same topology.

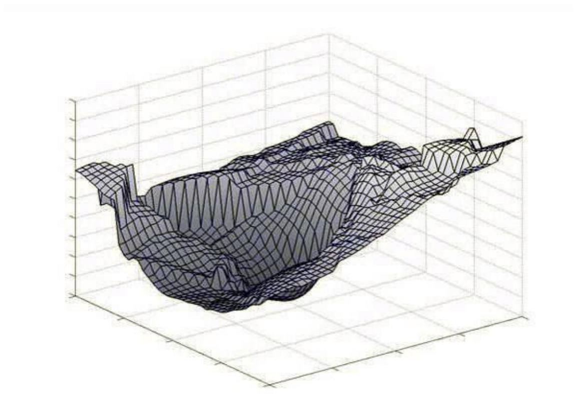


Figure 4.19. The fitness landscape for Poisson's ratio.

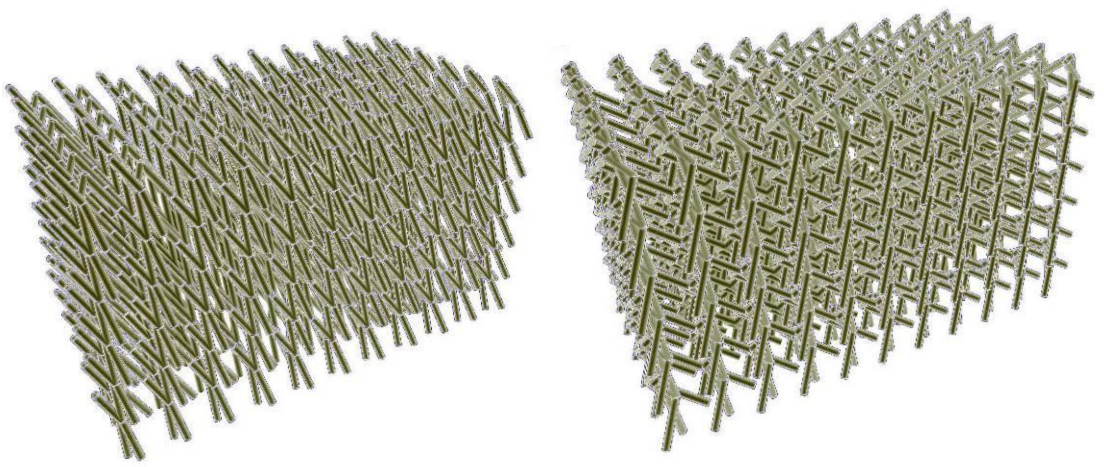


Figure 4.20. The same topology optimised for positive and negative Poisson's ratios.

4.5 Modular optimisation by discrete, finite element approximation

The optimisation of a larger structure is based on the strategy of simplifying units by means of an isomorphism/homomorphism. The method first makes a large scale finite element approximation of the form using unit cubes as the elements. After analysis each simple element is replaced with a more detailed structure found to meet local requirements.

4.5.1 METHOD

The proposed method can be split up into five steps, summarized as follows:

1. *Analysis of the object as a whole*[*]*:* The total system is first analysed under the simulated load as a complete object. This continuous system is approximated by the finite element method (FEM), modelled as a set of 8 node hexahedral solids, each with 24 degrees of freedom, on a regular 3-dimensional grid. A simple cantilever (figure 4.21) was used as a proof of concept, with one end fixed and the other end loaded vertically. FEA determines the resulting forces at each point within the volume of the object—the range in colours from dark to white represent the range of stresses present within the object.

2. *Division of the object into unit cubes:* The volume of the object is divided into a three dimensional grid of unit cubes, each to contain a modular unit of structure. This grid may correspond with the FEA grid, but in many cases the analysis is performed at a lower resolution and the results interpolated to the unit cube scale. The analysis provides the relative deflection for each of the x, y and z-axes, which vary gradually from unit to adjacent unit. The geometry of the structural units will change in response to this, so that adjacent units will contain a very slightly different structure. In the object as a whole the geometry of each cube will vary considerably, but do so gradually and continuously over its volume in response to the local stress (Figure 4.22).

   Topology will not change, which accomplishes two things necessary for scalability. First, it avoids abrupt breaks in the continuity of the structure, where adjacent units of

[*] This is simply an analysis of an assumed homogeneous solid, showing the regions which require more or less structure in any axis. A further stage of optimisation (e.g. Bendsøe and Sigmund, 2003) would ideally be employed here, particularly for more complex objects, but in practice the initial analysis was found to be a good approximation. If an initial optimisation is used, it would yield a field of desired material properties that would be used in the same way as the field of deflections above.

incompatible structures would be zones of weakness in the overall object. Second, it allows a single graph of connections to be evolved which can be applied to all modules. This allows an object of any size and any number of units to be evolved once, rather than requiring many runs of the GA for large objects or complex load conditions.

3. *Evolution of a suitable topology for a representative sample of stresses:* The method in section 4.3 evolves a suitable topology for a given range of local loads. A set of sample unit cubes are taken to represent the range of these loads, based on the relative displacements of the units in the analysis (steps 1 and 2). For each candidate solution, the sum of the mean displacements of the nodes for this sample set then forms the fitness of the topology. The GA thus evolves a topology that can best accommodate this range of stresses across the whole object.

4. *Repetition of the module, optimising geometry to local conditions:* At this point, the entire continuous approximation of the object is replaced, one unit cube at a time, by a set of discrete elements as defined by the chosen topology. In the application of the module to the object, each unit is individually optimised by gradient descent (section 4.4) to best support its local load (steps 1 and 2). Because the changes in strain are gradual across the continuous simulation this geometry changes gradually with small changes of the applied forces from unit cube to adjacent unit cube.

5. *Manufacture by stereolithography:* The module is advantageous also at this final stage of the computation. Objects with complex internal structure are problematic for standard CAD boundary representations of geometry (e.g. .STL format) due to the high number of internal polygons (Knoppers and Hague 2005). Standard software that operates on these also requires a check and generation of additional support structure which is unnecessary and costly for these structures. The format of the unit grid, however, lends itself to direct processing into the horizontal slices by which the machine actually operates at a lower level*. It could be used stream data directly to the machine, by generating the internal structure online during fabrication, for very complicated or large objects. Figure 4.23 illustrates how one unit cube is sliced formatting. The machine instruction consists of a series of 2-dimensional outlines of the geometry that determine the edges of the slice to be filled by solid material. These are easily calculated by centring a standard polygon on the intersection of each member

* Rosen (2007) uses a similarly low-level method, but units are not aligned to the grid.

with the slice plane, scaling it uniformly by the diameter of the member, and axially by the tangent of the angle.

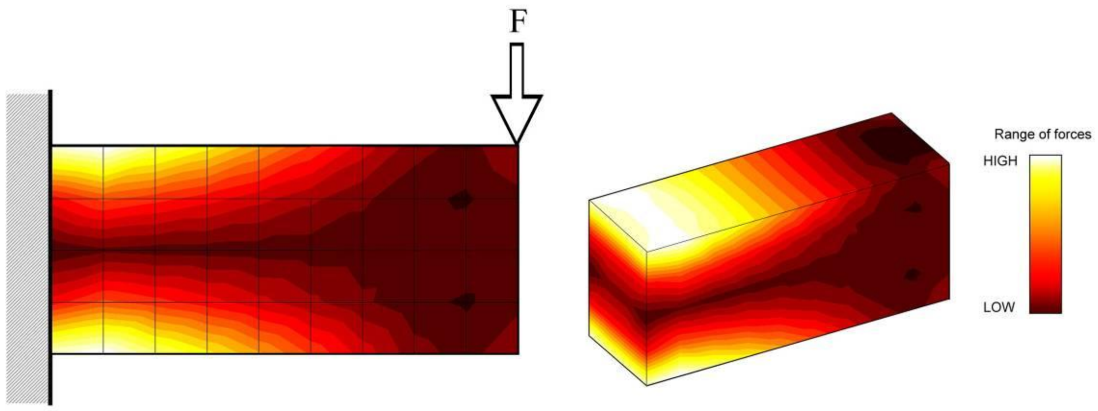


Figure 4.21. The cantilever as analysed by finite element analysis.

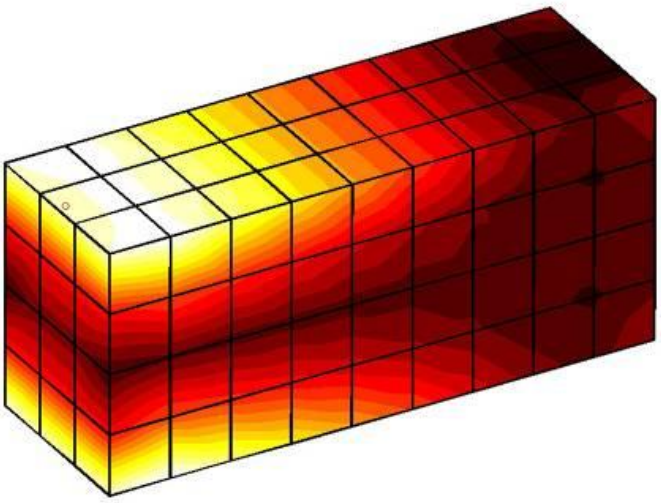


Figure 4.22. The cantilever beam is split up into unit cubes.

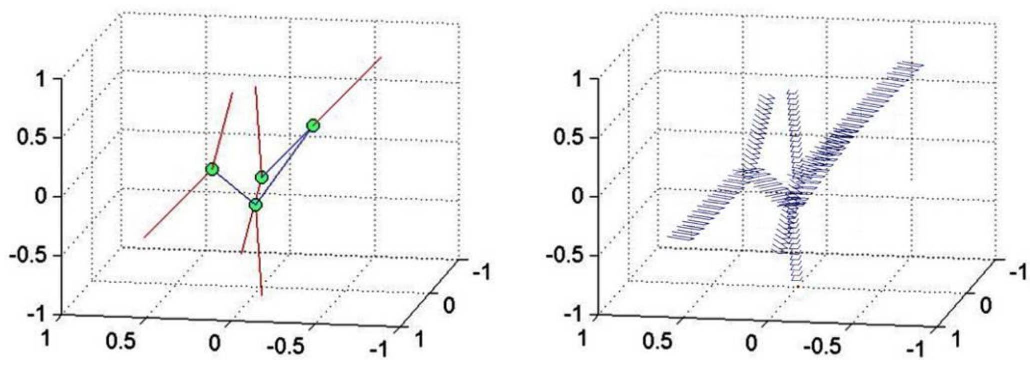


Figure 4.23. The structure of each unit cube is ‘sliced’ to .slc format.

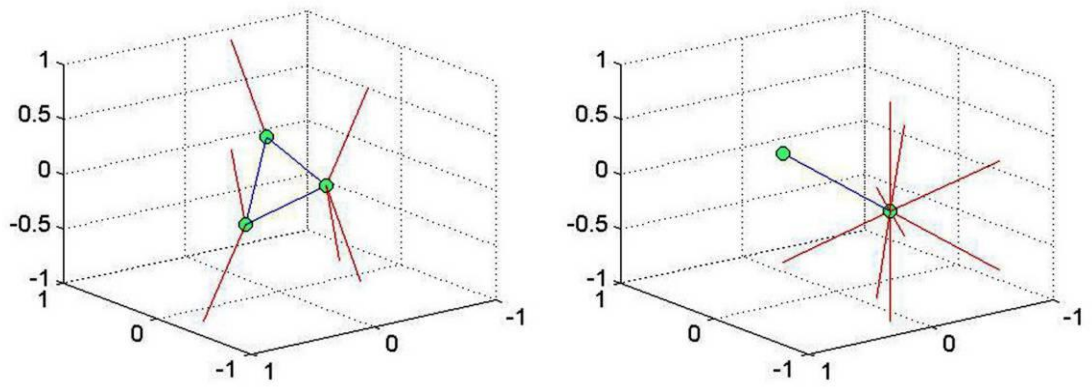


Figure 4.24. An evolved topology (left) and a simple, engineered version (right).

4.5.2 Comparison of the optimised structures

Structures were optimised for the cantilever to evaluate the GA optimisation of topology and the local geometry optimisation by gradient descent. To test the topology optimisation, two topologies were compared with one another, the first a manually designed example similar to the standard used for fine support structure, and the second evolved by GA specifically for the cantilever (Figure 4.24). To test the local geometry optimisation, each of these was employed in an overall structure in which the geometry of units was identical throughout, and then with geometry optimised to the local stresses calculated by the global FEA. Boundary and loading conditions, the total mass of material and gross dimensions were maintained across all test cases. Deflection of the overall structure under identical loads was calculated by FEA, and the comparative results of this are given in the length of one side of a unit cube.

First the topologies were replicated without any variation in the geometry of the units or diameter of struts. As can be observed in the following diagrams, the uniform repetition of both the engineered and the evolved topology perform relatively poorly under the cantilever loading conditions when all modules are the same. Maximum deflections of the structure for the engineered and evolved topologies are 2.052 and 3.623 units respectively. In this case the topology evolved by GA appears to perform particularly poorly, with a maximum deflection of 76.5% more than the engineered topology, and a marked asymmetrical twist in the deformed structure.

Local optimisation of units by gradient descent yields a marked improvement in stiffness. With geometrical variation of the structure, the total deflections for the engineered and evolved topologies were 0.303 and 0.053 units respectively, only 14.8% and 1.46% of the unoptimised deflections.

Looking again at the use of the GA, while the optimised topology performed poorly with no variation in its geometry, it now deflects only 17% as much as the engineered structure. The use of several force vectors in the objective function ensures this evolved topology was not designed for one specific load condition but for a range. It therefore only reveals this benefit when replicated throughout the volume of the beam with its geometry varied and struts thickened in accordance with the force conditions present in each unit cube. A closer look at the geometry of the more regular engineered module indicates that it is relatively inflexible in changing only in the thickness of the struts, whereas the evolved topology accommodates the variation in stresses also by changing the length and angle of struts throughout the structure. It is this greater ability to adapt geometry to the changing conditions of stress across the structure that increases the overall performance of the evolved topology.

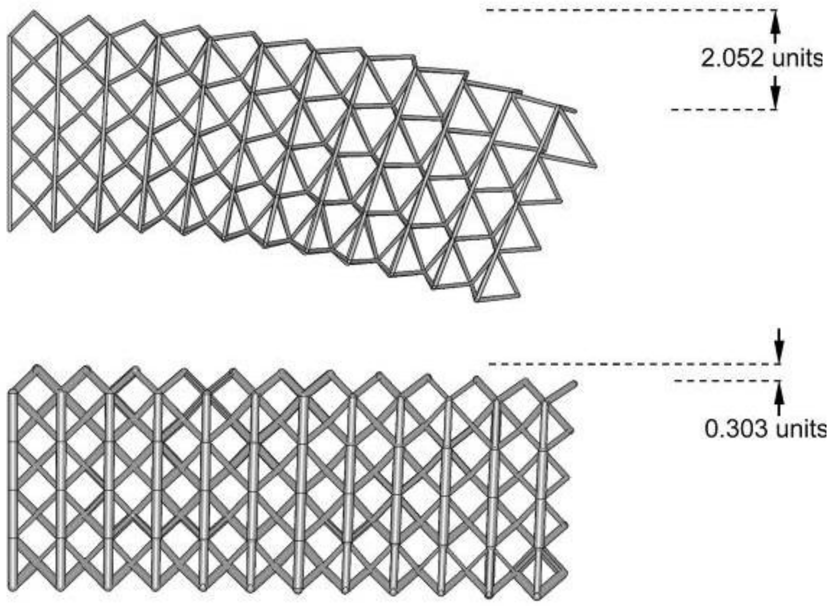


Figure 4.25. The engineered topology with identical modules (top) and optimized (bottom)

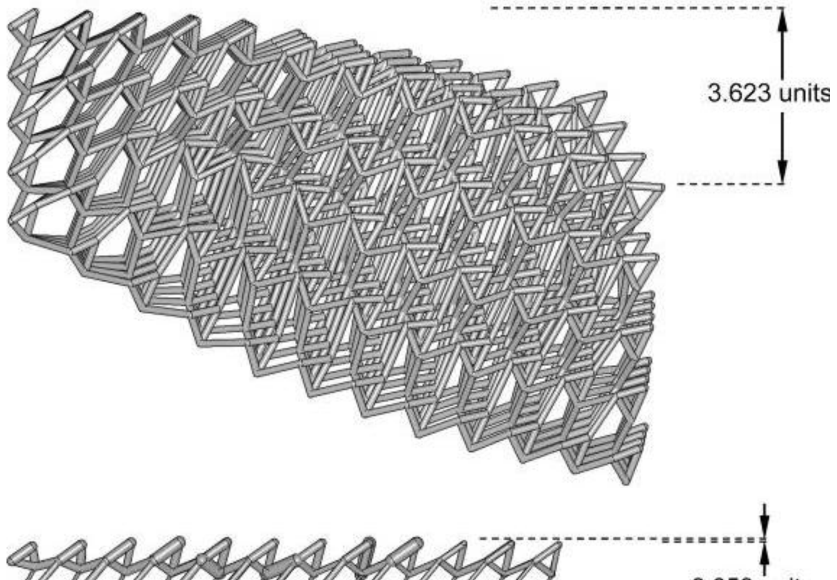


Figure 4.26. The evolved topology with identical modules (top) and optimized (bottom)

4.5.3 Analysis

The analysis of the generated designs under the specified loading conditions yielded favourable results. From the comparative analysis of the performance of optimised and unoptimised structures under identical loading conditions, several conclusions can be drawn:

- Structures in which the unit module is optimised to accommodate local force conditions outperform those that are uniform throughout. This is the case whether the topology is a generic grid or evolved for the specific stresses of the problem.
- During optimisation it is not necessary to simulate the entire structure in all its complexity, as an abstraction by finite element analysis allows the local stresses to be approximated at any point. Local sampling is an approximation that applies to the structure as a whole; so optimising each unit cube individually for these local stresses is a far more efficient method, shown by our results to be effective.
- Evolved topologies can enhance the optimisation by being more flexible. Particularly as applied forces become more complex there is a greater need for a non-standard topology. A greater ability to change geometry in terms of node positions and strut diameters allows a module to better accommodate local forces, and the GA can find topologies with this property.

This method treated a complex microstructure as though it were a continuous material with properties that could be changed to benefit the design. It demonstrated that materials can be fabricated, using a technique such as stereolithography, which are optimised at every point to the forces present there. This method of fabricating complex structures through the use of a flexible module has several benefits in terms of efficiency that are based on the saving in computation time and inherent to the production process itself. These are:

- The use of a module allows repetition indefinitely, limited only by the physical limitations of fabrication.
- The flexibility of the module allows a high degree of complexity of structure.
- Scalability to larger and more complex structures is easily accomplished. The speed of computation of the .slc files increases at same rate as scale of the object to be built and the fabrication time. Evolving a topology with the GA is performed only once at the outset, so the overall computational efficiency actually increases with larger scale structures.

4.6 Chapter review

The practical benefit of the method shown is the efficiency gained by the simplification of units. The initial FEA of the object scales approximately linearly with member size and model complexity. The substantial saving is in the optimisation, as each unit module is kept to a constant number of members, and thus the algorithm scales only linearly here as well, rather than exponentially.

In terms of the complexity of the overall structural system, the technique employed an approximation of the low level parts of the system to simplify the model. First, a standard FEA model employed a simplification of units for the object as a whole, employing both an approximation of a supposed material by a set of simple hexahedral elements, and then optionally talking advantage of a change in resolution. Then, in design, the actual structural unit optimised was isomorphic to one of these far simpler elements. The isomorphism in this stage is likely to involve an increase in internal complexity, but it can be isolated from its surrounding units. The two sub-problems of topology and geometry to which optimisation was applied were thus made tractable for large structures by the modular nature of the task. Rather than performing a gradient search in thousands (or millions) of dimensions, for example, a module with only a few struts required only about a dozen.

The units were optimised in isolation, but this does *not* indicate that each unit can stand in isolation from its neighbours, or that the behaviour of each unit is not embodied within the environment of its neighbours. The continuous cubes of the initial analysis must be replaced with a unit optimised to those particular forces. A weaker or stronger unit in a certain location, or one with a different orientation, would effectively represent a discontinuity of material properties from the initial FEA step, and thereby affect all its neighbours. In keeping with the importance of interaction in complex systems, this isolation of units to be simplified was possible only because the topology remained constant, and therefore each unit interacts with its neighbours in exactly the same way.

The success of the optimisation indicates that an isomorphism or homomorphism is possible between the low resolution FEM approximation of the continuous material and the final designed modules—it is possible to produce the specific dynamic properties of one structure using a totally different structure. This hints at but does not fully address Hypothesis A, as it is still very much within a standard engineering framework—the behaviour of finite elements is well understood and explicitly built into the model. But by showing how the structure can be broken down into isomorphic modules, with interfaces that are complex but nonetheless specifiable, it does introduce the possibility of another method of simplifying the model of a complex system, in which its higher level behaviour is mimicked without simulating the underlying causes. Now that the modular topology of structural units has been established, the following chapter will make this second approximation to these using machine learning.

Chapter 5: Inductive learning for optimisation[*]

Summary: Machine learning is used to approximate the behaviour of each structural cell in the space frame. This provides evidence against Hypothesis A, in that an explicit structual model based on member geometry is no longer necessary to predict behaviour. Hypothesis B is less relevant to this problem to the extent that it involves a clear optimisation objective, however some evidence is demonstrated against this in that the approximations are founded on data communicated to the machine only as observations of structural behaviour.

Hypothesis A suggests the underlying structural behaviour of the space frame must be simulated, as was done by modelling the struts by FEA in Chapter 4. In both structural and spatial domains, this research proposes to use inductive machine learning to aid in designing with complex systems. Inductive reasoning (Harman 1992; Russel and Norvig 1995) derives a generalised conclusion from repeated observations of the same kind. Instead of making a repeated simulation and explicit analysis of many different solutions, a trained learning algorithm is able to make decisions based on prior experience. To falsify Hypothesis A in the context of the structural domain, this chapter uses a learning algorithm to build a higher-level approximation of the optimal structural system based only on observations, without knowing underlying causes. This will be tested by use in design optimisation. The isomorphism used in Chapter 4 has clarified the boundaries of the structural unit and its interface. The same 'unit cube' will be used here as the structural subsystem to be approximated.

Hypothesis B concerns communication with the machine. At this point the design objectives are quite simple but clear goals of stiffness and mass. Nevertheless, the learning algorithm will have no knowledge of them. This limited evidence against Hypothesis B will become more relevant in the spatial problems where objectives are less clear.

[*] Some of the research in this chapter (involving model A) was conducted in collaboration with Siavash Hwroun Mahdavi and published in Hanna and Haroun Mahdavi (2006). Much of the remainder has been published as Hanna (2007).

5.1 Learning of the structural problem by support vector machine (SVM)

The second stage of optimisation in Chapter 4 aimed to find the optimal geometry of a unit given a set of loading conditions. This geometry was clearly defined parametrically by node positions and the local load on a cube was defined by vectors in each of the three axes. The low variance between solutions found by gradient descent (§4.4) suggests that there is a single, unique optimal structure for each load, so this geometry can be considered a function of the load:

$$optimalGeometry = f\,(\,localLoad\,) \tag{5.1}$$

Even with efficient gradient-based methods, the repetition of iterative optimisation is time consuming for structures of many modular units. Defining this function can be a more efficient method of structural optimisation than the repeated search procedure.

A support vector machine (SVM) will be used to learn this function, replacing the iterative analysis and optimisation entirely (figure 5.1), and resulting in a very efficient and accurate optimisation procedure. Each training example for the SVM is an optimal solution found by the gradient descent algorithm in which each sample is a finite element analysis of the structure. These solutions can be calculated manually, and so the overall effect of machine learning is primarily to replace the simulation step, and speed up the output of a solution from the sampled forces that are present within the object. This can be seen in figure 5.1 as isomorphic with the entire process of simulation and geometry optimisation, sitting within the overall process of generating the structure.

The function itself has a three-dimensional vector input corresponding to the stress (in either tension or compression) in the three axes of a given unit cube, and an output consisting of the node point positions for the optimal structure. The nodes are also located in three-dimensional space, so for a topology of *n* points the output is a 3*n*-dimensional vector. A single topology will be used consisting of four node points per unit cube, resulting in a 12 dimensional output. Each training sample is created by generating a random input vector from the normal distribution with mean $\mu = 0$ and standard deviation $\sigma = 1$, and the node point outputs are found by the gradient descent method described above. A radial basis function kernel is used for all dimensions in the SVM with variance $\sigma^2 = 0.2$. The SVM is trained on the gradually increasing set of stress vectors and node points until the accuracy of the learned function no longer increased.

Validation of this accuracy is performed by comparing the output from the learned function with the actual point positions for stress/node pairs in a separate validation set.

This method has the purely practical benefit of speed. But learning offers another, perhaps greater benefit in that the model may incorporate a complexity beyond what can be modelled explicitly. It can thus model something that could not otherwise be simulated because it is not fully understood. Finite element approximations can be made to simulate such complex microstructures as those of wood (Astley et al. 1997) and bone. These are nearly homomorphic approximations of a structure's actual behaviour, but there is always some discrepancy due to the resolution of the model or heterogeneity of the material at the fine scale, so where:

$$actualBehaviour = g\ (\ modelGeometry\ ) + \varepsilon \qquad (5.2)$$

in which ε represents complexities present in the actual structure but not in an abstract model, the best we can do is to minimise ε. Much of this discrepancy is caused by digital fabrication methods too may introduce added complexity into a structure that does not exist in the initial model: geometry changes due to resolution; thickening of members at the nodes due to resin viscosity or raster surface irregularities; irradiation times; etc. Modelling the geometry of these irregularities would normally require a drastic increase in mesh resolution and corresponding computation time, but many appear to be dependent on the model geometry, e.g. the correlation of stiffness and member angle (§4.2.3), so Eq. 5.2 can be decomposed:

$$actualBehaviour = g\ (modelGeometry\ ) + e\ (modelGeometry\ ) + \varepsilon' \qquad (5.3)$$

where $\varepsilon' \ll \varepsilon$. By operating with a combined function $(g{+}e)(\cdot)$, ε is minimised. The function $f(\cdot)$ is learned from the overall observed *actualBehaviour* of many instances, and can thus incorporate $(g{+}e)(\cdot)$ implicitly. In this chapter, the function of member angle to stiffness (§4.2.3) is used again to simulate the additional complexity of $g(\cdot)$, but measurements taken directly from prototypes of the specific topology and machine might otherwise be used.

A third benefit concerns the fact that any optimisation algorithm is also subject to some error due to local optima in the search space. Changes in initial conditions or stochastic optimisation cause the same *localLoad* to result in different solutions to geometry, and even controlling these would occasionally result in adjacent units with a very small difference in *localLoad* to have a disproportionately large difference in geometry, weakening the overall structure. In learning the overall function (Eq. 5.1), these changes are treated as noise by the algorithm, and substantially eliminated. This will be discussed further in (§§5.3.2 and 5.3.5).

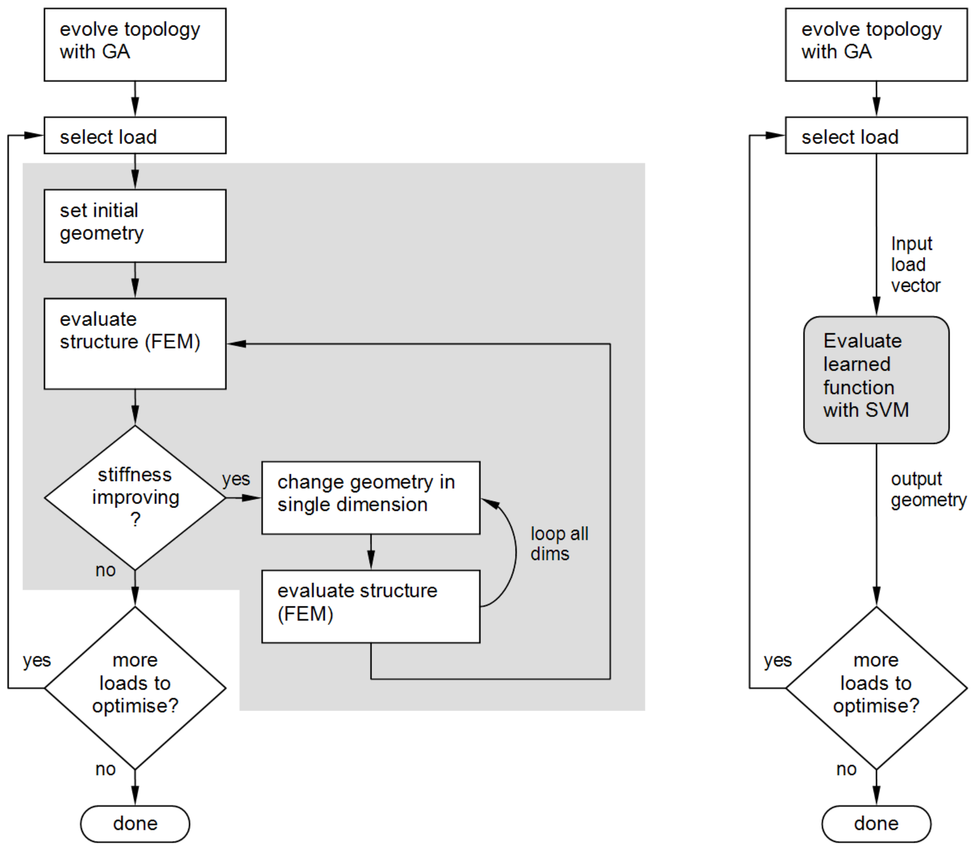


Figure 5.1. The iterative optimisation process (left) is replaced entirely by an isomorphic function learned by the support vector machine (right).

5.2 Learning methodology

The proposed method builds upon the optimisation procedure introduced in chapter 4, which uses a FEM analysis at two stages: first to estimate the global distribution of stress across the entire loaded object, and then to guide the incremental optimisation of each single module.

5.2.1 The optimisation model

This chapter addresses primarily the second analysis, in the optimisation of the individual structural modules. In the final object, each of these will be connected to a number of slightly varying neighbouring modules through which the loads will be distributed, but for the sake of efficiency and because the final structure is not known the optimisation deals with each unit in isolation. The struts internal to one unit of structure and all struts that connect it to its adjoining neighbours are used in the model, which due to the grid arrangement entails a duplication of each of the struts shared by one unit and its neighbours. These pairs of struts determine the boundary conditions: each component of the applied load (in the x, y, or z axis) is divided equally between all struts that connect to neighbouring modules in the positive direction of that axis, while the corresponding degree of freedom is constrained in the negative direction. If three struts connect to units above a module (positive z axis), for example, there will be three corresponding struts connecting to units below. For a total applied load of -0.6 units in the z axis, -0.2 would be applied to each of the upper struts at their extreme node, while the lower nodes of the lower struts are constrained to the x-y plane. The same is true of the other axes. The extreme node of a single unloaded strut is fixed in all three axes to restrain the whole system. Each strut is modelled by a 12 degree of freedom beam element with one node at each end. All joints between struts are fixed in both bending and rotation.

All optimisations are tested relative only to each other and so where possible no real-world units are used for dimension, force, etc. Actual material properties vary between resins used in stereolithography, but are kept at constant default values to simulate a standard isotropic solid with Young's modulus of $E = 7000$ kPa, a Poisson's ratio of $\nu = 0.3$ and density $\rho = 2500$. Adjustments made to simulate the effect of the manufacturing process on struts of varying angles (§4.2.3) were made after FEM analysis, by dividing the calculated deflections by the measurements taken in physical testing. The member sections are treated as cylindrical, with identical moments of inertia in each axis and torsional stiffness assumed from the polar moment of inertia for a cylinder $J = I_1+I_2$.

5.2.2 OPTIMISATION BY GRADIENT DESCENT

Gradient descent (GD) is again used as the optimisation method, yielding the node positions to minimise the total deflection in the structure under a specified load applied to the unit cube (as in §4.4). Simultaneous optimisation of member section diameters and nodes was found to be more difficult for gradient methods, so instead the nodes are first determined GD with a constant member section, and final member sections found as a second step as a simple function of the relative member stress. Simulation of stress and deflection is performed using the finite element method, using OpenFEM, an open-source finite element toolbox accessed via MATLAB.

5.2.3 Approximations in the model

Several approximations are made in the above model, particularly in the abstraction of a single unit of structure in isolation, but these approximations have been shown effective in practice and may be further refined without compromising the issue of learning presented here. This work focuses specifically on the ability of the machine learning algorithm to approximate the output of any optimisation model, and as such is concerned more with internal consistency. The SVM can be trained to learn several slightly different versions of the above description, and it is assumed that its output would only improve if the optimisation model can be made more accurate.

5.2.4 The learning algorithm

Support vector machines (SVM) (Vapnik 1995) can be described generally as a type of linear classifier that uses a non-linear kernel function to map input data to a sufficiently high dimension such that it can be separated by a hyperplane (Duda et al. 2001). The transform resulting from this kernel function ensures this hyperplane is non-linear in the original input space, and so the SVM can just as easily be used in regression to a non-linear function as in classification. They will be used in this capacity to learn the function of optimal structures.

Given a data set *D*, consisting of an input vector **x** and a response vector **y**, the function to be learned

$$\mathbf{y} = f(\mathbf{x}) \tag{5.4}$$

is approximated by the SVM by building a model $f'(\mathbf{x})$ based on *D*, that enables the estimation

$$\mathbf{y}' = f'(\mathbf{x}). \tag{5.5}$$

The type of SVM used here is a Least Squares Support Vector Machine (LS-SVM), in which the solution follows from solving a set of linear equations, instead of quadratic programming for classical SVMs (Suykens et al. 2002). A Gaussian radial basis function is used as the kernel.

5.2.5 Learning objective

The design objective to be learned is to find the best structural geometry for a single modular unit given the input of its external load. The task is simply this: for each set of loads, find the set of nodal points that represent the optimal structure (Figure 5.2). As a function (Eq. 5.4), the input **x** is the three-dimensional vector of external forces corresponding to the stress (in either tension or compression) in the three axes of a given unit cube. This is represented by the components in the directions of the x, y and z axes:

$\mathbf{x} = (x_{(x)}, x_{(y)}, x_{(z)}).$ (5.6)

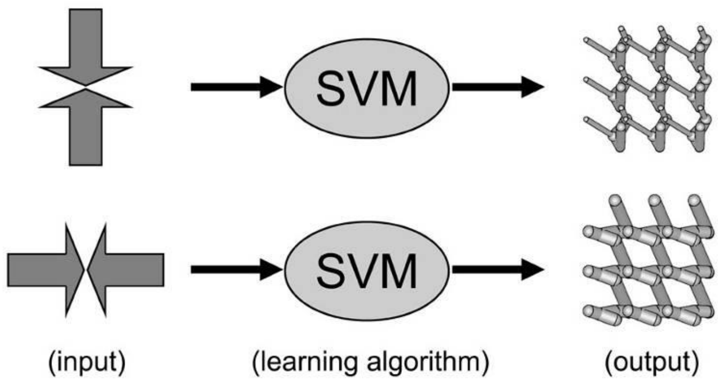


Figure 5.2. Different force inputs result in ideal geometry outputs.

The output structure **y** consists of the nodal point positions for the optimal structure as found by prior GD optimisation. This is the set of (x, y, z) coordinates for each of the nodal points y_i:

$\mathbf{y} = (y_{1(x)}, y_{1(y)}, y_{1(z)}, y_{2(x)}, y_{2(y)}, y_{2(z)}, \ldots, y_{n(x)}, y_{n(y)}, y_{n(z)}),$ (5.7)

The nodes are also located in three-dimensional space, so for a topology of *n* points the output **y** is a 3*n*-dimensional vector. The inclusion of section sizes and other optimisation parameters such as material properties would be a simple extension of the dimensionality of **y**, but only the nodes are used here.

5.2.6 Output complexities

As described in §5.1, the output vector **y** to be learned takes into account not just the behaviour of an ideal, homogeneous space frame, but also the complexities caused by the fabrication process as in Eq. 5.3. The nonlinear function in §4.2.3 was applied to each strut to simulate these. The function of node positions to resulting stiffness of a standard finite element space frame model

$\mathbf{s} = g(\mathbf{y})$ (5.8)

is illustrated in part in Figure 5.3 (left) where the horizontal plane represents a two-dimensional slice through the 3n-dimensional space of nodal points **y** and the vertical axis represents the stiffness **s**. When changes in member strength due to fabrication angle are simulated, a revised stiffness function

$\mathbf{s}' = g'(\mathbf{y})$ (5.9)

reveals the more complex behaviour illustrated in figure 5.3 (right). This incorporates the complexities ε (Eq. 5.2) or the function *e*(·) (Eq. 5.3), that form a crucial element of the function to be learned. It is the function **s'** on which gradient descent is performed to optimise stiffness. Although the function *g'*(·) used here is derived from iterative gradient on the modified finite element model, it is this function that could also incorporate samples taken from physical testing.

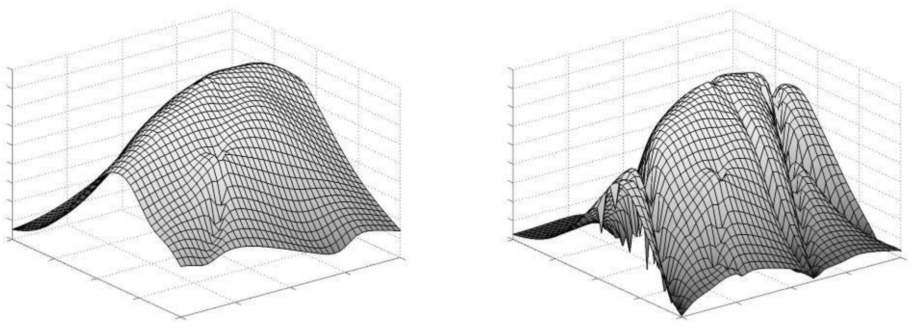


Figure 5.3. Samples of the stiffness function of a given topology for homogeneous struts (left) and for struts that vary in stiffness as a function of angle (right). In these plots increased stiffness is upward: the gradient is ascended in optimisation.

5.2.7 The data sets

Learning results of two different structural models based on the same unit module were compared. Model A is more constrained, having all unloaded boundary nodes fixed with pin joints and identical material properties for all elements. In model B, boundary nodes are fixed only in the axes in which they connect to surrounding unit cubes (§4.2) but are free to slide in the corresponding plane to more accurately simulate the flexing of the surrounding structure. Also, the effect of strut angle on elasticity due to the manufacturing process (§4.2.3) is simulated. Both of these result in the behaviour of model B more closely resembling reality, but make the search space more complex. For practical reasons, optimisation of model B was also terminated after 110 iterations, regardless of whether an optimum was found. While GD optimisation of model A always resulted in a near global optimum, the variance of solutions found for model B was greater (Figure 9), amounting to considerable increased noise in the function to be learned.

A single topology was used consisting of four nodes per unit cube, resulting in a 12-dimensional output y. For each of models A and B, the data set *D* was created not to uniformly sample the entire space of possible solutions, but for a normally distributed range of forces and the associated optimal solutions.

Each training sample is created by generating a random input vector x from the normal distribution with mean μ = 0 and standard deviation σ = 1, resulting in a range of approximately [–3:3] units of force in each of the three axes. The actual distribution of each of the components of x are plotted in Figure 5.4. The nodal point outputs y are found by the gradient descent method described in section 3.1.1, and result in asymmetrical distributions of nodal positions throughout the space of the unit cube. The distributions of each of the four nodal points in the three axes of space are shown in Figure 5.5. Although the positions of nodes are not constrained by the optimisation algorithm, the repeated nature of the structural modules implies a maximum bound on the search space of one unit for each of the components of y. The variance in the data set for each of the points is 0.72, 0.49, 0.53 and 0.56 units in this space respectively, indicating a large portion of the space was sampled in *D*.

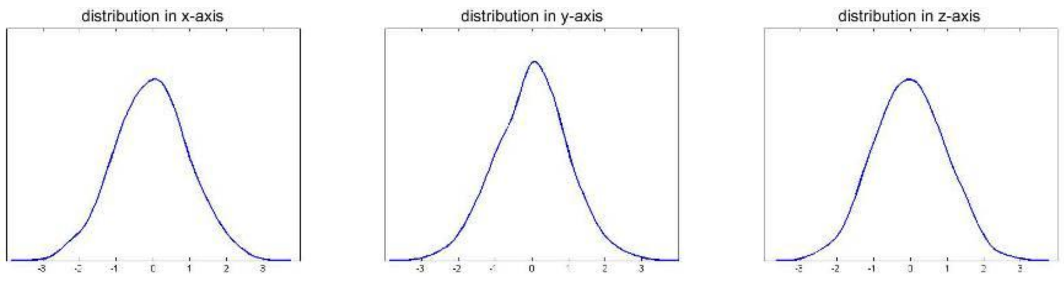


Figure 5.4. Probability distributions of the x-axis, y-axis and z-axis components of input force vector x are based on a normal distribution with mean zero.

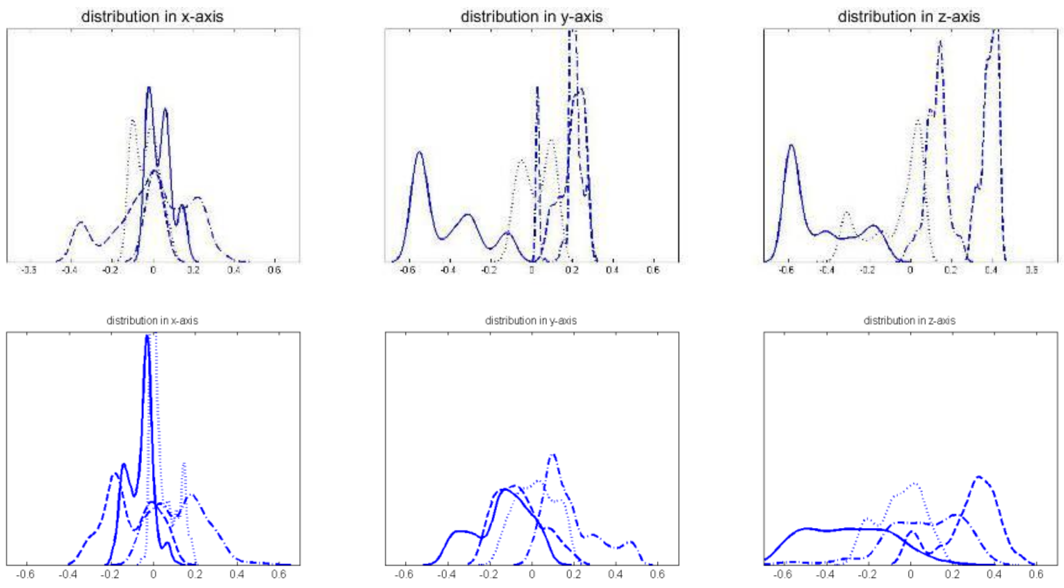


Figure 5.5. Probability distributions of the x-axis, y-axis and z-axis components of the nodal points y are asymmetrical in physical space, the result of optimisation. Model A is shown above, model B below.

5.2.8 Training

The learning objective set for the SVM is to predict the optimal geometries of a structure y given different force conditions x. Each training example is an optimal solution found by the iterated gradient descent algorithm in which each sample is a finite element analysis of the structure. The greatest computational cost is therefore in generating this training data, and so the proposed learning method in practice would be online, with a gradually increasing data set. Here, batch process was used so that tests of the method could be controlled.

Training of the SVM was performed with the gradually increasing set of stress vectors x and nodal points y until the accuracy of the learned function no longer increased. A radial basis function kernel with variance $\sigma^2 = 0.2$. was used to map all dimensions in the SVM.

5.2.9 Error estimation

Methods of error estimation have been systematically evaluated in (Reich and Barai 1999). The method used, hold-out, is the most conservative, in that it maintains a pessimistic bias toward the results.

The data *D* is divided at random into two sets: a training set *T* and a separate validation set *V*. The SVM is trained on *T*, and then evaluated on *V*, the errors in *V* indicating the generalisation error. For *D* of size *n*, the size of *T* is ideally 0.6n to 0.8n and *V* is the remaining 0.2*n* to 0.4*n*. While there are no general bounds for regression, the data *D* of size $n > 1000$ produces results with confidence more than 0.95 in classification problems (Reich and Barai 1999).

The following tests conform to these recommendations for accuracy. The performance of the SVM was evaluated for training sets of varying size, to a maximum size $n = 1300$. For all tests, the validation set *V* was a randomly selected set of size 300. The size of *D* for which the SVM will be considered in the tests to be fully trained occurs at $n = 1000$, which is the recommended size for 0.95 confidence, and errors for even smaller training sets have the most pessimistic bias of any estimation method. The results therefore display the worst case estimation of errors, and the true accuracy of the algorithm is likely to be greater or equal to what is reported in the following sections.

5.3 The trained algorithm: results and analysis

The structures produced by the SVM are compared in this section to those found by a standard benchmark of gradient descent for the same problem definition. The stiffness and stress of individual modules is examined first, followed by the performance of a composite structure of many units. The two optimisation methods are also compared in terms of speed.

To evaluate the results of learning, a SVM was trained on an increasing set *T* of samples (from 1 to 1000) while being tested against a separate validation set *V* of 300 samples. In the following sections, this performance is evaluated both in terms of how *similar* the solutions given by the SVM are to the ideal solutions on which it was trained, and how well those solutions actually *perform* when tested under given loads. Under both criteria learning was seen to improve steadily with an increasing training set until performance plateaued at a very high level.

5.3.1 Accuracy of the learned function

The performance, or error θ, of the algorithm trained with output y consisting of a single component y is often measured as a square loss function

$$\theta = \frac{\sum_{i=1}^{n}(y_i - f(x_i))^2}{n}$$

(5.10)

where n is the number of samples in the validation set *V* (Reich and Barai 1999). As the output vector y is 12-dimensional, this can be generalized to

$$\theta = \frac{\sum_{i=1}^{n}\left(\sum_{j=1}^{d}\left|y_{ij} - f'(x_i)\right|^k\right)^{\frac{1}{k}}}{n}$$

(5.11)

where *d* is the dimensionality of the output vector y and *k* is the exponent of the metric. The choice of *k*=2 (squared Euclidian distance) is appropriate for measurement of error in physical space, or *k*=1 (the Manhattan or city block metric) is suited to independent parameters. As the data in y is a combination of both—independent points in physical 3-space—the Euclidian metric of *k*=2 has been used.

This error θ then is simply the mean distance between the nodes in each of the ideal samples y and the nodes in the corresponding solution y'= *f'*(x) output by the SVM. Distance here is measured as the sum of squared differences in each dimension of the 12-dimensional output vectors. The graphs in Figure 5.6 display the accuracy of the predicted nodes during training with an increasing set *T* of examples and a separate validation set *V* of 300 examples from model A and model B. Both indicate a steadily decreasing error for training sets *T* up to a point (indicated by '○'), after which point there is little further perceptible change.

The number of training examples required (650 and 275) appear to be a result of the particular data sets, rather than inherent in the algorithm, and it is likely the required size of training set *T* would vary for different structural topologies. There is negligible variance in the resulting error θ when a different randomly selected set *T* is used in the SVM, or in the order of samples presented in training. While the observed plateau beginning after this point in training does not coincide with an error θ of zero, it should be noted that both the generalisation of the model f'(x) and the pessimistic bias of hold-out estimation will ensure a lower limit on the error. Training set sizes of 650 and 275 are likely simply to be the limits of learning for this problem.

The average accuracy of the functions at this point is approximately 0.025 units to the validation set, equal to the smallest possible manufacturing tolerance for a unit cube of 2mm. At this stage the function of optimal geometries as provided by gradient descent can be considered, for all practical purposes, sufficiently learned.

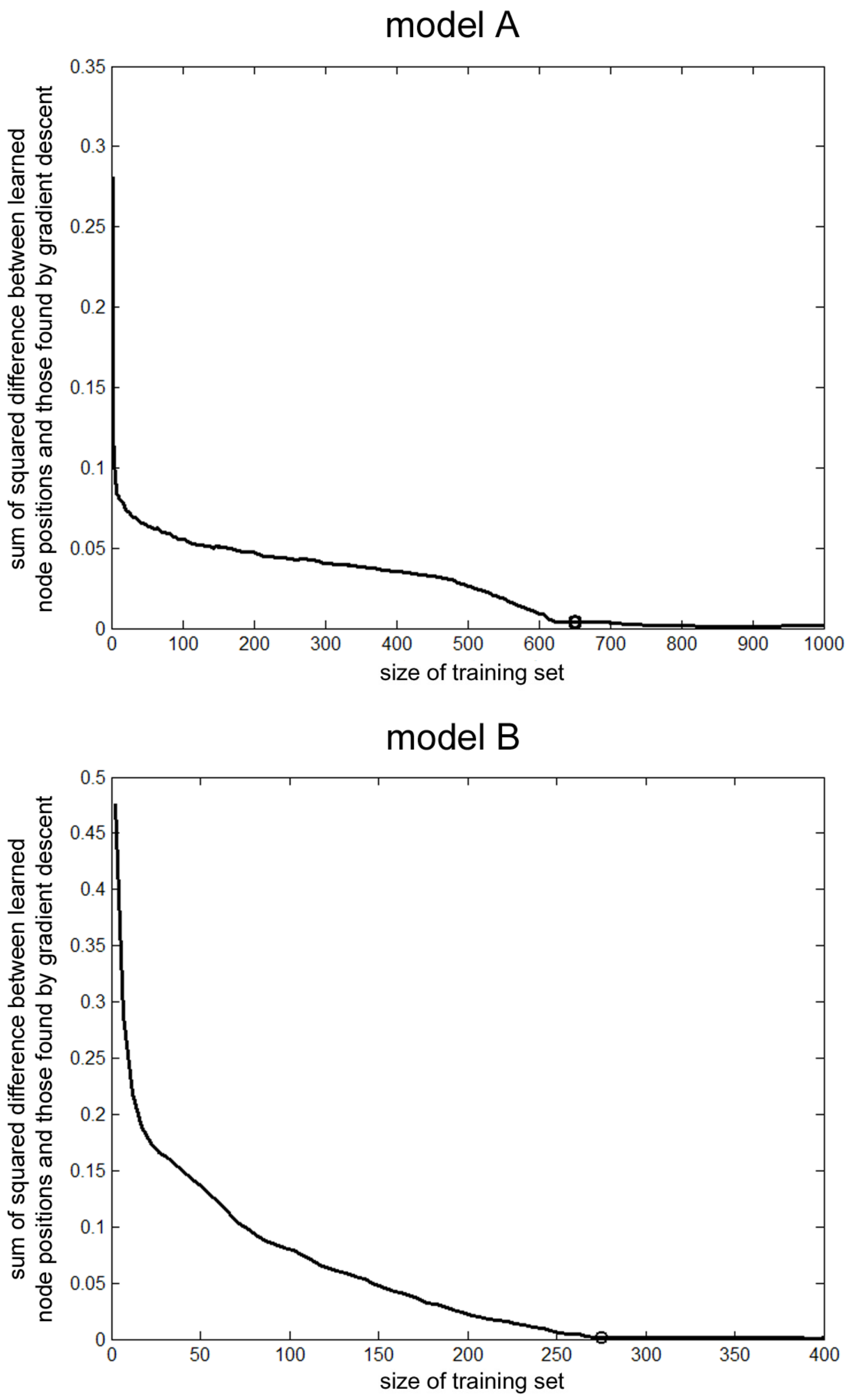


Figure 5.6. Accuracy of learning increases with increased training

5.3.2 Performance of the predicted geometries compared to GD

While the previous plots indicate the standard method of evaluating the accuracy of the function in terms of node distances, it is more relevant to our purposes to know how well the structures perform under their respective stresses. This can be determined for a given structure in the validation set by performing a finite element analysis on both the geometry y found by GD and the predicted geometry y'= *f*'(x) as found by the SVM. Both are loaded with the same input vector of stresses, and their stiffness under this load condition is measured as a total displacement of nodes when the load is applied.

This displacement between the original nodal points y and the resulting positions ŷ under simulated load is thus given by

$$disp(y,\hat{y}) = \sum_{i=1}^{m} [(y_{i(x)} - \hat{y}_{i(x)})^2 + (y_{i(y)} - \hat{y}_{i(y)})^2 + (y_{i(z)} - \hat{y}_{i(z)})^2]^{\frac{1}{2}} \tag{5.12}$$

where *m* is the number of nodal points, and the performance of the predicted structures y' is estimated as the average ratio of displacements

$$\delta = \frac{\sum_{i=1}^{n} \left(\frac{disp(y,\hat{y})}{disp(y',\hat{y}')} \right)}{n} \tag{5.13}$$

where *n* is the number of samples in the validation set *V*.

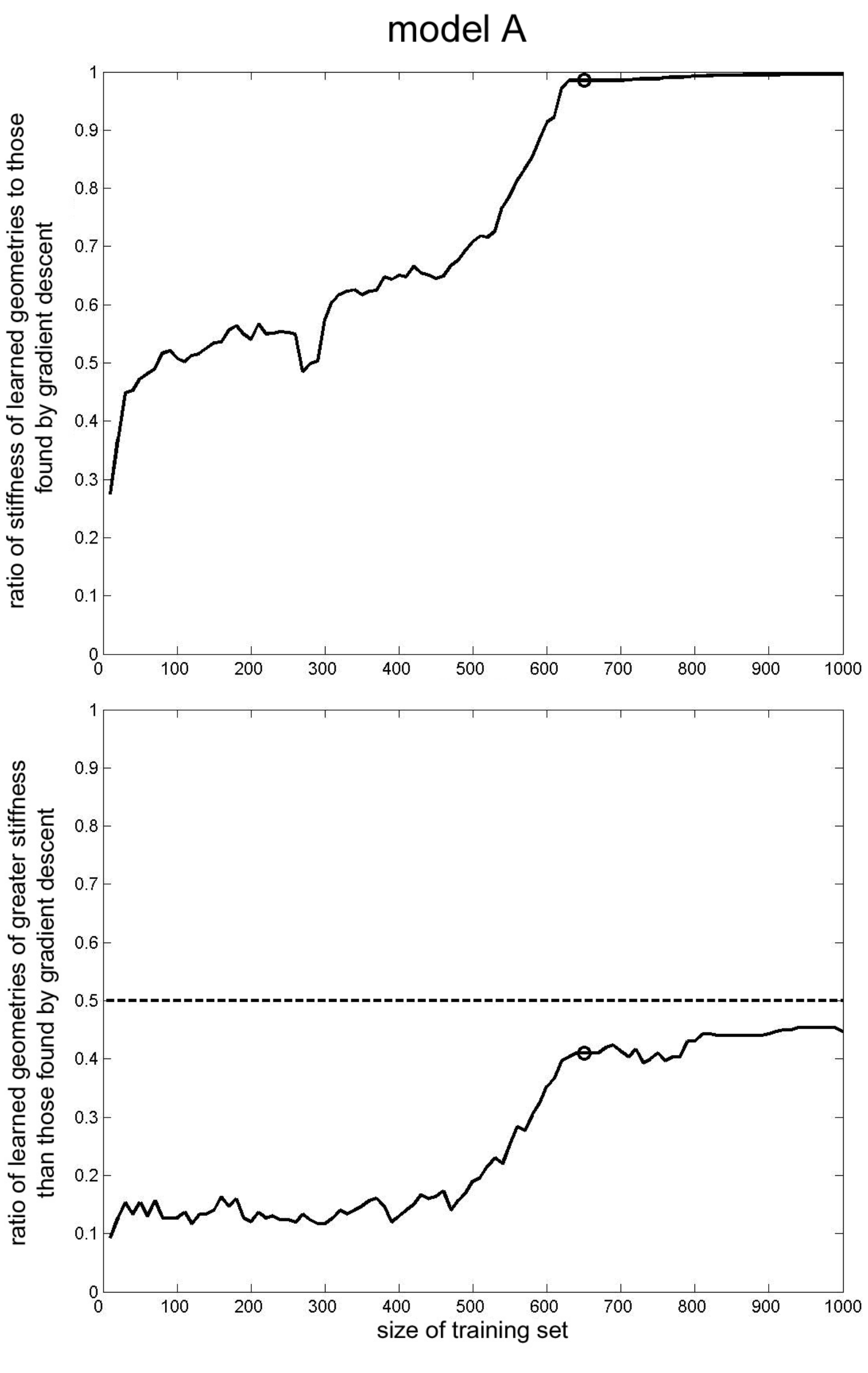


Figure 5.7. Performance δ and learned improvements over the training set.

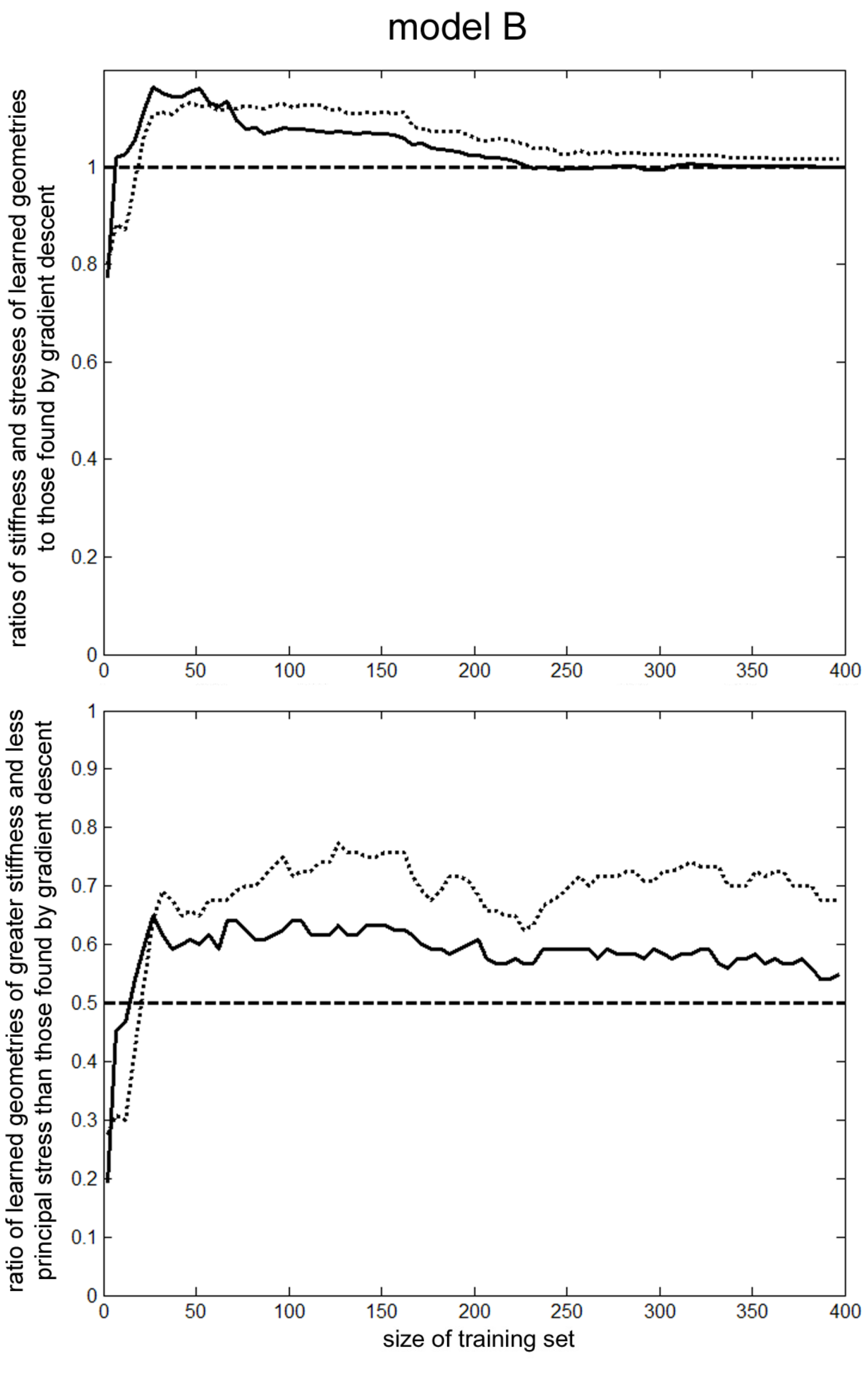


Figure 5.8. Performance δ and learned improvements over the training set. Deflection is shown as a solid line; principal stress is shown dotted.

Figures 5.7 and 5.8 (top) plot this performance δ of the predicted structures y' against the same validation set as in Figure 5.6. A ratio of 1.0 would indicate the predicted structures perform (on average) as well as those found by GD. As learning progressed, some predicted structures actually performed better than their equivalents as found by GD, and the ratio of number of structures of greater stiffness found by learning is shown in the lower plots of figures 5.7 and 5.8.

In model A, improvement with an increasing training set is evident over the same range of 1-650 samples, with a mean stiffness ratio of nearly 1.0 (equal to the GD validation set) when the function is fully learned. At this point the percentage difference between the resulting displacement of the original samples and the predicted geometries had dropped to 1.51%. The number of structures with lower deflection than their GD counterparts also increased steadily over this range. Where 50% would represent the maximum expected value of a perfectly learned function on data with noise, the SVM approaches this value at a training set size of 650 with 42% of structures having greater stiffness than the supposed ideal set.

The fact that many, or indeed any, structures can outperform the optimal structures in the data from which the SVM was trained can be explained by the method in which the data were generated. GD as a search method is itself prone to error due to local optima in the fitness landscape, and is thus not guaranteed to find the globally optimal solution, and some small variance in node positions has been seen in tests (§4.4.2). It is this variance that causes some of the optimized geometries in the training and validation sets T and V to fall slightly below the true optimal solution. It can be considered equivalent to noise in a set of examples collected from real-world measurements. In avoiding overfitting, the regression process performed by the SVM effectively 'smoothes out' the function learned so that some of these optimized structures lie either side of the function $f'(x)$. Thus, while the learned function may not be accurate enough to predict the exact node positions in the validation set, in these cases this actually is an advantage, providing an even stronger, more optimal structure.

This is even more evident in the results of learning in model B (Figure 5.8), in which the noise introduced by local optima is greater. Again the mean stiffness of learned structures is approximately equal to those found by GD after the function is fully learned at 275 samples, but for the range of training set sizes below this and beginning at approximately 15 samples the overall stiffness of the predicted structures is actually greater than those in the set on which the SVM was trained. The same is true when principal stress rather than deflection is taken as the measurement of performance, plotted in the dotted lines of Figure 5.8.

As learning progresses in a noisy data set, the closest approximation to the function generally occurs before overfitting, at which point the error θ of an independent validation set increases. Unusually, this did not happen for the validation error θ (Figure 5.6) but appears to be evident in the performance curves, which peak at about 50 training samples for stiffness and 120 samples for stress. At this point the learned function is roughly 10-15% better than GD optimisation before declining to equal that of GD. This would appear to be a result of the fact that the noise in the data is not random and uncorrelated, but a result of local attractors in the fitness landscape found by a deterministic GD search. Because the entire data set was generated by starting this search at the same point in the search space these same local attractors would recur many times in the data, rather than cancelling one another out as would random noise, and would be learned by the SVM as part of the function. This occurs at 275 samples where performance is seen to equal GD, but it is evident that the optimal performance is found by earlier generalisation with a much smaller data set. Two practical observations can be drawn from this: that models with more noise due to GD should be approximated more generally using fewer samples, and the performance of learning should be evaluated by testing the structures rather than measuring error directly on the function.

5.3.3 Assembling a modular structure

In addition to the ability of the learned function to outperform some of the structures optimized by GD, there is a secondary benefit offered by this smoothing that effects a composite structure formed of many unit cubes. The ideal situation for a complex arrayed structure (as described in section 4.5) is that stress conditions change gradually and continuously over its volume, and adjacent unit cubes under similar stresses will have similar shaped structure. With any optimisation process applied to individual unit cubes the variance in accuracy, or noise, will cause changes in node position or strut width to be more abrupt between some adjacent cubes. The repeated optimisation of many separate structures amplifies the discretisation caused by the initial sampling of the unit stresses, and these abrupt transitions result in weak points in the overall structure. By using the learned, continuous function to derive the structural geometry, the transitions between adjacent cubes are smoother, and the composite structure benefits in overall stiffness. As shown in the plots in Figures 5.7 and 5.8, some predicted structures perform better than what would be found by GD in instances where GD results in sub-optimal local optima. This section tests the overall performance of an array of these against a benchmark solution found by gradient descent.

The design problem is a simple cantilever like the examples in Chapter 4 for which modular gradient descent was used. The overall distribution of loads throughout the object was first determined using FEM on a regular grid of 8-node isoparametric volumes (e.g. cubes) to estimate the local relative displacements in each axis for a homogeneous material, and these then taken as the local loads x with which to optimise the structure of each unit cube. The positions of the nodes y were found by gradient descent, and also using the function y'= *f*'(x) estimated by the SVM using training set derived from model B. A training set of 50 samples was used, as this was seen (section 5.3.2) to result in the best performance of individual structures. These units were reassembled into the original cantilever and analysed by FEM under the original applied load.

As only the node positions are learned, the base solution for comparison has been optimised by GD for these, but member section sizes within a cantilever structure are constant. The force applied to the end nodes has been scaled so that the mean displacement of nodes in this structure is one unit. Also for comparison, another structure was assembled such that the strut section sizes in a given unit are scaled in proportion to the magnitude of the local vector x, an approximation to distribute more mass where stresses are higher that results in a stiffer overall structure. All structures have the same overall mass.

The overall assemblies found by the SVM outperformed the GD optimised versions in each case, as shown in the list of mean and maximum deflections in table 5.1. For structures of uniform strut thickness, SVM learning provides a slight improvement of 5% in mean deflection and 16% in maximum deflection, but this is far greater (approximately 80%) when the section sizes are scaled. A greater improvement can be made by changing the data set *D* on which the SVM is trained. Rather than using the generic normal distribution of input vectors x as in section 5.2.7, the final row of table 1 shows the results of training the SVM with samples taken from the same distribution as the local force vectors within the cantilever, which covers a more restricted range. The resulting learned function was thereby tuned to the particular problem and deflections were reduced over 90%.

These are far greater than the small improvements in individual structures in section 5.3.2. The greatest benefit to the overall combined structure appears to be the continuity of the learned function as compared to the noise introduced by GD. The structure benefits by a continuous functional estimation by producing a more gradual transition between adjacent unit cubes, avoiding potential weak points caused by recombining individually optimized structures.

Table 5.1. Performance of modular cantilever beams produced by GA and SVM.

	Uniform strut sections		Sections scaled to |**x**|	
	Mean def.	Max. def.	Mean def.	Max. def.
GD	1.000	2.186	0.500	1.080
SVM 50 ex.	0.949	1.833	0.111	0.215
SVM 50 ex. in cantilever distribution	0.216	0.443	0.039	0.079

5.3.4 Computational efficiency of learning compared to GD

Although gradient descent is not a computationally demanding method of optimisation in comparison with simulated annealing or genetic algorithms, the iterative sampling of gradient does require a large number of FEM analyses and the use of the learned function provides greater efficiency. The typical GD optimisation (implemented in MATLAB and using OpenFEM for analysis), required approximately 1000 such analyses and 58 seconds of computation time for each unit of structure. Training and evaluation time of the LS-SVM increases with the square of the number of training samples, but thereafter evaluation time increases only linearly with the number of evaluations. For a training set of 50, as used in the previous sections, total training time is 0.062 seconds and evaluation time is 0.0012 seconds for each structural unit, for a total time of far less than even one unit optimisation by GD. For a structure of 51 units in which 50 runs of GD would be required for training there is already some saving of computation time, but this method is intended specifically for much larger structures. In the time required to optimise the next unit for a total number of 51, over 48,000 units can be computed using the SVM.

Improved methods of optimisation may be more efficient or more accurate, with improved accuracy resulting in less noise in the function to be learned and requiring a larger training set for optimal performance. Even with a training set of 650 samples (as in model A), total training time is 16.5 seconds and evaluation time is 0.0037 seconds for each unit of structure, still negligible in comparison to optimisation by GD.

5.3.5 Observations

Several observations can be made regarding the ability of machine learning algorithms, in particular SVMs, to accurately predict the optimal geometries of structures, and thus be used as a substitute for a traditional optimisation algorithm:

- *The accuracy of the learned function approaches that of the GD optimisation*. Although the learned function is not as accurate as GD for optimisation, it does come close, with minimal error θ after several hundred training samples (section 5.3.1). At this point the function error was within the manufacturing tolerances of the structure and can thereby substitute as equivalent to gradient descent.
- *The accuracy is within tolerances dictated by the manufacturing process*. The small shortcoming in performance of solutions predicted by the SVM becomes negligible when fabrication is considered. The error of the function measured in node point positions was found to be 1/5th the finest resolution of the stereolithography machine.
- *The learned function is quicker for optimising larger structures*. While essentially equivalent in output, the SVM is thousands of times more efficient in terms of speed (section 5.3.4). Finding an optimal structure based on the learned function is far quicker than performing a full optimisation via gradient descent, as each sample of the latter requires a full finite element analysis, and one sample must be made for each dimension to calculate the gradient at each step. The production of the data set for learning is time consuming, but only in that a set of optimisations must be carried out in advance. In the example cases, from 50 to 650 fully optimized examples were required to learn the function at the outset. Many structural problems require the optimisation to be performed only once, but for those in which a similar structural optimisation is needed repeatedly, as in the case of an object composed of many modular units, the initial investment in learning the function of optimal geometries is justified. As this method of optimisation is meant to be scalable to objects of thousands of modules, the learned function represents a substantial advantage in speed.
- *The learned function results in a smoother and stiffer overall structure*. The discontinuities in an array of modules caused by separate optimisations are a disadvantage to the performance to the structure as a whole. The continuous function learned by the SVM ensures that the changes in the field of stress vectors are met with a similarly continuous change in structure. Overall performance is therefore improved by a greater degree than performance of individual modules in isolation (section 5.3.3).

The two structural models A and B differed in terms of complexity and constraints. The smooth search space of the first resulted in a function that, once learned, appeared to predict the same optimal structures as GD. More unexpected of the findings was that in generalising from the examples presented (particularly of model B), the learning algorithm was so often able to actually outperform the original optimisations on which it was trained. The characteristic peak in the performance curve indicates overfitting with increased training that is expected in noisy data, but the fact that it appeared only in structural performance measured by stiffness or strain suggests that the noise is not random, but inherent in the function that generated the data in the first place: the space searched by GD. The fact that structures outperformed those optimised by GD at this point indicates that the learned function was better able to approximate the true underlying optimal function even with noisy data.

It should not be assumed, however, that this function is truly optimal. The low numbers of training samples used to achieve it (50-120 as opposed to 650 for model A) indicate that it may still be a rather rough approximation, and could be improved by a more thoroughly optimised data set. Beginning the search at random start points, running GD for longer (it was terminated at 110 iterations), or using other optimisation methods entirely would be likely to improve it. It can be supposed from the number of training samples required for model A that such improved data sets would take longer for SVM training, but a closer approximation to the true optimal would be worth the small computational cost.

5.4 Chapter review

For this well defined structural problem in which the environment and topology are constant and the loads quantified by a continuous valued vector, it has been possible to learn the function mapping local vector to optimal structure. The results of the combined process of iterative simulation and optimisation were generalised into this function. The results compare favourably to traditional optimisation in quality of solutions, but the primary advantage in the case of the current examples is one of speed: the FEA model is complex and computationally costly to run, and therefore the learned function that replaces it is much more efficient. Rather than repeated sampling and evaluation, it is thus possible to make design decisions for this structural problem based entirely on learning from previous examples.

These results falsify hypothesis A in the context of the structural problem. A function was learned that was isomorphic to the structural optimisation, without any simulation of the structural geometry, material or underlying physical causes. This was tested by its effective use in the design of new structures.

The fact that the internal details can be treated as a ‘black box’, and simply mapped at their interface between desired behaviour and node points, appears relevant in two important ways:

- First, the approximation made in learning was seen to improve the performance of the structure by generalising a continuous function of geometry change (section 5.3.2). By approximating the overall behaviour of many different modules at a much higher level, the function was not subject to the discontinuities that result from optimisation with a full simulation.
- Second, it is only the observed results of this simulation that are used in training, not its structure. Although all experiments in this chapter were virtual, necessitating a finite element simulation of the structure’s behaviour from which the examples and finally the generalised function were drawn, if one had a decent set of observations the simulation itself is not necessary.

The second point suggests that empirically measured performance can be easily incorporated without even requiring explicit analysis or bottom up modelling. The first point suggests that the noise that would be expected in these observations is not a problem.

The relaxing of constraints between models A and B, and the simple function of member angles to stiffness stand in for real world noise and complexities. These could be extended to incorporate physical manufacturing inconsistencies, geometry changes due to resin pooling, and other details not normally included in the finite element model. These are not associated element properties at the resolution of the strut model, but may still be a function of relative point positions. They can therefore still be learned and improvements made over the GD solutions. Importantly, these can be incorporated directly into the same function whether they are derived from analytical methods or empirical measurement. One qualification should be kept in mind, as the actual behaviour of these physical structures may be different and more complex than the simulations tested here, and there may be increased noise. If so, there is no guarantee that the algorithm used to learn the optimal simulated structure will have the same success with the measured observations. Particularly in light of the differences in learning between models A and B, however, it seems reasonable to assume that fine tuning the function is simply a matter of degree, and a greater or smaller number of observations would be used in training.

But the inductive technique can be applied to systems that cannot be modelled as well, either because of an increased complexity or number of parts or because some of the variables are not known. In sec. 4.2.3 the fine scale structural changes affecting the strength of variously inclined members were incorporated into a finite element model as a function based on physical tests. It is a simple function, but the imperfections in the construction process may prove to introduce even greater complexities into the structure that have not been anticipated: excess resin may collect around the nodes causing increased stiffness, or a tendency of members to drift during construction may increase as diameter decreases. While these complexities could be explicitly accounted for in a progressively detailed finite element model, the function learned by the SVM would incorporate them explicitly, and with little or no increased computational cost. The method would allow examples to be taken directly from real, physical tests to incorporate all the complexity of the actual structure even when many variables are unknown.

The first evidence against Hypothesis B is offered in the fact that the algorithm does not need to have any kind of defined goals represented either in training or in production, but simply maps the function apparent in the data to which it is exposed, and then 'copes smoothly' with new situations in a manner noticeably different from how the explicit FEA optimization or a human engineer would. At the level of *processing rules* it is independent, but demonstrably produces the same *structure rules*. Competence is shown by the fact that the structures perform well. This evidence is somewhat limited by the constraints of the domain, however, as there is not much to communicate here, and stiffness and mass are relatively simple goals. 'Wicked problems' are very different, and will be addressed more completely in Chapters 7–9.

Chapter 6: Office planning tool

Summary: An initial office planning tool developed to the specifications of Foster + Partners is demonstrated as an optimisation application. This is found to be lacking in the context of design use, highlighting inherent problems with the assumption of clear objectives both in engineering and software design.

As a practical investigation into commercial applications in an architectural design environment, an initial project was proposed for Foster + Partners that met the immediate needs of the interior design team and as a foundation for future research. This was to be a straightforward rule-based system to generate floor plan layouts for office projects. The issue in this project was not to develop a very sophisticated algorithm, or incorporate new research, but rather to begin on a project which would be a collaboration with a design team so that their needs could be established and some insight could be gained into the commercial use of such software. It was also assumed to be a base on which further research could improve by involving more powerful optimisation techniques and machine learning.

The strategy was to attempt to explicitly codify a set of processing rules to generate the layouts in roughly the same way that the design team would go about it, these rules being derived by a combination of reflection and discussion with the design team and an informal analysis of plans they had produced. This is very much in the classical tradition of AI and in line with assumptions built in to Hypotheses A and B. The office as a system to be designed is not addressed, but it is assumed that the processes of the design team *as a complex system* can be understood and simulated (Hypothesis A). Communication from the designer to the software as to the design goals and parameters is via an explicit framework of symbolic or numeric variables (Hypothesis B). To the extent that this is possible, it would support at least the utility of working under Hypothesis A and B. If the tool cannot be made to address a designer's goal, it does not refute either hypothesis (it may simply be a bad tool), but it accomplishes its main task of indicating some of the issues to be addressed with respect to the design environment and task.

This is what can be seen in the course of development. The task chosen (of planning office interiors) consists of many subtasks of two different kinds. The harder aspects of the task, such as dealing with different work types, cultures or organisational subcultures, and interaction between people in the space are of a level complexity far above that of the more straightforward subtasks of optimising numbers of desks, net to gross ratios, fire escape distances, etc. All of these criteria interact in the design as a whole, but taken individually, the first are ill-defined, ‘wicked’ problems, while the second can be clearly stated and solved without much difficulty. Only the latter were dealt with here. This was not the intent of the project from the start, but in the successive attempts to make design processes explicit, these were the only class of problems that could be readily stated, while the others, being ill-defined, tended to be overlooked.

The tool that was developed incorporated basic optimisation and no learning algorithms. Its primary form was a graphical interface for a top-down parametric design tool. The exercise of its creation is presented here for the insights yielded into the requirements for such tools, and the discrepancies between the explicitly stated goals of designers and what they design.

6.1 Initial project outline

The design task did not at this stage require a complex algorithm. However, some optimisation was needed to choose, for example, the spacing of grid aligned units at macro and microscopic scales simultaneously. As an arrangement of modular units, a small change in desk type, grid size, alignment, etc. would be multiplied by repetition, and could thereby have a large effect on the overall layout as it interacts with planning grids, columns and walls at the large scale. The aim on the part of the interior design team was to have a tool in which to examine the implications of such small-scale changes as altering furniture or larger changes as modifying circulation or building footprint which would alter the overall capacity of an office plan, ratios of usable floor area, etc. At present the usual method for examining such options is to redraw entire sections or entire plans in sufficient detail manually, a time consuming process. The automatic layout of small scale design features to produce a finished plan immediately, and the further automation of statistical information such as population density by department or area was therefore the first requirement of the tool.

The outline of the software as determined in June 2003 allowed for the integration of the existing Bentley Microstation interface in use at Fosters with the new Generative Components[*] environment then being developed and considered for release in 2004. The software was to receive CAD and numeric input from the user, and generate a solution in the form of a finished plan. Through the Generative Components interface this could then be modified interactively while viewing updates of crucial parameters in real time.

[*] Originally *Custom Objects*. The name was changed before the first pre-beta release, but changes were made to file formats, etc. throughout the time this tool was being developed. The first outline looks very different from current versions of Generative Components.

6.1.1 TOP-DOWN DESIGN: RULES

The initial focus for the project was to implement a sort of expert system, to produce architectural drawings within the domain based on a rule set determined largely in advance in collaboration with the interior design team at Foster and Partners. No machine learning would be necessary at this stage; instead, a preset series of design guidelines was to be encoded into the software to build up finished drawings from graphical and numerical input. This rule set could function at any scale, and the order of rules might be flexible, but would be a top-down process in terms of method in that the rules would be pre-determined based on prior analysis of the problem domain. This analysis was begun early in the work, and yielded as a first attempt the following set of general guidelines for the arrangement of both desks and cellular offices within a given floor plan.

1. Know the planning grid - often 1.5m grid - desking generally runs parallel to this grid.
2. Desking preferably at 90 degrees to glazing on main elevation.
3. Desking should be 90 degrees to primary circulation (so no backs to the corridors)
4. Leave 1.5m corridor for primary circluation connecting lifts toilets and escapes.
5. Leave 1.8m minimum clear back to back between bench desks, or 0.8m between L shaped desks.
6. Leave 1.3m minimum between back of desk and walls
7. Desks are normally coupled.
8. Avoid rows of more than 4 desks long
9. Average of 1 storage cabinet per person - positioned along edge of desks or between clusters.
10. Leave 150mm minimum (500mm preferably) between desking and glazing.
11. Centre desking between columns.
12. Cellular offices - generally 2 x 3 planning grids i.e 3m wide x 4.5m deep on a 1.5m grid
13. Cellular space generally perimeter facing.
14. Partitions to line up on mullions.
15. Avoid columns within cellular offices
16. Pair doors to offices where possible

This set of rules constituted an explicit codification of the expectations and working method of the design team.

6.1.2 Schematic organisation

The required input would take three forms:

- NUMERIC (visual C dialogue box): selection of parameters i.e.
    - # desks/furniture each type
    - circulation width

- GENERATIVE COMPONENT:
    - desk/furniture description in .cs file contains internal constants: e.g.
        - desk width
        - clearance

- GRAPHICAL: plan (will require mdl utility to 'read')
    - core
    - planning grid
    - column positions

The optimisation stage was to be entirely self contained in a C or C# program consisting of the following:

- Layout algorithm
    - blocking/stacking
    - space syntax heuristics? (possibly)

And the output to the Generative Components environment would be:

- XML script including
    - core
    - primary circulation
    - planning blocks; including
        - desks, or
        - other furniture

Upon discussion with the interiors department of Foster and Partners, it was found that the generic problem of a typical floor plate could solved in a top down hierarchical manner by dividing the plan into a series of distinct blocks of differing function. These surround the core areas and are connected by a primary circulation zone of predetermined dimensions. The coding requirements for these blocks would include

- a pointer to the shared planning grid
- a pointer to the columns and boundaries
- the desk/furniture spacing ratio (e.g. 5:12, etc.)
- the desk/furniture type

The overall object hierarchy as produced by the software would then be:

1. overall plan layout (as XML script)

2. programme zones (predetermined Generative Components .cs files)
    - core
    - primary circulation
    - planning blocks

3. individual units (Generative Components from flexible library)
    - desks
    - misc. furniture

6.2 Planning tool implementation

6.2.1 DEVELOPMENT OF REQUIREMENTS

The rule set expressed in §6.1.1 is at first a reduction of the very rich problem of space planning to the set of relatively clearly defined easy subtasks. This may reflect the actual structure rules (and possibly even processing rules) of a designer at Foster + Partners to some extent, but falls short of a full description of space planning. It is likely that these rules more accurately reflect the process carried out the more isolated the designer is from the rest of the project, less so the more they are involved. Notably, all task rules refer exclusively to desk problems as unconnected to other needs.

These rules were abstracted further in subsequent analysis and discussion as to their application in the tool itself. In the course of development it became apparent that much of the knowledge required for the actual drawing composition tended toward geometric and pattern-forming skills (e.g. the regular placement of desks against a column grid) rather than the domain specific knowledge unique to those with extensive experience of office arrangement. It didn't therefore require an *expert* system with a database of explicit decision rules, but an effective algorithm for spatial representation coupled with an interface allowing for flexibility and fine-tuning by the designer.

An analysis of the rule list above reveals that rather than encompassing a complex system of specialized knowledge (as would be modelled by the expert system approach) the guidelines can be divided into three categories, each of which could be handled by a simpler and separate part of the software.

Numeric input: Many of the rules can be considered simply numerical parameters for distances or quantities, etc., to be specified by the designer. Those considered to be rules are of quite constant values, although subject to occasional modification, whereas other parameters not mentioned would change frequently. It was considered that a common method for setting these would be best. These cover the following rules:

4. Leave 1.5m corridor for primary circulation connecting lifts toilets and escapes.
5. Leave 1.8m minimum clear back to back between bench desks, or 0.8m between L shaped desks.
6. Leave 1.3m minimum between back of desk and walls
8. Avoid rows of more than 4 desks long
9. Average of 1 storage cabinet per person - positioned along edge of desks or between clusters.
10. Leave 150mm minimum (500mm preferably) between desking and glazing.

Graphical input: This contains the CAD data from which the design is generated. This is completely variable and must be determined by the designer. Some elements (e.g. planning grids, building perimeters) are unique to the project whereas others (e.g. standardized furniture) may be shared between many projects. In general these rules change less frequently than those covered under 'numerical input'. They include:

1. Know the planning grid - often 1.5m grid - desking generally runs parallel to this grid.

5. Leave 1.8m minimum clear back to back between bench desks, or 0.8m between L shaped desks.[*]

6. Leave 1.3m minimum between back of desk and walls [*]

12. Cellular offices - generally 2 x 3 planning grids i.e 3m wide x 4.5m deep on a 1.5m grid

Algorithm: With the above input set, the remaining rules can be incorporated into an algorithm for manipulating geometry which does not need to change. The rules below are either always used, or can be set by a parameter within this algorithm.

2. Desking preferably at 90 degrees to glazing on main elevation.

3. Desking should be 90 degrees to primary circulation (so no backs to the corridors)

7. Desks are normally coupled.

11. Centre desking between columns.

13. Cellular space generally perimeter facing.

14. Partitions to line up on mullions.

15. Avoid columns within cellular offices

16. Pair doors to offices where possible (see also #7)

6.2.2 Project Components

- Graphical input:
    - Microstation dgn. Files. Levels specific. Building layout
    - Generative Components desks/offices
- Floor plate analysis / top down areas: MDL/C in Microstation
- Layout generation / bottom up stacking: Generative Components and C#

6.2.3 Graphical input: modular units

It was decided that the creation of the individual furniture or office elements which form the building blocks of the layout would be more easily accomplished graphically than numerically or functionally. Generative Components allows the creation of an image with points and lines

[*] Rules 5 and 6 can be reduced to a 0.8m clearance as a numeric input and a 0.5m clearance in the graphical specification of the desk, hence they are included in both sections

from which parameters representing their properties in space can be read as points within their local coordinate system. Of particular relevance, for example, is the 0.5m clearance around the front of a desk mentioned in rules 5 and 6 above. This is easily represented graphically by placing a point on the desk drawing file at the imaginary boundary representing the clearance zone of the desk. In loading the file, the desk layout portion of the algorithm reads the coordinates of this point, and used them as the parameters dictating the area required to place that particular desk. Moving the point in another desk file simply tells the algorithm the new required clearance, and the rule is the same for rectangular, L-shaped, or curvilinear shaped desks.

The ease of use provided by the graphical solution does however create a few problems in terms of speed. The Generative Components created graphically are computationally more complicated than what might be coded manually, and when many such units are used together the update times tend to make interactivity somewhat less than ‘real-time’. Two solutions were proposed for future development: that of replacing the Generative Components furniture elements with standard Microstation cells, or of temporarily disabling the live redrawing of the plan while the user is making changes.

6.2.4 ALGORITHM: GRID RELATIONSHIPS

Two strategies were under consideration by the Interiors team at Fosters with respect to planning grids: the standard orthogonal grid, usually consisting of 1.5m units in both dimensions, and an ‘organic’ layout which appears to be undetermined by a grid, but in reality simply replaces the rectilinear grid with one consisting of hexagons. A hexagonal unit has six neighbours rather than four, which has the effect of allowing a freedom of several directions for each unit to take when constructing a string of modules unit by unit, rather than constraining modules to a straight line. This ‘organic’ grid is not used as a planning grid for any buildings at Fosters however, so it was decided that the project should represent some version of continuous grid lines from which orthogonal, and later radial grids could be derived.

The ‘intelligence’ required at this stage of the layout process consists of finding the best way to rationalise a pattern of regular modular units (desks, filing cabinets, offices, and required clearance and circulation space) with two larger grids of differing scales—the planning grid for window mullions, finishings, etc, and the structural grid of columns. This is quite simple if the units are multiples of one another but this is rarely the case. In the examination of approximately 100 case studies of layouts produced by the team it was found that a variety of different patterns were used to most efficiently create a rhythm of units within the grids. Rows of desks might be placed with every fifth row to coincide with every second column, for example, with a gap left in the rows at these positions. It was noticed that each of the patterns could be expressed by a ratio of planning grid units to desks (or offices, etc.) with a preference for ratios consisting of lower whole numbers (3:1, 2:1, etc.) More unusual ratios (as high as 12:5) were used less often and only when these provided a unique advantage in negotiating the three rhythms.

The process of selecting the ratio to use is deterministic and straightforward for the algorithm. The preference is for the lowest ratios and the total number of possible ratios is quite small, so the entire set of possibilities can be evaluated even within a real-time update. Limits are placed by the designer on the highest whole numbers to use for the planning and column grid units. These are determined by such motives as ensuring a cellular office is column-free. First the default ratio of one grid unit is set. The algorithm then calculates the number of rows of a given unit can be fitted to that ratio for successively higher ratios until the limit is reached. A higher ratio is chosen only if the gross number of units exceeds that of a lower ratio.

For each update, the following steps occur in real time:

- Review input parameters (grid, settings, boundary)
- For each allowed ratio against grids:
    - Generate layout
    - Evaluate efficiency
- Output layout graphics
- Calculate and display numerical efficiency

6.2.5 Demonstration of the tool

An implementation of the software was delivered in summer 2004. The graphical input stage of the algorithm consists of two processes, triggered by buttons within Microstation. The first, to read the initial input plan, scans the selected part of the drawing and divides the usable floor plate into the large scale blocks to be assigned to programme zones. These are indicated as closed shape elements in the live drawing file. At this point modifications to the blocks may be made using the standard Microstation interface. The second process generates an XML script file based on the block shapes and given parameter settings (described below) which is viewed with the Generative Components interface.

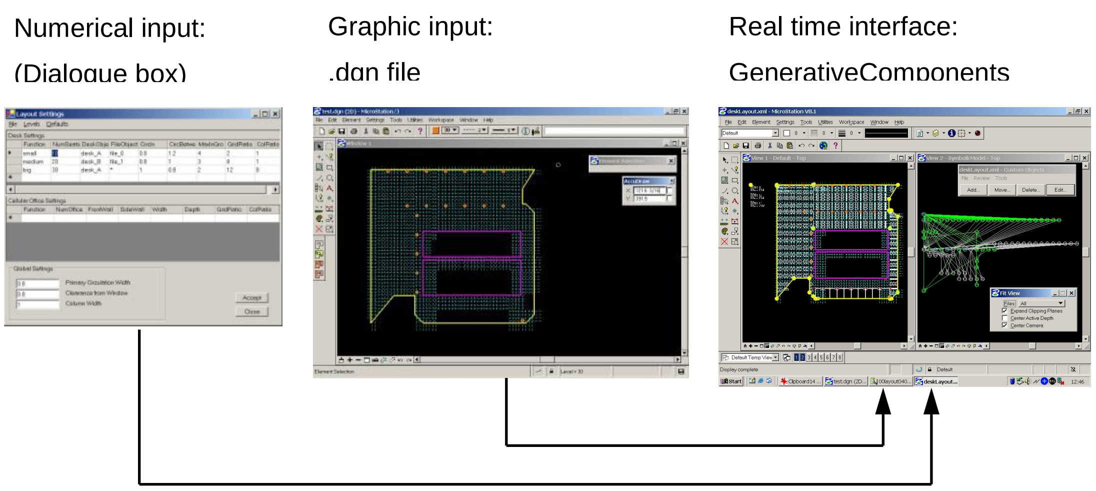
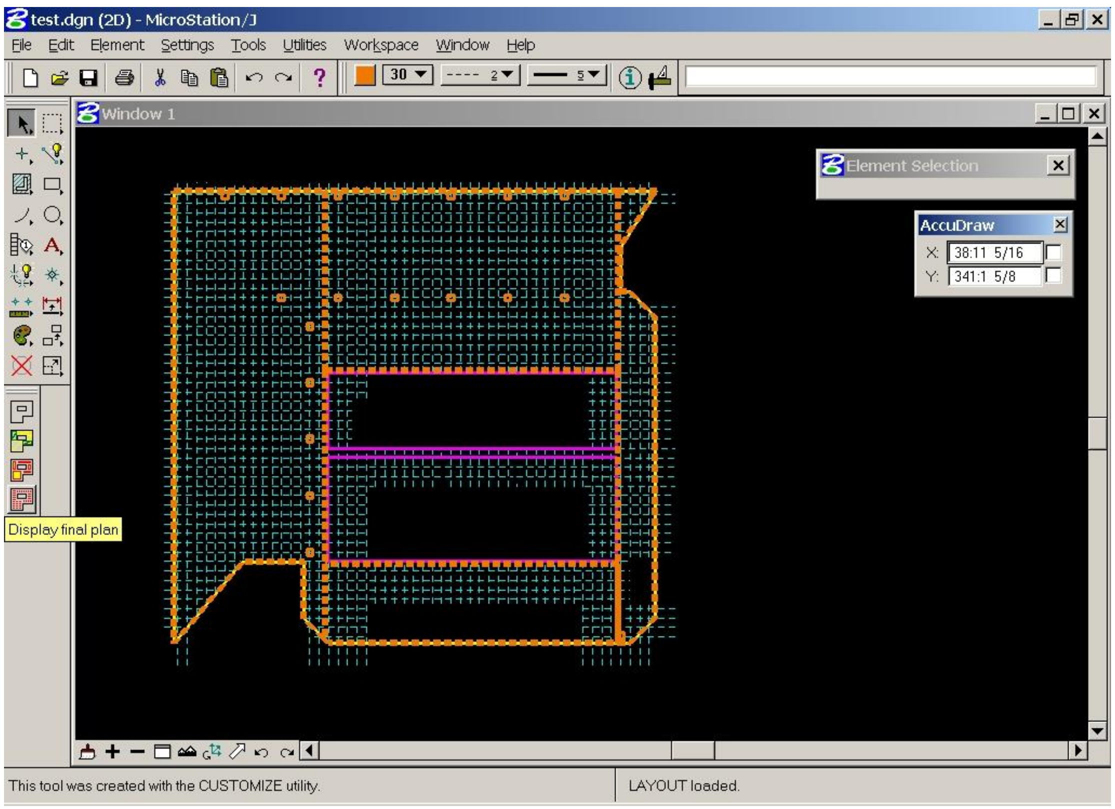


Figure 6.1. Three separate interface components of the tool.


test.dgn (2D) - MicroStation/J
File Edit Element Settings Tools Utilities Workspace Window Help
Window 1
Element Selection
AccuDraw
X: 38:11 5/16
Y: 341:1 5/8
Display final plan

This tool was created with the CUSTOMIZE utility.

LAYOUT loaded.

Figure 6.2. CAD input and gross zoning phase in Microstation.

The first stage of the algorithm produces the gross division of a plan (from a Microstation .dgn file) into a set of regions that best reflect the preferences set in the dialogue box. Primary circulation is drawn for access to all required points on the building core, and each planning zone is oriented to an exterior boundary. The interface at this step is standard Microstation as displayed in Figure 6.2.

The Generative Components interface is displayed in figure 6.3. View one (left) is the standard plan view of a group of desks within a given area of an office. These are placed with appropriate filing units as specified and are set out according to the grid and column positions of the input plan (hidden for clarity). The boundaries of the area are indicated by the yellow corner points, which can be dragged by mouse to make immediate changes to the zone layout. In addition, parameters such as circulation, spacing and group size can be edited via the Generative Components interface in either of the two views. View two shows the hierarchy of the elements in the design in graph form. Boundary points, grids and variables can be edited here via this alternate interface.

The full interface can be seen in the next image. The analysis of the entire building results in an interior layout of the blocks provided by the first step of the algorithm. Open plan desks are placed over the bulk of the plan, with a series of cellular offices along the lower edge of the plan. A live text representation of the relevant statistics is shown on the plan view, which updates automatically as corner points are dragged in the interface. The symbolic graph of the entire layout (right) is also more complex.

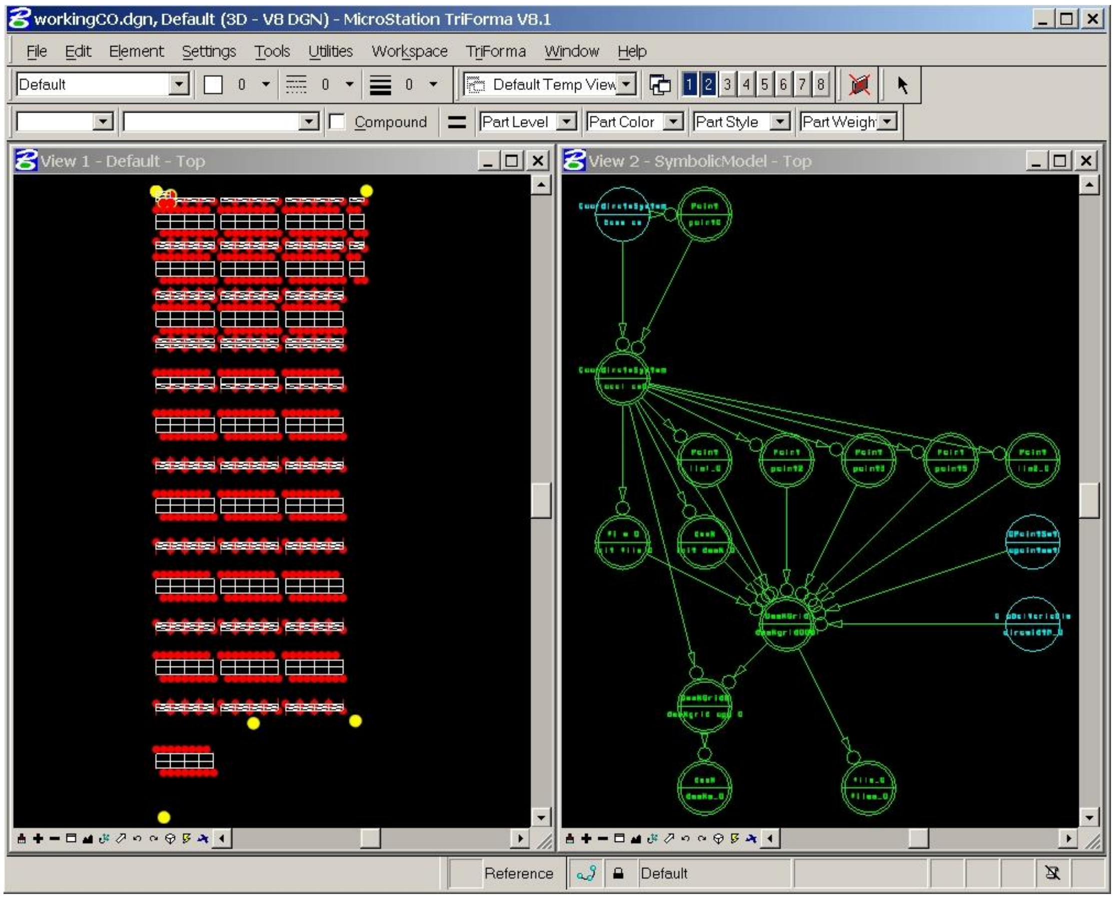

workingCO.dgn, Default (3D - V8 DGN) - MicroStation TriForma V8.1
File Edit Element Settings Tools Utilities Workspace TriForma Window Help
Default
Default Temp View
Compound
Part Level
Part Color
Part Style
Part Weigh
View 1 - Default - Top
View 2 - SymbolicModel - Top
Reference
Default

Figure 6.3. Real-time desk optimisation in Generative Components.

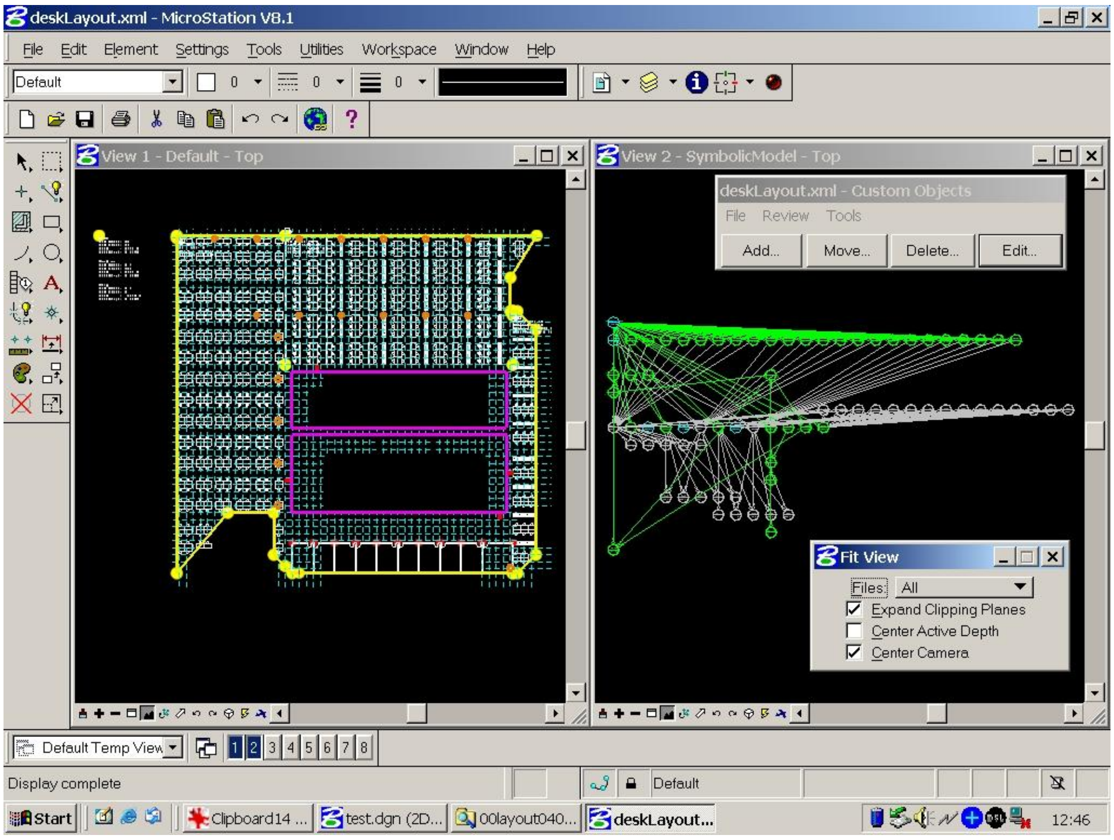
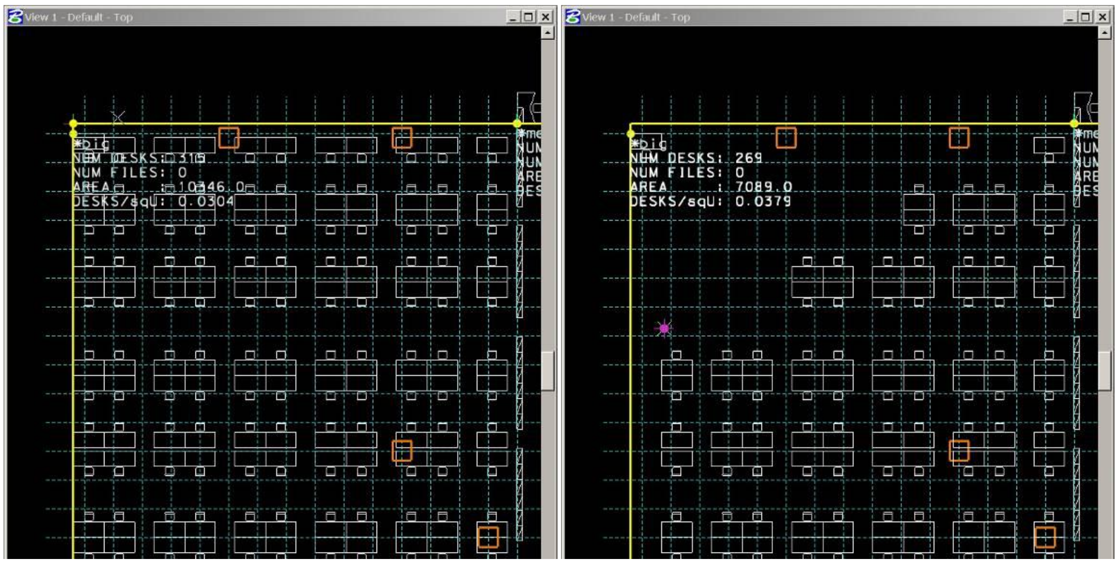

deskLayout.xml - MicroStation V8.1
File Edit Element Settings Tools Utilities Workspace Window Help
View 1 - Default - Top
View 2 - SymbolicModel - Top
deskLayout.xml - Custom Objects
File Review Tools
Add... Move... Delete... Edit...
Fit View
Files: All
Expand Clipping Planes
Center Active Depth
Center Camera
Default Temp View
Display complete
Default
Start
deskLayout...
12:46

Figure 6.4. The full plan in the Generative Components interface.



```
*big
NUM DESKS: 315
NUM FILES: 0
AREA      : 10346.0
DESKS/sqU: 0.0304
```

```
*big
NUM DESKS: 269
NUM FILES: 0
AREA      : 7089.0
DESKS/sqU: 0.0379
```

Figure 6.5. Statistics derived from the plan are updated in real time along with the desk layout.

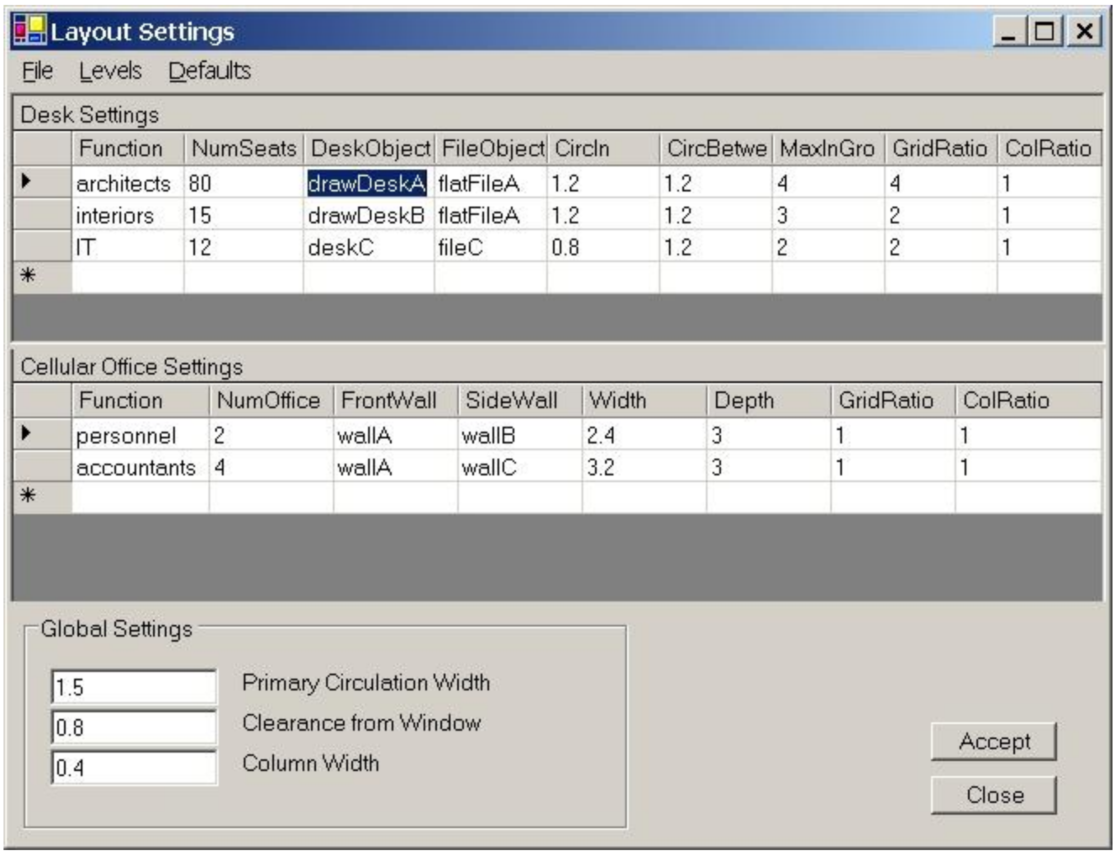
The Generative Components interface allows the resulting model to be updated interactively, for large or small manual adjustments to be made to the geometry in design exploration. The example in figure 6.5 recalculates the overall layout of desks within a zone as a boundary point is moved, but similar changes can easily be made to the various clearances around those desks or relationships of furniture to planning or structural grids.

6.2.6 Increased user interface

The need for user flexibility and an increased interface became clear during the development of the project. This was particularly evident in the case of graphical input, in which the user was given more flexibility by splitting the initial processing stage into two. Instead of generating the XML output file directly from the plan input, an intermediate stage involving editable shape elements gives the user more control.

The numeric input also required an interface which would be clearly legible, easy to use and fast. The design team initially requested a step-by-step ‘wizard’ style application, but this was found to be too time consuming for making quick changes or repeated runs of the program. A dialogue box was adopted instead (figure 6.6), allowing all relevant data to be visible simultaneously, and changes to be made rapidly in any order. In addition, the settings can be saved for later use and proposal generation on similar plans.

The bulk of the interface is based on a Windows Explorer style of list box, in which each programmatic zone is listed alone with the relevant parameters: number of seats, circulation dimensions, etc. Both open plan desk zones and cellular offices can be set in their respective windows, and edited via mouse click. Global variables such as primary circulation are set in the box at lower left, and a separate window (figure 6.7) can be opened to set the default Microstation levels on which relevant drawing information can be found.



Desk Settings

	Function	NumSeats	DeskObject	FileObject	CircIn	CircBetwe	MaxInGro	GridRatio	ColRatio
▸	architects	80	drawDeskA	flatFileA	1.2	1.2	4	4	1
	interiors	15	drawDeskB	flatFileA	1.2	1.2	3	2	1
	IT	12	deskC	fileC	0.8	1.2	2	2	1
*									

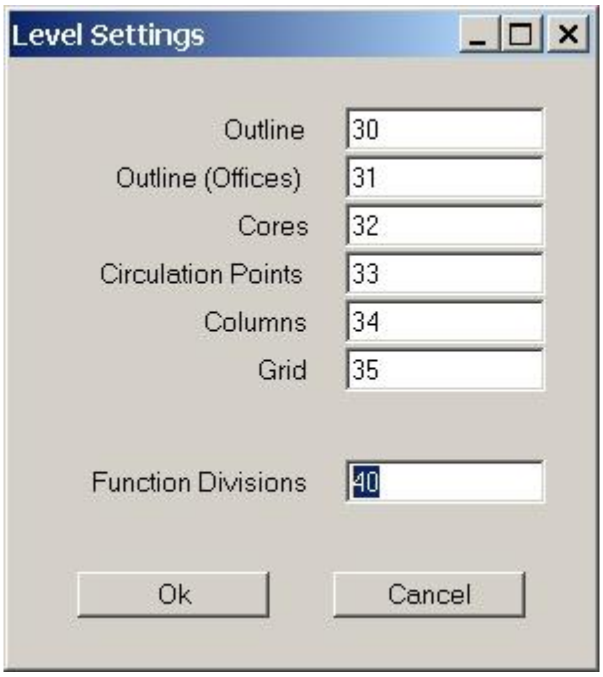
Cellular Office Settings

	Function	NumOffice	FrontWall	SideWall	Width	Depth	GridRatio	ColRatio
▸	personnel	2	wallA	wallB	2.4	3	1	1
	accountants	4	wallA	wallC	3.2	3	1	1
*								

Global Settings

1.5	Primary Circulation Width
0.8	Clearance from Window
0.4	Column Width

Figure 6.6. Main dialogue box for numerical interface.



Outline	30
Outline (Offices)	31
Cores	32
Circulation Points	33
Columns	34
Grid	35
Function Divisions	40

Figure 6.7. Default drawing level settings are set in a second dialogue box.

6.3 Observations drawn from the initial project

This project was primarily intended as an investigation into the needs of the design team, without a great deal of new research into the optimisation methods to be used or the quality of solutions produced. Several points should be noted regarding the issues arising in its development, as these affected the direction of subsequent research.

6.3.1 Increased interactivity

The first and most obvious change above the initial brief is the desire for increased interactivity. More control and flexibility in the design loop was requested by the design team, and was deemed desirable even when balanced by a decrease in speed.

The design of user interface became much more focused on openness, and ability to make changes at many stages of the process. While the initial proposal included a single algorithm to read the input drawing and generate the output the developed project used a two-stage process: shape blocks were drawn first, and then read. This allows for the modification of the blocks as suggested at the intermediate stage, the manual drawing of the blocks in specific cases, or the correction of mistakes unforeseen by the relatively straightforward algorithm.

The overall structure of the tool is thus a modular process. First, the gross work areas and specific unit types are set, and then a more flexible rudimentary optimisation procedure chooses the best finished arrangement. The setting of parameters in the dialogue box does not indicate that this is a constraint-based system. Indeed, these are the goals of the update step, but they are soft constraints in that they do not have to be satisfied. Numbers of desks required are only ratios between sections and treated as general guidelines toward which to optimise, and are rarely met exactly. The only fixed, internal constraints are the gross zones set either by the first part of the algorithm or specified by hand by the user. Any of the elements—zone marking, optimisation, desk placement, real time boundary update—could be changed or removed under the designer's control. While this entailed more manual effort to run the algorithm through from start to finish, the increased openness of the process was considered a distinct advantage for the user.

This modularity is not the same as Simon's (1969) 'near decomposability', but in fact quite the reverse. Rather than the modules having internal complexity with well defined and relatively simple interfaces, the modules were internally well defined (by the algorithms implemented in the tool) and broken apart at their most complex points—where the interface is graphically and CAD based. The advantage to the designer is that they can interfere with the normal course of the algorithm at this point, making decisions and changes that are not easily coded.

6.3.2 Graphic communication

On the broader issue of computational representation of architecture, it became evident on viewing the rules in section 6.1 that their specification should be primarily graphic, not verbal. The first rule list as compiled by the design team was in an inappropriate form for actual use on a project, but became quite straightforward when translated into an algorithm to manipulate geometry.

6.3.3 Reduction of the task

The rules in section 6.1 are also generally local, quite simple and reductionist. None appear to have any effect beyond local structure, other than the simple alignment to overall planning grids. Certainly none have anything to do with the more difficult and subtle aspects of space planning such as higher level social organisation.

These rules were given after some deliberation and discussion by an interiors design group that deals with such office designs daily. This highlights two potential problems. It is possible, first of all, that these rules actually do represent the total design strategy of the Foster + Partners design group as they currently practise, in which case the kind of tool needed would be one to support the integration of more complex aspects of the task. A tool might explicitly analyse more complex spatial relationships so that they may be considered in the design, for example. A second, and more likely, problem is that the bulk of the designers' work actually does occur at a level that is more subtle and complex than these rules indicate, and that this is difficult to state explicitly. This very important subset of the designer's skill was simply overlooked when forming the explicit list of rules, precisely because it consists of the kinds of decisions that are largely unconscious, and only made when the designer is actively engaged in the process of designing. If much of a designer's strategy cannot be explained verbally even within a consultation group of other designers, it is clear that the highly formal requirements necessary to set up an optimisation pose a similar problem.

6.3.4 MOVING FORWARD

The tool functioned well in that it satisfied the agreed brief and rule set. In this sense it was a successful example of engineering to well-defined objectives. Unfortunately, these objectives appear to have been too tightly constrained[*]. Much of the space planning task is not explicitly discussed and is likely a 'wicked problem', and no clear way of addressing these aspects became evident in the development of the tool. This gives no evidence to either corroborate or refute Hypotheses A and B, but does suggest something about their usefulness in dealing with these aspects of design. The fact that designers themselves were apt to ignore many aspects of the design problem, and likely their own processes, when stating its rules suggests that they do not explicitly think in terms of the underlying structure of the system they are designing, at least to the point where it need be modelled (Hypothesis A). The request for more points of graphic interface, to make interim changes that can't be coded a priori suggests that there are many aspects of design that should not be communicated explicitly (Hypothesis B).

The observations suggest the interface between this sub-problem of desk packing and other, more difficult and ill-defined, aspects of space planning is likely to be far more complex than was acknowledged initially. Space planning as a whole is not 'nearly decomposable'.

This exercise suggests at least two ways of dealing with this. The first is simply increasing the bandwidth, so the algorithm is dealing with a richer description of the requirements of the project. The second is to accept that information must be organised and reduced at some point—the isolated desk packing problem may sometimes be a valid subtask within a larger project—but that this organisation changes frequently. The designer's need for more flexibility in the interface is evidence of this. Rather than predetermining the way in which the data is organised and reduced at the interface (Hypothesis B), a strategy might exist by which the machine itself can do this. Again, the interface is more complex; reduction occurs within the machine. The machine might then switch representations and reinterpret this data as needed[†]. The following chapters explore this possibility. After the completion of this tool, the explicit approach was seen as inadequate to address the problem at hand, and research focused on methods by which design objectives may be communicated by example.

[*] The software delivered has not frequently, if ever, actually been used by Foster + Partners in design practice.

[†] Newell and Simon (1976) also mention the potential advantage of 'moving from one representation to another'.

Chapter 7: Flexible representation of plans by graph spectra

Summary: This chapter proposes a means by which plans may be represented by numerical data to be accessed by the machine. The representation is of sufficiently high dimensionality to capture enough feature information that the algorithm will be able to extract its own higher level approximations as required. This basic representation will form the raw data for the use of machine learning in subsequent chapters, and thus the basis for the evidence against Hypothesis A.

The structural problem (Chapters 3 and 4) involved complexity in modelling the behaviour of the space frame; the space planning problem differs in that many of the design goals themselves are complex and unspecified. Observations of the initial planning tool (Chapter 6) showed that generating an efficient packing of desks for a given plan is straightforward, but more difficult design objectives surrounding the complex social factors or space use patterns of a given organisation are not. This supports the view that these goals are not readily specifiable, but evolve in practice out of a process of 'reflection-in-action' (Schön 1983), 'hermeneutical' (Snodgrass and Coyne 1997) or 'critical' discussion (Popper 1979), or are drawn from a designer's extensive tacit experience.

If an explicit understanding of the underlying socio-spatial causes of an office's behaviour does not exist, how can a designer design for it? Hypothesis A would propose this is impossible. The following three chapters address this by showing that the computer can come to a similar kind of understanding, forming its own goals based on precedents that are known or judged to work well or poorly. Rather than providing a set of explicit goals and quantifiable measures, the user need only provide a pre-existing plan or set of pre-existing plans of, for instance, a previous office known to have worked well for the same organisation in the past. In contrast to the rules employed in the previous chapter, the question changes from "how can we explain the behaviour of the system?" to "how can we reproduce it?"[*] As with the black box of cybernetics, we do not necessarily have to understand the details of all the inner workings to do so.

To falsify Hypothesis A it must be shown that the complex underlying structure of existing designs—properties such as the socio-spatial involved in space planning—does not have to be

[*] This is identical to the replication of the genotype of a morphic language (Hillier and Hanson 1984) in that it is 'inverted' and transmitted through actual built form. The difference is that individual designers do not have to share any prior knowledge (§3.4.1, case iv).

fully understood (particularly by the machine) for the algorithm to reproduce it from precedents. Algorithms will be used to demonstrate this by training on existing designs and generating new ones that have similar properties. Testing appears to pose a methodological problem however—*intersubjective verification* requires a basic statement (Popper 1959) that is unambiguous, and if the complex underlying structure cannot be explicitly described it is impossible to determine whether it appears in examples produced by the algorithm*. Instead of the assumed socio-spatial structures, therefore, more obvious and clearly described geometrical structures† are used for testing, as they are immediately apparent to anyone evaluating the output.

There are two possible objections to implementing and relying on this method, directly related to the problem of induction. These were discussed more generally in Chapter 1, but can be stated more specifically with respect to the current method, along with how they will be dealt with:

- *Just because design precedents are known to have worked well in the past, it doesn't mean they will still work for new design tasks.* This is the classic problem of induction of Hume, Popper, et al., solved in practice by recourse to the best tested theory (§1.2.2). For this not to be the case presumes an improbable event distinguishing new design tasks from old, all else being equal. There are different ways of drawing precedents however, and how designers do this might be highly relevant. Unselfconscious designers, as Alexander (1964) notes, easily take advantage of the complex understanding embodied in the designs they use as precedents. What about self-conscious designers? His more codified 'pattern language' (Alexander et al. 1977), derived from such observations, attempts to catalogue an established precedent solution to particular combinations of sub-problems, but while these may be complex and elegant, the method assumes that the sub-problems are relatively fixed and can be mapped easily to the code. This change in method from unselfconscious to selfconscious is an event that distinguishes current from past cases.

  The reflective methods above (Schön 1983; Snodgrass and Coyne 1997; Popper 1979; et al.) avoid this symbolic reduction, and derive their design moves and goals from a potentially infinite domain of designs known or judged to work well or poorly (either previously realised and tested in use or tentatively proposed and discussed), and can

* Space syntax measures might alternatively be employed to measure an assumed underlying structure, but the test would be less obvious as dependent on assumptions built into the measures.
† More subtle socio-spatial properties will be used (Chapter 8) when examining existing buildings in which properties surrounding their design—function, date, etc.—are known objectively, but these are not testable in newly generated examples.

reinterpret them at will. Care will be taken to allow intersubjective verification of the tests.

- *Just because it learns clearly defined geometric patterns, doesn't mean it learns far more subtle socio-spatial patterns.* This objection is more easily supported, given the domain change. There are two parts to it. Regarding the change from geometric to social or spatial patterns, there is the danger that the computational representation and learning algorithm are good at finding patterns in the geometric domain, but conclusions can't be extended to the very different socio-spatial. This will be controlled by using a basic representation* that is socio-spatial to learn and test for geometrical patterns. These patterns are thus only easily verifiable for a human judge, but less straightforward for the machine—they must be learned to be reproduced.

  The clarity of geometry as opposed to the subtlety of the socio-spatial is a matter of degree. The change in representation ensures that much clear geometry is not immediately clear in the graph representation used. This will only become an issue in Chapters 8 and 9, when machine learning will be used.

After the initial planning tool it was found that the interface was too constrained, in part because of the constraints on how data was represented. This chapter focuses on another strategy for representing plans as precedents, not in the sense of symbolic representation but as an analogue metric to allow the measurement of differences between plans. The next chapter will use machine supervised learning algorithms to allow the user to set the implicit design objectives by deriving features common to labelled plans. Chapter 9 will then combine both with an optimisation algorithm to produce new plans to the set objectives.

7.1 Goals for a generic representation of office spaces

Beginning with the representation might well appear to be falling into the trap of classical AI, in which finding the right representation (Newell and Simon 1976) or ontology (Hayes 1979; Lenat and Guha 1990) was thought to be the biggest part of the problem. Worse, it might appear to make a refutation of Hypothesis B impossible. Neither of these is the case, as the representation is not 'common' to both user and computer, and will not be used for communication.

* This is not in the sense of a classical representation, as explained in the next section.

What is required is the equivalent for the computer of basic sense data, precisely because existing representations of plans are inappropriately symbolic. Graphic elements in a CAD file are stored digitally as a one-dimensional list, but are meant to represent a drawing as we might perceive a real drawing— a collection of objects with invariant spatial relationships to one another (no matter how we scale or rotate the drawing). Predetermining an alternative representation is therefore justifiable—the basic senses are innate even in nature,[*] and the underlying, raw form of this data is always given. The strategy here will be to make this representation as generic as possible, so that higher order 'representations within the representation' may be extracted from it, autonomously, by the machine. As would be required to refute Hypothesis B (§3.4.1: case iv), these higher order representations will be non-classical, distributed representations (§3.4.2) that apply only within the context of action in design (§3.4.3).

This representation of raw sense data can be approached in one of two ways:

- An attempt could be made to simulate our own senses, particularly vision. This is problematic in that vision is already complex and far from understood, and many assumptions must be made as to the underlying processing rules. If these assumptions are wrong, processing rules become predetermined arbitrarily, making the classic, symbolic interpretation of Hypothesis B impossible to test. If the assumptions are correct, testing the 'embodied' interpretation of Hypothesis B is compromised, as we would be starting with a known identical system of processing.
- Hypothesis B is best refuted by effective communication with a machine with a very different system of understanding from ours. This second approach is taken here. The representation will then be tested by seeing whether the machine can make the same kinds of decisions about a plan as we would, despite this difference.

This chapter deals specifically with the issue of measuring and representing the plan in a way that is both meaningful and rich enough to capture the essential spatial relationships between parts. Graphs are proposed, from which feature vectors may be quantified to encode the plans for input to the machine. These provide a numerical, metric point in a high dimensional feature space, by which a plan's relationship to other plans may be measured. Two methods are explained in section 7.2 for extracting the feature vector: scalar (mainly Space Syntax) measures, and graph spectra. These are evaluated in section 7.3 in terms of expected

[*] Learning does occur in tuning the senses to relevant stimuli, etc., but the senses themselves are in place for learning to occur. This is analogous to the learning approach to follow in Chapter 8.

classifications, and the spectrum is chosen as the appropriate representation. Section 7.4 examines the ways in which different spectra are affected by global and local variations in a set of test plans. In Section 7.5 a genetic algorithm (GA) is used to generate plans to match a given arrangement, thereby evaluating the effectiveness of the representation within an optimisation algorithm. The objective function is set to minimise the Euclidian distance between the generated population and a set target as represented by the graph spectra.

Graphs may be initially derived from a given plan in several ways, each one encoding different features of spatial structure. Both axial line and boundary graphs (Hillier & Hanson 1984) are evaluated as methods of representation, to determine which is the more appropriate basis for measurement. The GA optimisation is also used to examine which graph types are necessary to capture the arrangement of the plan both in terms of topology and specific shape at both the local and global level. The main question is whether the space can be measured by the spectrum in a manner that yields a representation that is generic enough. This is stated in as the more readily testable:

- Is it possible to get enough information from the combined boundary, axial and convex graphs to describe a design well enough to allow an optimisation algorithm to duplicate it?
- Can a solution be evolved to match the features of an existing plan?

Evolution is seen to progress toward the initial targets over time, even in very large search spaces, indicating that the resulting representation is sufficiently rich to be used not only for detailed statistical comparison, but also in an optimisation algorithm to generate plans with similar spatial configuration.

A definition of the graph types is given below, followed by a description of the spectral feature vector.

7.2 Representing and measuring between plans

The use of visibility graph and axial line analyses are the spatial counterpart to the FEA used in the structural problem (section 3.2.3). Various graph types, including adjacency, axial and boundary graphs (Hillier & Hanson 1984; Turner 2005), have been used to effectively represent spaces, the comparison of which is then normally achieved by set scalar measures such as integration, control (Hillier & Hanson 1984) or clustering coefficient (Watts & Strogatz 1998) derived from these. These measures capture locally that particular quality of the space, but are not sufficient to identify the graph as a whole uniquely. For detailed statistical analyses, database search, and applications that may refer to the overall structure of the space a richer quantification of the entire space may be desirable.

Two methods of *measuring differences between* graphs are presented for comparison in this section. The principles of the above measures are briefly reviewed, and graph spectral analysis is introduced as a potentially richer technique to automatically represent spaces by graphs derived from the plan.

7.2.1 MEASURING LOCAL PROPERTIES

Both visibility graph (VGA) (section 3.2.3; (Turner et al. 2001) and axial line analysis (section 3.2.3; Hillier et al. 1983; Hillier and Hanson 1984; Turner 2005) employ unobstructed lines of sight, and both represent the structure of these as a graph. In VGA, the lines are represented by graph connections, and in axial line analysis they are the nodes, but for now it is the similarity between the methods that is important. Both methods employ a series of predetermined scalar measures of these that indicate the properties of nodes. The *degree* of a node (referred to as connectivity in space syntax) is the most straightforward; it is the number of connections to adjacent nodes and is a purely local property. Other measures such as *integration* (Hillier and Hanson 1984) have been developed to give some idea of the larger structure of graph, but these are still associated with the properties of this structure as seen from particular nodes. In analysis, each node is given a value.

In VGA the nodes correspond to a regular grid, so these measures can be visualised as a raster plot. In analyses of Frank Lloyd Wright's Fallingwater, each reveals different properties of the space, but all generally indicate a more connected set of spaces connecting the two large rooms at the top and left of the plan (Figure 7.1). Alternatively, agent simulation (Turner and Penn 2002; Turner 2006) operates by exploring a similar graph stochastically with a series of random walks, or by calculating the steady state of such a system (Turner 2007). Eventually the state of system converges, and each local node is given a value in proportion to its level of traffic. This naturally incorporates global structure but ultimately still assigns a value to each node (figure 7.2).

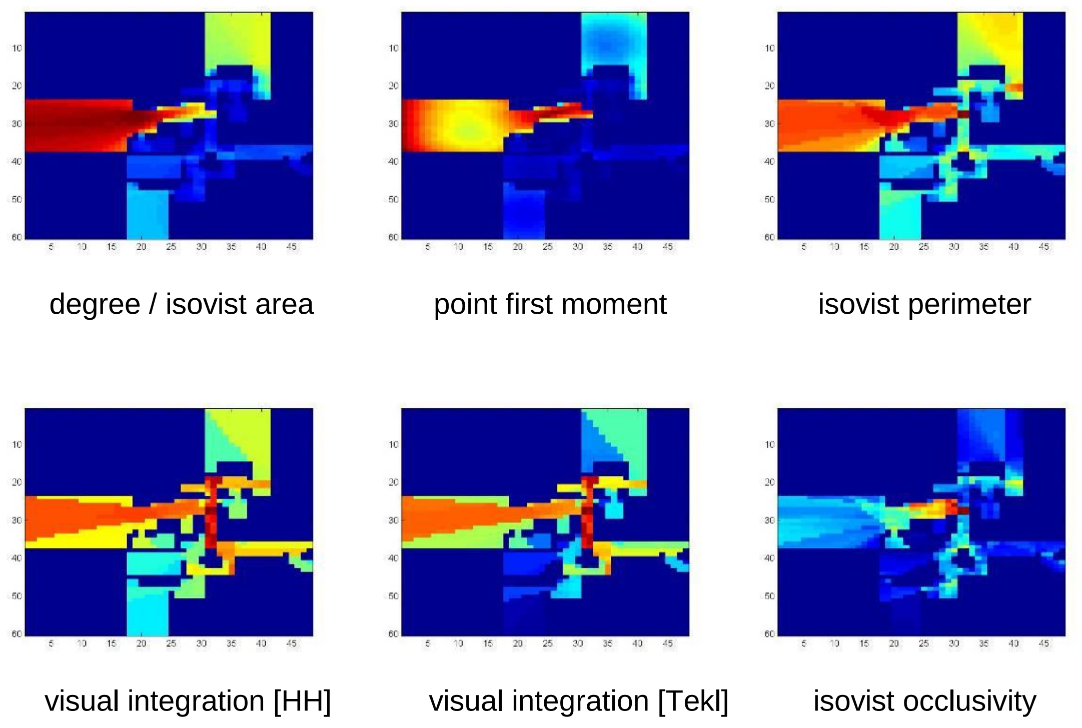


Figure 7.1. VGA measures of Fallingwater. Analyses performed by Depthmap.

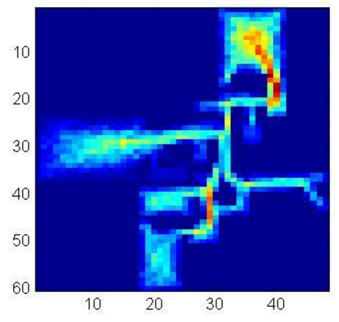


Figure 7.2. The result of agent simulation. Analysis performed by Depthmap.

There would seem to be a theoretical problem with the use of such measures to represent a complex system (§3.2), in that they are predetermined and by their nature reduce much of the structure of the system as a whole. This can be overcome somewhat by using a variety of different measures quantifying different aspects of the graph, rather than just one.

There is a more pressing practical difficulty in using such measures to compare graphs in their entirety is that there is not always a clear correspondence between nodes on one graph and nodes on another, and therefore it is not always clear which values to compare. The regular VGA grid is some use in this, and in this chapter all plans will be scaled to grids of identical size, but node to node comparison is impossible for axial line and other graphs, even if the number of nodes is the same. While degree is entirely local and integration considers deeper connections at local nodes, neither is particularly suited to quantify the global structure for the comparison of graphs in their entirety.

7.2.2 Measuring graph spectra

Several approaches to similarity measurement of whole graphs have been based on small graphs of adjacency or connectivity of spaces in plan. Conroy-Dalton and Kirsan (2008) use the edit distance between two graphs to measure the similarity between buildings, and have shown that these correlate with cultural similarities and differences. Jupp and Gero (2003) suggest an analysis based on similarity and complexity measures of semantic graphs. With very large graphs as generated by axial lines (Turner 2005) or visibility graph analysis (Turner et al. 2001), calculation of similarity becomes more difficult.

Graph spectra have been used in image analysis and pattern recognition to effectively index, classify and retrieve complex, high dimensional data (Luo et al. 2003; Robles-Kelly & Hancock 2003) and are used here in a similar manner. To strictly and fully represent spaces, ideally a procedure would exist whereby any given plan can be mapped to exactly one spectrum, and also that the resulting spectrum can be mapped to exactly one plan. This second criterion is known not to hold true for graphs of small size, but it is thought that almost all graphs have unique spectra, increasingly so as the number of nodes increases to the level of detail in real plans (Van Dam and Haemers 2002)

7.2.3 Defining the spectrum for plan representation

The spectrum of a graph, or ordered set of eigenvalues of its adjacency matrix A, can be used to represent the graph as a single feature vector. This spectrum is useful as a representation of the graph because it is invariant under all permutations of the original matrix, and therefore identical for all isomorphic graphs (Zhu and Wilson 2005). As such, the problem of finding corresponding nodes (section 7.2.1) disappears. While it is possible that two non-isomorphic graphs can share the same spectrum, it has been suggested that this occurs less frequently as the graph size increases (Zhu and Wilson 2005) and therefore almost all graphs, particularly of the sizes yielded by plans, may be uniquely determined by their spectrum (Van Dam and Haemers 2002).

For any graph with a set of nodes V and a set of edges E, the most straightforward way of representing the graph in matrix form is to use the adjacency matrix A, a $|V| \times |V|$ matrix defined by:

$$A(i,j) = \begin{cases} 1 \text{ if } (i,j) \in E \\ \text{or} \\ 0 \text{ otherwise.} \end{cases} \tag{7.1}$$

The spectrum of the graph is found by taking the eigendecomposition of the matrix representation. The eigenvalues λ and eigenvectors φ for A are given by solving for

$$A = \Phi \Lambda \Phi^{T}, \tag{7.2}$$

where the matrix $\Phi = (\varphi^1 \mid \varphi^2 \mid \dots \mid \varphi^{|V|})$ contains the eigenvectors as columns and the matrix $\Lambda = \text{diag}(\lambda^1, \lambda^2, \dots , \lambda^{|V|})$ contains the eigenvalues as diagonal elements. The spectrum is defined as the set of ordered eigenvalues

$$\{ \lambda^1, \lambda^2, \dots , \lambda^{|V|} \}. \tag{7.3}$$

7.2.4 ASSEMBLING THE FEATURE VECTOR: ORDERING OF THE SPECTRUM

Several approaches may be taken to assembling the spectral feature vector from the above set (Eq. 7.3). It is essential that this be ordered consistently for all graphs, and that it be a constant length.

While graph nodes and eigenvalues of the adjacency matrix have no intrinsic order, the spectrum has to be sorted such that any isomorphic graphs will have the same order of eigenvalues. In many analyses (Luo et al. 2003), values (and corresponding vectors) are sorted by absolute magnitude, such that $|\lambda^1| > |\lambda^2| > \dots > |\lambda^{|V|}|$. To ensure a constant length the spectrum may be truncated (Luo et al. 2003) to yield a vector composed of the n largest values

$$S = (\lambda^1, \lambda^2, \dots , \lambda^n)^{T}. \tag{7.4}$$

This can be problematic when the set of eigenvalues contains several values that are of the same magnitude, either positive or negative, and the resulting sort yields a different order for identical graphs. The plot in Figure 7.3 shows three spectra sorted by magnitude, superimposed to show the false discrepancy that would be measured even for values of relative similarity. Sorting by actual value, including the sign such that $\lambda^1 > \lambda^2 > \ldots > \lambda^{|V|}$, avoids this problem and is the method used here.

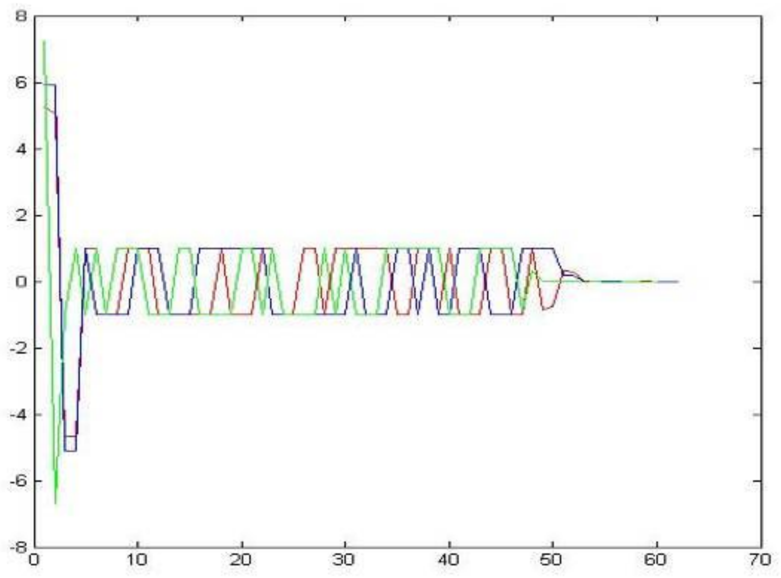


Figure 7.3. Three different versions of the same graph spectrum, sorted by absolute magnitude. Many identical magnitudes with different signs are sorted arbitrarily.

7.3 Classification by visibility measures vs. graph spectra

The two types of representation—VGA measures and axial line spectra—were tested by classifying a set of building plans. A set of 100 sample plans (Figure 7.4) was used containing examples of two building types: modern offices, and neoclassical museums. The experiment tested the degree to which the machine classification corresponds to our own, human decisions, and so these groups were specifically selected by the author as sets that can be distinguished by a human judge. Distinct building types were used, rather than simply contrasting styles, to avoid any ambiguity as to the distinction of style at this point, and avoid as much as possible the experimenter's bias.

Each group consists of ten topologically distinct plans, each repeated under various rotational and scaling transformations for a total of 50 samples. In overall shape the instances of these two types are quite similar to one another. Many, for example, consist of a ring of usable floor plate around one or two central voids: in the case of the offices this void represents a central core of lifts and services, in the case of the much larger museums based on neoclassical models it is a courtyard emitting natural light to interior galleries. This similarity in shape between the two groups ensures that the classification is based on more subtle details.

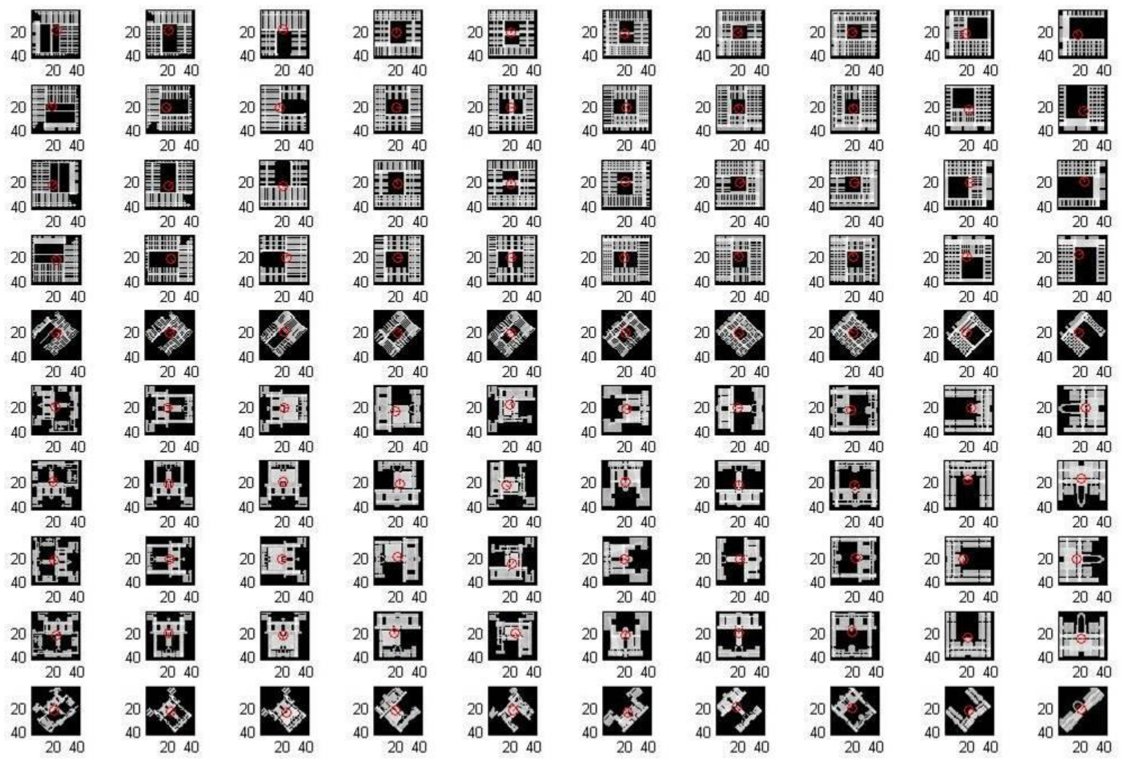


Figure 7.4. 50 plans representing museums (upper), and 50 offices (lower).

Both options for defining the feature vector of the plan utilised Depthmap. VGA output is typically in the form of a grid based raster image displaying one of several spatial measurements in pixel intensity, similar to the raster visual images used in the examples of robot navigation and facial recognition (section 3.5.1). These were translated directly into high-dimensional feature vectors pixel by pixel in the same way. Alternatively, the graph constructed by an axial line analysis may be used and the feature vector found by spectral embedding (Robles-Kelly and Hancock 2003). In this case the spatial measures provided by Depthmap were ignored, and only the minimal axial line graph was used, its spectrum becoming the feature vector. Each of these methods was tested in this section and evaluated on their ability to capture the distinction between the two classes of plans.

7.3.1 Classification of plans by VGA data

Image processing and machine vision techniques were modified to use VGA data as raster images. The output of a visibility graph analysis consists of an image (of a plan), in up to ten dimensions, each of which represents a measurement of visual integration, mean depth, etc. rather than light intensity. The same kinds of techniques used to evaluate visual images can thus be extended to deal with the 'sense' data provided by a visibility graph, and to classify a plan based on these characteristics.

VGA was performed on each of the 100 scaled samples, and this data—in effect a ten-dimensional raster image—was taken as the computer's sense data or feature vector for each of the spaces. Because comparison between samples must be done node to node, each was scaled to an equivalent 22 pixel square. The seemingly very low resolution of the samples is due to the amount of processing required to then deal with each sample as a higher dimensional vector. Even at this small size the solution space becomes 22×22×10, or the number of pixels times the number of VGA measures, for a total of 4840 dimensions. (The fact that even this reduction of detail is adequate to describe the essential features of the spaces is encouraging, and much higher resolutions can easily be accommodated.) These feature vectors represent each plan as a single point in a 4840-dimensional space, but 2-dimensional, reduced versions of this are plotted in Figure 7.5 with each axis determined by the two principal components of the data set.

The two groups of plans are reasonably well classified. The offices, marked by red '○'s, are distributed mainly over a region in the left part of the plot, and the museums, marked by blue '×'s, are toward the right. The left plot shows the set of plans plotted against the first two components on the x and y axes. There is a near separation of the set into two distinct clusters, and the analysis has picked up on the rotation of the plans in that similar transformed versions share an identical point in these two components. The third component—shown on the vertical axis (right)—picks up differences in rotation.

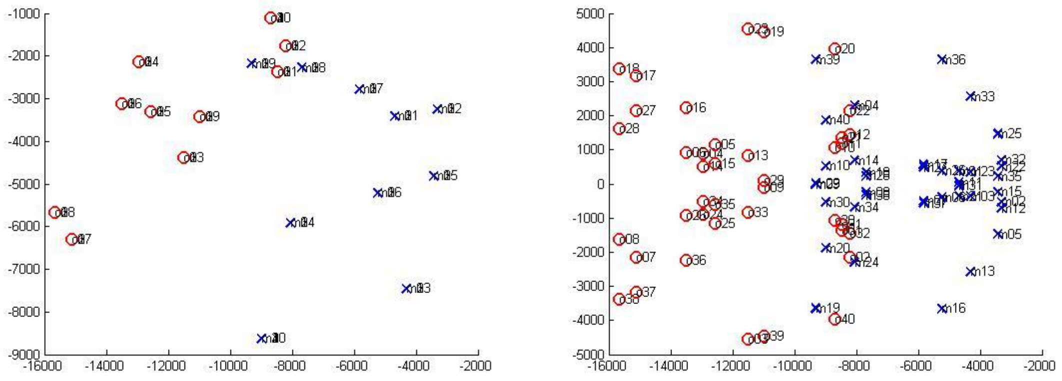


Figure 7.5. The set of museum and office plans plotted against the principal components of VGA. Left: components 1 and 2; Right: components 1 and 3.

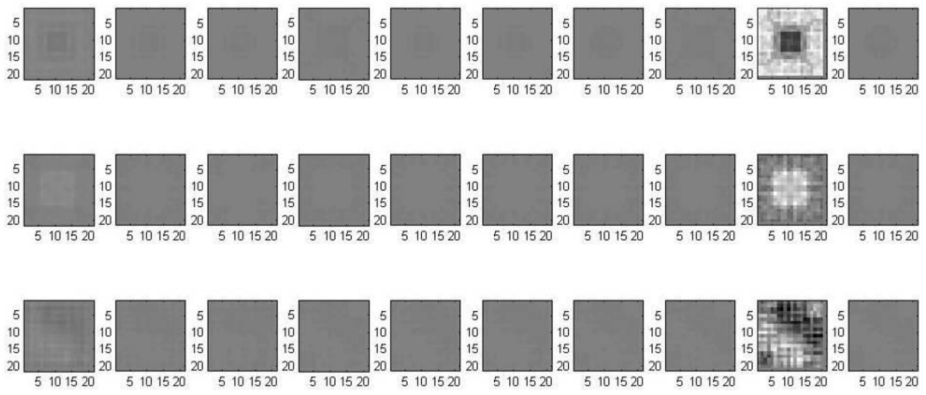


Figure 7.6. The first three principal components viewed as raster images.

Viewing these three principal components (Figure 7.6) as raster images on their original grid highlights two potential means to improve the classification. First, the first two components display biaxial symmetry corresponding to the regular rotations made to the plans in the data set, and the third does not. But real data are rarely so clearly formatted, and initial rotations may be arbitrary, a situation far more problematic in classification. Second, the variance of the set is clearly dominated by the measure of connectivity (second measure from the right), purely because of the units of measurement. To solve these problems, a small amount of processing was done to eliminate any human prejudices from the set of data to be examined and put all samples on a roughly equal footing.

- *Rotation:* An estimate was made of the principal axis of each sample and the plan rotated to this alignment. Each was then cropped and scaled to an equivalent resolution: in this case 22 pixels square.
- *Dimensional scaling:* Finally, the data within each of the VGA measures was scaled to a similar overall range. The units used by Depthmap result in vastly different values for

each measurement: Isovist moment of inertia, for example, might have a value of 0.001, while degree of connectivity measures 100, thereby biasing any analysis highly in favour of differences in connectivity. The measurements of all plans were scaled uniformly to the same variance to reflect this difference in units.

The entire set is plotted with these adjustments in figure 7.7.

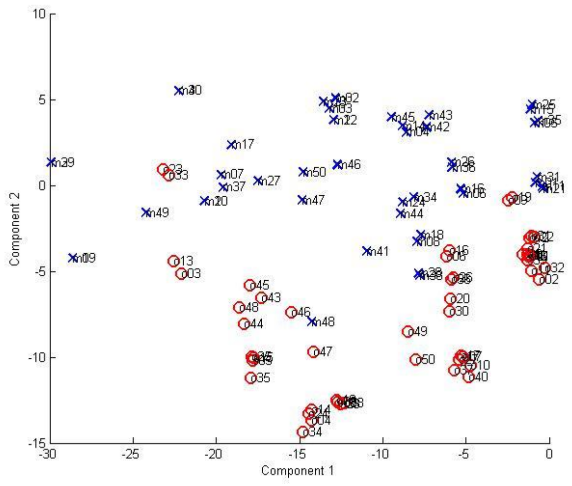


Figure 7.7. The set of museum and office plans plotted by principal components, after processing by rotation and dimensional scaling.

There is still some overlap of isolated points in the two groups, but in general these are separated into distinct groups based on building type on the second principal component. Notable outliers are those with very similar looking samples for both offices and museums, such as the ninth column in the plans diagram above, some of which are indeed on the boundary between the two groups. Offices 09 and 19 can be seen toward the right of the diagram close to the museums, and museums 09 and 19 are at the left.

The realignment to eliminate human bias when positioning the plan, however, is not perfect. Rotational similarity in the original data set is still picked up in the analysis, although it is less clear, as can be seen by the proximity (rather than overlap) of plans that are rotational variants of one another: offices 02, 12, 22, 34 and 42, or museums 01, 11, 21, and 31. In general, the chosen principal axis for each plan is not necessary aligned with any element of the plan itself, and could, in effect, be causing some confusion to the set by causing plans with similar principal axes to be grouped together. The ideal result would be similar to Figure 7.8, a plot of the first ten plans of each set, analysed in their original orientations, which shows an even clearer distinction between groups.

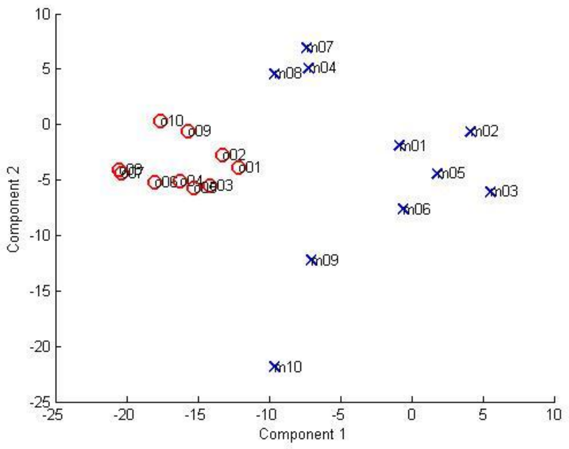


Figure 7.8. The first ten examples only of museum and office plans, plotted by principal components

7.3.2 Classification of plans by axial line analysis

The use of the axial line graph is a potentially more attractive option because it eliminates the problem of rotation by dealing with topological connectivity rather than location on a raster grid, and because this connectivity is an essential local feature of spaces to be classified. This method attempts to deal directly with connectivity in the axial line graph by using the spectrum of this graph to form the feature vector. Principal components analysis is performed again; however, the plans are analysed not as raster images, but as a collection of lines that make up the minimal axial line graph, as provided by Depthmap.

Two options were tested: the first combines the graph with the same Depthmap measures as above, by applying them to the axial graph; the second uses the basic graph without analysis. To do so, the adjacency matrix was used in both a weighted and unweighted form. Weighting was implemented by multiplying each connection entry by the measurements of integration, moment of inertia, etc., thus yielding ten separate spectra for each plan. The combined set was used as the feature vector. The unweighted graph alone is represented by a binary adjacency matrix consisting only of 0 and 1.

The previous data set of distinct museum and office types was used, but all rotationally similar examples were removed from the plots as their adjacency graphs are identical. The resulting, reduced set consisted of 20 museums and 20 offices. The PCA embedding of these plans with weighted graphs is shown in Figure 7.9, followed by a similar embedding based on the adjacency graph only (Figure 7.10, left) and an embedding of the adjacency graph by multi-dimensional scaling (Figure 7.10, right). It can be seen in all cases that the two groups of plans can be almost entirely classified even on only the first principal component (horizontal axis).

There is a clear separation into two clusters in each case. The museums, marked by '○'s, are distributed over a region to the left of the principal axis, and the offices, marked by blue '×'s, are toward the right.

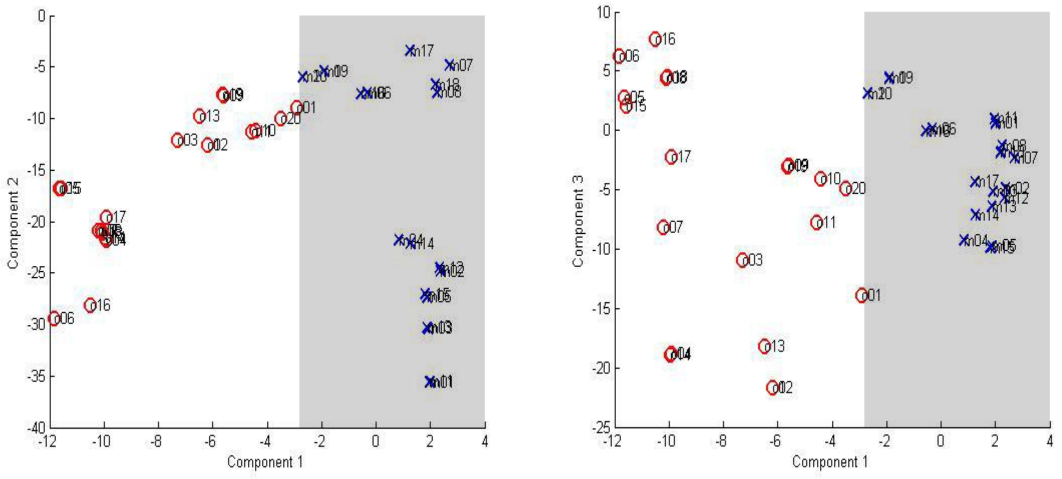


Figure 7.9. The plans are classified by the machine by PCA using weighted graphs to derive the spectra.

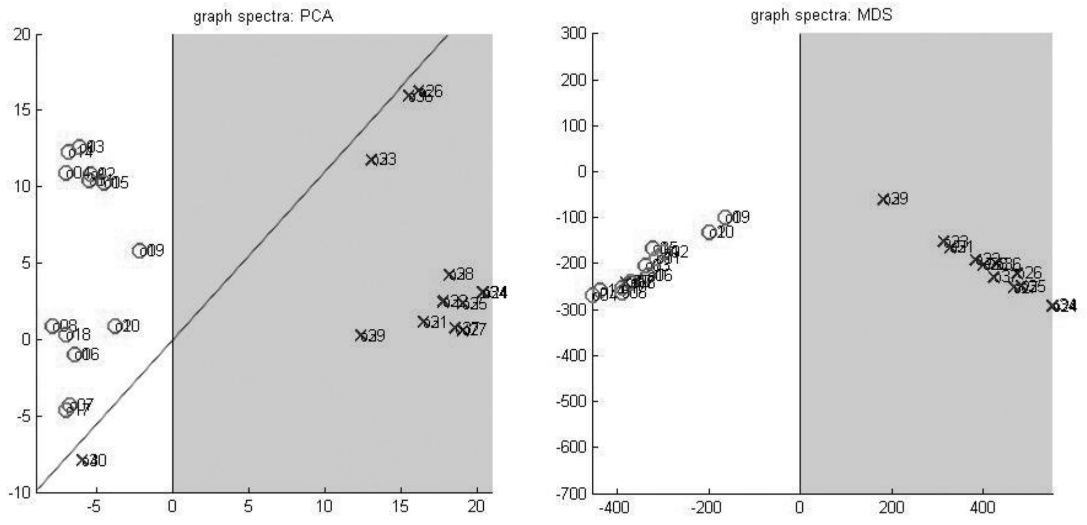


Figure 7.10. The plans are classified by the machine by PCA (left) and multi-dimensional scaling (right). The horizontal axis is the principal axis of the data set; the vertical axis is the second.

The combined method of weighting the graph by each of its spatial measures resulted in a space of two groups separable along the first principal component, but only by very little. The point of separation seems arbitrarily placed at approximately -2.9, and does not correspond to the separation that would be found by an unsupervised algorithm.

The embedding using only the adjacency graph is much clearer (Figure 7.10, left), with a large margin between groups. A single outlier from the office group can be accurately classified as well using another dimension of the second principal component. Except for this outlier, the two sets are easily separable along the first component at the origin (the mean of both sets taken together), and the margin between them at this point is greater than the distance between points of a single class. A multi dimensional scaling method (Figure 7.10, right) was also used as an alternative to PCA, resulting in an even clearer distinction between the two classes. Unsupervised clustering or simply a default setting of a boundary at the origin would therefore give the desired classification result.

The combined method, using the spectrum of an axial graph weighted by the Depthmap measures, appears unnecessary. The use of the binary adjacency graph alone outperforms that of the weighted graph with a clearer distinction either side of the origin of the principal component, and with a wider margin (Figure 7.10). These extra measures are derived initially from functions of the adjacency graph itself, and so are more likely to act as noise, rather than adding new information to the feature vector.

7.3.3 Practical evaluation: visibility measures vs. axial spectra

The use of the axial graph spectra was found preferable to the use of VGA measures both in terms of better classification, and in terms of computation time (there were a maximum of 68 dimensions for axial spectra compared with 4840 for VGA). There are three main reasons for its use:

- The graph data is more generically represented by the spectrum. VGA is dependent on a number of predetermined algorithms for processing the scalar measures, each already embedded with spatial theory. The use of many of these helps, but to some extent their selection is still arbitrary.
- Arbitrary rotation of initial plans is not an issue, as is the case with human representations of spaces, e.g. physical drawings of a plan.
- Connectivity is an essential feature best revealed by the graph.
- Spectral analysis of axial graphs appears to result in a cleaner distinction into classes that a human judge is likely to recognise.

The use of spectral embedding techniques on simple adjacency matrices is used throughout the remainder of this research.

7.4 Deriving initial graphs from the plan

Both VGA and axial line graphs were used above to analyse the plan set for embedding. Although this is a suitable general method for analysis, a refinement of the method of graph generation was desirable for three reasons:

- The analysis required to generate such a representation using Depthmap currently takes from several seconds to several minutes (on a typical desktop PC running at 2.0 GHz), far too long to be of practical use in a GA requiring hundreds to thousands of fitness evaluations in a reasonable period of time.
- Axial lines are visual, defined as straight, and do not directly represent access and movement patterns in spaces that may be curved[*] (Dalton et al. 2003; Figuieredo and Amorim 2005). Other options should be investigated.
- Also, the analysis empty space alone does not adequately capture the relationship between desks that face one another. The situation of a desk in relation to others highlights the fact that there are two types of interaction involved at each unit: the connection of the chair to an open space of the floor, and the possible connection of the desk itself to other desks through which documents, messages and conversation might pass in the course of normal office activities.

The third reason especially relates to important aspects of a person's embodiment in an office: how posture relates to engagement with media on the desk or people behind it; the fact that our forward vision determines this to some degree; the communication to others given by such body postures. These 'real world' issues are part of the socio-spatial pattern that the architect deals with in designing. The computer, never having sat at a desk, has no way of understanding these in the same way, and constructing an explicit ontology to do so is the assumption built into both Hypotheses A and B. Instead, additional map types will be used (in addition to axial) that make the desk relationships themselves visible to the algorithm.

Three map types used by Hillier and Hanson (1984), the axial, convex and interface maps, result in graph representations of space, and form the basis of the plan features to be represented. This drawing of space alone does not capture the relationship between desks that face one another, so

[*] The empirical success of axial analysis indicates that the connections between straight axial lines do capture the properties of curved space; however, these properties are emergent and dependent on the method of processing. It may be possible to extract them from the spectrum, but what is being sought is as generic a representation as possible, without having to provide a means of processing. Providing the computer another map type *in addition to* the axial simply provides a wider base from which it can interpret, as suggested by results of tests in §7.5.3.

an additional graph of desk adjacencies will be introduced. Three graph types and some properties of their spectra are investigated in this section.

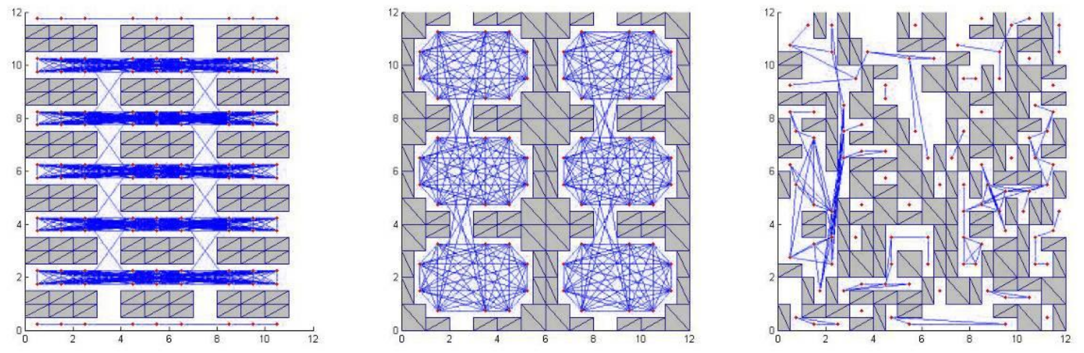
7.4.1 Selection of plan features

Three map types have been used, two for open space and one for desk connections. Each captures different features of the plan, and their use alone and together will be compared. Common to all three is that the basic unit of spatial division is the desk or chair grid point rather than the more generic notion of the convex space. This follows roughly from the terminal nodes of buildings used in the interface map (Hillier and Hanson 1984), which are essential in expressing connection in systems consisting of two types of space, and is also a standard which is economical to implement, particularly for iterative optimisation. On this basis three graphs are constructed: a visibility graph between seats, and two versions of a modified boundary graph: one for open spaces and another for desk groups.

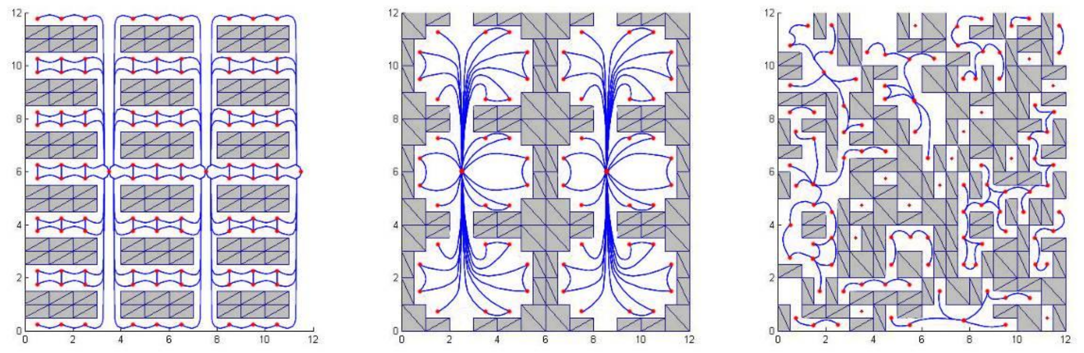
- *Axial / visibility graph features*: An algorithm was implemented to generate an axial (visibility) map from chair positions in a plan, in which graph nodes are not the lines themselves, but the chair points. This differs from the true axial line map defined by plan vertices (Turner A, 2005), and is more like Depthmap's grid visibility analysis, but the use of the set modules with chair points as the only possible nodes greatly reduces the computation time necessary to generate a graph. The algorithm simply draws links between from each chair node to every other chair node that can be connected by an unobstructed, direct sight line through empty space. Three examples of the resulting axial graphs are displayed in Figure 7.11 (a).

- *Boundary graph features*: The boundary graph treats as a unity all space within a given boundary, regardless of its shape, and therefore captures all immediate or continuous spatial connections, regardless of sight. The basic node is still taken as the individual chair point, but empty space is also considered and links are drawn only between nodes that are directly face-wise adjacent. A chair point is connected to any other immediately behind or to the side or to an open space onto which it backs. All adjacent void spaces are then grouped together into one single node, regardless of size, representing all continuous space within the boundaries formed by the desks. This use of the term boundary graph is not typical in space syntax literature—its use is roughly equivalent to the interface map (Hillier and Hanson 1984), as opposed to the perimeter boundary mapping of Psarra and Grajewski (2001). Boundary graphs are shown in figure 7.11 (b).

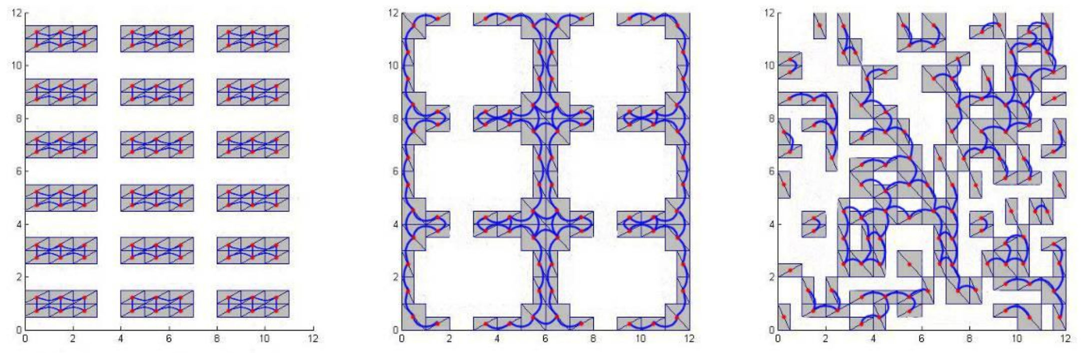
- *Desk graph features*: Desk graphs capture the relationships between adjacent desks. They are generated by the same method as the boundary graphs, except the graph nodes are the desks rather than chairs, and these are connected to all other adjacent desks. The resulting graph will generally not be unified, but segmented into discrete sub-graphs each corresponding to a connected group of desks. The desk graphs for three plans are shown in Figure 7.11 (c).



a) axial graphs



b) boundary graphs



c) desk graphs

Figure 7.11. Axial, boundary and desk graphs for different plan configurations.

7.4.2 The effect of local/global features and scale changes on the spectrum

The ultimate goal of the optimisation in design will not be to replicate a given plan, but to reproduce learned features of plans, usually in a new space that is likely to be of different shape and size. Furthermore, in the learning of features from a set of plans of differing shapes and sizes, it would be inappropriate to have the training dominated by the overall plan size, ignoring local desk configuration. Plan spectra were examined to determine the effect of global and local changes to the plan on the feature vector.

- *Capturing local features in the spectrum*

  The spectra of two plan types were compared to judge the effects of global vs. local changes to each. In the plots below (Figure 7.12), the same *local* arrangement of desks is repeated a different number of times for both the straight rows and the outward facing clusters, but the *global* configuration is changed by increasing the plan area and total number of desks. The spectra of the boundary graph (left) and axial graph (right) are shown, with axes scaled to the same overall length for ease of comparison. In both cases the overall distribution of values in the spectrum and their magnitudes appear identical, but are spread over a wider number of values when the plans increase in size.

  The spectra have a total number of values equal to the number of nodes in the graph, thus doubling the graph increases this. Truncating the spectra to a common length is typically done for comparison (Luo et al. 2003; Robles-Kelly and Hancock 2003) but these plans indicate it may be useful to interpolate the spectra of various graphs to capture the similar local arrangements in such a situation. Plans of very different sizes can be compared in this way. Truncation is used in this chapter, but a deeper investigation of the two methods will be given in section 9.1.

- *Capturing global features in the spectrum*

  Local changes to a simple pattern of desks affect both the values in the spectrum and their distribution. In the plans in Figure 7.13, extra desks are added across the horizontal rows in the desk clusters, which increase the values of the spectrum and its overall length. Although the precise relationship is not immediately obvious, values are roughly proportional to spatial integration—in particular the largest magnitude eigenvalues, which increase as the number of desks adjoining the open space grows.

  The number of positive or negative ‘steps’, or distinct values in the spectra of the boundary graphs shown, is always equal to the number of nodes of distinct connectivity (degree) values in the graph, and the number of units within each ‘step’ is equal to the number of nodes of that type. In the bottom plan, for example, the first step is of two values of very high magnitude, corresponding to the two main open spaces. This is followed by four smaller steps corresponding to the four levels of integration along the rows of desks. These steps decrease appropriately in the smaller plans above.

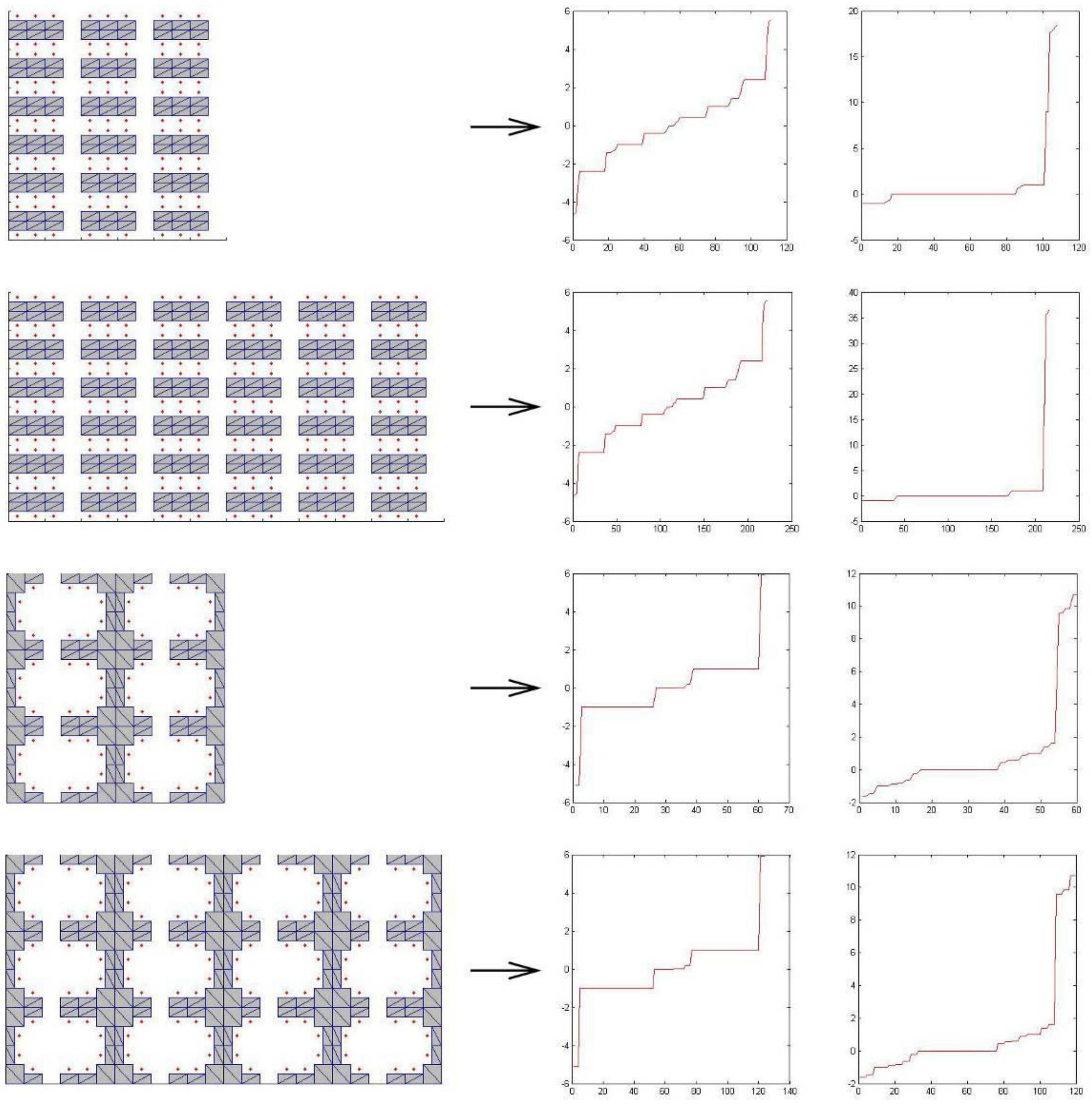


Figure 7.12. Global changes to the plan increase the length of the spectrum, but not the distribution of values. Spectra from boundary and axial graphs are shown.

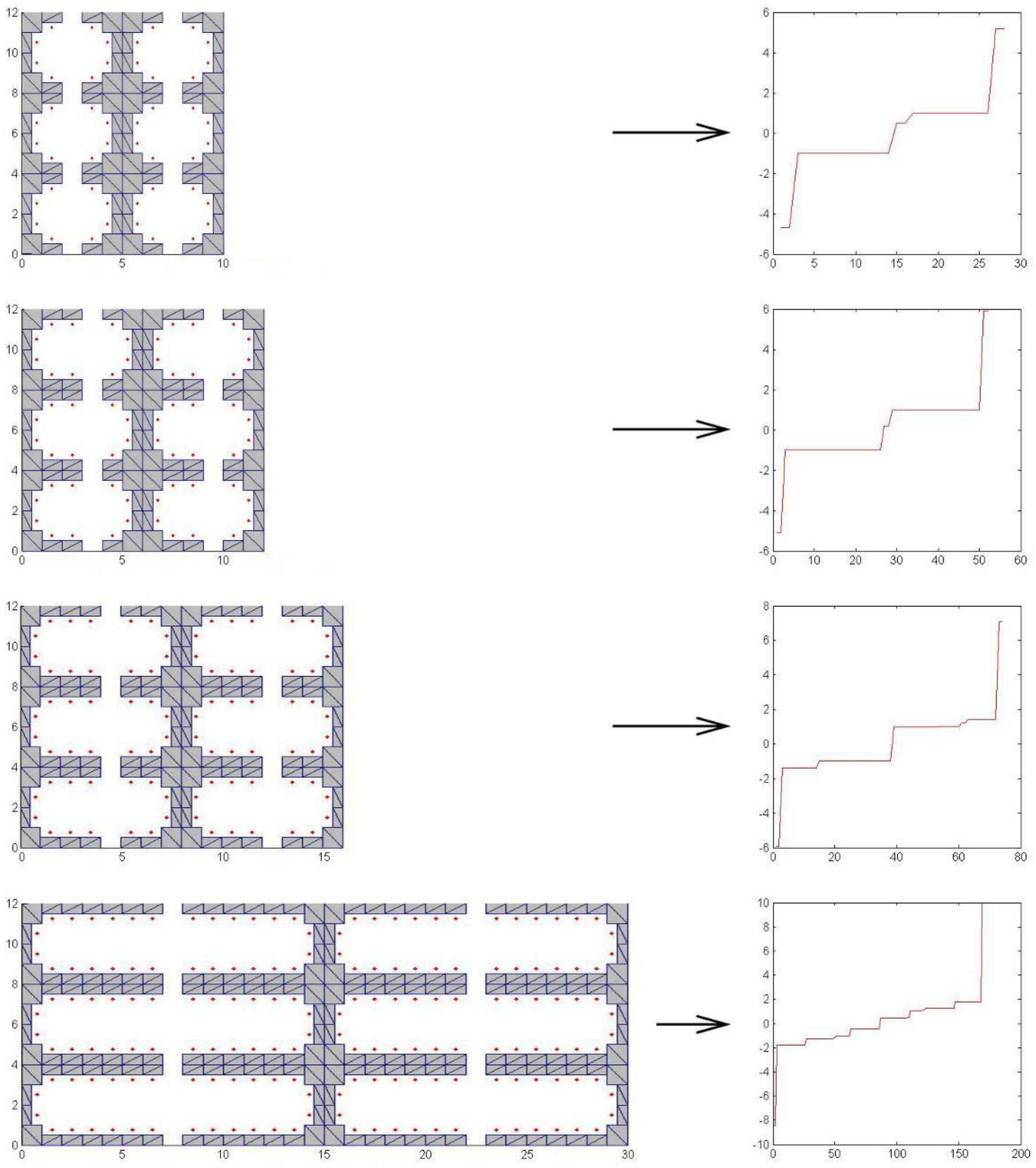


Figure 7.13. Adding nodes locally to the plan increases the magnitude of the spectral range, and the number of distinct values. Spectra from boundary graphs are shown.

7.5 Test of the method: optimising to set plans

To be of use as a metric representation of plans, each spectra must correspond reasonably closely to a unique plan (section 7.2.2; Van Dam and Haemers 2002), and also measure similarities between plans in such a way that they can be used to search for a particular plan. This section tests both criteria by performing optimisation by GA toward a preset goal of a given plan. The major question is whether the search space defined by the spectra enables the algorithm to produce a plan that is similar to the initial target. In doing so, the performance of the three graph types above is of particular interest. The results of such a search reveal the kinds of features represented in each spectra. These are compared to determine which are necessary to capture the relevant features of plans.

7.5.1 Algorithm specifications

The target plan can be considered a *prototype*, as it is a single, real example that the optimisation algorithm is set to match. Evaluation of success in this case would thus be a simple comparison of how similar the result was to the initial goal plan, and so the objective function is a distance measurement in the *n*-dimensional space of the spectra. More precisely, the fitness is taken to be inversely proportional to the distance between a given plan's graph and that of the given prototype:

$$f(i) = 1 \Big/ \sqrt{\sum_{j=1}^{n} \left[S_i(j) - S_{goal}(j)\right]^2} \tag{7.5}$$

where *S* is the spectrum as given in Eq. 7.4.

An underlying orthogonal grid was used to represent the layout of desks within a plan, assuming a planning grid similar to those used in the planning tool in chapter 6. On this grid, desk/chair units were represented by an integer value (1–6) as a filled/void pair within one grid square. These may be in one of four orientations, as well as completely empty or completely filled. The method only produces orthogonal arrangements as implemented, but could be used with plans that contain several non-orthogonal grids or in principal extended for use with hexagonal or other planning grids if required.

Three genotype specifications were devised to represent the layout of desks within a plan, varying in the amount of constraints used and the corresponding size of the search space, although only genome representation A, the simplest of the three, was used in this chapter as it is the least constrained, and it was hoped to avoid as far as possible building in bias in a general test of the method. The base plan grid is simply represented as a matrix of the same dimensions as the grid with each entry encoding the position of a desk using one of the above values 1 to 6 (Figure 7.14). As a result the search space involved is vast ($6n^2$ for an n × n plan or roughly 10112 for even a small plan of n=12), and seemingly simple changes to plans as rotation of sections, mirroring, etc., can not be expressed except by individual changes to desks. It is highly unlikely that a matching plan will be found using this representation in a reasonable length of time, but the analysis of the results can indicate what plan features are being captured by the graphs. Boundary graphs only are investigated in the following section, and the results then compared with the use of other graphs.

All three genotypes are described in Appendix A.

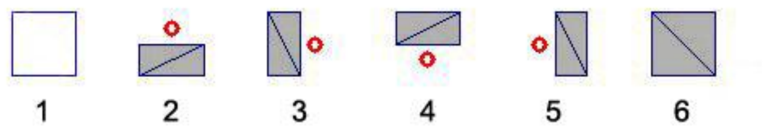


Figure 7.14. Possible desk types and positions in the grid.

7.5.2 Plan Matching with Boundary Graphs

The algorithm (a GA with genome representation A and a population size of 35) was tested with goals of two generic plan types: an arrangement of simple rows, and a linked set of outward facing clusters. Figure 7.15 shows first the prototype plans (left), then the result of the fitness evaluation over time counted in generations, and the resulting final outputs when the optimisation was terminated.

The results initially appear unpromising, except for the replication of the gross topology of the second plan: a series of two separated spaces divided by a central row of desks. A closer inspection reveals that the finer details of desk adjacency have also been duplicated, however, inasmuch as they are captured by the boundary graph alone. The graph represents all connected open spaces as a single node, and each chair as a separate node, with connections between face-wise adjacent neighbouring squares. The graph in the second test plan consists of two unconnected sub-graphs, each of which has a single node (of white space) connected to 12 pairs of adjacent desks and 6 single desks. The resulting optimised solution has a similar structure, except the number of pairs is only 10 and the number of connected single desks is 5 or 7. The structure of the first test plan is less clear, in part because the boundary graph is inherently unable to represent the axiality that is so prominent a feature, but the general arrangement of three main open spaces joined by groups of desks (in many cases they are three, but not all) is captured. In both cases the number of desks or ratio of empty space is also approximately correct.

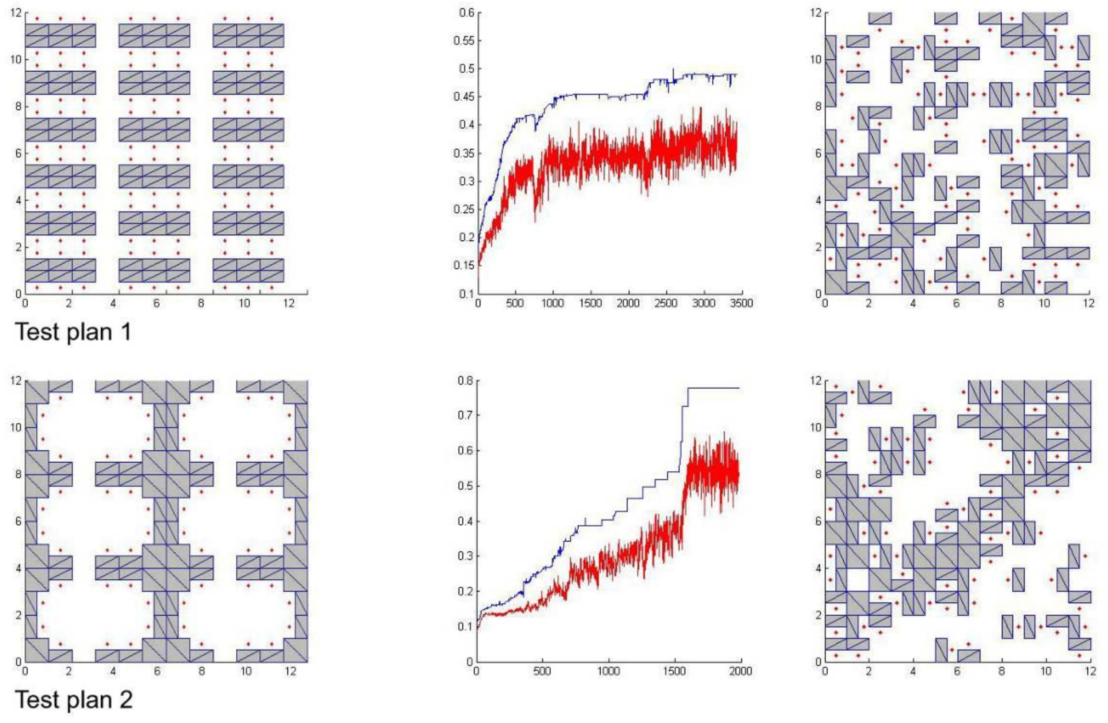


Figure 7.15. Results of a GA search for a boundary graph spectrum taken from straight rows (above) and convex desk groups (below).

7.5.3 Correlation between Topology and Fitness Level

A sharp increase in fitness is evident in the evolution of the plan to match the second test above. Fitness jumps from about 0.5 to 0.8 in just a few generations around generation 1600. Because of the simplicity of the genome representation used, it appears likely that this was due to a sudden arrival at a large feature match like the overall topological division into two sub-graphs. This is examined more closely by comparing the spectra and fitnesses of very similar plans that do or do not display this topological division. The initial prototype plan is shown in Figure 7.16 (top), with its spectrum to the right. Below this are the plan produced by the genetic algorithm, and finally the same plan with several desks removed to connect the two separated spatial regions.

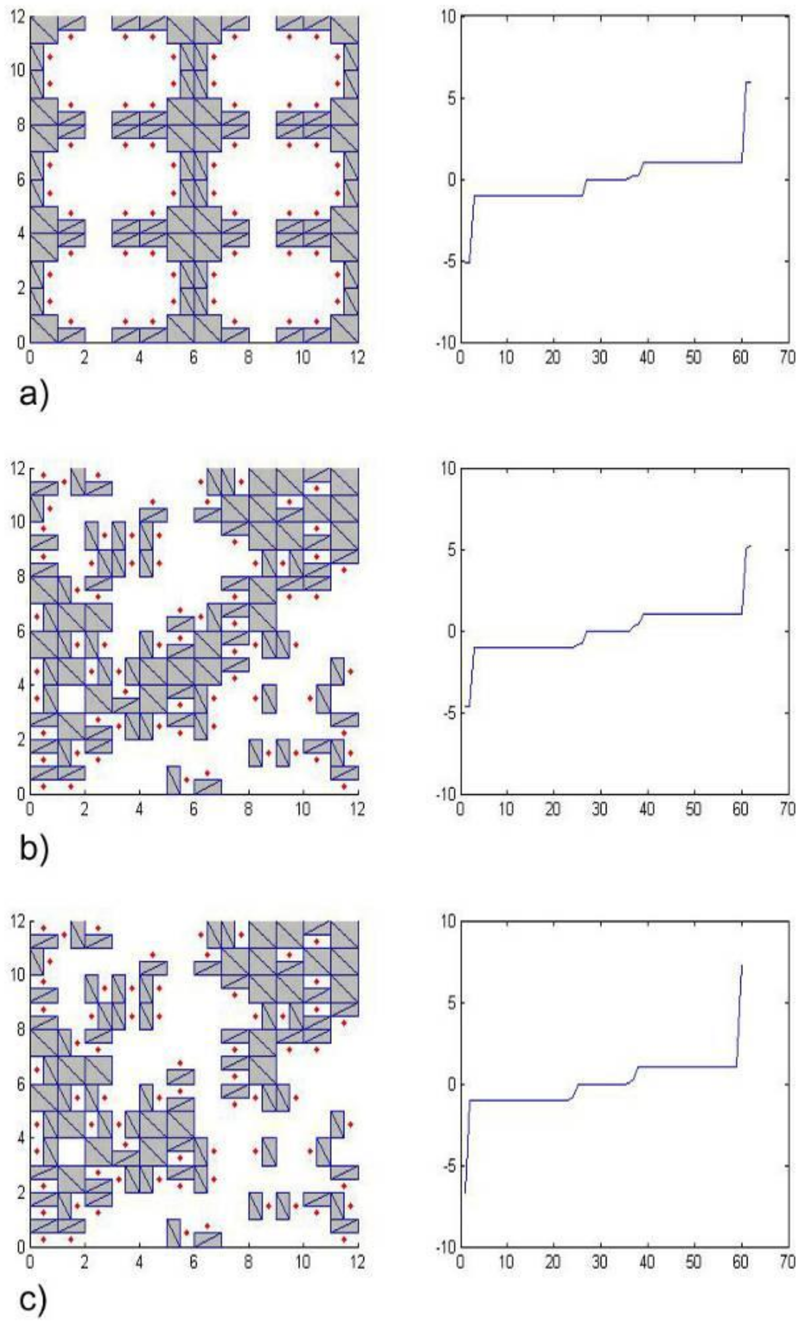


Figure 7.16. Spectra of Topologically Similar (a & b), and Distinct Plans (c).

The spectra appear very similar to one another in overall shape and magnitude, but in fact the numerical difference between the two lower plans is significant. The distance of the optimised plan (centre) from the goal is 1.2953, resulting in a fitness of 0.7780, whereas the distance of the linked plan below from the goal is 7.8061, resulting in a fitness of only 0.1281. This difference is largely due to the change in the eigenvalues of second greatest magnitude (plotted first and last on the horizontal axis, above), which correspond to the division of the graph into two distinct sections.

The overall form of the plan in each case looks unlike the initial goal, in part because of the vast search space and relative simplicity of the genome to efficiently represent patterns. But the plan similarities and the degree to which the spectrum of the evolved graph resembles its goal indicate this is due to the inherent limit of what the boundary graph alone can represent. As a method of fitness measurement, the distance between spectra appears to be an appropriate indication of overall topology, and the optimisation appears successful in capturing the essential topological features represented by the boundary graph chosen.

7.5.4 Using Multiple Graphs and Spectra for GA Search

The use of the other graph types was used to clarify the results obtained in §7.5.2 and §7.5.3 above. Graph representations based on unobstructed lines of sight, either as axial lines or grid visibility graphs, are the most prevalent in space syntax analyses and the type represented in Depthmap. While the boundary graphs used above capture only the topology of the space, axial graphs record what can be seen from a given point, and therefore the shape of the space: the typical minimal line axial map produced by Depthmap, for example, represents each convex space by a retained axial line. On initial consideration of the two graph types to be implemented, axial and convex, it appeared likely that the axial would contain all of the information available in the convex graph, and more. Rather than including a redundant graph in the computation therefore, an optimisation or learning algorithm would ideally make do with just the richer, axial representation. It appears on further consideration that some features may not have been represented by the axial representations chosen. Because only the chair points are used, the axiality of major corridors in the plans above is not necessarily captured by the axial graph alone. The following tests compare the result of plan matching by GA with the use of the spectra of axial graphs as described in section 7.4.1, with those combining both axial and boundary graphs.

A further refinement to the representation is also tested. Axial graphs do not typically have self connected nodes (i.e. $diag(A) = 0$ for the adjacency matrix), which effectively removes nodes that have no other connections from the graph entirely. Isolated desks as appear in the results of section 7.5.2 are therefore invisible, and ignored in the similarity measurement that determines fitness. The results are compared for both versions of the axial graph: without self-connected nodes (Figure 7.17 a, b) and with self-connected nodes (Figure 7.17 c, d, e).

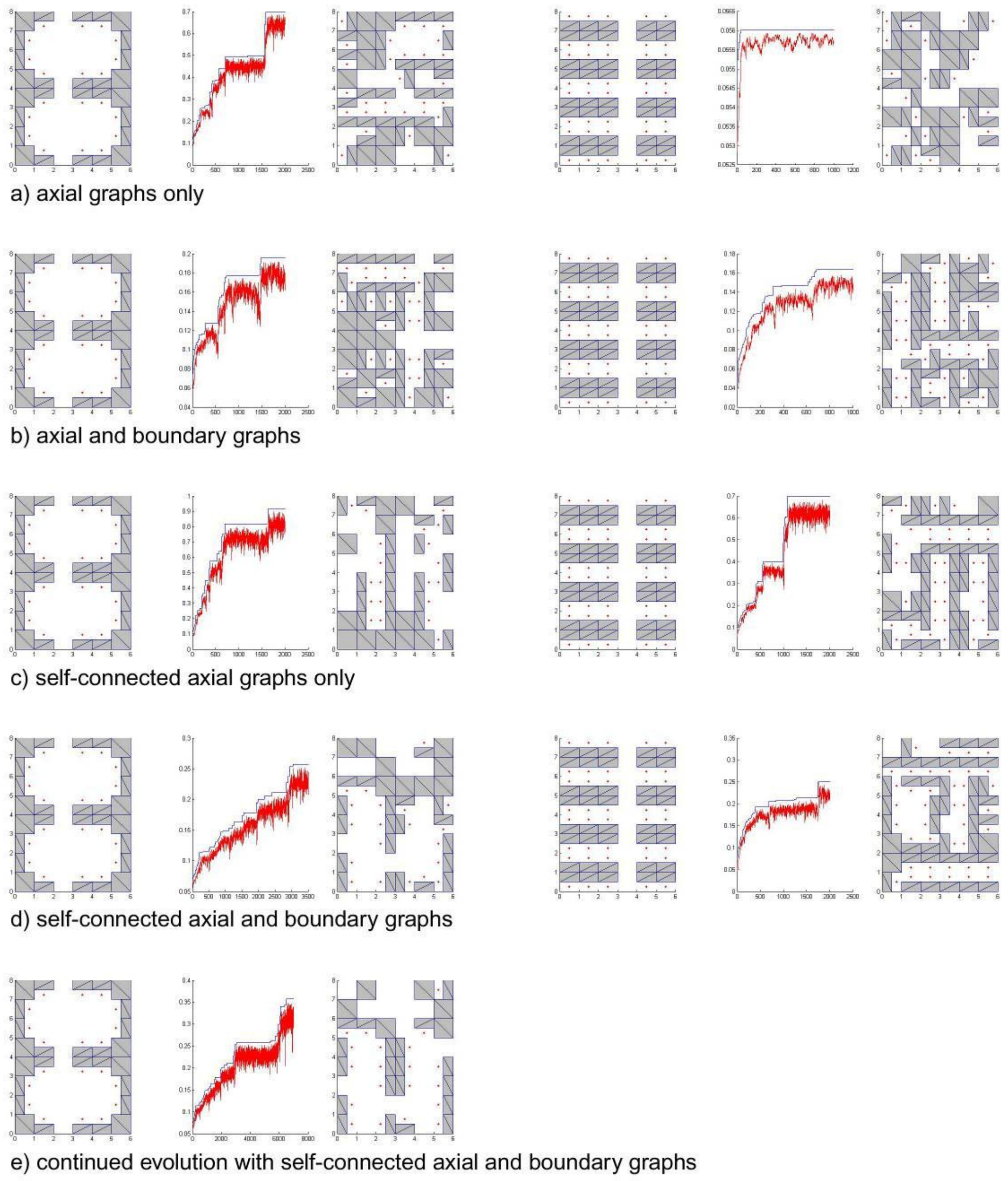


Figure 7.17. Comparative results of a GA search for two plans using various graph types: (a) axial graphs, (b) axial and boundary graphs, (c) self-connected axial graphs, and (d, e) self-connected axial and boundary graphs.

It appears from the evolved plans that the axial method is far better at characterising the space than the boundary graphs alone, yielding clearer reconstructions of the initial plans. Both methods combined appear better still, although with a somewhat longer computation time.

The largest improvement in terms of the characterisation of the open spaces (and removal of unconnected nodes) came from connecting each node to itself, thus allowing unconnected nodes to appear on the resulting graphs and thereby providing an automatic penalty to the fitness.

In none of the examples is the original plan duplicated exactly, but as mentioned in section 7.5.1 it seems unlikely that genome representation A would do so even with a graph that captures all the features of the plan, simply because of the vastness of the search space. In most cases the fitness measurement rises initially but appears eventually to plateau at an incorrect local optimum, an expected result of the genome only expressing individual desks and not larger scale patterns. The rise in fitness appears smoother however when two graph types are used than with one.

The use of both graph types together also produces better evolved plans. It appears from the GA output that the axial method is far better at characterising the space than the boundary graphs alone, yielding clearer reconstructions of the initial plans. The plan in Figure 7.17 (d, right) very closely resembles the convex group arrangement of its goal. If evolution is continued for several thousands of generations the plan improves to quite closely approximate the target plan (Figure 7.17 e). The addition of both the boundary and desk graphs to the axial appears to yield marginally better results, although with a further increase in computation time. This similarity and correspondence of measured fitness to perceived plan similarity indicates that the essential features of the plan are indeed captured by the combination of plan spectra.

7.5.5 Evolving plans with targets of different global shape

In addition to enabling the matching of an existing plan, the representation should allow for the reproduction of the features of a plan in a different context, in particular a plan of a different overall shape and size. The properties of spectra of differing plan size were discussed in section 7.4.2, but a brief test was also made of the GA to search for plans of varying size. In this case a target plan based on a 12×12 grid provided spectra (axial and boundary) to be matched by a 6×6 plan.

The first plan (Figure 7.18, top) is a set of back to back, linear rows each containing a total of 8 desks and joined by breaks to form aisles. The result approximates this with rows of an average of 7 desks, but little connection between them. The most striking global feature of the second plan (Figure 7.18, bottom) is the topological division of both boundary and axial graphs into two distinct areas. This same division is evident in the GA result, although with only a 6 × 6 grid there is no space available to subdivide each into a series of convex spaces. There are 10 desks in the convex spaces of the original plan, and 9 or 10 in the result.

In both cases the results give a good approximation of the overall large scale features of the plan, and also the size of small groups within, but the constrained size restricts local topological features—particularly the division into local subgroups. The way in which these do not match their targets may be partially explained by the way the spectra are compared. Because of the difference in size between the target and result, the spectra used in optimisation were truncated for comparison, whereas the observations in section 7.4.2 (Figure 7.12) suggested local features are better captured by interpolation. Both methods will be compared in more detail in chapter 9.

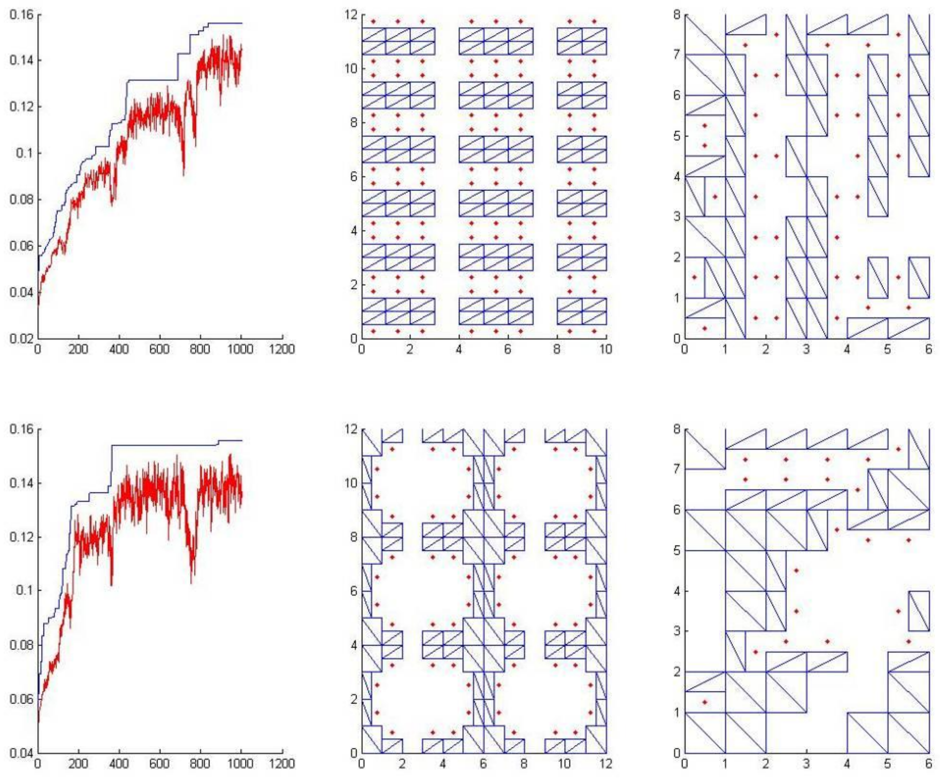


Figure 7.18. Results of a GA search to match plans of a different size: target plans 12×12, resulting plans 6×6. Axial and boundary graph spectra were used.

7.6 Chapter review

As a prerequisite for machine learning to follow, this chapter has developed a generic representation of office plans—individual phenotypes of a morphic language. Several possible methods were tested and evaluated based on the geometrical features of plans that they were able to capture and reproduce. Both predetermined visibility measures and graph spectra were examined, and graph spectra have been shown to be the more effective. For all graph types, the spectra were seen to capture local patterns of spatial arrangement even as global size is varied, and thus may be used in comparing plans of differing overall scale. They constitute a reliable metric of plans, in that similar plans have spectral vectors that fall close together in a high dimensional space, while very different plans fall farther apart. In GA search, even with a large search space and an intentionally restricted genome, the resulting plans resembled the goal sufficiently well to suggest that the spectrum encodes almost all of the spatial structure of the plan.

It appears essential, however, to use more than one method to derive the graph from the existing plan. Several graph types were examined, and results indicate that boundary graphs alone can capture the gross topological qualities of a space, but axial graphs are needed to indicate local relationships concerned with the actual shape of spaces and lines of sight. Using both the boundary and axial graphs together better represents the overall geometry of the space, and allows for a finer and more effective measurement of similarity. The result in a GA search appears to be that the fitness increase is smoother and the final plan more closely resembles the goal, as it is less likely to become trapped in a local optimum.

The most important reason for this is to provide a broad and generic basis for the subsequent learning of representations of relevant features. Cilliers (1998, pp 75-78) warns against pre-processing of input data in neural networks when dealing with complexity, primarily because this reduces features and introduces a bias that may not be helpful. The ‘wicked’ aspects of space planning ensure the designer will not know in advance which features may be relevant, and so minimising this bias is essential. Falsifying Hypothesis A will also require this. Graph spectra in themselves are a richer metric of structure than scalar measures (§7.3), but the use of several underlying graph strategies is best to ensure a broad range of features are available (§7.5).

Target plans were never duplicated exactly in the GA search due to the extreme size of the search space of genome representation A. The search was also hindered by a lack of higher structure that makes the kinds of useful design changes corresponding to convex spaces nearly impossible, particularly in later generations or by mutation alone. Of the three possibilities, genome A was chosen precisely to avoid a predetermined bias and because the results were being used to explain the features inherent in the spectra, rather than as a viable plan. Results were drawn at various stages of evolution based on progress along the way. In the work to follow (Chapter 9), final outputs will be examined, so the more constrained genome representation C will be used.

Chapter 8: Design through learning and mimicking archetypes

Summary: Machine learning is used to derive essential features from the raw data of plans, thus providing the definition of an archetype. The fact that this higher order concept can reconstructed by means of the data alone, indicates that it can be communicated using only this data, without being defined explicitly. This provides evidence against Hypothesis B.

The office space problem is one in which the complexity of the system makes the objective difficult to discern. Rather than explicitly stating a goal by a set measure or with a pre-existing target plan, this chapter outlines how an objective might be implicit in a set of examples, and how this might be quantified. When the desired features of a complex design problem are unclear, the proposal will be to use machine learning to find them, so that they may be used in the optimisation of new designs. The present chapter concentrates specifically on outlining the principles of a feature subspace and how inductive machine learning is used to define it.

This approach is based on the use of norms and types, where types embody the desired features of the design solution sought. Part of Hypothesis B is the proposition that meaning is predefined, as a universal and unchanging set of types might serve to convey. The approach here does not take the features of these to be predefined, however. At its core, type can be anything that distinguishes one group of works from another. For the purposes of design, types will be represented in a way that is able to capture the way they can evolve over time (§2.2.2), or be reinterpreted.

Hypothesis B can be falsified by communicating a complex concept through examples of existing designs. This is in fact a test of competence in a morphic language, in which the ‘inverted genotype’ is readily extracted from examples. This chapter will outline how such higher level structural rules may be repeatedly extracted and communicated across different systems, not as a ‘genotype’ but as an ‘archetype’. If the processing rules are not to be predetermined, this competence will then be tested by judging whether different sets of processing rules are capable of producing similar structures from the same archetype.

The approach is tested in the context of architectural plans. The archetype is introduced in §8.1 and then demonstrated in §8.2. This section deals with analysis, and begins by providing methods by which plans of real buildings can be embedded in a feature space such that those that are similar fall near to one another. This yields a way in which desired features might be understood and represented by a computer, which is not based on any predefined symbolic representation, but allows for reflection and interpretation. §8.3 refines this spatial embedding, and combines a very simple generative process to synthesise designs. It will show that the archetype allows competence in a morphic language to be achieved by allowing the resulting output of one system to be produced by another that is structurally different.

8.1 The archetype

Types are often imprecisely defined. This is considered difficult, if not impossible, if by definition “everything is more or less vague in the type” (Quatremère de Quincy 1832). This vagueness refers only to the fact no single explicit or ostensive definition is possible, however, because a range of varied examples may all equally exemplify the type. Wittgenstein (1958, §67) proposes the tacit notion of a ‘family resemblance’ as the best alternative, for example in the definition of ‘games’—for which there is no single essence or characteristic set of features. Here types will be defined with precision, without sacrificing this property.

For the production, as opposed to the analysis of a type, many approaches utilise a generative rule system, by which examples of the style may be built. Just as a generative grammar (Chomsky 1957) is capable of producing all well formed expressions of a language, these rule systems are capable of producing examples of designs. As an approach to representing style, a given rule set is often thought to encode the style (Hersey and Freedman 1992; Koning and Eizenberg 1981; Stiny G and Mitchell WJ 1978), such that it produces designs that are examples of it, and can therefore serve to represent it. This is a useful analytical and pedagogical abstraction, but gives limited insight into how we actually design, transmit or communicate types, or how they can change. Linguistic grammars themselves are simply formal methods, and do not necessarily represent the actual cognitive processes involved in understanding or producing language. In language, however, it is not the grammar that defines style, but specific choices within it – the same English grammar, for instance, is capable of producing romantic, utopian, realist, symbolic or an innumerable number of different styles of literature. Similarly, the idea of a type, like the prototype plans in chapter 7, will allow a separation of the type definition from the generative rules themselves, by guiding such choices within a larger rule system.

The archetype is an ideal model comprised of the features that exemplify the type. This concept differs from the predetermined description of a type as category into which particular examples can fall, and from that of a prototype, precedent or case, which are actual instances on which later examples can be modelled[*]. The target plans in chapter 7 were prototypes—specific plans to be matched, but an archetype is a generalisation that cannot exist materially, yet matches and is compared to many actual instances. This is almost certainly not a real example, but an abstraction made up of only those features necessary to differentiate it from other archetypes.

If one imagines a particular style or work of a particular architect, that of Palladio for example, certain features such as axial symmetry, classical orders, etc. come to mind immediately because they are common to the works. Others, like the colour of the roof tiles perhaps do not, either because they are not shared (the Villa Rotunda is red, San Giorgio Maggiore is white), or because they are ubiquitous (the majority of Italian roofs have always been terra cotta). Yet even this set of features cannot be predetermined, as it may differ from style to style. In human terms, it is this set of essential features that constitute an archetype. This section proposes a computational and algorithmic equivalent.

The use of a feature space agrees with our own intuitive ability to evaluate designs as being stylistically nearer or farther from one another, and is commonly applied in machine learning. The nearest neighbour algorithm (Duda et al. 2001), for instance, classifies an unknown example of data by simply measuring its distance to previously known and labelled examples, or prototypes. Clark (1993, p. 20) describes connectionist representation as utilising prototypes, in that the ‘typical’ example of an item in a particular category can represent that category as a statistical ‘central tendency’ of features. Thus it is possible that this prototype may be between real examples and may or may not actually exist. Measurement of any actual instance is by a ‘semantic metric’ (ibid. p. 19), or distance in sub-symbolic representational (e.g. weight) space, with similar patterns naturally having a smaller semantic distance.

The *archetype,* as described here, also uses such a central tendency and a semantic metric, but differs from a *prototype* in that it is comprised only of a subspace of features (dimensions) and could thus never exist as a real example. This archetype is based on the mapping of design examples in a high dimensional feature space (Φ), and uses methods of dimensionality reduction of this space to yield an archetype that describes the common features, *and only those common features*. A large variety of real examples can thus be equally exemplary instances of the archetype (having a semantic distance of zero) while differing in their irrelevant features,

[*] The choice of terminology is meant to draw the distinction between the *arche-* ‘ruling’ *type* from the *proto-* ‘first’ *type*, which by definition precedes all other examples. The archetype can be defined after its precedents.

and these may still be measured with precision[*]. The archetype can be used to classify, and as a measure to generate new designs.

Two innovations are proposed over such existing methods. The first is that the archetype is a generalisation that combines both the concept of the ideal example and the particular space in which it is measured, consisting only of the features relevant to that particular style: it is comprised of both a mapping to a feature space and a point within that space. The second innovation concerns the synthesis of new designs, and incorporates the notion of affordances (Gibson 1979), to consider the design process as a continual evaluation and choice between possible alternatives as the development of the design progresses. These choices can be made by repeated measurement against the ideal that the archetype represents. Figure 8.1 presents a schematic diagram of the method to be described in the following sections.

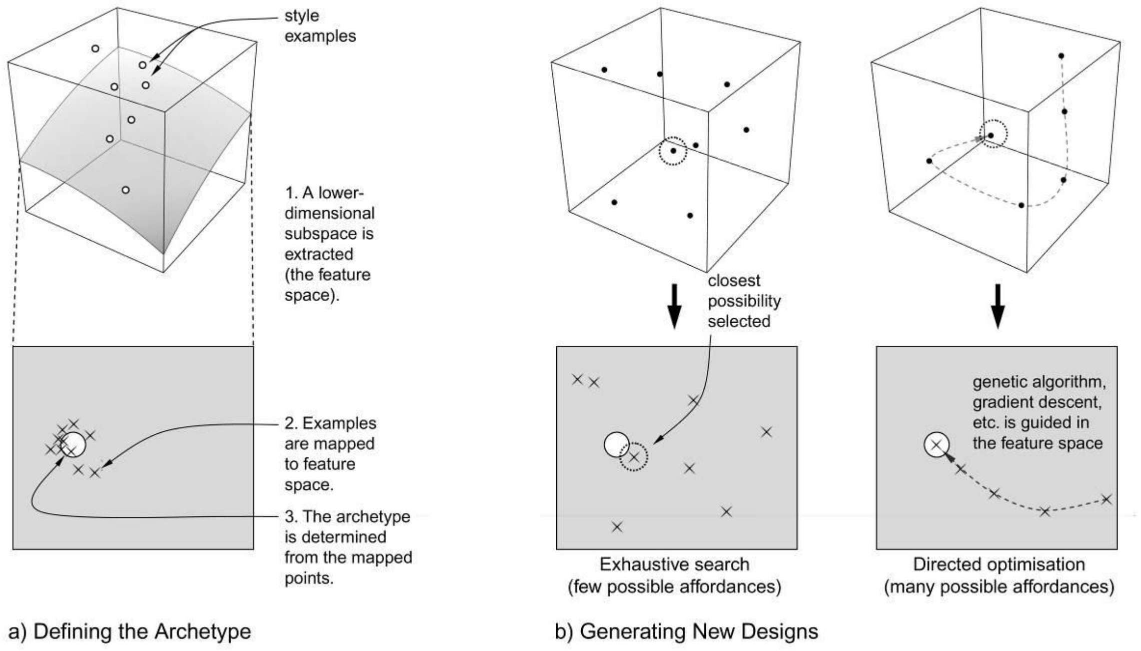

style examples
1. A lower-dimensional subspace is extracted (the feature space).
2. Examples are mapped to feature space.
3. The archetype is determined from the mapped points.
a) Defining the Archetype
closest possibility selected
genetic algorithm, gradient descent, etc. is guided in the feature space
Exhaustive search (few possible affordances)
Directed optimisation (many possible affordances)
b) Generating New Designs

Figure 8.1. Diagram of the method of (a) deriving the feature space and archetype from examples, and (b) generating new designs by selecting affordances in the feature space.

[*] Zwicky and Sadock (1975) explain the difference between *ambiguity* and *lack of specification*, demonstrating that a statement can be perfectly precise but yield many possible interpretations of events with respect to details not mentioned. The sentence “My sister is the Ruritanian secretary of state.” is unspecified with respect to unstated details of her age, the duration of her post and whether she is right-handed, whereas the sentence “They saw her duck.” is ambiguous in having two possible semantic (and syntactic) interpretations of “duck” (ibid. p. 2–3). Archetypes are similarly underspecified with respect to features not relevant to the type, but are precise, not ambiguous.

8.1.1 Properties of types

In deriving a method for determining and measuring these features then, several properties evident in real design processes are desirable:

- *A method should exist for deriving types from examples*: In designing, we often cannot state a goal explicitly, only point to examples. Types are only discernable after several similar designs exist, and it is through the emulation of examples, not by recipe, that a type is communicated.
- *Types are not clearly bounded, and are subject to evolution and change*: Even the seemingly clearly defined classical orders, before being codified by Vitruvius (2001), were the result of centuries of experiment and still subject to debate on particular proportions throughout the Renaissance.
- *Types are divisible to ever finer distinctions*: There is a continuum of stylistic differences (Gombrich's (1960) 'flux of experience'), in which a type may be distinguished by period, movement, designer, and successive distinctions down to the level of the individual work.

The method proposed here is intended to reflect these properties.

8.1.2 Generating designs

A simulation of the design process in all its detail may be impossible, but complexity studies indicate that in modelling such systems it is the basic structure of interaction that is required (Arthur et al. 1997). Ideally this is reflected in any generative method chosen to produce designs, as in the choice of a bottom up aggregation rather than a top down division to model cities that are built over time (Hillier and Hanson 1994; Duarte et al 2006). To allow flexibility inherent in the properties above (Section 8.1.1) there are two possible structures through which the generative method and type representation can interact:

- *The type is represented in a generative rule set*: In this case the rule set must be constantly re-created, allowing it to evolve. The possibility has been suggested (Prats et al. 2006) that new grammatical representations, for example, can be created at any step in a continuing design process.
- *Type is determined by analysis*: If the rule set is generic, a single rule set is capable of generating many different types/styles, each of which is defined not by the rules but the choices made within it. Each possible choice would be evaluated with regard to producing the type.

These two options are recognisable as *instructive* and *elective* (Lederberg 1958; see §1.1.2) respectively, the first exemplified by classical generative methods such as shape grammars (Stiny 1976), and the latter by evolutionary theories and optimisation. They can also be identified, respectively, with *procedural* or *declarative* knowledge (§3.4.3), in terms of what must be communicated to the system in order for it to produce a given type of design. The second option would seem to have the only real mechanisms for interpretation and reflection required in design, however (§3.4), and is more amenable to the requirements of flexibility and fine distinction mentioned above. It is this determination of type by analysis that is implemented in a computational model in the next section. Design examples are evaluated as a group, then the relevant features determined automatically in the definition of the archetype. By so doing, both the processes of defining a style and building examples of it can be performed by a machine.

8.2 Example analysis: representation of an archetype in feature space

Theories of typology—whether or not they admit change and reinterpretation—generally agree that examples of a type are united by some 'family resemblance' (Wittgenstein 1958, §67). A possible effect of this is that, to some extent, examples of one type are objectively more similar to one another than to examples of another, and so the description of a type is inherent in the examples themselves.

This section examines how this can be quantified, using axial graph spectra to illustrate a feature space (Φ) for building plans, and then explains how the archetype is defined in this. The hypothesis is that there are a number of features that play a role in type—geometrical, socio-spatial or otherwise—and that these will affect a plan's placement in Φ. Real buildings can be used to test for these, because although the non-geometrical features are impossible to verify intersubjectively from the plan, objective properties such as date, architect and function are known.

The archetype feature space Φ must be derived initially from a generic representation (as Chapter 7) in which the type is not assumed a priori, i.e. it contains features relevant to any possible type, both desirable and undesirable. If the resemblance between examples is objective, a typological or stylistic description will not be needed a priori, and the algorithm should be able to make the same kinds of classifications that we would, without explicit training. In this section an initial feature space is found for a group of buildings by unsupervised dimensionality reduction. The examples were found to fall naturally into recognisable clusters within the space without the need for explicit labelling, and these correspond to the buildings' inherent similarities. This will provide the basis from which to derive an archetype from the examples.

8.2.1 Description of plan styles in a feature space

A set of 24 plans were used, taken from various twentieth century iconic buildings (Weston 2004). Axial graphs were constructed for each of the 24 samples, and this data – in effect a set of binary adjacency matrices – was taken as the computer's raw sense data, or experience of each of the spaces. Analysis was performed using Depthmap, which constructed a minimal axial graph for each based entirely on plan input and an objective algorithm. Figure 8.2 displays the lines of sight in Frank Lloyd Wright's Fallingwater shaded to indicate the degree of spatial integration. Darker lines clearly reveal the shallowest zones that link the rooms of the house. As in chapter 7, the spectrum of this graph was taken to form the raw feature vector so as to deal directly with the connectivity in the axial line graph. The reduced map was constructed by an algorithm (Turner et al. 2001), and no subjective choices were required between further measurements provided by the software (of integration, moment of inertia, etc.) or of the selection of units in which to measure. Reduced graphs ranged from a minimum of 8 to a maximum of 147 nodes (axial lines), but were equalised by padding with null entries (roughly equivalent to truncation[*]; see Luo et al. 2003) to yield feature vectors of 147 dimensions.

[*] Truncation of spectra is the usual method when all examples have higher dimensionality than the desired feature vector. Here, the smallest example of 8 dimensions was an anomaly, and a greater number of dimensions was used. They are roughly equivalent and have little effect on the results because the set of eigenvalues that comprises the feature vector typically contains a large number of near-zero or zero values. These dimensions therefore have little or no variance and may be removed or added to without affecting the principal components of the overall distribution used in PCA. When more sophisticated learning methods are used to extract the feature spaces in Chapter 9, other alternatives will be tested.

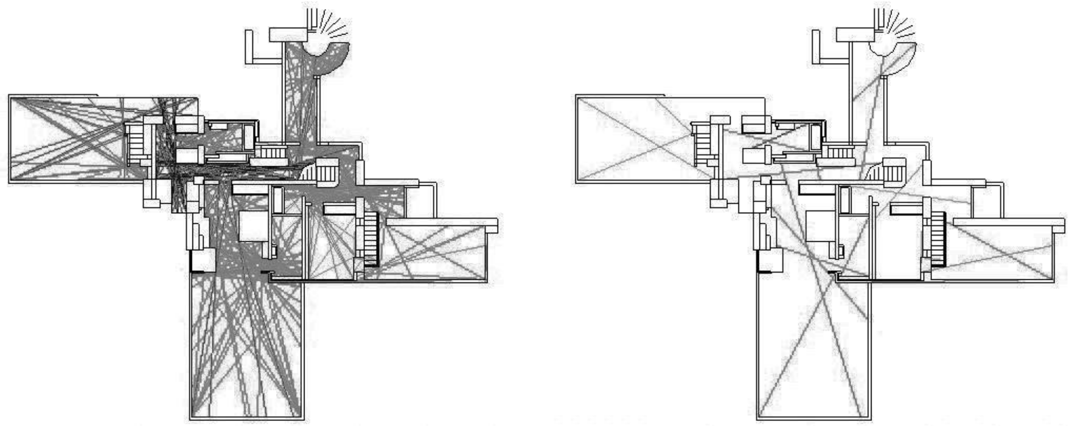


Figure 8.2. A complete axial graph of Frank Lloyd Wright's Fallingwater (left) and the reduced minimal graph (right).

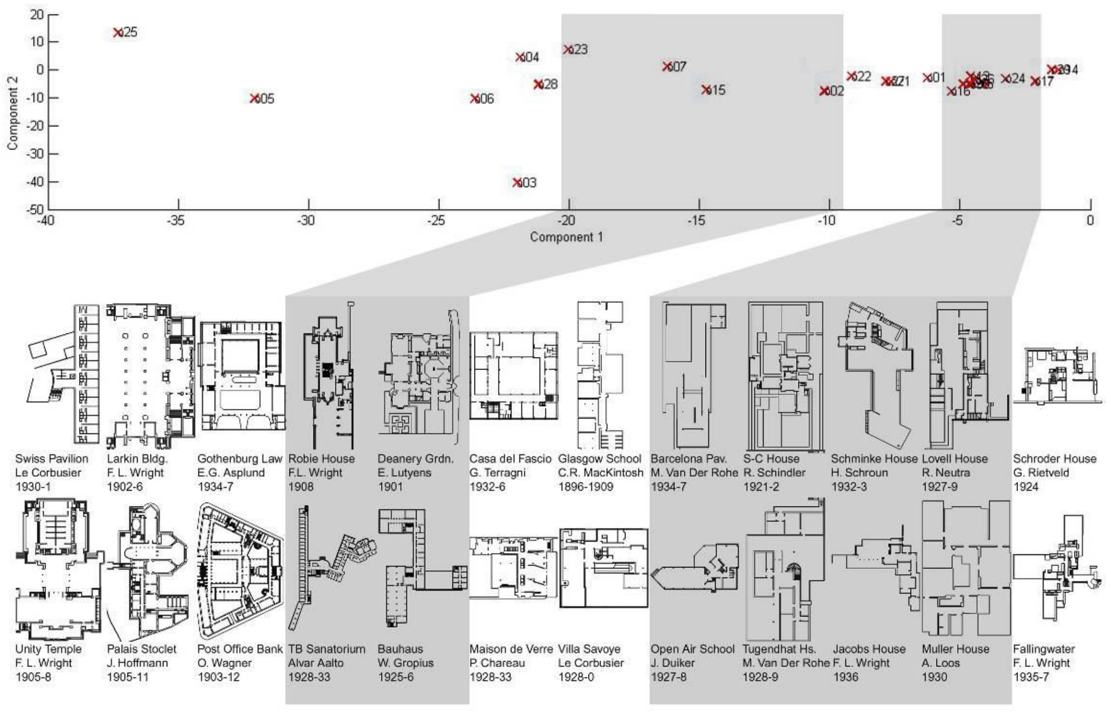


Figure 8.3. The example buildings plotted in feature space.

With the spatial analysis of each plan quantified as a feature vector, the example buildings can then be plotted in a high 147-dimensional feature space, with each value in the spectrum on a different dimensional axis. The task of deriving the reduced feature space Φ is that of determining which combination of these 147 dimensions are relevant and which are not.

At this point there is nothing to recommend any of these buildings over their counterparts, so the principal components are used as an initial test (as in chapter 7). PCA determines, for a given set of data, the combination of dimensions in which it is most likely to vary, and these are used as new axes in a reduced version of the space that captures the essential statistical features of the data set. A reduction of the plans' feature space based on the first two principal components of the set is shown in Figure 8.3. The dimensions of this new feature space are strictly computational, and are meaningful only in a statistical sense, rather than in the sense that they could be easily described. The first component, on the horizontal axis, represents a combination of the features in which the plans are determined by the algorithm to differ the most.

Yet these can be seen to include meaningful features such as program typology (most houses toward the right and larger, public buildings to the left) as well as general stylistic trends. The shaded groups indicate the proximity of most of the axially symmetrical, pre-Modernist buildings to one another, as well as rough zones of early, and of high modernist buildings, typically with spaces that flow into one another articulated by shifting orthogonal planes. There is an approximate chronological order from left to right, seen clearly in the buildings by Wright, the Villa Savoye and contemporary Maison de Verre are next to one another, and van der Rohe's two works are virtually overlapping.

The divisions in the diagram are shaded simply for clarity, and are not meant to suggest a division into distinct groups. The points as plotted here represent a continuum in a uniform space. It is true that van der Rohe, Le Corbusier and Wright can be considered to have quite different styles, as can different buildings by Wright alone, but proximity in this feature space is meant to suggest those buildings that are more similar to one another by degrees. The only significant outliers in this regard seem to be those caused by program (the private houses vs. the public buildings) but at this point no attempt has been made to draw the distinction between those features that are important and those that are not. (The machine learning algorithms to be described in section 8.3 will allow this.) The fact that similar buildings do fall near to one another in the reduced feature space confirms that the features indicated by the principal component are at least statistically related to a notion of building types.

8.2.2 Defining archetypes by the clusters

Suppose there were clusters of buildings within this feature space that one wished to mark out as clear types—the pre-modernist institutional buildings at far left (*typically* with defined internal spaces and little connection to the exterior) and the late modernist houses in the fourth group (*typically* with fluid interiors and strong connections to the landscape). Rather than describing their typical traits in symbolic fashion, the region in the feature space provides a clear metric definition derived *from* the precedents. Although these are plotted in a continuum of other buildings, by removing all points not in these groups, the continuum separates into more clearly defined clusters with a wide margin between. The examples of museums and offices in section 7.3 also displayed this clear margin.

The setting of an archetype from such a clearly classified plot requires only the selection of the point that best describes the region of reduced feature space in which the examples of that style lie. Nearest-neighbour algorithms for analysis define prototypes as the generating points of the Voronoi tessellation that separates the classes (Figure 8.4, left) (Duda et al. 2001), but for the synthesis to follow in section 8.3 this would bias the result when these lie close to the edge of a cell. Also, for only two classes there are infinitely many such points and for only one class there is no separating boundary at all. In the nearest-neighbour algorithm, *proto*types precede the partitioning of the space, whereas the archetype should be defined after. The point that minimises the distance to all examples in the cluster is simply the mean,

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \frac{\varphi(\boldsymbol{x}_i)}{n} \qquad \forall\, \boldsymbol{x} \in S, \tag{8.1}$$

which can be applied as easily for two type-clusters or two thousand. This mean and the mapping to the reduced feature space

$$\Phi = \{\mathrm{z}_1, \ldots, \mathrm{z}_j\} = \varphi\,(\,\{\,\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_i\,\}\,), \tag{8.2}$$

where $j < i$, together constitute the archetype (Figure 8.4, right).

The lower dimensional feature space Φ that results allows a description of style that is convenient for analysis and measurement—in that any plan example can be evaluated, and compact—in that only a few dimensions need be used. Because most of the dimensions have been removed, the space itself comprises only those features that are relevant to differentiate examples of one type from another, and the mean point of each of the clusters above can be said to be the archetypal ideal of each group. Like any archetype it is a generalisation, and examples that fit it exactly may not ever be found in the real world, but it can serve as a model to clearly measure example designs and to guide the creation of new ones.

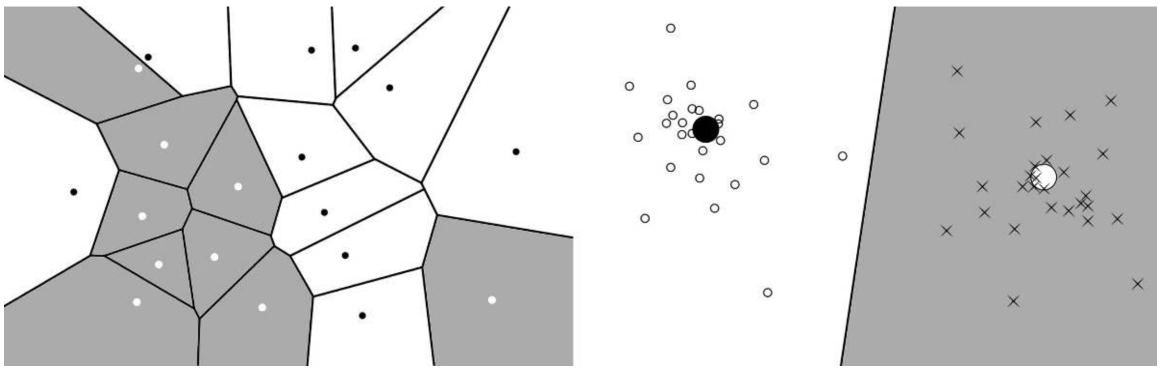


Figure 8.4. Left: Nearest neighbour prototype points define Voronoi regions of the original feature space. Right: Archetype points $\boldsymbol{\mu}$ at the mean of the examples define regions of the reduced feature space Φ.

8.3 Synthesis: Production of new designs

For most subtle distinctions, the 'family resemblance' being sought in the archetype is unlikely to be so objectively clear in the PCA subspace, which is determined completely by the overall distribution of the examples, and does not necessarily reflect preferences a designer might have about certain buildings. The archetypes are clear only if there is a wide margin between relevant clusters, but much less so if these happen to fall close together in the feature space, or overlap. In such cases the mapping to Φ must be refined so examples within a desired type are closer together, and well separated from other types. In this section more sophisticated classification algorithms are used in place of PCA to derive the features, and methods for improving stylistic fidelity will be investigated. The use of *supervised* learning with *labelled* examples will imply a reduced feature space that is not just statistical, but meaningful.

By doing this, the archetype becomes a means for communicate the complex meaning of the type it represents. It is the concept of a certain class of actual designs. If it can be shown to be recreated faithfully in different machines simply by sharing those designs, this concept will be communicated without prior, formal definition. This therefore sets up the falsification of Hypothesis B. The archetype will be used here to arrange desks in an arbitrary pattern communicated only by labelled examples, showing that the results of one system can be produced by another that is structurally different.

The declarative knowledge provided by the archetype is combined with a generative method to synthesize new designs (the *hybrid approach* to action, §3.4.3). A similar design domain to Chapter 7 is used, with a basic grid and paired units of solid desk with chair in front. Instead of optimisation, a simple generative method is used that could be seen as analogous to others, including shape grammars. The aim is to show the applicability of a general method before reintroducing the GA in the next chapter. The basic form of this algorithm is borrowed from the use of the grid model to represent the larger scale process of building aggregation in towns built over time without a central plan (Hillier and Hanson 1984), which results in configurations of built and public open spaces that might resemble the desks and chairs in an office. It therefore approximates a real-world instance of collective design, arising through the actions of a number of people considering their own needs individually, yet resulting in an emergent a global order.

8.3.1 A basic open generative system: building aggregation

The generative algorithm is historically related to the axial graphs used in section 8.2. Axial analysis was developed initially to analyse the variation in settlement layouts, particularly in French and English stylistic variants of what Hillier and Hanson (1984) termed the ‘beady ring’ form. This form itself was found to be the natural result of a very simple but open generative system of aggregation, proposed as an exposition of how many small decisions generate the recognisable order (Hillier and Hanson 1984). As growing hamlets reach a certain size, a regularity appears in the configuration of the public spaces between the buildings. It takes the shape of a rough ring of joined spaces around a central clump of buildings facing outward, and several inward-facing groups of buildings around the perimeter. While this pattern appears in each of the hamlets studied, and continues to grow with the town, it need not be the result of central planning, as the simple aggregation model shows.

A grid is used, with two types of objects represented in its squares: closed cells with an orientation defined by an entrance in one side, and open voids. These represented built and open urban space in the original model, but also take the form of desk/chair units. The minimal unit is thus made up of two face-wise adjacent squares, with a closed cell facing on to an open space in front. The model allows these pairs to aggregate such that each new pair must join its open cell to at least one other open cell already placed, and the closed cell does not join another closed cell only at the vertex. Other than these two rules, the position and orientation of each new unit is completely random. Each time the model is run, the emergent global structure forms that of the beady ring settlements studied, with a chain of open spaces onto which inner and outer groups of buildings face.

The grid is able to represent configurations of very different patterns by the choices made in aggregation, and so it can stand as an analogy to more sophisticated generative methods to demonstrate the principles of this chapter. For this reason and for its historical roots in the development of the axial analyses in section 8.2, the overall structure of this grid aggregation model will be used below to show that types can be learned from examples and used to emulate the original pattern.

8.3.2 Design as selection of affordances

In this case the aim is not to examine general properties of the beady ring. More important for the question of types are the specific differences between different layouts. Hillier and Hanson noted the differences between the beady rings of France, and their counterpart villages in England that tend toward a more linear arrangement. These cultural differences in global form are also a result of the same uncoordinated local actions over time, yet the decisions of building placement that lead to a circular or a linear arrangement seem somehow to have been instilled into the individual members of the culture, not as contrasting sets of rules but as contrasting choices of application of the same rule set. "It is not simply the existence of certain generators that gives the global configurational properties of each individual [design]. It is the way in which variations in the application of the generators govern the growth of an expanding aggregation." (Hillier and Hanson 1984, pp. 84-85)

The model was initially never run with anything but random choices, but these decisions of application can be made instead by the evaluation of each possibility against an ideal specified by the archetype, and the choice of that possibility which fits best. Learning to build to a particular type is possible because each time a unit is built, the configuration of the new cell pair relative to its surrounding neighbourhood gives an example of an ideal for an individual to follow, another example to build up the archetype.

For a designer, temporally embodied in the world (section 3.4.3), design involves a time-extended process of making and then examining from a different point of view. An individual agent in the context of this aggregation needs only to make a selection from a set of possible affordances. This supplements the simple aggregation process with an elective design step. Assuming every act of construction is selfish and uncoordinated*, the motivation behind the

* This is not to say that agents building a real city are necessarily selfish and not coordinated in any way beyond the interaction this model suggests. One of the limitations of the random aggregation model is that as it grows to much larger scales it begins to diverge from the structure of a real city. There are several likely reasons related to human planning abilities. Real builders can behave with future intent, anticipating possible development opportunities for themselves or others. These affordances are several

decisions of building placement would be to maximise some particular qualities considered to be important, such as direct access to the public space of the town and the economy of sharing a wall with a neighbour in the rules above. Each of these is an affordance of particular vacant building sites available at any given time. Rather than predetermining which of these qualities are most desirable however, they might be relative and change from culture to culture. For each unit built, the configuration of the surrounding neighbourhood relative to the cell pair gives an ideal example for an individual to follow—another example to define the archetype.

8.3.3 TWO TYPES OF AGGREGATION AS EXAMPLES

Two artificial norms were chosen to be easily distinguishable from one another, and a simple algorithm written to aggregate open/closed pairs of units in the manner of each. The first is a strict arrangement of straight rows, and the second is a random arrangement of units joined open cell to open cell (Figure 8.5).

To learn the two ideals, a classification algorithm was trained on the units as they were placed. Rather than using axial graphs (as section 8.2) or combinations of several graph types (as Chapter 7), this simplified grid model allows samples to be taken directly. Each time a new pair is placed in the plan, its relationship to the 7×7 cell neighbourhood surrounding the open half of the doublet is taken as its input. The 49 cells, each containing either a closed building (indicated by a filled square or 1), a public open space (a dot or -1) or yet unbuilt (empty or 0) are used as the computer's sensory experience of that particular built example.

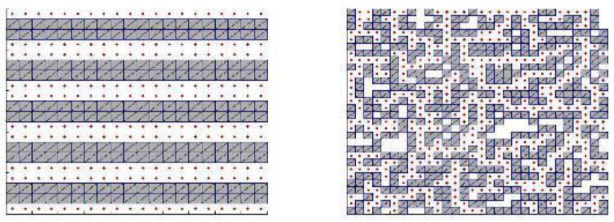


Figure 8.5. Two types: strict rows and random aggregation.

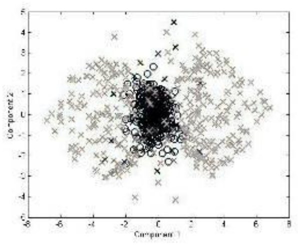


Figure 8.6. Examples from a 49-dimensional feature space.

As in the case of the plan graphs, these neighbourhoods are points in an initial feature space. Each unique example can be represented by a point in a 49-dimensional space, a 2-dimensional

steps removed from the immediate action, and may either be considered explicitly or embedded in received cultural building practice. This section acknowledges the latter possibility.

projection (by principal components) of which is shown in Figure 8.6. Neighbourhoods of the straight rows are indicated by '×', and the random style by '○' markers in the centre.

8.3.4 Learning and building to an archetype

Significant overlap is evident in Figure 8.6, and the means of each cluster are roughly in the same location, making the definition of the archetype impossible in the PCA feature space. The family resemblance expected of the two classes is not objectively clear in the data as it was in §8.2. A supervised learning algorithm was therefore used to provide the mapping $\varphi(\cdot)$ in Eq. 8.2. In the resulting space, the mean $\boldsymbol{\mu}$ of the cluster represents an archetype of that style to be used in a straightforward building algorithm: at every step a given number of positions and orientations are available to be built, and the decision is simply the act of choosing which one of these affordances is closest to this ideal:

$$\min(\| \varphi(\mathbf{x}) - \boldsymbol{\mu} \|). \tag{8.3}$$

Three experiments test the method of learning an archetype from examples and building to that archetype. The first (§8.3.5) tests basic morphic language competence of the machine to reproduce plans of the desired type from the archetype, with the hypothesis that clearer clustering in Φ will lead to a better resulting generation of the type. The second (§8.3.6) investigates whether the archetype can be made clearer by specifically labelling only the examples in that class, suggesting that a different feature space Φ should be used for each archetype. The third (§8.3.7) tests the hypothesis that the results of construction are independent of the choice of learning algorithm and particular representation of the archetype.

8.3.5 Clustering in a feature space and clarity of style: training by Support Vector Machine

The most crucial hypothesis to be tested was that the type can be learned at all, and that this would allow designs to be produced of a given type. It implies there should be a direct correlation between clear clustering in the feature space Φ and the strength of the type in the resulting design. A support vector machine (SVM) (Vapnik 1995) was used for the initial classification because its easily tuneable parameters allow its mapping tolerance to be adjusted to test this hypothesis.

SVMs operate by finding a maximally separating hyperplane between the two labelled classes in a higher dimensional representation of the input, and that representation is given by a non-linear function with a parameter that can be used to adjust the fit to the data—in this case the width of a Gaussian. Figure 8.7 shows the results for $\sigma^2 = 5$, 15 and 25 respectively. The SVM output is plotted in the left column with row examples to the left of random examples, such that the vertical axis represents the single dimension of SVM output. The effectiveness of the classification is indicated in the second column images by the shading of each sample, where samples near the mean of the rows type are shaded in light grey and those of the random type are black. Clearly there is a clearer classification in Φ for the higher values of σ^2.

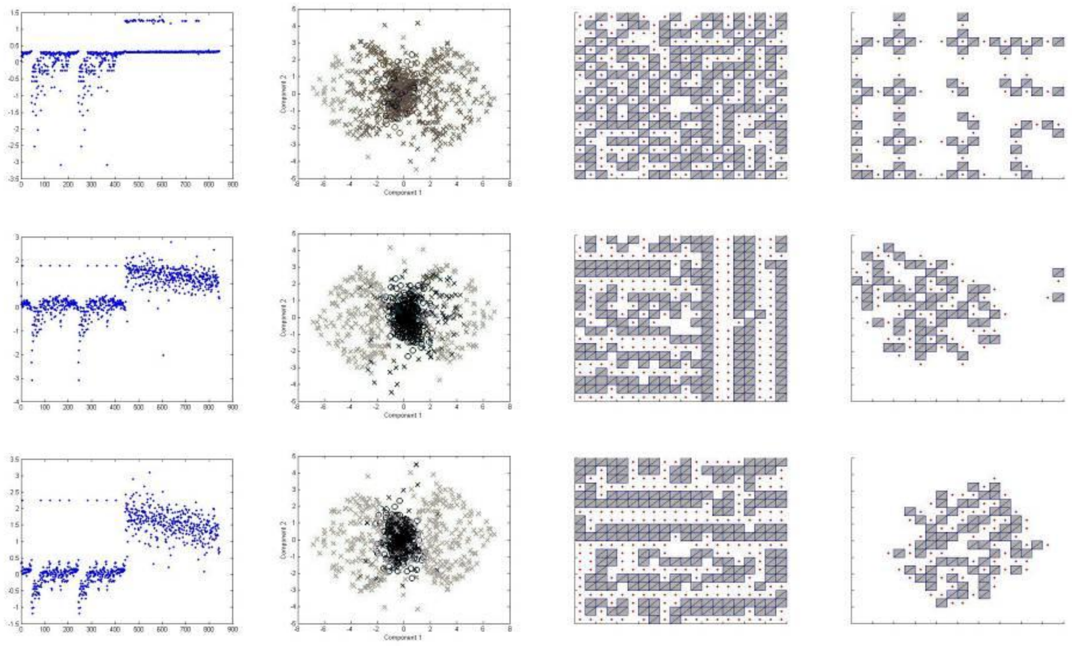


Figure 8.7. Building results for algorithms trained with a SVM: σ2 = 5 (top), 15 (centre) and 25 (bottom). The first image on the left shows the mapping of 900 examples against the vertical axis (Φ). The second indicates apparent membership in each cluster by the shading of the points. Resulting building patterns follow emulating rows, then random aggregation.

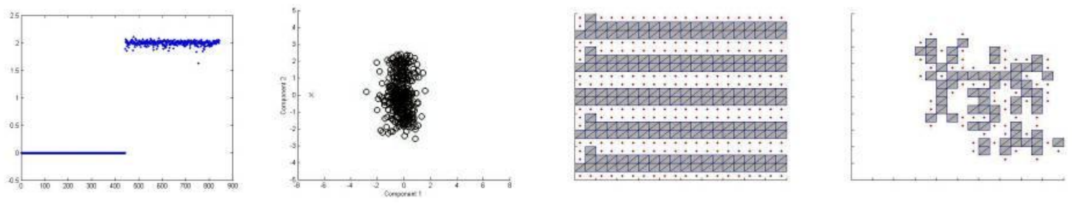


Figure 8.8. The same training on a set of 'ideal' examples.

It is evident from the results that as σ^2 increases, the cleaner separation between the two groups by the algorithm results in a clearer construction, as shown in the images to the right. At each construction step, the possible sites and orientations are evaluated by the SVM, and the one closest the mean of either type as learned is selected. The completed constructions over a period of time are shown, one emulating the rows style as learned and the other the random arrangement, and the overall patterns are most easily seen for $\sigma^2 = 25$, particularly for the straight rows.

The initial hypothesis is confirmed, but the separations in Figure 8.7 are never quite enough, and the classifier can only produce adequate rows with an artificially created set of 'perfect' examples of row neighbourhoods. These are all identical, so that each is exactly perceived as the ideal archetype, and consequently the perfect classification of the two groups results in a stronger expression of the type. (Figure 8.8.)

8.3.6 Clarifying the archetype feature space: training by neural network

The method thus far performed one analysis for the relevant components of all types, and so presupposes a common feature space Φ. It would yield appropriate archetype definitions if all types differed in the same features, and thus could be classified in the same space, but this is unlikely. It is against the 'family resemblance' of overlapping features that tends to describe a type. Rather than merely classifying two types, the benefit of the clear archetype in Figure 8 suggests the choice of a feature space fit to a single type yields stronger results. In this section a unique feature space is found for a single type by training a neural network to find a space in which the points are clustered closely together as differentiated from all others.

A neural network was used to learn the rows type only, with the random examples serving as mere background from which to differentiate the relevant features. A Feedforward Multilayer Perceptron (MLP) (Rosenblatt 1958) was used, with 49 input nodes corresponding to the state of the neighbourhood, 50 nodes in the hidden layer, and a single, linear output that rates each example in Φ. Training was conducted by exposing the network to 450 examples from each of the two test types and backpropagation of errors. Because the goal is to learn the features of the rows type only rather than to classify both, a variation on the typical error function was used. Normally, the error J in classification problems is a function of the difference between an output $\mathbf{z}$ and a specified target $\mathbf{t}$. As there was no need for a target for examples outside the style in question however, the target for the samples of the rows was set to 0, and the reciprocal of the error used for all other examples,

$$J_k = \begin{cases} \frac{1}{2}|\mathbf{z}_k|^2 & \text{if } k \in S \\ \frac{1}{2}\left|\frac{1}{\mathbf{z}_k}\right|^2 & \text{if } k \notin S \end{cases} \tag{8.4}$$

causing the error to fall as examples appear farther away. The advantage of this modified error function should be a large separation in Φ of all the random type examples and an output in Φ very close to 0 for most of the linear rows type. Unlike the earlier classification however, the random type examples may be scattered anywhere in Φ—their clear clustering is not relevant to the definition of the rows archetype.

Results of this type-specific feature space were superior to those of the SVM in figure 8.7. In Figure 8.9 each of the examples is shown as a single dot in the vertical axis Φ corresponding to the value of the network's single node output. After training, most of the first 450 examples along the horizontal axis (the row units) appear at 0, and most of the others (the random aggregations, to the right) as far away (note the extreme scale of the output axis). The resulting aggregation of open and closed cells produced by the placement algorithm very closely resembles that of the original rows from which it was trained.

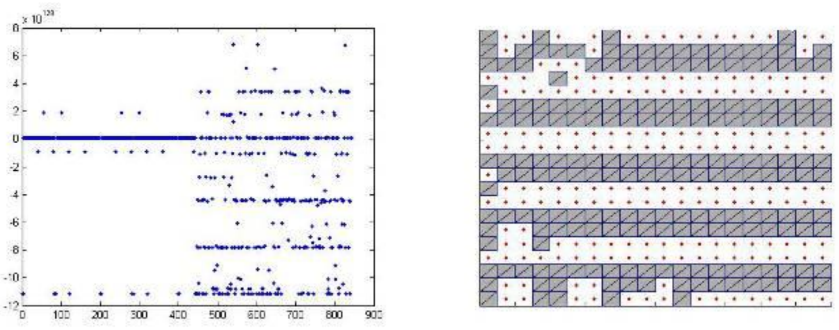


Figure 8.9. Training of a three layer network on the row samples.

8.3.7 Variations on the representation to learn the same type

Because the type is described by a feature space rather than symbolically, the actual method of feature space mapping in the archetype should be arbitrary. If the 'embodied' interpretation of Hypothesis B is false, and different processing rules can result in the same structure (§3.4.3, case iv), the choice of classification algorithm used to define the type should not matter. This section tests that it can be changed and still lead to recognisable output.

The results indicate that this is the case. One type can be described in many different ways, or feature spaces of different dimensionality. Figure 8.10 shows the result of several very different learning algorithms exposed to the same set of examples, each resulting in a very different mapping of features (Φ, left) but very similar overall construction of rows (right). First is a neural network similar to the one in Figure 8.9, except that only the nearest examples of the random style were used in training. Below this, a different technique is used to train the network: errors from both groups are measured from the mean, but rather than adding the weight updates at each step for the examples from the random type, they are subtracted. The last example is a differently structured network entirely: a Kohonen self-organising feature map (Kohonen 1982). The subtraction training and the Kohonen feature map were found to be the most successful at replicating the overall pattern for this test type.

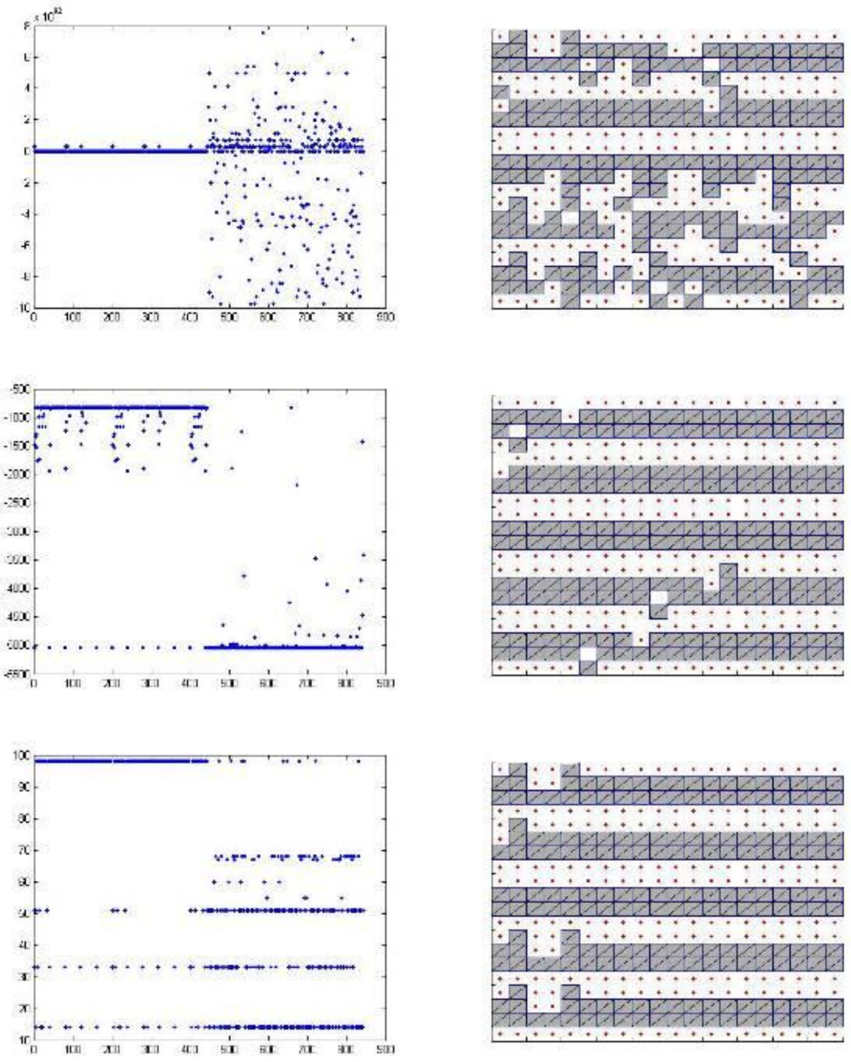


Figure 8.10. Three completely different algorithms (two double-layer neural networks and one Kohonen network) result in different feature spaces (left), but make similar evaluations and similar constructions.

The similarity of the final constructions indicates a type can be represented many different ways. Even with the constrained grid morphology of design space, there is a drastic difference in the feature spaces Φ (Figure 8.10, left). These networks have different structures and different training methods, but produce similar results to one another. Feature spaces may differ in detail and even dimensionality, as each of the algorithms is capable of mapping to an arbitrary number of dimensions. The only necessary common process is that each forms an archetype in its unique feature space based on the examples of that group.

8.4 Chapter review

The archetype provides a clear and precise definition of type. This agrees with the notion of a fluid concept that is always subject to change, and therefore suited to a flexible representation. What is suggested here is that it can nevertheless be accurately represented, communicated and reproduced. This chapter has presented an algorithmic method for both deriving a stylistic definition automatically from examples, and using it to generate new designs. Architectural examples were used, and were investigated primarily in terms of their spatial features, but it is intended as a general model in that other forms of input and classification algorithms may be used. Likewise, axial analysis and the aggregation model are not essential to the method, but the principles of feature space reduction and archetype should apply to a variety of analysis and synthesis techniques.

The concept of the archetype proposed is of a defined ideal and of a space in which to measure example designs. As such, archetypes do not correspond to individual buildings in all their detail, but contains only the features most relevant to define that type. Instead of clear symbolic definitions, one can measure an example's similarity in degrees, on a continuous scale. The complexity of the feature space derived from a neural network or similar algorithm allows it to refer to many instances related only by a 'family resemblance' just as words in a language refer to similarly overlapping concepts (Wittgenstein 1958, §67). Boundaries are indistinct between archetypes, but they are no less useful for that reason. This results in a definition of type that is flexible, can evolve, and is based on examples as per the criteria in section 8.1.1.

The reflective (§1.1.2) process of design has two requirements—the capacity to change the frame of reference through which decisions are made, and an ongoing process of engagement of the designer with the developing design. The archetype provides the first of these by being reinterpretable from new sets of examples at any time. The second is an embodied approach to cognition (§3.4.3) that operates in this case through an elective method of choosing affordances.

The archetype is in some sense a representation, but it is an ‘action oriented’ representation (Clark 1998) that must be embodied. It can not be extracted from the generative algorithm, but is valid only in the context of an agent’s design process. Examples convey a meaning, but only in the context of how they are employed within this (after Wittgenstein 1958, §71).

With respect to the social aspects of human perception and design, it is evident from a comparison between §8.2 and §8.3 that some archetypes may be immediately obvious in the data itself, while others may require significant training to extract. This difference corresponds roughly with Clark and Thornton’s (1997) distinction between type-1 and type-2 regularities in data (§3.2.6). Both situations certainly occur in human understanding of types, styles or design concepts, but the more complex type-2 is likely most common, and thus the cultural means of transmitting relevant structures are necessary. Labeling of plans as members of the type has here been a rough substitute for a wide range of social practices that would normally convey the meaning of the type, and the structure by which it would be learned by a designer. The details of this mechanism with respect to the social practice of collaborative design will be further explored in chapter 10.

Chapter 9: Optimisation to generate plans

Summary: A full demonstration of the above methods is made using a genetic algorithm (GA) to generate office desk plans. The algorithm produces plans of a particular desired form without being given geometric information or a model plan, and thus falsifies Hypothesis A. The fact that communication of the archetype is through exposure to example data only falsifies Hypothesis B.

The previous three chapters have outlined the domain and approach to the workplace layout problem. The development of the initial tool (Chapter 6) highlighted the need to look beyond efficient packing to the more difficult aspects—social and spatial relationships that appear ill-defined. The following two chapters each developed a method by which these relationships could be defined, numerically and computationally for the machine. Spectral embedding was proposed as a generic representation to allow the machine to evaluate existing plans (Chapter 7), and an *archetype* as an 'action oriented' or embodied representation was derived from this (Chapter 8). This can serve as a goal in design and can be communicated implicitly via a set of plans provided by the designer. This chapter combines these two methods to generate office plans with desired structure.

Hypothesis A will be falsified in this design domain by showing that the algorithm can produce plans with a given structure, without modelling the underlying causes of this structure. The archetype will be used instead to extract a high level approximation of this from existing precedents. The form of this approximation (topological) will differ from that of the structures to be reproduced (geometrical). Hypothesis B will be falsified by communicating no processing rules for this structure, and avoiding explicit, a priori, formalised knowledge representation. The archetype will be communicated only by the precedents themselves.

This chapter investigates the method in three stages. The possibility of learning arbitrary structure is addressed first (§§9.1–9.3). The ability of various algorithms to learn specifically from plans is tested, and the results used to suggest guidelines for implementation. Second (§9.4), optimisation places demands on the archetype feature space in addition to the improved classification sought in Chapter 8, so refinements to the feature space are explored to improve the search. Finally (§9.5), the combination of learning and optimisation is tested by evolving plans to achieve a similarity to existing sets. Here, a GA is used to generate novel designs not from matching pre-existing plans as in chapter 7, but to goals learned by the machine based on prior, labelled or classified examples of previous designs. The method is similar, except that the pre-existing prototype plans used as targets in the objective function are replaced by archetypes as defined in chapter 8.

These tests will demonstrate competence in a morphic language (Hillier and Hanson 1984), by which an 'inverted genotype' in the form of an archetype will be extracted from the design examples, and used to produce new ones. This learned archetype itself is impossible to describe (either to impose or recognise) explicitly, and thus as inaccessible directly as the processing rules of spoken language (§3.4.1) or the truth of a theory (§1.2). Wittgenstein (1958, §1.2) suggests the learning of a word in a language game is verified by a person's use of it; Popper (1959) similarly requires intersubjective verification in testing a theory. In design we are also concerned with action in the world—the thing that is actually produced is of primary concern, rather than the accuracy of a given representation of the world (§3.4.3). The test for competence must therefore be based entirely on an evaluation of the patterns it produces.

The 'wicked' social and spatial structures that are most relevant are by definition resistant to clear definition, making a test of the method by verifying these possible only via actual observation in real offices designed by the method. As such, arbitrary features that are clearly recognisable will be used to define the archetype; the test will show that these can be produced in new designs. Easily recognisable, geometrical patterns will be used[*]. Testing of plans will follow the following steps:

1. Set *original, arbitrary geometrical criteria* to represent the archetype.
2. Select sample sets of plans based on whether these criteria are present.
3. Use the *samples only* to communicate this archetype to the machine.
4. Produce new designs (the algorithm will be specifically prohibited from producing exact matches to the original samples).
5. Evaluate the new designs against the *original set criteria*.

The degree to which the structure proposed in (1) is present in (5) will demonstrate the competence of the method.

[*] As noted (Chapter 7, intro.), the fact that the underlying graph spectral representation is more suited to the socio-spatial patterns ultimately desired makes this is a valid test of the method. This suitability has since been demonstrated by the ability of unsupervised algorithms to make realistic typological distinctions of real buildings based on graph spectra (§§7.3 & 8.2).

9.1 Sample plans for learning

The arbitrary features that determine feature space Φ are to be determined by the design problem at hand. As a simple case, one might desire a general arrangement of groups of regular, outward facing desks as opposed to linear rows, inward facing groups or random clumps, not unlike the distinction between the two types in section 8.3. The size of these groups may be irrelevant, as may be their position relative to one another. Such a design goal is admittedly straightforward to describe, and to evaluate visually, but does not appear explicitly in the spectral analyses. A group of existing plans may be selected that have this arrangement of desks, along with another that has an undesirable arrangement of rows. The use of inductive machine learning to construct the feature space from these groups is investigated in this section, and methods proposed to improve learning in practice.

9.1.1 Plan features to be used in evaluation

The advantage of a user trained feature space is that it can be tuned to differentiate the features deemed appropriate by the designer, approximate the designer's own judgements, and thereby be of use both as an evaluation tool and in optimisation. It is assumed that the designer must be free to decide what features and classifications are relevant, that is the space must be capable of representing any arbitrary dimensions chosen.

As a source of plans for the experiments in this section, the *ClassPlan* data set was created manually from 128 permutations of five parameters. Three affect local desk arrangement:

parameter 1: convex groups or linear rows
parameter 2: groups aligned or staggered
parameter 3: local length of desk groups

and two represent global configuration:

parameter 4: space filled with desks or partially open
parameter 5: global plan shape and size (12 × 12, 24 × 12, 12 × 24 or 24 × 24)

All five parameters each take values in the range [-1, 1] and are illustrated in figure 9.1; the entire *ClassPlan* set is shown in Appendix B. In section 9.2, the GA will use genotype representation C (Appendix A) to evolve new plans to match the archetype derived from these, so all *ClassPlan* examples were intentionally created always to slightly differ from those which can be created by this genotype.

The features of these five variables are arbitrary and mutually independent. The data set is plotted in figure 9.2 against its first and second principal components, with each of the classes determined by these variables indicated in a different colour. Clearly the classes indicated by variable 1, convex groups and linear rows, are easily differentiated along the first principal component, but the other four variables are not. These other variables therefore do not have any clear, specific relationship to the inherent features captured by the graph spectra—it is what Clark and Thorton (1997) term a type-2[*] relationship—but this may be learned by using these classes as target, or output dimensions.

For several of the tests a more continuous sequence of plans is desired. A second set, the *SequencePlan* set (figure 9.3), was created based on the variables 3 and 5 above, which can be broken down into finer increments than the *ClassPlan* set. Four basic plan types were created, different from those in *ClassPlan*, and examples were created in 10 different overall sizes (from 12 x 12 to 30 x 30) and 10 different desk group lengths (generally 1 to 11). Each plan type, therefore, was varied in both global features. A full description is given in Appendix B.

[*] The type-1/type-2 distinction was actually proposed with respect to supervised, rather than unsupervised learning. The use of type-2 to describe a pattern that is invisible by an unsupervised algorithm but which will later be revealed by supervised learning gives the impression that the distinction maps easily on to the unsupervised/supervised learning distinction. This is not the case. It is used here in part because Clark and Thornton (1997) suggest that social and cultural structures provide the codings to transform type-2 to type-1. The labels to be used here should also be considered socially transmitted.

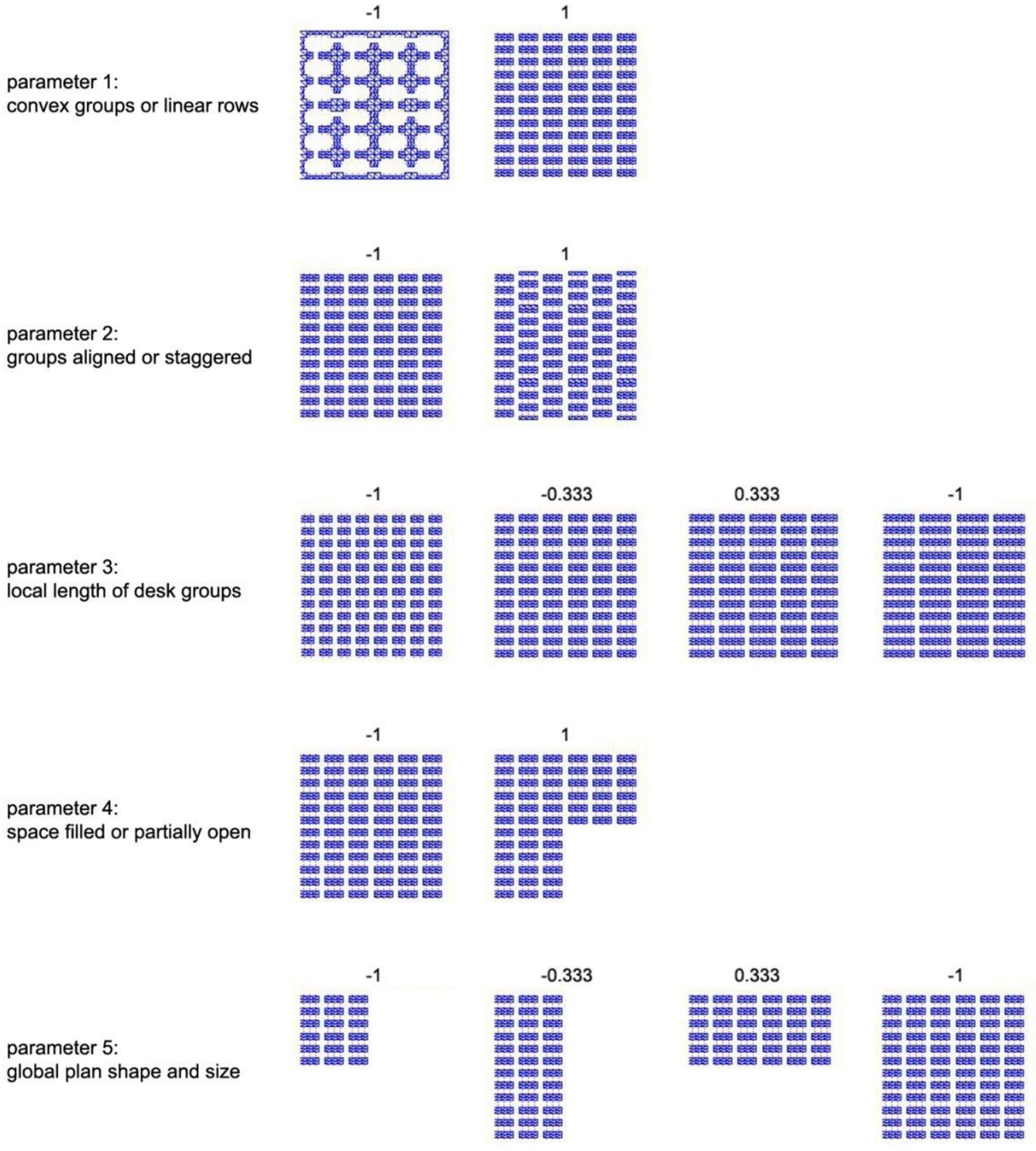

-1
1
parameter 1:
convex groups or linear rows
-1
1
parameter 2:
groups aligned or staggered
-1
-0.333
0.333
-1
parameter 3:
local length of desk groups
-1
1
parameter 4:
space filled or partially open
-1
-0.333
0.333
-1
parameter 5:
global plan shape and size

Figure 9.1. The *ClassPlan* set is constructed of all possible permutations of five parameters. There are 128 plans in total.

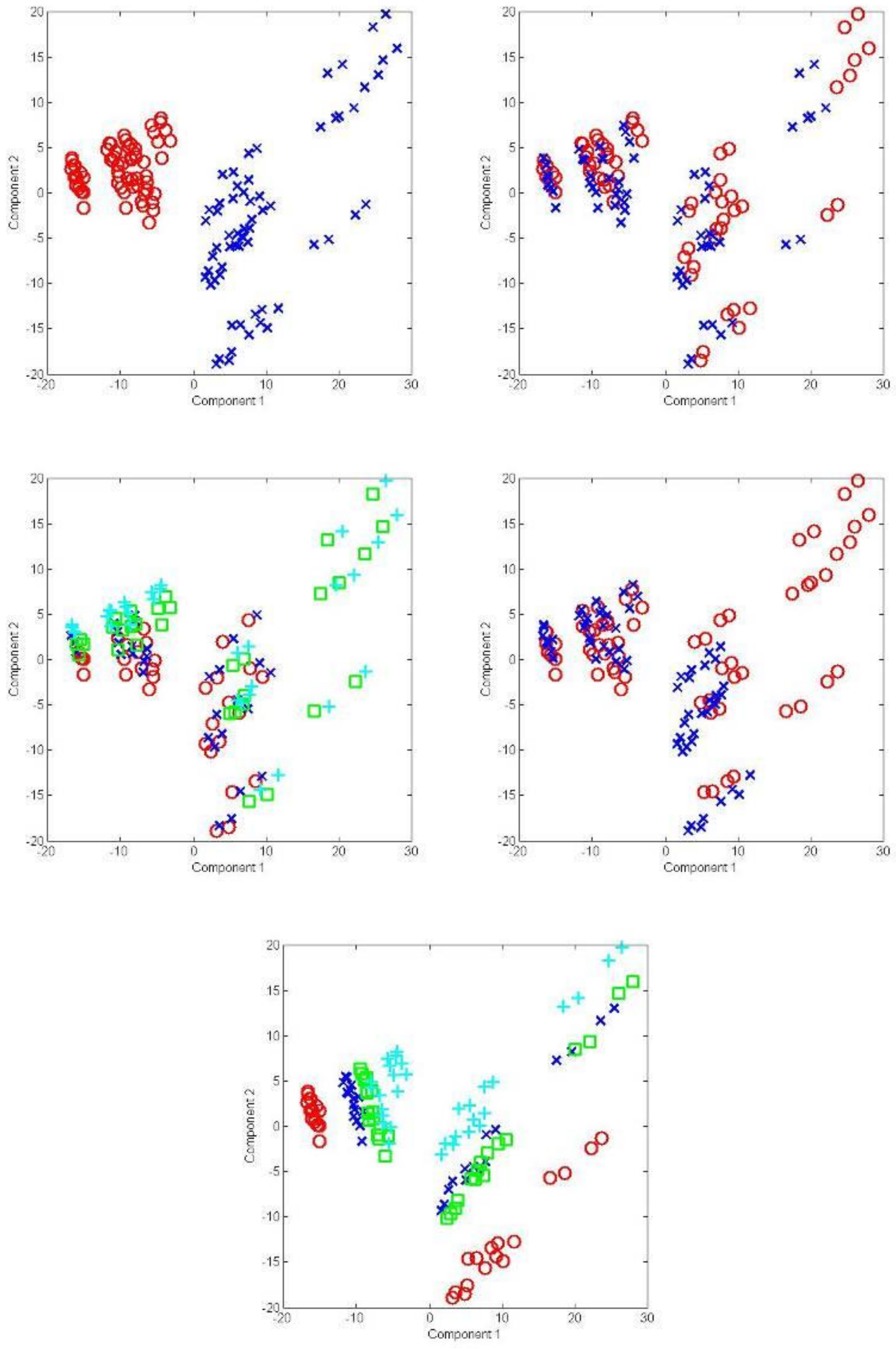


Figure 9.2. PCA of all *ClassPlan* examples. Parameter values are indicated by colour.

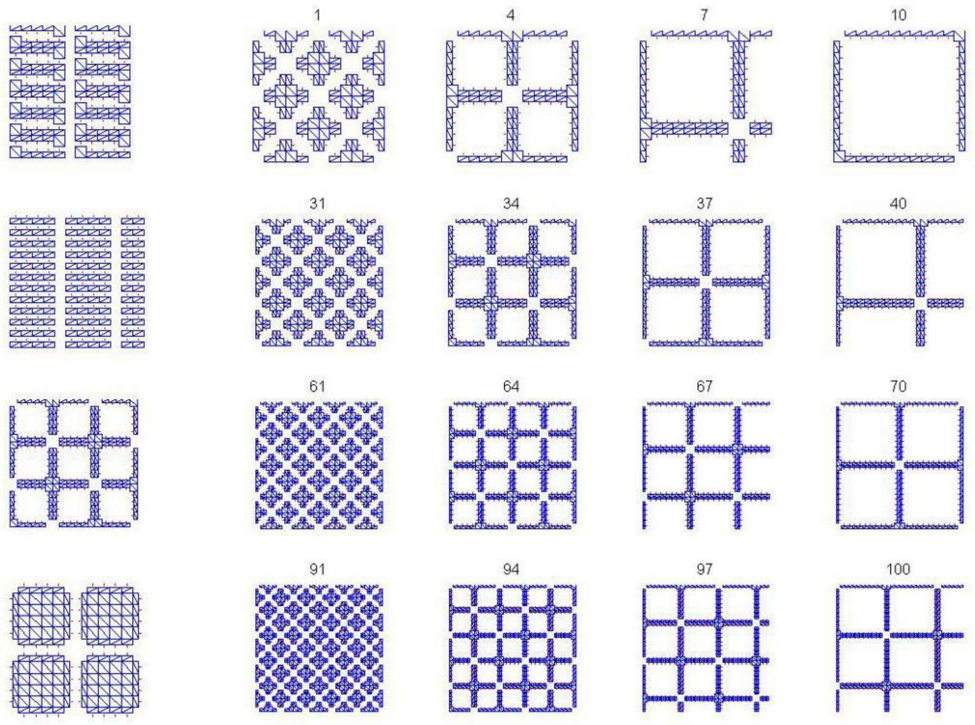


Figure 9.3. The *SequencePlan* set consists of four basic plan types (shown left) which are modified by two parameters: group size and overall plan size (samples shown in the grid at right). There are 400 plans in total.

9.2 Pre-processing the spectrum for machine learning input

Several options are available in terms of how the spectral data is presented to the machine learning algorithm. One was introduced in chapter 7, where three types of graphs were defined as initial methods of deriving the graph from the base plan, and it was observed that the use of a combination of these was necessary to capture and replicate features of target plans. In all cases, a concatenation of spectra from axial, boundary and desk graphs (section 7.4.1) were used in the experiments to follow.

Two subsequent choices are investigated in this section. First, after generating the graph, there is the question of how the spectral feature vector is defined from this—a choice between using the adjacency matrix or the Laplacian, and how to ensure all vectors are of equal dimensionality. Then there is a question of pre-processing this vector for presentation to the learning algorithm, in which normalising the mean and variance of each input is proposed. Both of these choices are tested in terms of how they affect learning of the above plan features.

9.2.1 Choices of Graph Spectra: adjacency vs. laplacian

As an alternative to the adjacency matrix, the Laplacian is often used to represent the graph. Spectra of the Laplacian or its derivitaves have been shown to be superior to the straight adjacency matrix for graph representation and classification (Zhu and Wilson 2005). In particular, they have been shown to have a higher correlation to graph edit distance (the number of edges deleted or added to change one graph to another) and result in fewer cospectral graphs (graphs which have identical spectra). Where the elements of a diagonal matrix $D = diag(deg(V_1), deg(V_2), \dots deg(V_{|V|}))$ indicate the degree of each of the nodes and the Laplacian is constructed from this and the adjacency matrix A (Eq. 7.1), such that

$$L = D - A. \tag{9.1}$$

The spectrum of the graph is found by taking the eigendecomposition of the matrix representation. The eigenvalues λ for A are given by solving for

$$A = \Phi\Lambda\Phi^T, \tag{9.2}$$

for the adjacency, or

$$L = \Phi\Lambda\Phi^T, \tag{9.3}$$

for the Laplacian, where the matrix $\Lambda = \text{diag}(\lambda^1, \lambda^2, \dots , \lambda^{|V|})$ contains the eigenvalues as diagonal elements.

9.2.2 Choices of Graph Spectra: truncation vs. interpolation

The spectrum of the graph is defined as the set of ordered eigenvalues of this matrix, sorted such that any isomorphic graphs will have the same order of eigenvalues. Sorting by value, such that $\lambda^1 < \lambda^2 < \dots < \lambda^{|V|}$, the full spectrum will be a vector composed of the $|V|$ eigenvalues:

$$S = (\lambda^1, \lambda^2, \dots , \lambda^{|V|})^T. \tag{9.4}$$

The full graph spectrum has $|V|$ values, or the number of nodes in the corresponding graph, but a feature vector of constant dimensionality is required even when comparing different sized graphs. To ensure a feature vector of constant dimensionality *n*, the spectrum may be derived from this either by truncating to the *n* largest values

$$S' = (\lambda^1, \lambda^2, \dots , \lambda^n)^T, \tag{9.5}$$

or by performing a cubic spline interpolation of the $|V|$ values in S to a new vector

$$S' = (l^1, l^2, \dots , l^n)^T. \tag{9.6}$$

and thus all spectra may be mapped in the same n-dimensional space (8.2).
Luo et al. (2003) suggest truncation as in Eq. 9.5, however the similarity of spectra as changes are made to global configuration was observed in §7.4.2 (figure 7.12). This would suggest that interpolation is also appropriate. The two methods are examined below with the spectra of both the adjacency matrix and the Laplacian.

9.2.3 Correspondence between parametric and feature space

The first test simply measures the fit between measures in the feature space described by the choice of spectral vector and those in an artificially created model. Zhu and Wilson (2005) measure the effectiveness of spectral representations by comparing sequential distances between spectra with known edit distances when creating the graphs themselves. A similar method is adopted here in a comparison of the spectra of plans that differ by a standard amount. Both global and local changes are considered. The *SequencePlan* example plan set was used (appendix X), consisting of four groups of 100 plans. Each group varies monotonically in ten steps of two parameters: one local (number of desks to a group), and the other global (overall plan size). Each of these four plan sets has a different general configuration of desks.

The distances in parameter space are taken as the number of additions to an initial plan: with the linear addition of between one and nine extra desks along a group, or the increase in plan size in increments of two units from 12 x 12 to 30 x 30. Spectral feature spaces that best describe the incremental progression should be those in which the Euclidian distances between spectra have the same linear progression – i.e. the ideal output in the plots in figure 9.4 will be linear.

The spectra for each of the three analyses (axial, boundary and desk adjacency) were either interpolated or truncated to 100 dimensions, and then concatenated to form a vector of 300 dimensions in total. Spectra from both adjacency matrices (Eq. 9.2) and Laplacians (Eq. 9.3) were compared. Each plot in Figure 9.4 displays these distances for each of the four plan types in the set (in blue, red, green and black) for a single plan progression (solid line) and for the mean over ten of a similar type (dotted line). The mean relative deviation was measured for each. Progressions are shown for both the global and local parameters.

All four spectral methods display roughly linear approximations to the increasing distances in parameter space. The choice between the use of the adjacency matrix or the Laplacian for the spectrum does not appear to make a great deal of difference, although the Laplacian slightly outperforms the adjacency in three out of four cases.

When global changes are made (plan sizes), the interpolated spectra perform reasonably well as suggested by the tests in section 7.4.2 (figure 7.14), however local changes (desk groups) are represented far better by truncation. Truncation yields a lower deviation than interpolation in each case.

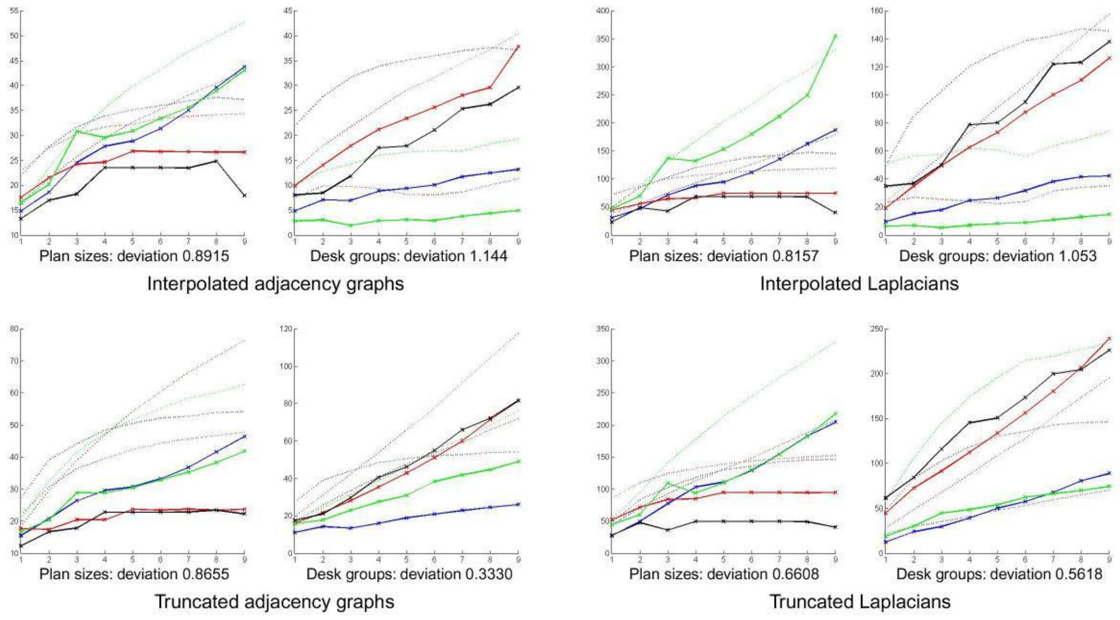


Figure 9.4. Incremental plan changes are plotted as distances in four spectral feature spaces.

9.2.4 Minimal errors in learning

The second test measures the suitability of the feature space to machine learning and classification. The reduced feature space mapping $\Phi(\cdot)$ in which to measure the plans:

$$\{z_1, \ldots, z_i\} = \Phi(S') = \Phi(\{l^1, l^2, \ldots, l^n\}). \tag{9.7}$$

To do so, a support vector machine (SVM) was trained on each of the five parameters in the *ClassPlan* data set (appendix B), using a one dimensional output z to match the parameter setting. Targets t of [–1, 1] were used for the binary parameters, and [–1, –0.33, 0.33, 1] for the others. Error was calculated as the squared difference between the SVM output **z** and the original parameter setting **t**.

The SVM was trained with varying values of σ^2, and the error estimated by 5-fold cross validation (Reich and Barai 1999). The results are displayed in figure 9.5 along with the mean error over all five data sets shown in dotted line. The minimum mean error is indicated by a '○' in each plot.

There is very little difference in the overall error, and all four methods provide spectra that enable the SVM to learn the classes to a reasonable degree of accuracy (the error is a sum over 128 examples). In contrast to the results of the measurement of Euclidian distance, the performance with the interpolated spectra is marginally better at its optimum with error rates of 5.4 and 4.8 for interpolation and 4.6 and 7.8 for truncation.

Of the interpolated spectra there is little difference between the use of the two matrices, but the minimum mean error is slightly less for the Laplacian, and the error rate on the worst classified data set is lowest for the interpolated Laplacian spectrum than for any other method. The use of the Laplacian has also been shown more effective than the adjacency by others (Zhu and Wilson, 2005). While compelling arguments can still be made for truncation, particularly with regard to the deviations in section 9.2.3, the interpolated Laplacian was selected for general use throughout the following sections.

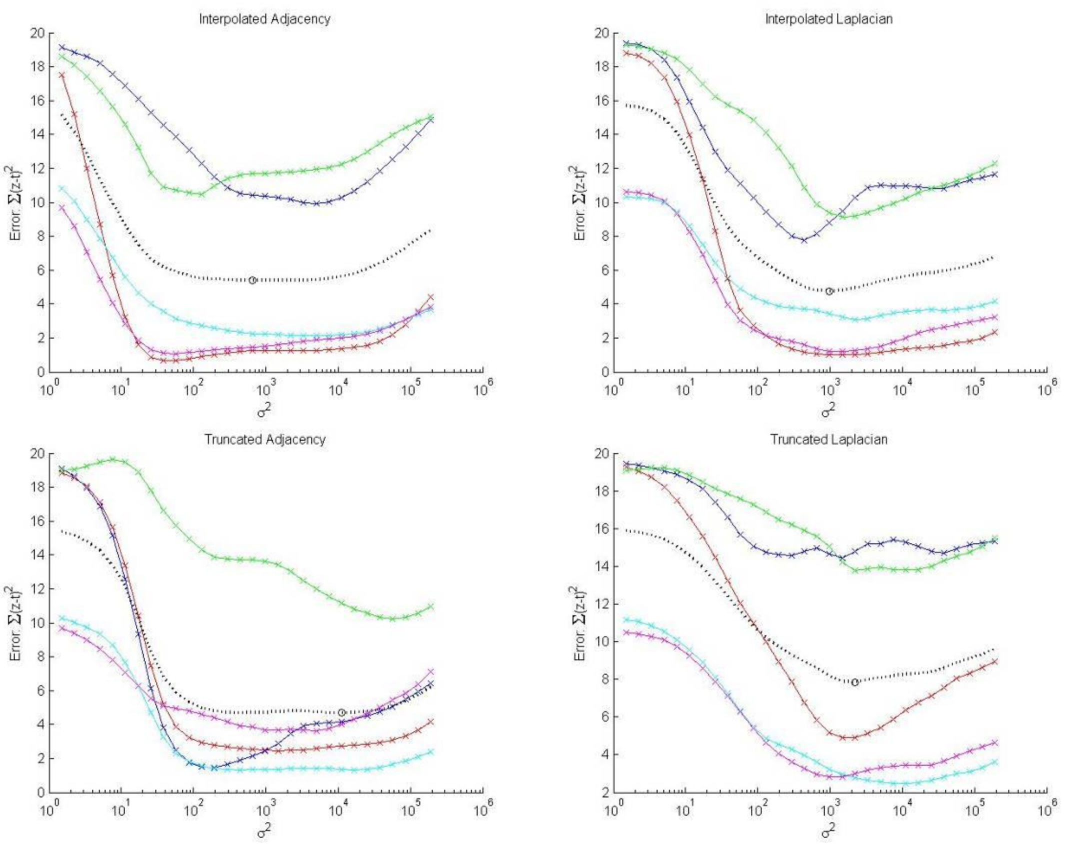


Figure 9.5. Error estimates in classifying each of the parameters in *ClassPlan*. Error is not binary classification, but total real valued difference from target value $\Sigma(z - t)^2$.

9.2.5 Input normalisation

When the units used in determining the input result in values that differ drastically in magnitude from one input dimension to the next, the training of the learning algorithm is driven mainly by the greater dimensions while the smaller valued dimensions are largely ignored. Input scaling can be employed to ensure a standard mean and variance in the data across all inputs (Bishop 1995; Duda et al. 2001), as was found necessary for VGA measures in section 7.3.1[*]. Cilliers (1998, p 75) explicitly warns against pre-processing in connectionist learning of complex problems, specifically because the data reduction often employed in cutting down the number of input dimensions may remove relevant features and build in an unwelcome bias. In this case, however, a pre-processing step of normalising input dimensions does not reduce input dimensions, but effectively increases them by eliminating the bias that would otherwise weight neural processing toward only the extreme values in the spectrum.

[*] The same is also recommended for output in regression problems (Bishop 1995), but the targets that have been chosen as for a typical classifier [–1,1] already fit this requirement.)

In the case of the graph spectra all input dimensions use the same units (the eigenvalue λ, or the interpolation from the spectral set *l*), but because of the ordering of the eigenvalues within the feature vector the first inputs within the spectrum are all negative with high variance, those in the middle near zero with little variance and the last all positive with high variance. Feature scaling can therefore be used to standardise the data set before learning.

Before training, the mean (μ) and variance (σ^2) of each dimension in the input set are determined and used to create a pre-processing weight and bias for the inputs

$$w_i(i) = \sigma_{required} / \sigma(i) \qquad (9.8)$$

$$b_i(i) = -(\mu \times w_i(i)) + \mu_{required} \qquad (9.9)$$

Using these to scale the initial training set

$$\mathbf{x} = w_i\, \mathbf{x} + b_i \qquad (9.10)$$

results in a new set in which the mean of every input is $\mu_{required}$ and the variance is $\sigma_{required}^2$. The values for w_i and b_i are saved for use with new data points.

The same five data sets used above were classified first with the inputs not scaled, and then scaled as above. The mean errors of each class are shown in table 9.1 along with the overall mean of the five sets. The classification error drops by approximately 1/3 when input scaling is used, indicating that the pre-processing step of input normalisation is a significant advantage for spectral input.

Table 9.1. Classification errors for each class using unscaled and scaled inputs.

	Inputs not scaled	Inputs scaled
Class 1	0.177	0.031
Class 2	0.436	0.419
Class 3	0.469	0.303
Class 4	0.250	0.185
Class 5	0.300	0.134
mean	0.326	0.214

9.3 Optimal learning algorithms and settings

All members of the *ClassPlan* set were used in training several algorithms to test whether the classes themselves were inherently learnable, and then to determine which learning algorithm and settings should be used.

9.3.1 Learning of the various classes

The algorithms were trained and validated repeatedly by single cross-validation (Reich and Barai 1999; equ'n 5.8), and the number of misclassified plans noted in the table below.

The feature space mapping Φ(·) was created in which to measure the plans as in (8.2), this time using either a SVM, or one of four feed-forward multilayer perceptrons (MLPs). The algorithm, was trained to classify two sets of 64 plans, one consisting of convex groups with a single target output of z = –1, and another of linear rows with a target of z = 1. The resulting mapping places the *n*-dimensional spectrum *S'* in a one-dimensional feature space Φ in which all convex group plans are clustered near a single point regardless of their other features of size, etc. The major consideration in the configuration of the network is the size of the hidden layer, which determines ability to discriminate or generalise, and was found to affect classification accuracy by up to 15% between 10 and 120 nodes. A hidden layer of 20 nodes was found by cross-validation to be the optimal configuration for classifying the plans in *ClassPlan*, and was used for all subsequent experiments.

Table 9.2. Misclassified plans for each class using different learning algorithms.

	SVM	MLP 120	MLP 100	MLP 20	MLP 10	mean
Class 1	0	0	0	0	0	0 (0%)
Class 2	30	24	25	29	22	26 (20%)
Class 3	13	7	9	6	14	10 (7.7%)
Class 4	39	35	34	20	25	31 (24%)
Class 5	2	16	11	7	8	9 (6.9%)
Total	84	82	79	62	69	

There is not a major difference in the learning ability of the two algorithms. The use of fewer neurons in the hidden layer was found to be beneficial. The MLP was found to outperform the SVM somewhat, and the MLP with 20 hidden layer neurons was considered to be an acceptable and fast choice for use in the GA.

In all cases classes 2 and 4 posed the greatest difficulty for the algorithm, followed by classes 3 and 5. Class 1 was trivial to learn, and no misclassifications were made by any classifier. While this clearly indicates that the global features of plan shape and size are more difficult for the learning algorithms to generalise, the worst of these was still learned by the best perceptron within a 23% error.

An analysis of the plans in which classification errors were made indicated no significant overlap in the error sets of all the runs. In the plot in figure 9.6, the errors of the 100, 20 and 10 hidden unit MLPs are shown against the ordered errors of the 120 hidden unit MLP, and very little correlation is visible. That is, the errors made were simply a result of the particular generalisations made in training and not inherent in the plan sets themselves.

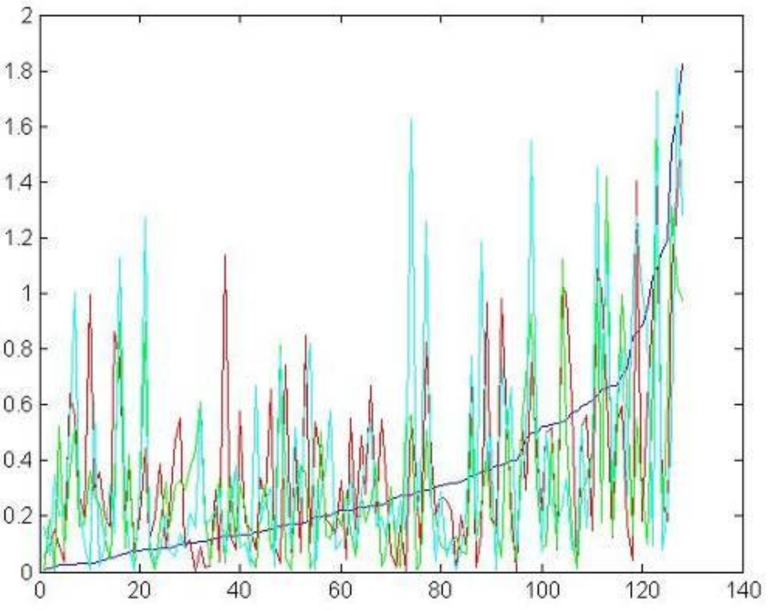


Figure 9.6. Errors between runs show little correlation with particular samples. Errors from 100, 20 and 10 hidden unit MLPs (with apparently random fluctuations) are shown ordered against those of a 120 hidden unit MLP (the blue line rising monotonically from left to right).

9.3.2 Overfitting and validation

The training of a MLP to a given data set minimises the overall error to the given examples, but can be in danger of overfitting to these particular examples and thus losing the generalisation useful for classifying new data. This can be tested by determining the error on a hold-out validation set as described in (§5.2.9), known to be a conservative estimate of true error (Reich and Barai 1999). As the MLP is trained over time the validation error initially decreases along with the training error, but then begins to increase when overfitting occurs. If this is likely, the training of the network can be stopped at the minimum of the validation error rather than the convergence of the training error (Duda et al. 2001), but in practice this can be difficult as learning rate settings any larger than the optimal η_{opt} could lead to false minima. A 20 hidden layer MLP was trained with a 20% hold-out validation set (26 samples) and the validation error plotted over time (figure 9.7) to compare with the training error for the five data sets.

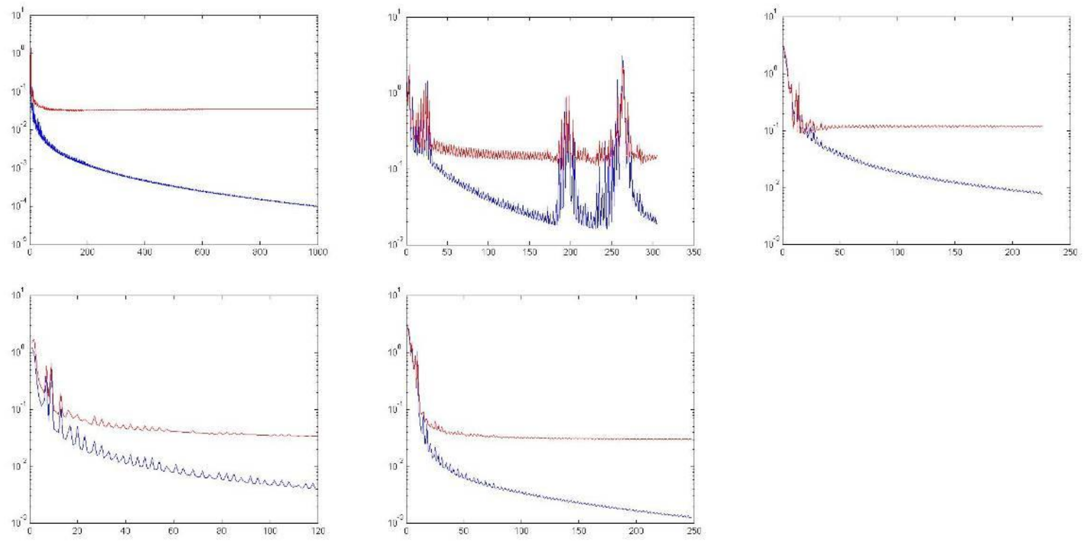


Figure 9.7. Training (lower blue line) and validation (upper red line) errors plotted over time.

In the third set there is a very slight increase in the validation error after an early minimum, but in most cases the validation error continues to decrease along with further training, although at a reduced rate. Overfitting would be more likely to occur with increasing numbers of hidden layer neurons as the flexibility to learn arbitrary functions is higher, as it would be for data sets which are more difficult to learn. Based on the set of plans tested and a MLP with 20 hidden units the danger of overfitting appears to be unlikely.

9.3.3 Learning rate

The MLP training performs gradient descent in the training error J. The learning rate η determines the size of the step, with rates below the optimal slowing convergence and higher learning rates causing the MLP to overstep the minimum error. Duda et al. recommend an initial learning rate for many problems of $\eta \approx 0.1$, however in tests of the algorithm on the plans the optimal learning rate was found to be far smaller: $\eta \approx 0.0005$ (figure 9.8).

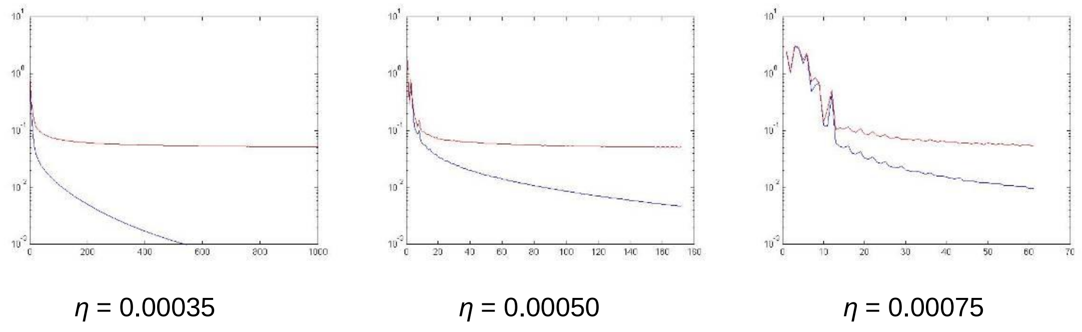


Figure 9.8. Training (blue) and validation (red) errors for different learning rates.

9.3.4 Hints and training on multiple features

In the above examples all classes were learned independently of one another, however the use of multiple output dimensions would allow all five classes to be learned simultaneously. In some problems it has been shown that hints, or the use of ancillary features during training, can improve classification accuracy (Abu-Mostafa 1990). In this method, output units are added to address a related classification problem during training, and these are then discarded for classification. This method could be used for any sets of correlated features.

The simultaneous training of all class dimensions together was compared with that of each dimension independently. A total of 100 hidden layer units were used for each test, together in a single network for the case of the simultaneous learning, and in five separate networks of 20 hidden layer units for the independent learning. The mean misclassifications of five runs of cross validation as in (§5.2.9) are shown in table 9.3.

Table 9.3. Misclassifications for training all dimensions simultaneously compared with training each dimension separately.

	all 5 dims together	Each dimension
Class 1	0.2	0
Class 2	22.0	22.8
Class 3	9.8	8.0
Class 4	31.2	25.0
Class 5	13.2	7.4

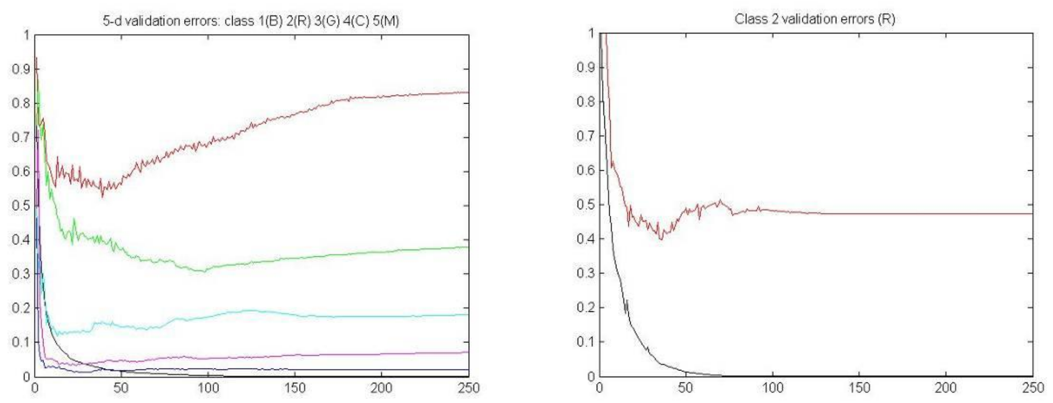


Figure 9.9. Validation errors of each class with simultaneous training (left) and class 2 when trained alone (right). Training alone reduces the minimum error and drastically reduces overfitting.

The plots in Figure 9.9 show the validation error J_{val} over time for the five target dimensions along with the mean training error in black. When all dimensions are trained simultaneously the problem of overfitting to the data set occurs, resulting in an increase of validation error as training increases, particularly for class 2 (in red). The same class trained alone (right) results in both an improved validation error minimum and greater stability with increased training.

As the data set was specifically constructed in such a way that there is no correlation between any of the separate classes, it can be expected that there will be little or no improvement from the hints of simultaneous class learning. In fact there is a drop in classification accuracy when all dimensions are learned together. It appears that the use of all hidden layer units to learn five independent and orthogonal functions simultaneously prevents the specialisation that should happen in the hidden layer as training progresses, and allows overfitting to the training set by providing more hidden layer neurons than are necessary (as in section 9.3.1). This suggests that the structure of the network may be tuned to the problem at hand by separating the hidden layer neurons into distinct groups each connected only to one output when the output dimensions are known to be independent.

The method chosen for doing so involves the separate training of networks on each of the class dimensions, using 20 hidden layer units and 1 output. These are then assembled into a larger network after training in which these five groups of 20 units form a hidden layer but each group is connected only to one of five outputs. This way the error criterion and training time can be set independently for each of the five classification dimensions, leading to a more accurate overall output.

9.4 Methods for improving the search

In applying machine learning to classification problems, the goal is to find the function that provides the discrete labels originally given to the initial data set. In applying learning to regression problems, the function provides the original real-valued output. In contrast to both of these, the present work is not concerned with an accurate mapping of the original data set, but with a function to guide optimisation.

This is not to say that the function should not be precise, only that a recreation of the original data is not the main issue. In classical artificial intelligence, representations are concerned with a faithful reconstruction of data for general purposes of further computation (Wheeler 2005). Computer vision applications that follow Marr's (1982) theory of vision, for example, use edge detection, stereopsis and similar algorithms to construct a 3-d model from 2-d images, with emphasis on the accuracy of this model. Embodied approaches to robotics (Brooks 1991; Durrant-Whyte 2004), in contrast, are concerned only with what is necessary to guide action in an environment. The latter are not only less computationally expensive, but more robust and suited particularly to the task at hand. Data in these approaches is no less important, but is in a form required for action rather than dictated solely by the original source.

This section examines what is required of the fitness data for the particular actions required in searching through the space of possible solutions. To do this, the more compact genome representation C (Appendix A) will be used. This adds to the definition of convex spaces the assumption that there is a regularity to their arrangement based on the repetition of desk group configurations. A parametric model is used to determine the shape size and connectivity of spaces, but these can be repeated over any sized plan in a manner similar to the parametric layout of desk rows in chapter 6.

The total size of the search space is no greater than 2^{97} or roughly 10^{29}, far smaller than genomes A and B. Because of the structural limitations taken to reduce this, there are some plan arrangements that are outside of the search space. The ClassPlan set used to set the optimisation goals in all tests to follow was composed of plans that are impossible to represent using this genome, to ensure that all solutions found by optimisation would only approximate the original targets by degrees.

9.4.1 Optimising to learned features

Optimisation toward a given specified goal is much the same as in chapter 7, except the fitness is judged not as a distance to an objective prototype plan in the full feature space of the spectrum but as a distance to an archetype in a reduced feature space Φ. In this case the archetype is derived from the classification of convex groups and linear rows in ClassPlan, using the MLP discussed in section 9.1. A genetic algorithm is again used, but all evaluations

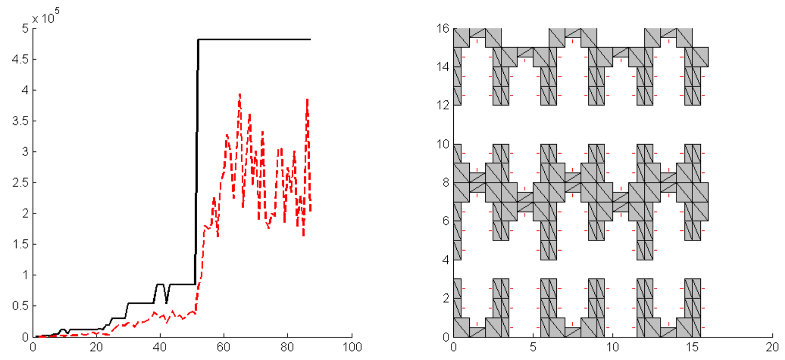
$$f(i) = 1/\sqrt{\left[\Phi(S'_i) - \Phi(S'_{goal})\right]^2} \tag{9.11}$$

are measured in this space. The goal is set either to the measured position of the mean of the ideal plans, or simply to the original target of z = ±1. With a trained network these values are similar; in the following example the means are –0.996 and 0.982, and so the targets –1 and 1 were used.

Figure 9.10 shows the results of two typical runs of the GA to evolve examples of a given class in the above feature space, either convex groups or linear rows. The algorithm has a population size of 40, a mutation rate of 4% and employed fitness proportional selection. Both results are recognisably of the target class but the designs evolved are novel: neither were present in the initial data set on which the MLP was trained.

9.4.2 Clarifying the search space / perception space

The use of a classification methodology for assigning target values to plans coincides well with our intuitive ability to distinguish between sets of objects, to see clearly in terms of polarities (section 10.4.3), but in constructing a search space in which optimisation will occur this can be a disadvantage. For a GA to conduct an efficient search, it is desirable to have changes to the genome affect corresponding and ideally proportional changes to the fitness of the phenotype (Bentley, 1999), but with an ideally trained two-class classifier, the response of all inputs within a class will be identical and equidistant from all examples of the opposite class. There is thus no variance in fitness unless the goal class has been achieved and no capacity to direct the search by fitness. While a MLP can not generally be trained to a zero-error, the standard configuration with sigmoid activation that has been used for training does bias the fitness to be disproportionately similar within a class and disproportionately varied between classes. The plots in (figure 9.10) indicate the usual pattern of fitness change in the search: a relatively small change in fitness for the initial generations, followed by a rapid and extreme jump after which no further improvement is made.


Class 1, target –1 (convex groups)

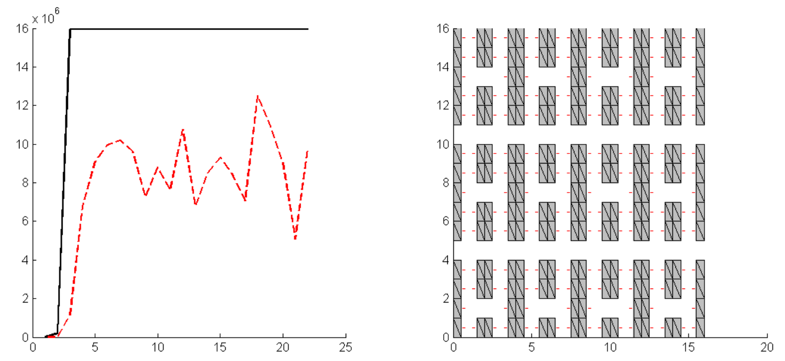

Class 1, target 1 (linear rows)

Figure 9.10. Plans produced by the GA for opposing objectives.

Two ways in which the perceptual space in which fitness is measured can be made more amenable to GA search were examined. These also deal with two issues of usability of such a network in actual design. First, while a continuous fitness evaluation is desired, it is often easier for a user to make a binary classification than to assign accurate continuous values. The possibility is proposed of developing a continuously variable feature space by training on a set with discrete targets. Second, it may be advantageous to begin training the network from a previous useful state.

For effective learning of difficult type-2 problems, algorithms can be assisted in acquiring the appropriate recoding of the data both by altering their structure during training and by prior manipulation of the data set (Clark and Thornton 1997). Elman (1993) gives examples of both in the context of supervised learning—a periodic resetting of units in a recurrent neural network allows a network to selectively focus on simpler, more relevant structures in the data, as does the pre-sorting of the data into batches of successive complexity. In unsupervised learning, the auto-encoder (Duda et al., 2001) yields a principal or non-linear component analysis in a hidden layer, which is made explicit by the subsequent removal of the output layer in the validation or use phase. The methods suggested here aim to improve the training of the network not to reproduce a data output set but specifically to yield a more *navigable* fitness landscape. The first is to tune the α parameter of $g(\cdot)$ to saturate at the class targets and the second begins training from weights initialised to the principal components of the data. These are tested below.

9.4.3 Tuning the activation function g(·)

Alteration of a neural network between training and evaluation to access learned internal representations—as occurs in the auto-encoder—is examined here to refine the feature space beyond the labels used in training. Classification of data toward discrete, opposing targets is appropriate given the construction of the *ClassPlan* data set and makes initial labelling easier for the user, but is detrimental to the later GA search. It is therefore desirable to allow the machine to be trained on a binary classification but make its evaluations on a continuous scale. It is the sigmoid activation function (Duda et al., 2001) in the MLP

$$g(net) = \alpha \tanh(\beta\, net) = \alpha\, (e^{+\beta\, net} - e^{-\beta\, net}) / (e^{+\beta\, net} + e^{-\beta\, net}) \qquad (9.12)$$

that ultimately determines the placement of a sample in Φ, and the parameter α limits the range of activation values to $\pm\alpha$, where all extreme values saturate at the function's asymptotes. If the network is trained with such a function and then used in fitness evaluation with the linear activation

$$g(net) = \alpha\, net \qquad (9.13)$$

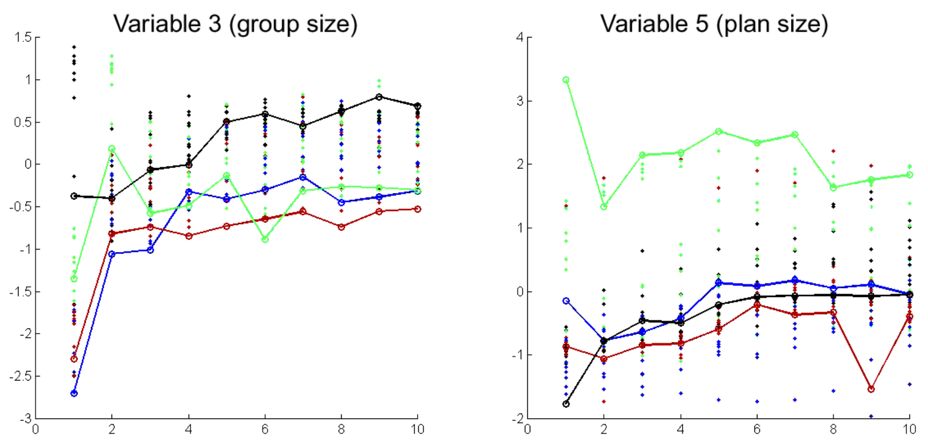
in the output units only, this will disperse the values formerly saturating the asymptotes at $\pm\alpha$ over the entire linear range $\pm\infty$.

This can be enhanced by tightening the training function even further. A typical setting of α = 1.716 (Duda et al., 2001) results in a $g(\cdot)$ that is nearly linear in the target range [–1, 1]. If α = 1.0 is used so that $g(\cdot)$ saturates at the classification target values ±1, almost all values for $net > 0$ result in $g(net) \approx 1.0$ and values for $net < 0$ result in $g(net) \approx -1.0$. This produces a more continuous space for evaluation.

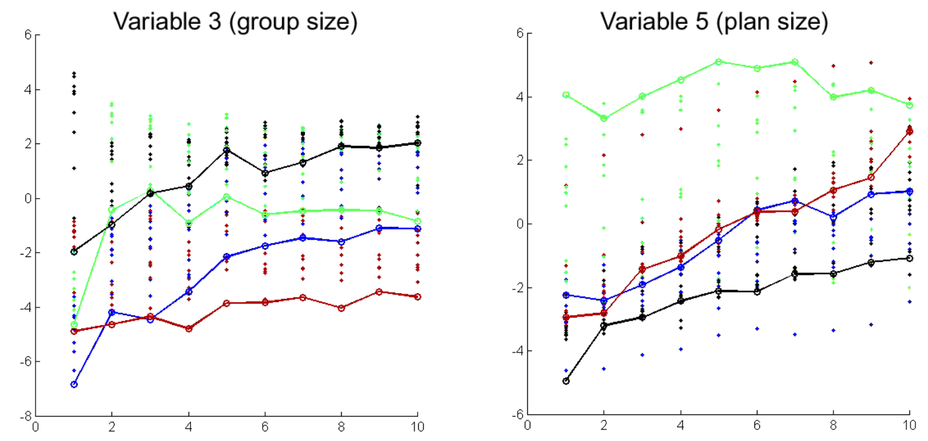
9.4.4 Validating the Continuity of Φ

Zhu and Wilson (2005) measure the effectiveness of spectral representations by comparing sequential distances between spectra with known edit distances recorded when creating the graphs themselves. Similarly, the continuity of the feature space Φ can be judged by comparing distances against changes to a known parameter in the plans. The *SequencePlan*, set was again used, consisting of four groups of 100 plans, each of which varies monotonically in ten steps of two parameters. One of these parameters is local (number of desks to a group) and corresponds roughly to variable 3 in the *ClassPlan* set, and the other is global (overall plan size), corresponding to variable 5. Each of these four plan sets has a different general configuration of desks. As in Section 9.1.2, the mapping $\Phi(\cdot)$ was created based on variables 3 and 5 of the ClassPlan set and then used to evaluate the corresponding features of the sequential data set. Ideally, even while trained only on opposing examples, the placement of sets of sequentially varying plans will still be incremental in Φ. The utility of the resulting space is then judged in terms of how close to linear the resulting output is.

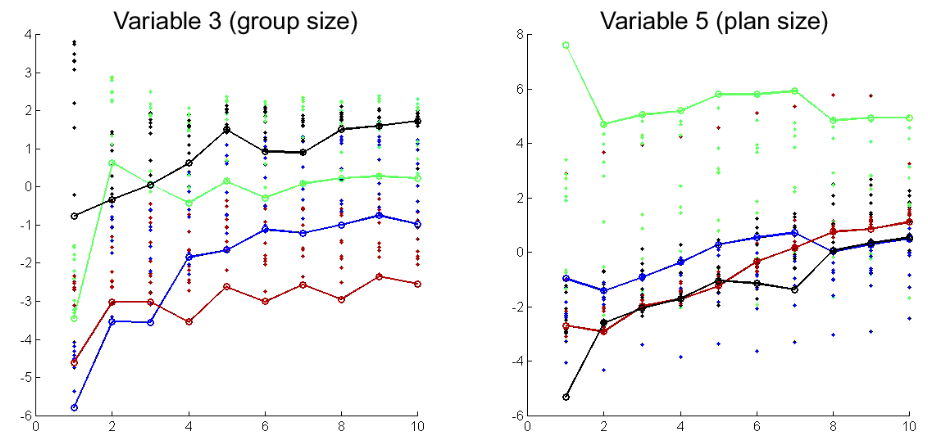
Evaluations of 40 sequences of 10 plans each from *SeqPlan* were made, 10 sequences of each plan type. Results of training with the 'tight' sigmoid of $\alpha = 1.0$ are displayed in Figure 9.11 (b) in comparison with a larger $\alpha = 1.716$ (figure 9.11.a), and a general straightening of the sequences of plans is evident. The corresponding mean relative deviations are displayed in Table 9.4, where deviation is taken relative to the whole set, or roughly equivalent to the scaling of the plots in (Figure 9.11). The overall relative deviation is seen to decrease by more than one third when trained with the 'tight' sigmoid $\alpha = 1.0$, reducing in all cases except for that of the plan type 3 (the green line on the plots).



a) Normal sigmoid function training



b) Tight sigmoid function training



c) Tight sigmoid function training initialised with principal components

Figure 9.11. Uniform changes of parameters plotted in various feature spaces.

Table 9.4. Mean relative deviations in Φ for various training methods.

	Random initial weights				Weights initialised from PCA			
	Normal training		Tight sigmoid [-1,1]		Normal training		Tight sigmoid [-1,1]	
	Class 4	Class 5	Class 4	Class 5	Class 4	Class 5	Class 4	Class 5
Type 1	0.394	0.410	0.120	0.375	0.191	0.363	0.159	0.351
Type 2	0.468	0.348	0.133	0.300	0.205	0.200	0.170	0.194
Type 3	0.749	0.571	0.208	0.750	0.418	0.581	0.317	0.590
Type 4	0.453	0.344	0.166	0.341	0.311	0.269	0.257	0.284
Mean	**0.516**	**0.418**	**0.157**	**0.441**	**0.281**	**0.353**	**0.226**	**0.355**
Method mean	**0.467**		**0.299**		**0.317**		**0.291**	

9.4.5 Initialising the weights with the principal components

A network trained on the two to four set classification given by *ClassPlan* must interpolate to a much finer degree to evaluate a complete set of ten plans in *SeqPlan*. With the initialisation of network weights at random, it is possible that an inappropriate initial setting is not adequately ‘unlearned’ by the limited examples in the training set. In the use of such a neural network for evaluating data on which it has not been trained, it may be useful to begin training with weights that are a decent first guess. The principal components of the data set by definition capture most of the variability of the examples, and their orthogonality ensures that there will be no redundancy in the network. In this section, training the network using the principal components of the training set as the initial weights is compared to the use of random initial weights.

PCA is performed on the training set to yield the set of eigenvalues λ and eigenvectors φ of the covariance matrix, which are sorted in order of decreasing magnitude of $|\lambda|$ as in Eq. 7.4. The network is to be trained with 20 hidden nodes, so the first 20 $\{ \lambda_1 , \lambda_2 , \ldots , \lambda_{20} \}$ and $\{ \varphi_1 , \varphi_2 , \ldots , \varphi_{20} \}$ are used. The eigenvectors representing the orthogonal principal components are then used for the hidden layer weights, such that the net activation net_j at a hidden layer node represents the projection of a given sample onto that component, and the eigenvalues are used for the output layer weights to weight the output in favour of the largest components. Because the projections can be quite large, and enter the saturation regions of $g(\cdot)$, these initial weights were scaled by 0.2 to maintain a *net* largely within the near-linear range of $g(\cdot)$. Each weight then is given by

$$w_{ij} = 0.2 \times \varphi^{j}(i) \quad (9.14)$$

$$w_{jk} = 0.2 \times \lambda^{j} \quad (9.15)$$

for the input and hidden layers respectively.

Again, the performance was measured in terms of relative deviation of the sequences in data set *SeqPlan*. The output of the network trained on the ‘tight’ sigmoid (Eq. 9.12) with weights initialised as in Eq. 9.14 and 9.15 is displayed in 9.11(c) and the results of training with both activation functions are listed in Table 9.4. The mean relative deviations are seen to decrease with the use of principal components as the initial weights, and this occurs for most of the sample sets in both classes. The best overall definition of Φ appears to be given by training with both the tight sigmoid and initialising the MLP with the principle components of the data set.

9.5 Testing the process: optimisation to generate plans

The method was tested first in optimising to match individual plans as goals within the feature space Φ, then by attempting to emulate these same plan features within a space of a different size, and finally to particular objectives defined as archetypes by a series of plans. For purposes of visual evaluation these objectives were set equal to particular known plans from *ClassPlan*, although it is relevant to note that because of the intentional mismatch between the genome representation and the *ClassPlan* set it is impossible to exactly duplicate a particular *ClassPlan* example, so optimisation is intentionally restricted to approximating these plans.

Optimisation was by GA as in §9.4, with one crucial difference in that all evaluation is performed in the *learned* feature space: a space only of five dimensions rather than hundreds. All five variables were expressed in the feature space, for a five dimensional measurement of the objective function (Eq. 9.11). The GA was run with a population size of 40, each time for 200 generations, to produce plans constrained to a 12×12 grid. Fitness is calculated as inverse distance to a target in Φ as in (9.11). In all cases, examples of the *ClassPlan* set were used as the examples from which to derive the objective.

9.5.1 Matching plans in *ClassPlan*

Although any labelling of example plans could be used to create the feature space Φ in which fitness is measured, the effectiveness of the algorithm is more easily judged when the goal is visually recognisable. The variables initially used to create the data set were used as objectives for the GA as each easily provides two opposing classes of 64 plans. The first test attempted to match targets for all five variables simultaneously, after each was trained separately as an independent dimension of Φ. This way, the plans produced could be visually compared against an actual plan in *ClassPlan*.

The optimisation was performed with targets set to match eight different plans in *ClassPlan*. Of the five distinct variables available, only three (group shape, alignment and length) could be represented adequately by the genome in representation C, while those for overall plan shape and size are predetermined by user preference. The polar targets of –1 and +1 were used for each of these variables in combination to give eight separate plans (shown in figure 9.12, left). The GA was run three times for each objective, and the results are displayed in figure 9.12 with the distances in feature space of the resulting solutions displayed above.

A visual analysis of the resulting plans confirms the similarity of the matches to the targets chosen. The first variable, contrasting convex groups with straight rows is easy to see in the examples, and is clearly emulated in all 24 cases except one: an attempt to match plan 81 which produces large convex groups rather than short rows. It might be argued that these convex groups are actually comprised of smaller, separate, linear segments, only that they are arranged on perpendicular axes. The distance of 0.5 units in feature space is noticeably higher than the majority of solutions (usually below 0.2), indicating that this solution is simply a result of an inadequate and incomplete search of the solution space rather than an unsound evaluation.

The second variable contrasts aligned groups (plans 1, 4, 65, 68) with staggered (plans 17, 20, 81, 84). Although the novel group arrangements introduce some subjectivity into the validation, between 13 and 18 of the 24 results are members of the appropriate class.

There is again some subjectivity in the judgement of the third variable of group length, as the ‘short’ example plans (1, 17, 65, 81) have two to three desks in a ‘typical’ line while the ‘long’ examples (4, 20, 68, 84) have five or six. If one attempts to find the typical length of a row of desks in any of the resulting plans and counts 2, 3 or 4 as ‘short’, and 5, 6 or 7 as ‘long’ then the number of plans that accurately meet their target is between 18 and 23 of the total 24.

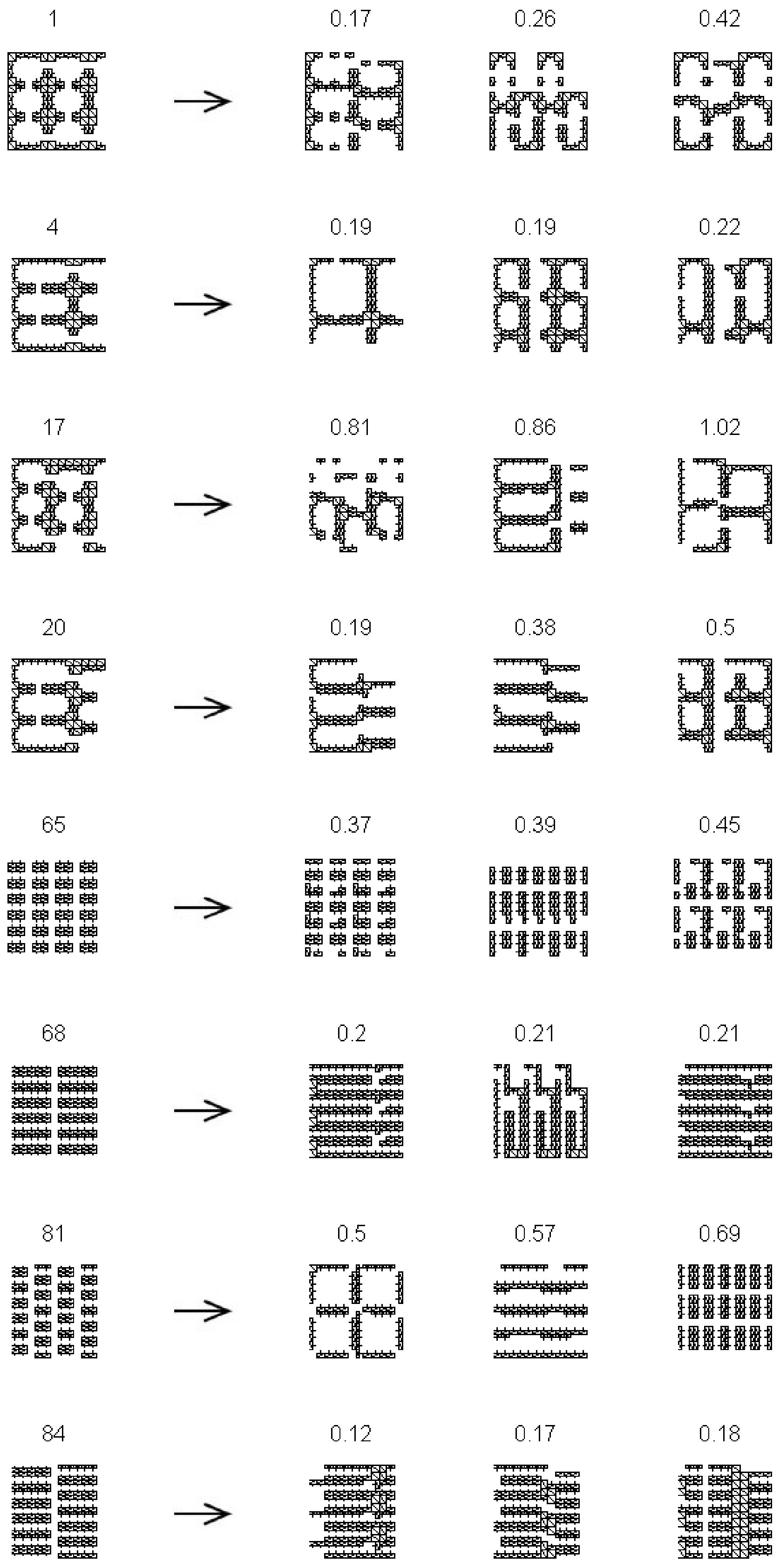


Figure 9.12. Results of the GA searching for a match to a plan in a multidimensional feature space Φ.

The varying ability of the GA to produce plans to fit each of the classes should be unsurprising, as it correlates with the varying ability of the MLP to learn each of the variables (table 9.2). As indicated by the cross-validation in sec. 9.1.3, the distinction between convex groups and straight rows is easily and most accurately represented, followed by group size, with staggering of rows the most difficult to discern. This is exactly the order in which the number of perceived inaccuracies have occurred in plan generation.

The inclusion of five dimensions of criteria in the objective function results in a more difficult search than in section 9.2.1. Evaluated by the same method the statistical accuracies over 24 runs are slightly lower:

variable 1. convex groups or linear rows:	96%
variable 2. aligned or staggered groups:	67%
variable 3. small or large groups:	83%

The varying ability of the GA to produce plans to fit each of the classes correlates with the varying ability of the MLP to learn each of the variables as indicated by the cross-validation in sec. 9.3.

9.5.2 Changing plan shape and size

In the application of optimisation to real design problems, it is most likely that the plans used for training will be of a different overall shape and size from those to be designed. The exercise of optimising to match the targets of an existing plan as in section 7.5.4 was again performed to produce plans of a size not in the original data set. The *ClassPlan* data set used to train the MLP contains plans of 12 × 12, 12 × 24, 24 × 12 and 24 × 24 units, and the GA was set to evolve a plan of 16 × 16 units. Based on the correlations of spectra with local features in chapter 7 and the comparison below, the hypothesis was that the differences in global plan size would not greatly affect the evaluation, and the plans resulting from the optimisation would have the same features as their targets at different sizes.

The spectra of the axial graphs of four plans are displayed in figure 9.13, with the original plan spectra in solid lines and the same plan cropped to a maximum size of 16 units in dotted lines. The plans chosen differ in local group size (the blue lines as opposed to the red) and overall size (plans 5 and 7, left, are 24 × 12 units; plans 13 and 15, right, are 24 × 24 units). In all four cases the change to the plan size affects the spectrum very little, although the change is slightly greater when the 24 × 24 plans are cropped in both dimensions (figure 9.13, bottom) than when the 24 × 12 plans are cropped to 16 × 12 (figure 9.13, top).

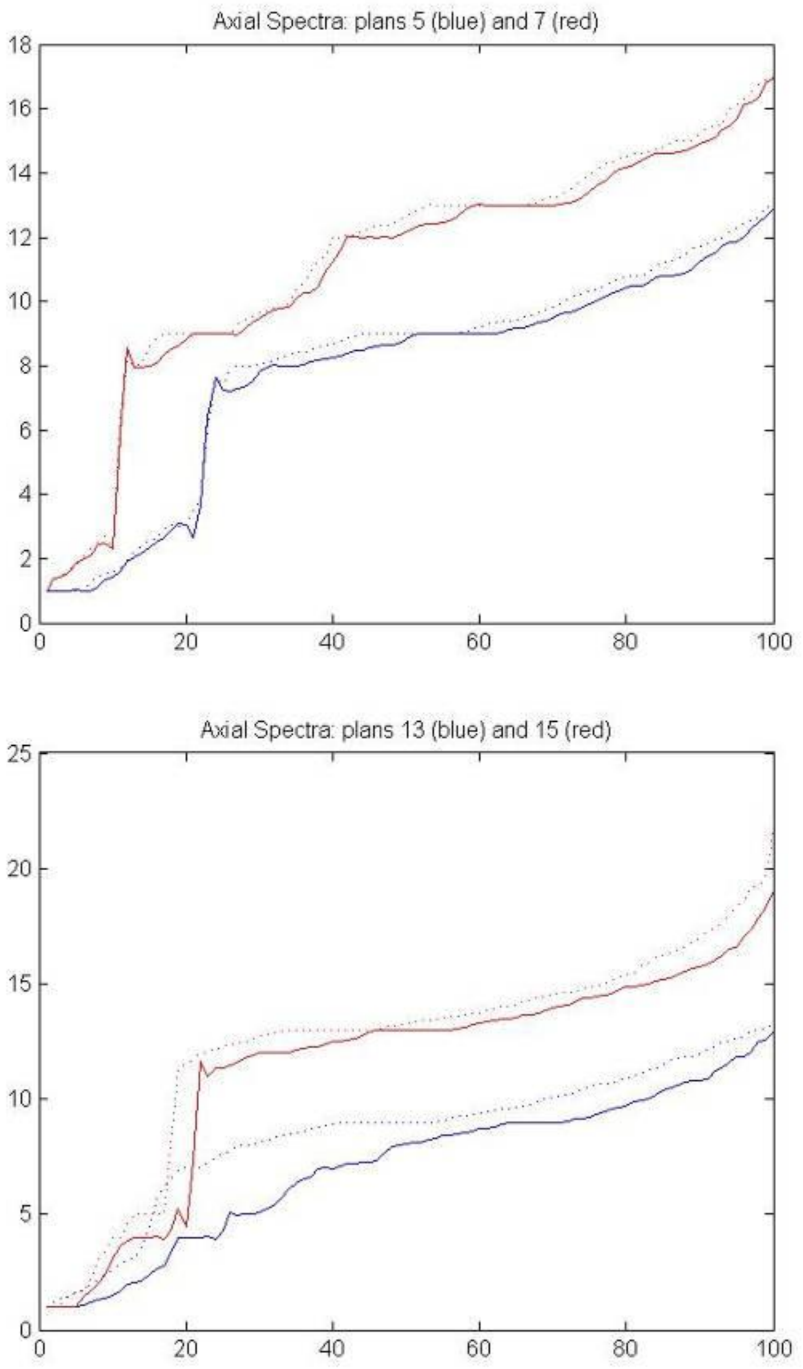


Figure 9.13. Changes in overall plan size have only minor effects on resulting spectra.

The same MLP as in section 9.3.1, trained on the examples in ClassPlan, was used to determine the fitness for the GA, and the targets were set equal to 16 plans representing all permutations of variables 1,3 and 5. The GA was run for 60 generations with a population size of 35; five solutions were provided for each setting. These are displayed in figure 9.14 with the distances in feature space indicated above each plan.

An overall visual examination of the results indicates a general similarity between the results and the target plans. As in section 9.5.1 the evaluation of the results is somewhat subjective, but they can be judged by the same criteria. For the first variable, distinguishing convex groups from straight rows, between 64 to 69 out of 80 results match the target. Variable 3, distinguishing aligned from staggered is comparatively poorly represented with only 41 to 45 accurate samples. Four values for the group length variable have been chosen for this experiment, making the results rather more difficult to discern, but if only the extremes are evaluated as in §9.5.1 then 28 of 40 samples are correct.

While still subjective, a more numerically precise indicator of the algorithm's efficacy can be given by a correlation coefficient between the expected target values and the GA results. For the two class variables, classes are assigned as either –1 or +1 when clear or 0 when the solution is not clearly of either class; an estimation of group length is given by the mean number of adjacent desks when groups of different sizes exist. Figure 9.15 plots the correlation between values of group lengths, and the correlation coefficients for each variable are as follows:

variable 1:	0.706
variable 2:	0.077
variable 3:	0.504

The difference between the three variables parallels the results of optimisation in section 9.5.1 and the learning in section 9.3. Variables 1 and 3 have moderate correlation coefficients, while variable 2 is very low.

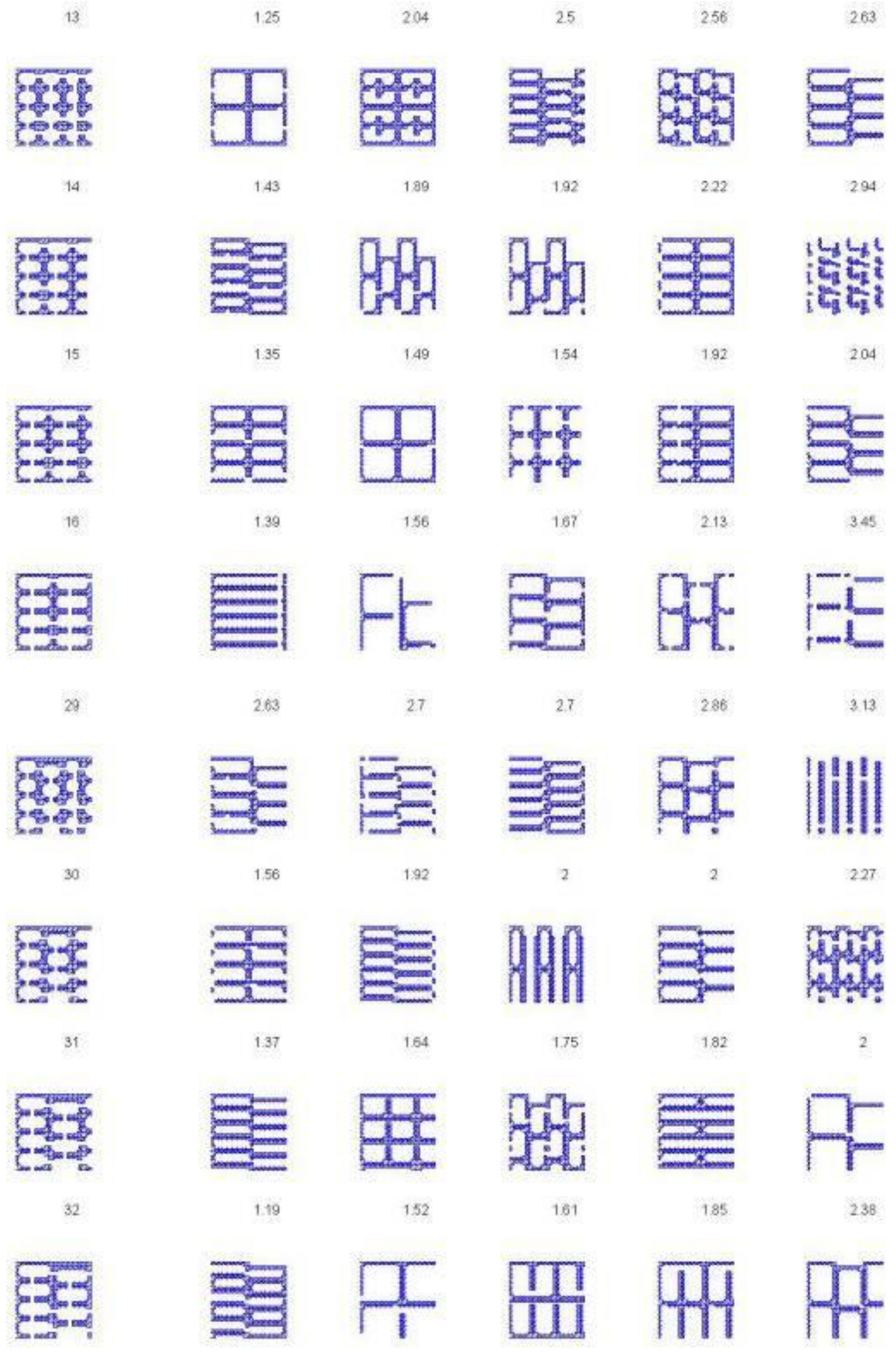


Figure 9.14. Plans generated at a size of 16 × 16 units, a size not in the original training set. Five examples to the right of each row were generated to match the model at left.

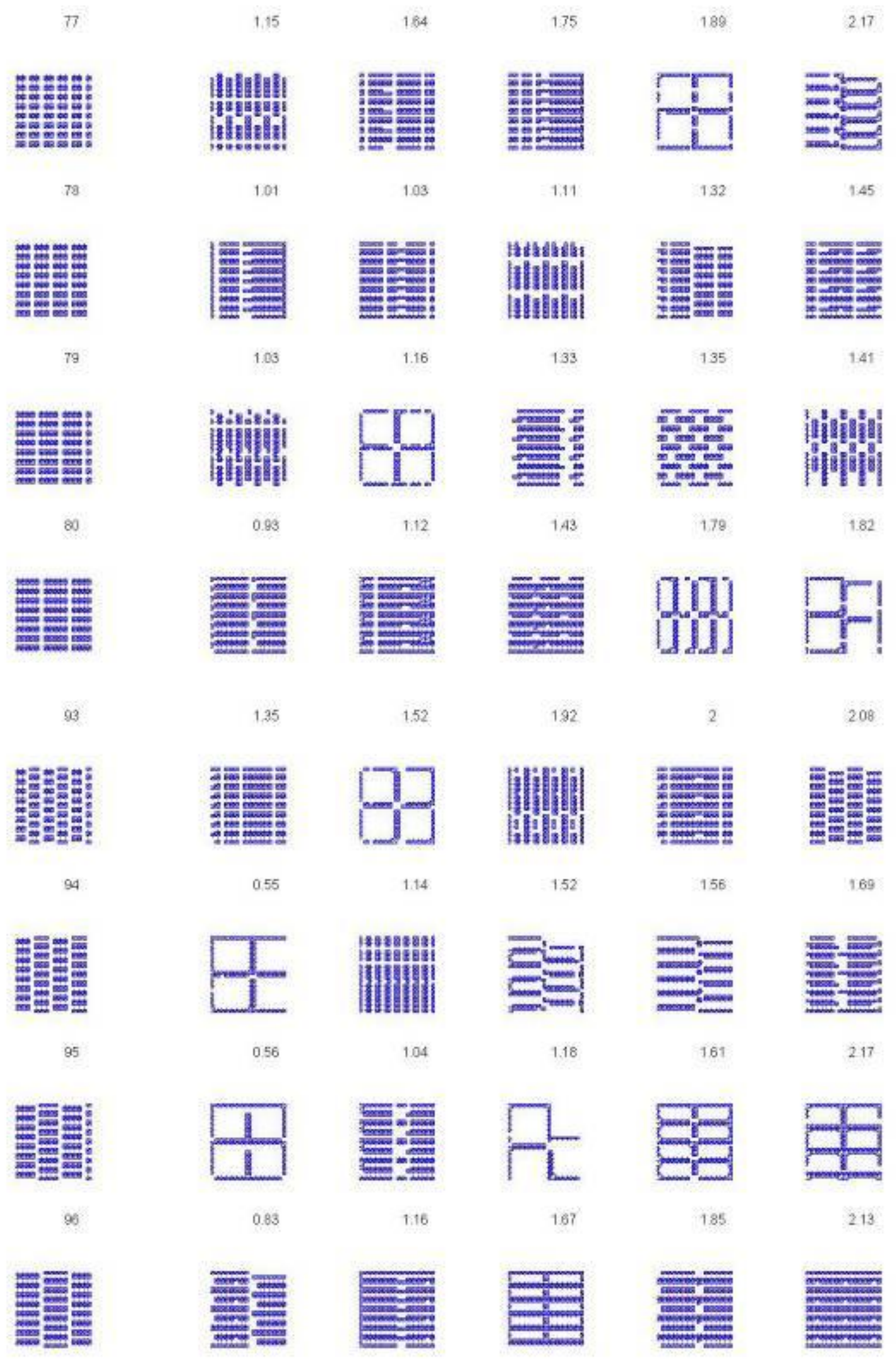


Figure 9.14 (continued). Plans generated at a size of 16 × 16 units, a size not in the original training set. Five examples to the right of each row were generated to match the model at left.

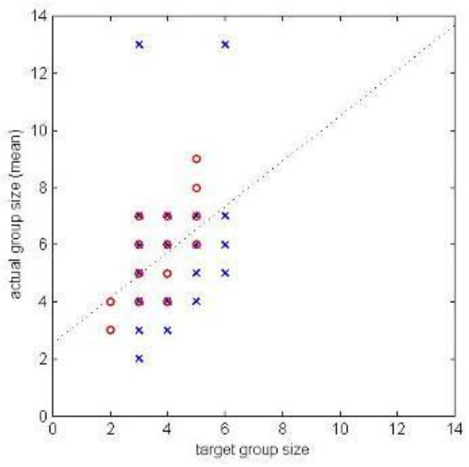


Figure 9.15. Correlation between desk group size in the target plan and those produced by GA.

The fit of the results is less successful than in section 9.5.1. There are several likely reasons for this, purely based on the structure of the experiment. The first is the GA: even with genome representation C the search space is large and so cannot be searched exhaustively, and the run was limited to 60 generations in this experiment (rather than 200 as in §9.5.3 or an average of 788 in §9.5.3). The distances in feature space confirm that the search could still be carried on, as most are between 1.0 and 2.0 rather than the 0.2 reached in the earlier experiment. It is also a limitation of representation C that due to its modular generation it can not actually represent the majority of the target plans exactly. Many can be nearly approximated, but with subtle differences in the graphs, so a distance of zero in the feature space is almost impossible in this experiment.

More relevant conceptually for both the current results and those in section 9.3.1 is that most of the designs produced do not fall neatly into the categories represented by the five variables or the ClassPlan data set. The importance of this goes beyond the subjectivity of evaluation. By training the MLP on a set of examples that uniformly cover a [–1, +1] range in five orthogonal dimensions it is implied that all plans will fall within this limited 5-dimensional subspace, and this is clearly not the case. The majority of plans that occur even in the relatively orderly genome representation C, those with multiple group shapes, isolated islands or irregular, intermittent rows, exist outside the hyperplane mapped by ClassPlan and are just as difficult for the MLP to classify as they would be for us. This highlights the need for training using as broad a data set as possible, varied in as many dimensions as can be done by the genome. The ClassPlan set was created artificially and therefore limited. In addition, a randomly generated set should be evaluated by the user and used to supplement this. Solutions produced by the GA, whether considered appropriate or not, should be labelled and used to continually refine the learning algorithm through continued training.

Given the shorter run time and the reasonable success of achieving particularly two of the three variable targets, it appears that the algorithm is in principle capable of reproducing the local details of plans on a different scale. Longer searches tend to give better results, and it is primarily the further refinement of the GA optimisation to enable larger and faster searches that would improve the technique for practical use.

9.5.3 Optimising to Learned Objectives

The final optimisation tested the ability of the machine to generalise archetypal features from given labelled sets of plans. Each set was chosen by one of the initial variables, but because global size was restricted to 12×12, only the local variables were used.

Figures 9.16, 9.17 and 9.18 show the results of five runs of the GA for objectives set to each of the extremes of variables 1, 2 and 3. Two example plans from the training set are shown at left to represent the objective defined by Φ, although these represent a much larger group of 64 plans of varying arrangement and overall size. At right the best solutions found are drawn with their distance to the goal indicated.

Each set of evolved plans can be seen to indicate the features common to the example plans: straight rows, small group size, etc., indicating both that the objective has been defined and that the GA was able to satisfy it. It is also evident that for each run the features not included in Φ are irrelevant. In Figure 9.16, the top row of plans contains only non-linear, convex groups of desks, but these range in size from small clumps of three or four (first plans at left) to large room like spaces of more than twenty (last plans on right).

Some variation exists in the apparent accuracy to which the GA meets the supposed objectives. Although these can only be judged empirically, this accuracy may be graded by counting each plan recognisably in the correct class as a value of 1, each one not as a value of 0, and those undecided as 0.5. Over a trial of ten runs, the statistical accuracy for each of the classification variables was judged to be the following:

variable 1. convex groups or linear rows:	95%
variable 2. aligned or staggered groups:	70%
variable 3. small or large groups:	90%

This appears to match the ability of the MLP to learn the distinction between the initial sets on which it was trained. The accuracy of the classification as judged by cross validation on these three variables was 100%, 92% and 93%, differing in value from the above figures, but ranking the variables in the same order in terms of clarity or distinctiveness in the feature space. As indicated by the small distances to the objectives in Figure 9.17, these inaccuracies in the aligned and staggered group plans are due to the difficulty in initial MLP training to this variable, rather than the GA optimisation. Of the three, this also is the most difficult feature for the human eye to discern, and it is likely that a different, perhaps larger, training set of example plans would be needed to improve the results.

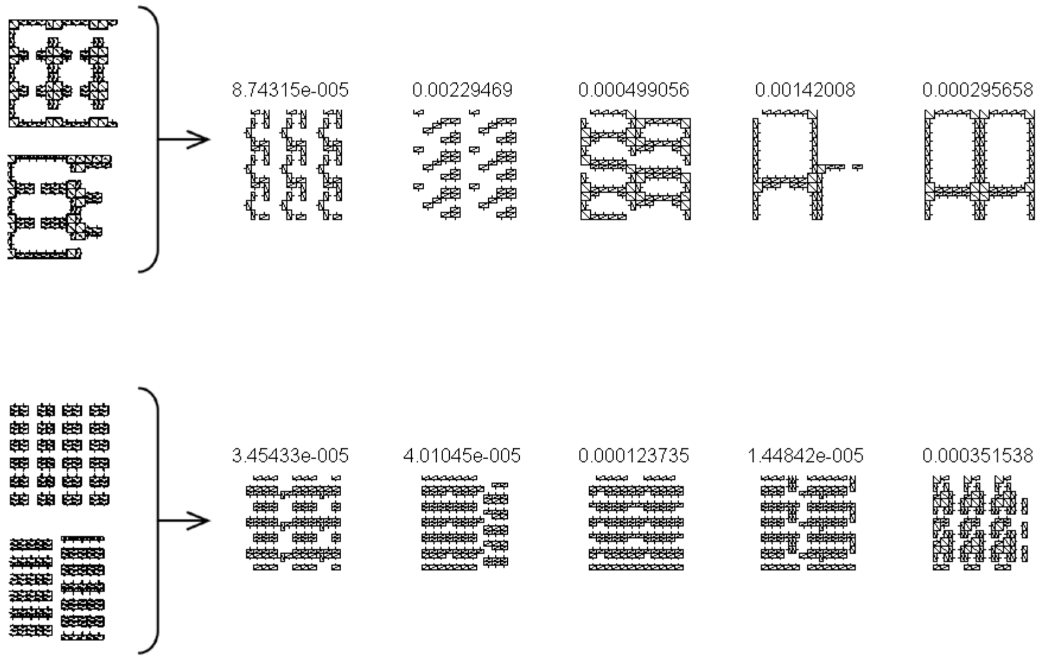


Figure 9.16. Results of the GA producing convex groups (top), or linear rows (bottom). Sample plans representative of the training sets are shown at left.

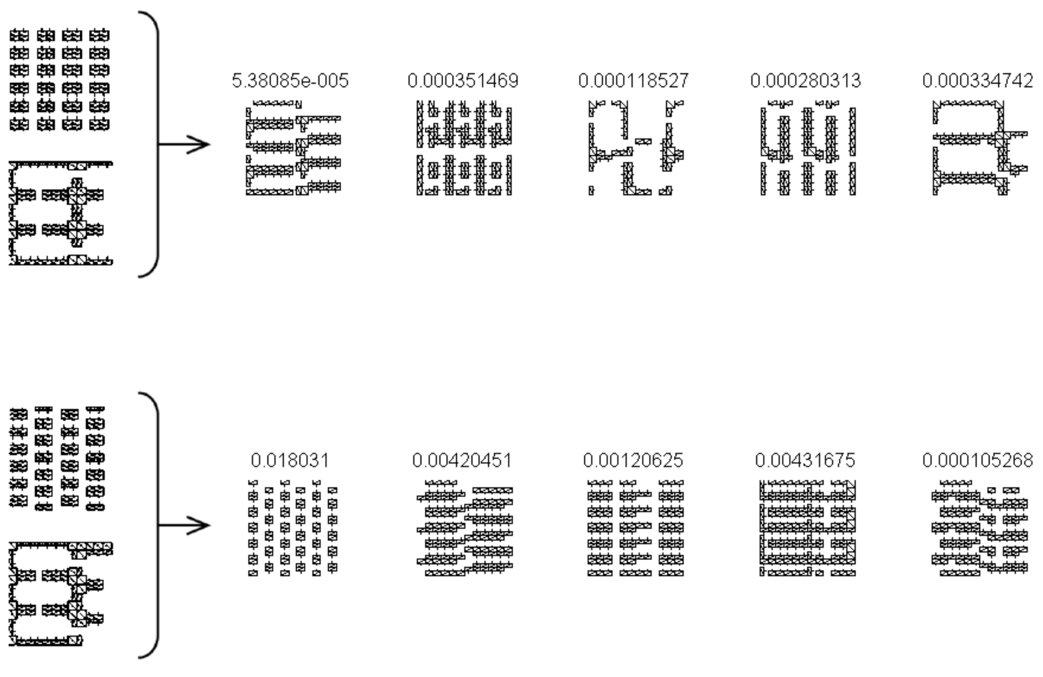


Figure 9.17. Results of the GA producing aligned (top), or staggered groups (bottom).

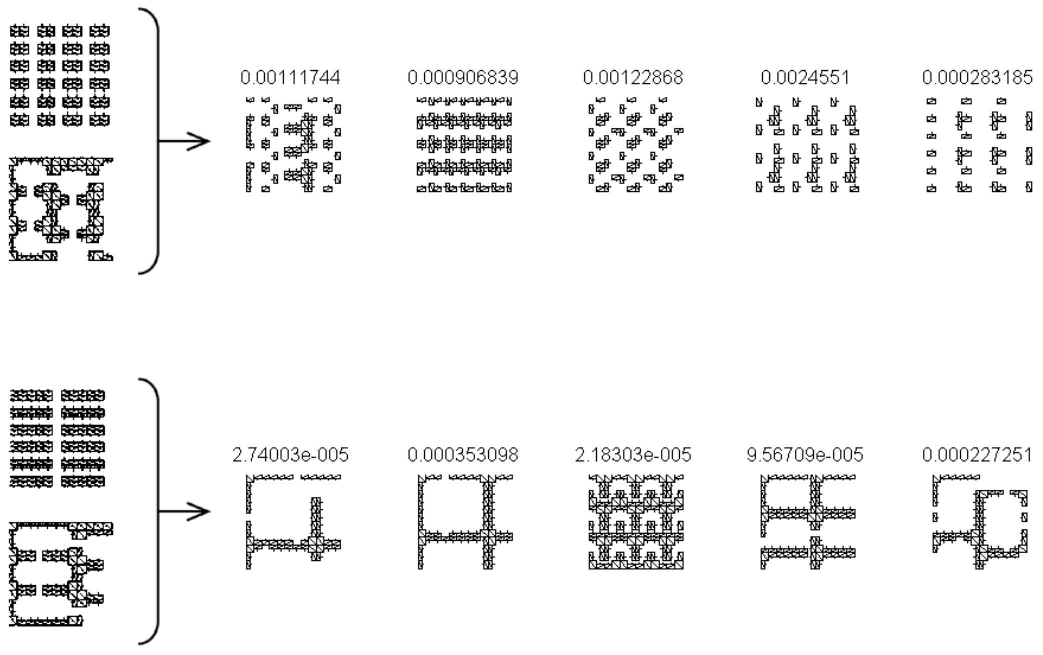


Figure 9.18. Results of the GA producing small (top), or large groups (bottom).

9.6 Chapter review

Optimisation has been used here to test the algorithm's competence in extracting and reproducing the structure of office plans, as given by *arbitrary geometrical criteria*. By finding solutions that approach the initial target features, this indicates that these features selected to form the feature space Φ were captured accurately enough to be able to drive the search. The fact that radically different plans did not result indicates that these features were captured uniquely and unambiguously.

Two separate processes were involved: the learning of the archetype, and optimisation of new plans. Machine learning on the plan spectra was seen to be effective. In sections 9.2 and 9.3 the *ClassPlan* set was well classified using a set of five independent variables. Some degree of error was revealed by cross validation, but this error was seen to be consistently related to the variables themselves, rather than the learning process, indicating that for the machine, just as for any two human judges, some categories are inherently more ambiguous than others. The training was seen to capture the sequence within two of the variables of a separate set (*SequencePlan*) on which it had not been previously trained (section 9.4.4). The two algorithms tested, a SVM and a MLP, were both capable of learning and determining a feature space capable of distinguishing arbitrarily chosen classes.

This space was then used in combination with GA optimisation to evolve new designs within those classes. The result of this in all cases was an improvement in the similarity of the plans produced to the target. Plans were successfully produced to correspond first to multiple selected features (sections 9.5.1 and 9.5.2), and then to a single selected feature (section 9.5.2). Differences in global scale or plan shape were effectively produced by the algorithm. In section 9.5.2 a similar arrangement was produced on a plan of different size without any explicit changes to the MLP or gene representation. Such a method could be used to reproduce features from one building within the constraints of a differently shaped plan for another.

9.6.1 Practical improvements to the optimisation

Methods of improving the algorithm were examined with a view toward practical implementation of such a method. These involve either the feature space developed by learning of the MLP or the search procedure by GA.

Training on multiple output dimensions simultaneously or the use of these as hints (section 9.3) does not appear effective. Dimensions of a feature space should be trained separately.

Regarding the selection of the training data itself, it appears the training data set should be as broad and multi-dimensional as possible. Section 9.3 indicated that overfitting is not particularly a problem, and in actual use training should continue until the change in J_{err} is sufficiently low. Any data set created intentionally for clarity is likely to miss out on several possible dimensions, a common danger of pre-processing of data in complex problems (Cilliers 1998). In generating the data set, the use of randomly generated solutions would likely have benefits over the purely orthogonal parameters used in *ClassPlan*. Better still, in practice, would be a means to direct this search to query the user to label areas of the possible feature space in need of definition. Concepts could then be negotiated by a process of trial and response, as in language learning (Wittgenstein 1958, Steels 2000). Such a directed process would be analogous to Kuhn's (1963, Chapter 8) crisis state, in which attention is focused on a particular area that appears most likely to fail and falsify the paradigm. This would rely less on the volume of evidence (as with random samples) than on getting the most appropriate samples.

The weakest part of the process is not the training of the machine with example plans but the generation of new plans by GA optimisation. Results were seen to improve over time, but particularly with genome representation A this process can be slow. The use of a more constrained generative system as in plan representation C greatly reduces the search space and benefits use in real design situations. The present focus on ill-defined aspects obscures the fact that most design domains also have many clearly definable constraints, and these might be built into the system. There is scope for additional investigation into the alignment of the gene and feature spaces, and enabling the optimisation handle larger and faster searches would improve the technique for practical use.

9.6.2 Inductive learning and ill-defined objectives

The aim of this chapter has been the use of inductive learning to handle ill-defined aspects of a design task in which objectives are not clearly stated, and thereby the falsification of Hypotheses A and B in this context.

Hypothesis A supposes the underlying structure of the system being designed is known and simulated. This was not the case. The group of samples was treated as a 'black box' (Ashby 1956; §3.2.5), in which the *arbitrary geometrical criteria* that defined it were invisible, and it was only observed at a higher level of its resulting plans. Learning was used to implicitly capture these arbitrary features. The approximation used to represent these criteria was also of a very different form from the criteria themselves—the spectral feature space was based on topological graphs rather than geometrical patterns. In this case, evaluation and labelling were performed at a high level only (e.g. 'convex groups'), but were used to discern low level features (e.g. actual placement of individual desks), and successfully optimise to these.

Hypothesis B assumes that communication of these criteria with the computer is only possible if they can be stated explicitly in the context of prior definitions. This was also shown to be untrue. A predetermined measure of fitness was not used, nor were the objectives communicated in an explicit statement. Instead this was learned independently as an archetype derived from the examples themselves.

Chapter 10: Discussion: collective creativity

Summary: This chapter argues for the importance of Hypothesis B, in that the dynamics of creative dialogue necessitate communication without standard, a priori definitions. Creativity is discussed as a collaborative process, and interpretation is proposed as the essential creative act. An agent based model is implemented to demonstrate how the machine learning and the methods in preceding chapters may provide a means for the computer to participate in this process.

Until this point, existing examples have been used and reapplied. In their use of precedents, types or concepts, the previous chapters have been essentially *conservative*. This one chapter will focus entirely on the innovative or, as Schön (1963) termed it, *radical*[*]. It will address the creation of new precedents, types and examples, deal with creativity in groups of designers, and examine how the previous work fits in the creative process. Having falsified both Hypotheses A and B, it will make the point that not only do their assumptions about modelling and communication not necessarily hold, but in creative design, they should not hold. The use of approximations derived from precedent (A) and communication via archetypes and examples (B) appear to be valuable, if not essential, parts of the creative process.

Addressing creativity is important because computational processes so easily lend themselves to standardisation of design. For self-conscious designers, even when the demand for innovation is greater, standards are more often set explicitly. The more novel the task, the greater is the perceived need to state these goals and standards in communicating the problem, the less there is on which to reflect, and the more closely it comes to resemble an optimisation problem. A linear design process, from design intent to rationalisation to fabrication, is easily implemented, in these cases. But design is largely (and perhaps increasingly, as Chapter 2) *non-standard*. Traditionally, for non-selfconscious designers, such a linear process does not exist, and standards are formed directly by precedents, allowing gradual modifications of process or form. Is it possible to communicate without standards? The previous results suggest that it is, if the machine can understand precedents.

The social context is therefore also important. Creativity will be discussed as a collective process, rather than that of a single individual. The conservative role of such communication in

[*] Schön (1963) divided his *Displacement of Concepts* into two sections, the first dealing with the *radical* function, i.e. “the emergence of new concepts” (p. 53), the second with the *conservative*, i.e. “carrying over what is old in our theories” (p. 111). The order is reversed here.

design creates the hermeneutical community of “those who share tacit understanding” (Snodgrass & Coyne 2006 p 122). The intersubjective testing (Popper 1959) of design propositions, like any theories, is inherently social. This chapter will attempt to show—in their radical role—how induction and interpretative reflective design are related and that they are crucial for creativity.

A minimal model is thus proposed for emergent creativity (§§10.1–10.3). This deals again with the fact that competent agents can use different underlying process rules, or ways of seeing, but rather than demonstrating the production of similar structure (as in previous chapters dealing with competence), it will be used to test the proposition that this is a sufficient condition for the creation of novelty. A simulation of artificially creative agents is used to explore communication among designers by the production of examples. This will suggest that creativity at a social level is not a result of many individuals trying to be creative at a personal level, but emerges naturally from the social interaction between comparatively simple minds embodied in a complex world.

A discussion of the interface will follow. Complexity has thus far referred to the systems that are the subject of design, but at a higher level the groups of people that participate in the design process itself also form a complex system. As Arthur et al. (1997) argue, parts of a system may be changed, providing the interaction structure is maintained. Computers are not often considered valid creative agents, but their interaction structure at our interface with them is typically as narrow and reduced as Simon’s ant (Simon 1996; and §1.1.1). The methods of communication by precedent and archetype are far more subtle, and preserve the interaction structure we use as designers—of communicating by drawing. The ability to ‘understand’ and create design precedents can arguably give the machine a useful role in the creative group. This prospect will be illustrated (§10.5) in the form of a practical design tool and an outline for how it might work.

10.1 Background: defining creativity

When it is defined with respect to a kind of problem, researchers distinguish creative tasks from non-creative tasks by a lack of knowledge (Rosenman 1997; Cropley 1999); either “knowledge of the problem, of the means of solution, [or] the nature of the solution” (Cropley 1999, p. 517) is absent. It is clear these definitions refer to the same kind of ill-defined, ‘wicked’ problem to which inductive learning has been applied in previous chapters. The creative act itself has been defined in a number of ways and at a range of levels of precision. This section will review these.

10.1.1 A WORKING DEFINITION

Almost all definitions emphasise the generation of *novelty*. Occasionally this is the only requirement, as is the case in a strategy of contrarianism (Cropley 1999; Runco 1999), in which one intentionally runs against the norms of society. This is apparently at odds with the use of norms in Chapters 8 & 9; however, creative work is “more than just original. It is original and fitting.” (Runco 1999, p. 368) The most common definition therefore maintains that creative acts consist of both *novelty* and *utility* (Boden 1990; Gardner 1993; Cropley 1999; Sosa and Gero 2003, Sarkar and Chakrabarti 2008). This utility may be determined objectively by the practical requirements of a design’s function, but it may also be determined by social pressures, fit to fashion or the design’s function in communicating a message (e.g. Saunders and Gero 2001). In the context of Schön discussion, novelty might be considered the *radical* function, and utility the *conservative*.

These goals are often modelled as a two step ‘generate and test’ procedure, in which novelty is created randomly, and solutions are then tested for their utility. This resembles natural evolution and is thus a powerful image, but there is good reason to suspect that creativity in human designers is a very different process. The teleological difference is often acknowledged (natural evolution has no design intent) but models such as genetic algorithms also apply fixed or evolving goals to mimic those assumed in a designer. A more subtle (and possibly more profound) difference is in the relationship between novelty and utility. Natural evolution does produce novelty at random, originally by asexual mutation, which makes relatively small changes to the organism. After several billion years this was augmented by sexual crossover of genes, again at random, which allows larger genetic changes. Design adds yet a third mechanism, unlike the first two in that it does not operate entirely at the level of the gene. As in a morphic language, the past chapters have shown how the *phenotype*, not the genotype, can be reinterpreted at each stage of design production. This adds not only the possibility of further change than is possible via mutation and crossover alone, but also tends to ensure that each solution generated is inherently useful. Popper captured much of this in his observation that “our theories die in our stead” (Popper 1973, p.7), but he was concerned specifically with the act of testing theories that might still be generated at random. Experienced designers would appear to see solutions immediately (Lawson 2006; Akin and Akin 1996; Schön 1963), thereby preventing many unfit ‘theories’ from even being born. In design, then, novelty can be produced faster and it is also—often—assured of utility in the same step. This is very different from the teleological definition of design, as a clear *telos* is unlikely in the case of ill-defined problems, and is not required for reinterpretation*.

For the model to show the importance of inductive learning, the working definition of creativity will be this combination of *novelty* and *utility*. Agents working within a norm or type ensure utility, so the model will need to show that the same processes can also provide sufficient conditions for novelty. The rest of this section outlines some of the mechanisms that have been proposed for this.

10.1.2 Social models

Many models acknowledge the relationship between a creative individual and a larger society. When the generation of novelty is considered, the view is often both that the creativity of the group is a straightforward accumulation of creative leaps of individual people, and that these individuals are varied in ability. Boden (1990), for instance, draws a distinction between the psychological 'P-creativity' of the individual, and the historical 'H-creativity' of ideas that are fundamentally novel for the whole of a culture. Referring to Alan Turing, Friedrich von Kekulé, Mozart, and other historical innovators, she proposes that the capacity for P-creative ideas means there is a good chance for H-creativity. H-creativity is wholly dependent on the P-creativity of these outstanding individuals.

The active influence of society on the individual is often noted in models considering evaluation and judgement of creative ideas. Theories of societal pressure or social impact (Latané 1981) are essentially conservative, in describing the increase in conformity as group size grows, but this creates the norms against which innovative solutions may be judged. As Cropley (1999, p. 519) notes, "'creativity' is not really a property of products or processes at all, but that it is a category of judgement in the minds of observers". Distance in a notional space is typically implied, which is greater between the norm and more innovative ideas or individuals. Aleinikov (1999) suggests this may be measured in different ways, in that creativity is embodied in: a) the individual, b) society, c) instruments, and d) reality, but the picture is of society as a gradually expanding sphere, with individual creative acts pushing at the outer boundaries (Figure 10.1).

* It is possible that 'design' may still more closely resemble natural evolution on this point alone, and differ primarily in the reinterpretation of the phenotype. Creationists would have to abandon use of the term 'intelligent design'.

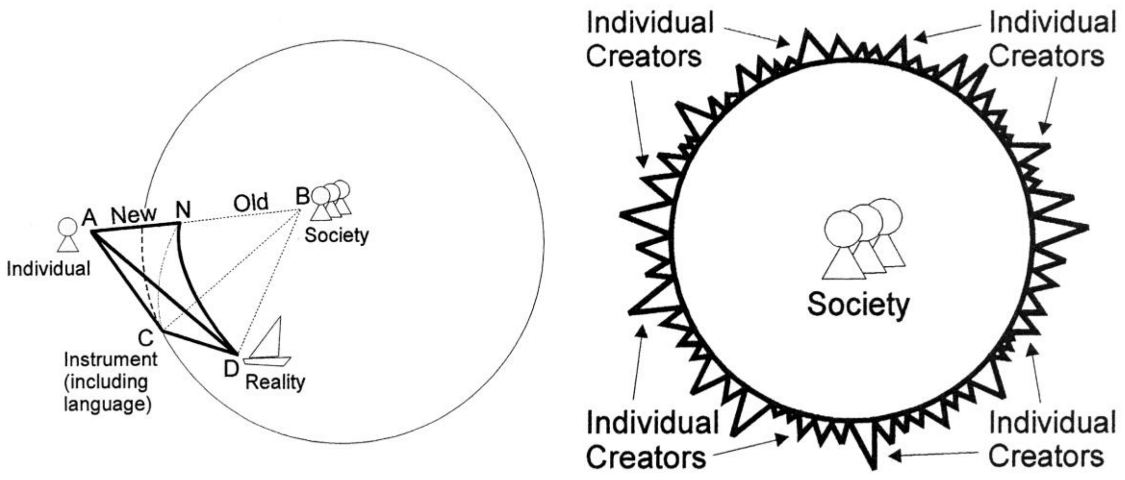


Figure 10.1. Creative acts as expanding the boundaries of existing norms. From Aleinikov (1999).

Such spaces facilitate interaction in computer agent models. The interactive creativity of large groups of designers can be seen as a type of swarm behaviour involving the thoughts of agents (Kennedy & Eberhart, 2001), after social models of the coordinated dynamics of bee swarms, ant colonies (Dorigo, 1997), schools of fish and flocks of birds (Reynolds 1987). These all appear in motion to have a sophisticated group mind. The rules of the algorithms differ in their specifics, but all interactions between agents take one of two basic forms: they either *attract* or *repel* one another in space. The update of an artificial bird's velocity in (Reynolds, 1987), for instance, would be of the form:

$$v_{t+1} = \mu v_t + (1 - \mu)(w_{avoid}v_{avoid} + w_{match}v_{match} + w_{centre}v_{centre}), \qquad (10.1)$$

where v_{match} and v_{centre} cause the agent to imitate the others and v_{avoid} to keep away from its neighbours. Attraction allows for a focused local search and exchange of information with similarly inclined neighbours, and repulsion simply keeps individuals at a minimum distance. While this social cohesion in flocks appears to be highly conservative overall, the model in §10.2 will attempt to demonstrate that a drive to deviate is not necessary to explore new space. Such flocking can therefore appear as both Schön's (1963) conservative and radical functions.

Such techniques have been employed in the study of creativity from the bottom up, by modelling artificial creativity. This is typically seen as a process of interaction between many individuals in a social context, and reveals some of the outcomes expected of real creative groups, such as innovation and cliquing (Saunders and Gero 2001). Axelrod's model of the dissemination of culture (Axelrod, 1997) allows agents in fixed locations to communicate with one another with a probability equal to their cultural similarity, resulting in a polarisation of cultures as these similarities are strengthened. Individuals split into stable non-communicating groups.

In number of more complex models, agents' influence on one another is mediated through embodiment in a shared world, reflecting designers' most obvious influence on one another via their work. Nehaniv and Dautenhahn (1999) suggest an algebraic framework for imitation in which dissimilar bodies can imitate one another by producing similar *effects on the environment*. Individual actions, or internal representations are not important, but rather the ability to meet a series of sub-goals such as covering a wall with paint when imitating the task of painting. Steels' (2000) 'Talking Heads' show that even these goals can be determined, in a point and guessing game played by robots that evolve a language. The two communicate through a shared environment consisting of coloured geometric shapes and a white board. Words and syntax are generated from the need to express concepts the robots may hold differently in their minds, the creativity springing from an interruption in full communication between the two agents. Hutchins' (1995) research with parallel constraint satisfaction networks suggests this interruption is actually beneficial. In highly connected networks of individuals, his populations reached poorer solutions than networks in which individuals were connected only moderately to one another.

10.1.3 CZIKSZENTMIHALYI'S SYSTEMS MODEL OF CREATIVITY

Many of these simulations highlight a possible conflict between novelty (the desire for more distant ideas) and utility (the desire for less distant ideas). By posing a single space of finite dimensionality in which to measure acts of creation, there is little choice but to propose an ideal balance between the two, as suggested by the Wundt curve adapted by Berlyne (Fig 10.2 and 10.3) and used by Saunders and Gero (2001)[*]. This is likely an oversimplification, as there

[*] Wundt's original observation was of the degree of perceived pleasure or displeasure (*Gefühlstones*) as the intensity of a stimulus (*Reizgrössen*) increases (Wundt 1874 p 452). The bell shaped curve gradually increases from an initial threshold of zero to a maximum point, then returns to zero and eventually to a maximal (infinite) displeasure at the point of maximum stimulation. Berlyne (1971) noted examples of the phenomenon in aesthetic judgements to basic stimuli such as musical tone volume or frequency (Vitz 1972) or the size of simple shapes (Martin 1906), but proposed in addition that "the horizontal axis presents not merely stimulus intensity but arousal potential, [including] collective properties like novelty and complexity." It is thus used to explain (e.g.) the historical change in acceptability of a piece of music (Skaife 1967), from unpleasant to pleasant to indifferent with repeated exposure (Berlyne 1971 p. 193). Berlyne also notes the similarity to curves explaining novelty in McClelland et al. (1953). Berlyne's construction of the curve is based on the identification of hedonic centres in the brain, numerous observations that the aversion centre inhibits the reward centre (ibid. p. 84), and that the aversion centre has a higher activation threshold (ibid. p. 88). Under the assumption that neural thresholds in each are normally distributed, the cumulative distribution function of each is an ogival curve (nearly a logistic sigmoid, but with thinner tails), the sum of which produces a curve resembling Wundt's, except levelling off asymptotically to a stable negative hedonic value rather than increasing asymptotically toward an infinite displeasure at a maximal stimulus intensity. It is a variation of this curve (with logistic sigmoids) used by Saunders and Gero (2001).

seems little justification for rejecting a highly useful solution in reality as uncreative simply because it is ‘too novel’—utility and novelty need not necessarily be measured on the same scale.

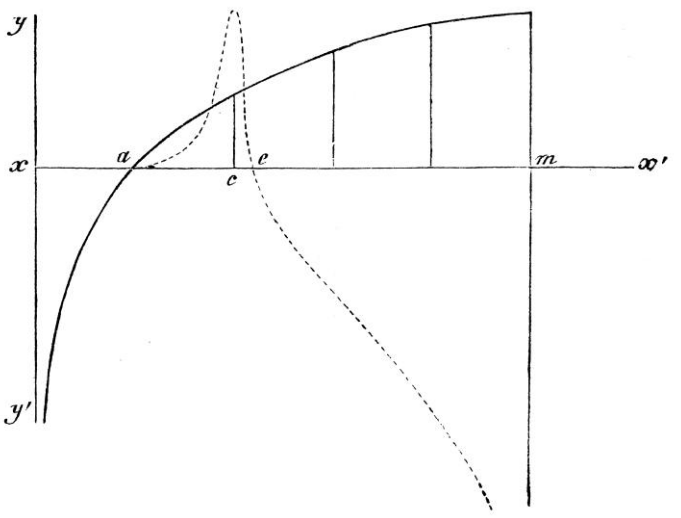


Figure 10.2. Wundt’s (1874) diagram of pleasure and displeasure with increasing stimulus intensity. Pleasure increases to a maximum at point *c*, then decreases to *e*, after which displeasureable sensation increases asymptotically to infinity at the maximum stimulus *m*.

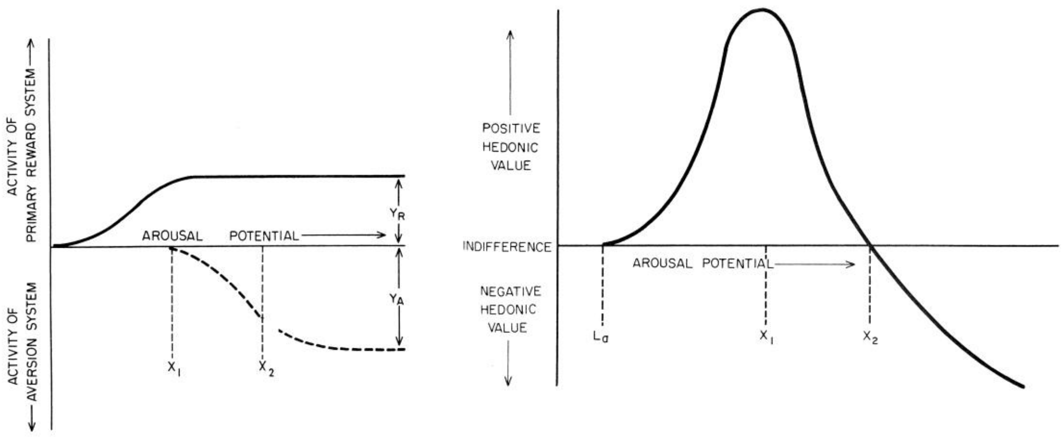


Figure 10.3. Berlyne’s (1971) adaptation of a similar curve has been used to explain preferences for moderate amounts of novelty. The curve is constructed from two separate effects of reward and aversion (left) which sum to yield the bell shape (right).

Instead of relying exclusively on a single curve, however, the derivation of the curve from several disparate sources in the brain suggests that different dimensions of stimuli might be relevant, and the conflation even of two curves into one is a simplification that Berlyne frequently unpacks in his description of specific phenomena. A multidimensional description may be more valuable. Moreover, the use of the curve to explain novelty has been criticised (Machotka 1980) as it requires reading the curve backwards over time, and for a lack of clarity in experimental results. Machotka (ibid.) suggests instead a more subtle set of hypotheses for different levels of complexity, an alternative that also appears similar to the use of more than one dimension of measurement.

An alternative structure suggests that different measurements may be made of an idea's novelty or utility by different parties, and that these do not necessarily conflict. Czikszentmihalyi's systems model presents a dynamic account of the process and structure of creativity within a broader environment of other individuals (Figure 10.4). This models the flow of ideas and interaction between a *person* (the creative individual), the *field* (the group of individuals that act as arbiters of creative output) and the *domain* (the collection of embodied work and symbolic representations deemed relevant by the field) (Czikszentmihalyi 1988). These three parts are each very different in structure, and consequently different in function. Communication is in one direction, and the structure and purpose of messages also changes according to which parts of the model are communicating.

As Czikszentmihalyi makes clear, the creative act is not an occurrence within the mind of an isolated individual, but an interaction with the domain and field, both of which are spaces outside the individual's private perception, and both of which may be shared by other individuals. Activity in each section of the triangle thus takes place in one of three very different spaces, each of which can be mapped into the next as indicated by the arrows. As opposed to seeing social H-creativity as an aggregate of individual P-creativity, all creativity is an emergent result arising from the process of communication from one to the next. The model to be proposed here (§10.2) will take this form. In mapping from one distinct, abstract space to the next, the simulated system should appear to exhibit creative behaviours—namely coalescence to socially established norms, clique differentiation, and cultural innovation.

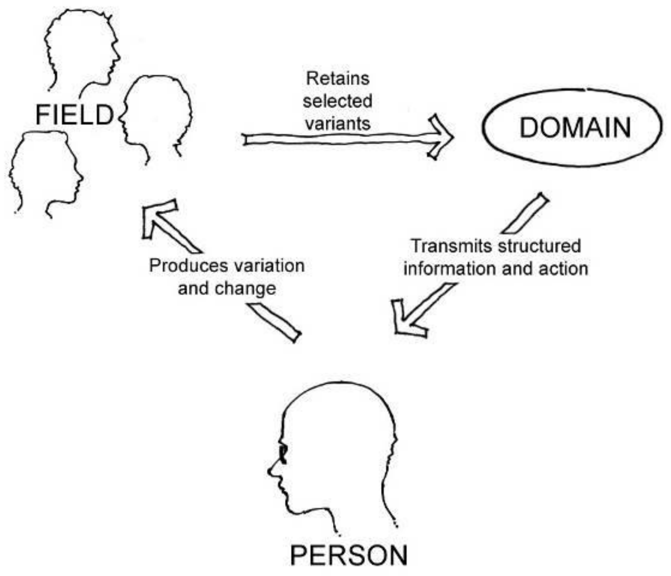

FIELD
Retains selected variants
DOMAIN
Transmits structured information and action
Produces variation and change
PERSON

Figure 10.4. Czikszentmihalyi's systems model of creativity.

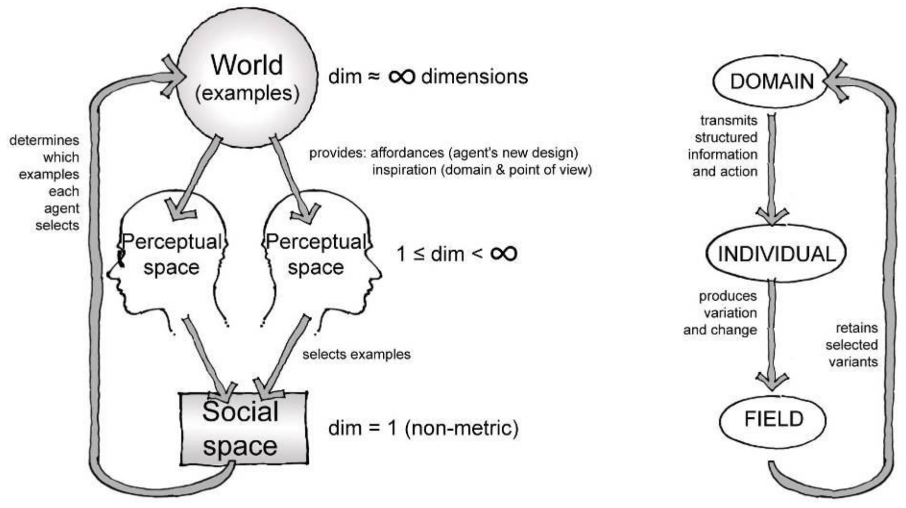

World (examples)
dim ≈ ∞ dimensions
determines which examples each agent selects
provides: affordances (agent's new design) inspiration (domain & point of view)
Perceptual space
Perceptual space
1 ≤ dim < ∞
selects examples
Social space
dim = 1 (non-metric)
DOMAIN
transmits structured information and action
INDIVIDUAL
produces variation and change
retains selected variants
FIELD

Figure 10.5. Three spaces of the model (left) paired with Czikszentmihalyi's (right).

10.1.4 Small paradigms

Even if it is emergent and dependent on the structure as a whole, the mechanism of the creative act still requires explanation. Two types of creativity are often considered separately, as everyday 'small-c' creativity versus 'big-C' creative breakthroughs (Gardner 1993; Boden 1990). The latter is the more obvious but not necessarily the most important. Metaphor is frequently cited as an example of an everyday creative act (Gibbs 1999), and possibly as the core of all understanding (Lakoff and Johnson 1999). Described as a new mapping across existing domains (Lakoff and Johnson 1999), it resembles the changing of frames of reference (Akin and Akin 1996) or displacement of concepts (Schön 1963) in that a real example is seen in a new way. A 'big-C' breakthrough is what Kuhn (1962) describes as a change in paradigm, which causes a great deal of phenomena to be seen in a new way. Kuhn apparently draws a qualitative distinction between the big-C and the small in pointing out that (small-c) differences of opinion don't violate the paradigm itself, and are within the practice of a different kind of activity he calls "normal science" or "puzzle solving".

But the difference between big and small-c may be simply one of scale. Different points of view may be just small paradigms, as apart from naming the smaller activity 'puzzle solving' Kuhn draws no definite threshold between the small and large differences of viewpoint. Kuhn's account describes the social process of paradigm change, in that paradigms can only be overthrown when challenged by a competing paradigm. Likewise, a smaller scale personal point of view will likely persist unless encountered by a different point of view, communicated explicitly or via examples of phenomena. As Wittgenstein (1958) et al. describe, a theory-hypothesis-meaning is never actually shared exactly by two individuals, just as Kuhn (1962) acknowledges that a paradigm may be interpreted somewhat differently at various levels by those working on ‘normal science’ within it. It would appear that the distinction between a point of view and a paradigm is linked in the same way to language—that is, if the shared conventions and means of communicating about phenomena are not violated by the differences of opinion, then it is simply too small to affect the paradigm; smaller still, and it may not even be communicable at all. To a significant extent, this distinction then is only in the eye of the practitioner.

10.1.5 Seeing-as: ‘novelty + utility’ is not a paradox

The points emphasised by Kuhn relate to *seeing* and perception, rather than production. One of these is the role the paradigm plays in our ability to interpret phenomena: “something like a paradigm is necessary for perception itself” (Kuhn 1962, p.93). In this sense, the low dimensional feature space Φ describes a paradigm. Its low-dimensionality is inherently limited to only partial descriptions of phenomena in the world just as a theory or individual point of view can not cover all dimensions of reality. “Paradigms determine large areas of experience” and therefore make it impossible to speak about observations in a pure and objective language (Kuhn 1962, Chapter 10).

Also of relevance is the manner in which a paradigm is discovered and communicated. In the absence of a pure language of observation it is impossible to speak of events impartially, but what is communicated are repeatable phenomena of key experiments. It is these example phenomena that determine a paradigm, just as examples determine via supervised learning the construction of Φ. In neither case is there anything resembling Popper’s (1979; §1.2) ‘bucket theory’ of mind, but at each observation examples are seen in the context of an existing paradigm, point of view or Φ, and that way of seeing can ultimately be traced back to the previous history of example phenomena. The creative moment in all of this, the creation of a new paradigm in the mind of the scientist, is described as emerging all at once (Kuhn 1962 p. 74) but otherwise likely to be ‘permanently inscrutable’. This is not so difficult a problem if one considers it not a process of making (which requires time) but a simple act of seeing. With slightly varied individuals looking at a crucial phenomenon, one is likely to already have a point of view that is predisposed to understanding it in a different way. If this different way happens to be more effective at explaining other phenomena, it may well become a new paradigm. Kuhn’s following observation that “almost always the men who achieve this have been very young or new to the field" (p. 74) and similar evidence by Gardner (1993) et al. support this, in that these are exactly the people most likely to bring with them an alternative way of seeing.

This addresses several of the otherwise mysterious or outright paradoxical aspects of creativity. Cropley (1999, p. 522) lists the four commonly noted phases of the creative process as: (1) Information; (2) Incubation; (3) Illumination; and (4) Verification. Points two and three are the most private, so the least understood. The third, Illumination, is the point mentioned above at which the idea appears seemingly all at once. Incubation is claimed to require 'intuition' and 'meta cognitive processes' to narrow down an otherwise extremely inefficient search (p. 522), but these intuitive judgements would be natural for someone with access to consistent ways of seeing outside a dominant paradigm, and the more the better. Cropley (p. 524) lists a number of apparent paradoxes, a few of which are:

> " a) creativity involves difference from the everyday, but is found in everybody;
> b) novelty, the single essential element in creativity, is necessary but not sufficient to define it;
> ...
> d) creative production requires deep knowledge, but freedom from its constraints;
> ...
> f) creativity requires deviating from social norms, but doing this in a way that the society can tolerate."

His suggested resolution is again based on the above phase model, in that conflicting parts of paradox simply occur at different times in process of creativity. However, none of these are in fact paradoxical if creative insight is simply a result of interaction between differing points of view on observed phenomena.

Central to the explanation is the premise that something considered totally novel by the rest of the group might be the natural result of one person's differing viewpoint (Φ) and therefore seem completely normal. There is thus no paradox and no conflict between novelty and utility. Creativity is not simple novelty, because every novel idea has a set of perfectly valid reasons for it, based on the phenomena the formed that creative person's point of view in the first place. That person simply convinces others of these. In science this is done by empirical testing of experiments; in the arts the reasons are perhaps less clear and require greater discussion. In both cases the creativity of the idea is not judged by the artist or scientist who proposed it, but by those who later change their way of seeing to reflect it.

10.2 Social agents: structure of a creative system

Because an individual person is so complex, only “the interaction structure through which they act” (Arthur et al. 1997, p. 9) of a creative group can be modelled. A systems approach to creativity (Czikszentmihalyi 1988) is used here to inform this structure, and then to draw some conclusions as to how the learning and optimisation methods developed thus far might be applied to form useful design tools. It will test that simple differences in (Φ) are sufficient to generate novelty. The proposition is as above, that group innovation is not based on randomness (as in stochastic optimisation), or even an internal drive to generate novelty (as Saunders and Gero 2001), but an unintentional change in how an agent perceives the world around us and how it interacts with others.

The cellular grid of open and solid spaces used in previous chapters is used again, as are the same learning algorithms. The difference is that each agent is able to interact with others and contribute to an overall building pattern in a shared world. A training of the network is necessary in supervised learning, but the explicit labels that accompany the data set in earlier implementations can be considered to occur quite naturally in a social setting. For problems in which the goals are obvious (e.g. structural stiffness) they can be immediately seen in the performance of examples, but for more subtle design situations these labels may actually come from a variety of social cues, from verbal statements, body language, situational context in which examples are seen. It is this subjective, social communication that the model aims to simulate.

10.2.1 STRUCTURE OF THE MODEL

Three distinct spaces, connected in the manner of Czikszentmihalyi’s model, form the arena for the system’s dynamics. They correspond generally with its *domain, person* and *field* (Figure 10.5), but are defined somewhat more broadly in terms of what they can contain:

- *Space of the world*: Beginning with our shared, physical reality, this contains the set of embodied artefacts or events that can be experienced by all. They are the only ‘objective’ part of the model. This differs from the *domain* in Czikszentmihalyi’s model, as the domain is only a subset of accepted works. In a multi-agent system however, there may be multiple fields, thus multiple domains containing different artefacts. In addition, the relative value of a range of work, good and bad, is capable of influencing an individual, so all works will be retained. The whole space is called the *world*, leaving *domain* to refer to selected sets within it.

- *Perceptual space*: Next, a subset of this reality is experienced by a given individual as subjective experience in a second space of lower dimensionality, or *perceptual space*—identical to the lower dimensional feature space Φ in previous chapters. An individual's perceptual space is unique, and therefore yields a unique picture of the objective *world*.
- *Social space*: The third space is the space in which social dynamics can be seen: the contextual space of a shared culture. This is not necessarily a metric space, but can be represented as a graph of distances between individuals. The group of individuals closest to an agent would be that agent's *field*.

Communication occurs between these three spaces in the direction indicated by Czikszentmihalyi. By doing so, creativity will not occur in any one in isolation, but in the cycle of mapping from one to the next.

As this communication occurs, a point in one space can be mapped to the next space in the model. Due to the structure and dimensionality of the spaces, however, this can only occur in one direction. Beginning with the potentially infinite dimensionality of the space of the world, an example artefact can be mapped to a point in the lower dimensional perceptual space in the familiar manner. In the next, social space, dimensionality has been reduced further to a single linear distance. At this point, the task of mapping back to the original, high-dimensional world to complete the loop would appear problematic, but not if the act of design is again considered as a selection of affordances as introduced in section 3.6.3, and implemented in chapters 8 and 9. Communication between one individual and another requires the completion of a full cycle, the first using the conventions of their social space to make an example in the world that the second individual can see with a different perceptual framework.

10.2.2 Space of the world

The grid aggregation process introduced in Chapter 8 is used as the design environment. The example units placed in their 7×7 square neighbourhoods are the possible items in the domain. Because the sensory input of each agent is an identical 49 square grid, each unique example can be represented by a point in a 49-dimensional space, a 2-d projection of which is shown again in Figure 10.6. All example neighbourhoods are projected onto the first two principle components of the set: neighbourhoods of the straight rows are indicated by '×', and the random culture by '○' markers in the centre. This full space of all possible examples is the space of the world. Its high dimensionality is a chief advantage of being embodied in the world—in our world an object in the domain such as a painting or sculpture can be revisited again and again and always be seen as something new.

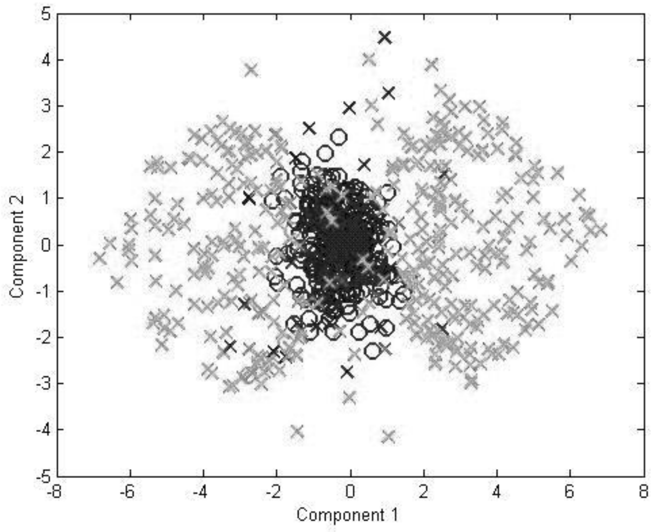


Figure 10.6. A projection of examples in a 49-dimensional world space.

10.2.3 Perceptual space

The mapping of examples to a feature space Φ, as in the previous chapters, corresponded to a particular way of seeing or measurement of each example—the paradigm or *perceptual* space. Perceptual space, at any one time, is necessarily a lower-dimensional abstraction of the world space, which determines what we actually *see* in the world. In the case of the design agents in sections 8.3 and 9.5, it is the one-dimensional output of the SVM or MLP. Any example in the real world can be mapped (via the learning algorithm) to a corresponding point in the space of an *individual's* perceptual space, which is different from every other individual's. Unlike the shared space if the world, this perception is completely private.

A perceptual space is represented in us by the state of our brain when experiencing the example, and so the mapping is not an 'internal representation' in the strict sense (Clark 1998, p. 168). The nature of this space is like what has been termed *semantic space* by linguists, in which words and concepts are mapped such that words with similar meanings are located near one another. The Hyperspace Analogue to Language (HAL) model (Lund & Burgess 1996, Burgess & Lund 1997) constructs such a space based on the proximity of words to one another in Usenet discussions in which nouns such as 'dog' and 'cat' fall into one cluster, 'china' and 'america' together in another in a manner that coincides intuitively with many people's experience of the concept. This perceptual framework is the subjective counterpart to the space of external examples, the interior space of qualia or private meaning. A thing may exist in the world independently, but is actually experienced in the perceptual space of the mind.

Neurons react to *changes* in stimulus (Adrian, 1928), and perception is therefore at root an act of differentiation. The minimal requirement of attempting to see or understand the world as clearly as possible drives an agent to distinguish between observations just as classification algorithms attempt to find a clear division between clusters. The differentiation of examples can be implemented here in the perceptual space via a feedforward MLP as used in previous chapters. Training of the network serves to usefully illustrate this movement, as the function of every neuron:

$$\mathbf{y} = \mathbf{wx} + w_{0}, \qquad (10.2)$$

is a simple linear function, and can be visualised as a hyperplane constantly moving in a high dimensional space in an attempt to separate the two classes of input examples. The agent's point of view can be imagined as aligned to this hyperplane and moving with it as though it were a standard projective view of a moving camera. The final output of the MLP is the agent's perceptual space, and may be of arbitrary dimensionality, but was implemented in this case as a single dimension as in the archetype yielding examples in section 8.3.

10.2.4 Social space

The group of other agents with which one is associated, and which determines examples in the domain, constitutes the *field*. It is not a single, constant group of judges, but a fluid, changing collection of individuals selected based on how similarly they see the world. To measure these connections between individuals the model utilises a third, *social space*, actually a simple one dimensional measure of distance between agents.

In models with very simple agents (e.g. Axelrod 1997) the likelihood of interaction is based on a measurement of an agent's single internal state or location. Here, the perceptual space is used. The perceptual spaces display a measure of distance between what the agent judges to be the current mean of its culture (0) and any given sample. After normalising these to have an identical variance of 1, the distance between any two agents' perceptions can be measured based on the mean of squared differences between each of the example points:

$$D(a_1, a_2) = \sum_{i=1}^{numExamples} \frac{\left[p_{a_1}(ex_i) - p_{a_2}(ex_i)\right]^2}{numExamples}$$

(10.3)

All learning algorithms that result in a perceptual mapping can be compared in this way, regardless of their internal workings. To illustrate, Figure 10.7 shows the result of several very different learning algorithms exposed to the same set of examples. Although each may differ in the details, each individual shares with the others the ability to perceive examples in the world. These are the same algorithms compared in section 8.3.4: an MLP trained only on a subset of samples, a second MLP in which inverse weights are used for non-archetypal samples, a Kohonen SOM (Kohonen 1982) and a SVM (Vapnik 1995). All of these different algorithms, when trained on the same examples, result in different outputs in their individual perceptual spaces (which may even have different numbers of dimensions) but each is alike in that the resulting perceptual framework allows the individual to make similar decisions about examples regarding their distance from the mean.

The group of other agents that each considers to be its *field* is determined by the distances measured as in (eq. 10.3), each selecting the agents perceived to be the closest. Table 1 shows the distances between the perceptual spaces of the four agents in Figure 10.7. Based on these distances, the second neural network agent and the Kohonen agent would each select the other as a member of their fields, being the closest of the possible choices. The similarity is also evident in the visual appearance of their construction outputs (Figure 10.7, right).

Kuhn (1962) emphasizes the role of a field's shared conventions and language both in facilitating internal communication, and in isolating it from outsiders. As the boundaries of a field are defined by shared examples that define it, the field itself can actually be considered to have a perceptual framework of its own: the perceptual space that would be defined by those examples (Figure 10.8). If two individuals each select a set of six or seven examples as representative of their individual ideals they might each have one or two unique samples, but also a general overlap in the remaining 5 which will define the individuals' shared field, and its agreed perceptual space. Also, because the field ignores examples outside those chosen, its distance to the perceptual spaces of each individual could be zero, even if these individuals would differ in mapping examples out of the field; in Kuhn's terms neither would be challenging the paradigm. This field's perceptual space therefore coincides to a fairly high degree with the perceptual spaces of each individual in the field, providing the common ground that makes communication and mutual understanding possible.

It is the conservative function of the shared examples that allows individuals' perceptual spaces to become more aligned. The radical function is a result of their subtle misalignment, and the unshared phenomena, which can result in innovation. The model should demonstrate this.

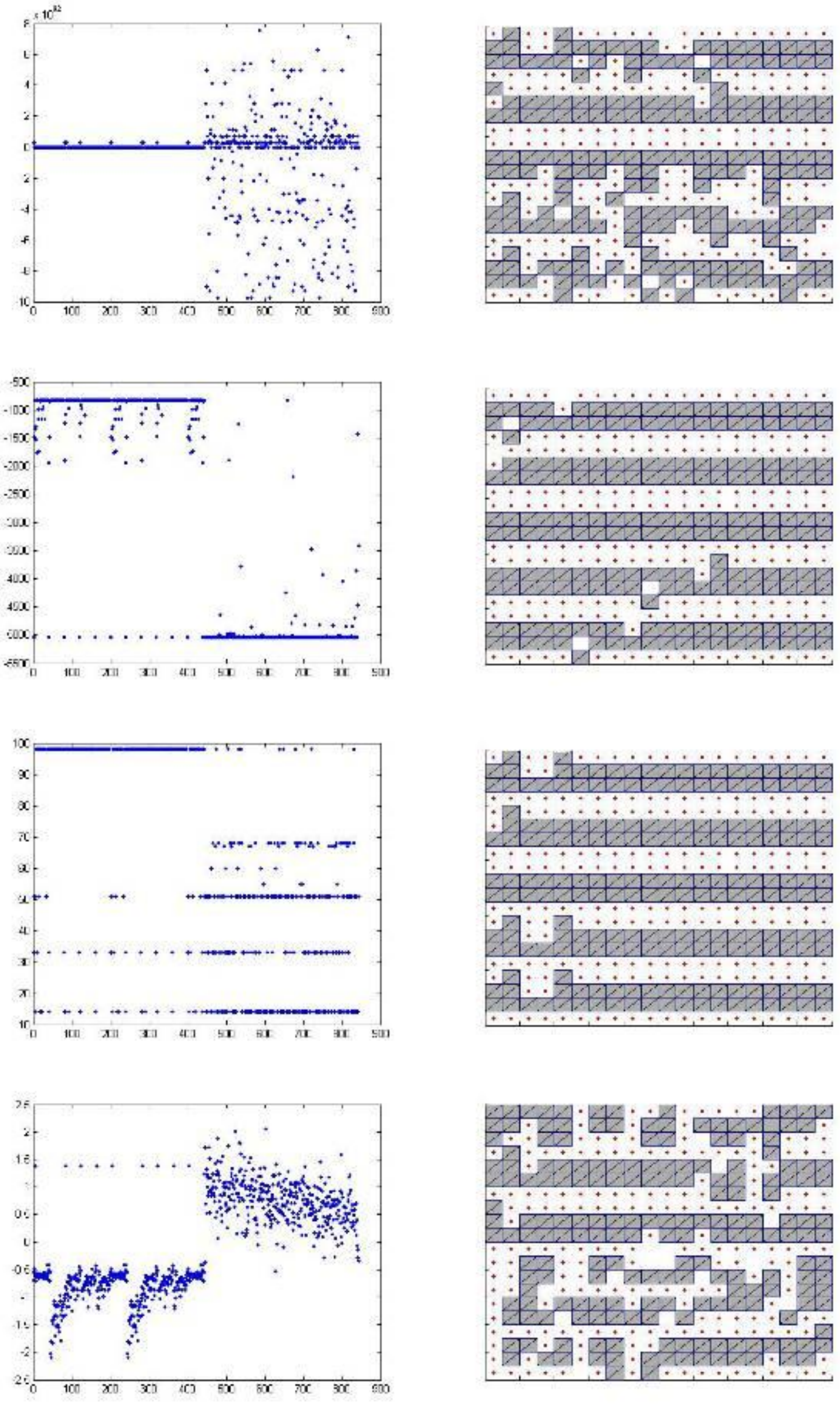


Figure 10.7. Agents trained with completely different algorithms (two three-layer neural networks, one Kohonen network and one SVM) have different perceptual spaces (left) but can make similar evaluations and similar constructions (right).

Table 10.1. Social space distances measured between the above perceptual spaces.

	neural net 1	neural net 2	Kohonen	SVM
neural net 1	-	1.47	1.54	1.62
neural net 2	1.47	-	0.44	1.95
Kohonen	1.54	0.44	-	1.90
SVM	1.62	1.95	1.90	-

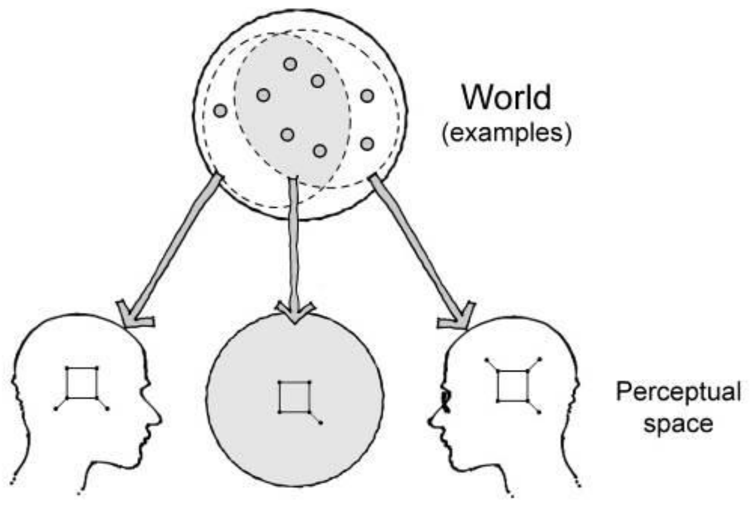


Figure 10.8. The field's shared perceptual space is defined by shared examples.

10.3 Putting agents together: behaviour of the model

At the macroscopic level, creativity is expected to coincide with certain behaviour of large communities. The behaviour of a group of agents in the model should exhibit certain characteristics associated with creative social systems: innovation, local coalescence toward accepted ideals, and clique formation.

- *Novelty as distance*: Innovation has two requirements, the most obvious being the generation of novelty in the world space of possibilities, a prerequisite for both P and H-creativity (Boden 1993). Over long spans of time, this inevitably results in a shift in cultural norms, fashions or styles, as an accumulation of many small innovations. If one imagines a theoretical space of 'all possible designs', (illustrated schematically in Figure 10.9) then the artefacts thus far produced in the history of humanity would fill only a small cloud, leaving vast expanses of the space still unexplored. But that cloud is always expanding around its periphery (as in Aleinikov 1999), and over time the system should expand in the world space of possibilities, to explore it broadly.

- *Novelty as unpredictability*: The second requirement of creative innovation is that the results are not just new, but also unexpected or surprising (Boden 1993, p.30). Many swarm algorithms (Dorigo 1997, Kennedy and Eberhart 2001) and cultural simulations (Axelrod 1997, Saunders and Gero 2001, Sosa and Gero 2002) incorporate randomness to generate novelty, whereas others (Reynolds 1987, Wolfram 1994) produce a complex or chaotic overall behaviour from the interactions of deterministic agents. Because this model is proposing an emergent creativity in the interactions *between* groups rather than explicitly novelty-seeking agents, it follows the second approach. Although no stochastic algorithms are used for sampling or training, viewed over time the system should display apparent randomness and unpredictability, both in the examples chosen (or created) by each agent, and in changes of field in the social space.

- *Social influence*: Like the agreement predicted by Latané's (1981) theory or the polarisations seen in Axelrod's (1997) models, subgroups in a social system will be expected to coalesce toward some local consensus. This is the basic behaviour of swarm models. This swarming tendency among people is necessary because we are social – there is a biological need to coalesce and an intellectual need to understand one another.

- *Cliques*: Finally, cliquing tendencies also split us up, often causing the bitterest of disagreements between seemingly close individuals. Saunders and Gero (2001) demonstrate the influence of groups of artificially creative agents on one another in the creation of fractal art, showing that agents with similar desires for novelty tend to form cliques. The effect is also revealed in genetic evidence: Bodmer and Cavalli-Sforza (1976) note that 93% of genetic differences occur within races, and the 7% genetic differences between them are weighted toward visible characteristics. These visible differences are explained by a sexual selection that penalizes traits identified in one's neighbours outside the group, thereby accentuating the visible differences of adjacent groups and increasing the cultural divide.

This divergent behaviour the creative social system should display is often pictured as a branching tree, such as that of languages (Figure 10.10) or the increasing specialisation of scientific disciplines from a common trunk of enlightenment natural philosophy, but there is also the possibility of merging branches. Kroeber (1948) notes this pattern as characteristic of cultural, as opposed to organic, phylogeny (Figure 10.11). English, for instance, has had major contributions from neighbours on the Latin, Britannic and Northern Germanic branches as well as influence from many others, and there are the so called interdisciplinary fields like biochemistry, which combine major branches of science. Rather than the clearly defined branches of the diagram then, the whole is a diverging and converging collection of loosely connected individuals clustered around individual foci, and this is how a system of agents is expected to behave over time.

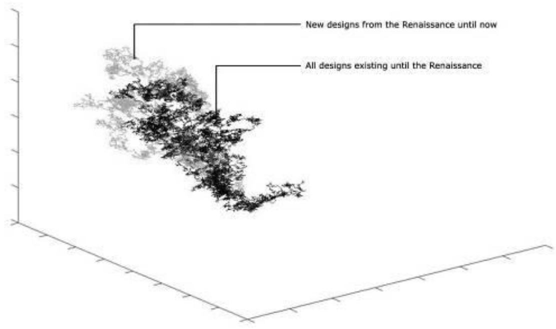


Figure 10.9. Examples in the space of all possible designs.

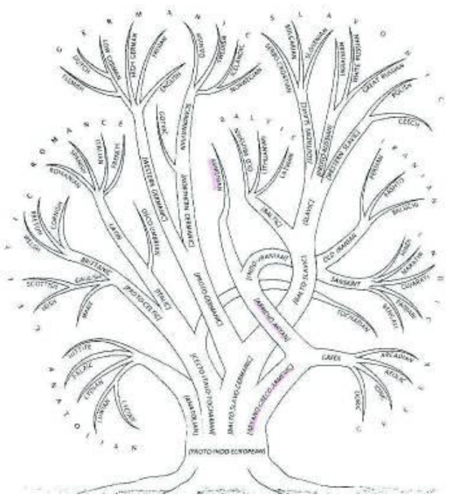


Figure 10.10. Indo-European languages. Image Gamkrelidze and Ivanov (1990).

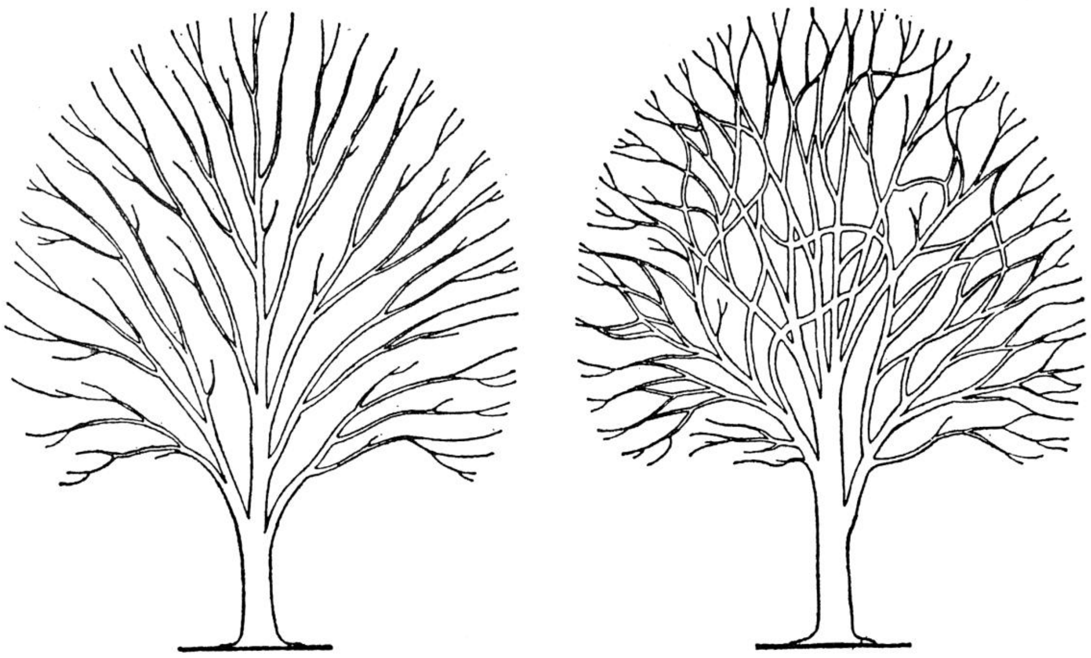


Figure 10.11. Organic phylogeny branches over large time scales, but cultural traits can merge with one another. Image: Kroeber (1948).

10.3.1 Testing the swarm

Agents interact by sharing examples in a common world, and indicating their perceptions to others with whom they are close in the social space. The interaction of a group of agents is tested in this section. Each agent cycles through the following steps, one in each of the spaces described in section 10.2:

1. Get a new set of n_d examples from the *world* that determine the agent's domain, and a set of n_o examples outside the domain.
2. Train the neural network so that it distinguishes the chosen n_d domain examples from the other n_o examples as the agent sees them. This adjusts the *perceptual space* of the agent. It can communicate its current position to all others by selecting an example from all available affordances in the world space nearest to what it perceives is its current ideal.
3. The similarity in the *social space* determines which agents communicate with one another. An agent selects the new set of samples for step 1 from the set pointed to by this group (the field), including the other agents' newest designs in its new domain.

The above steps accomplish the mapping between the three spaces. Examples in the world are mapped to a lower-dimensional perceptual space, and distances between perceptual spaces are mapped to a single dimension in the social space. The cycle reiterates as proximity in the social space determines, or points to, high-dimensional examples in the world. This repeated process causes agents to constantly adjust their perceptual framework to accommodate new examples in the domain. The results of the model test indicate this motivates an overall cultural change, as represented by variation in the building output of a particular field of agents.

10.3.2 System behaviour: innovation and cliques

When the agents operate within the shared constraints of the building aggregation model, the results of their building preferences can be examined in relation to their social field. Figure 10.12 shows the result of a group's interaction over time according to the above rules. It plots an arbitrary (one-dimensional) projection of the domain means for each agent in their shared world on the vertical axis against time on the horizontal. The details of this appear quite different depending on the axis of projection chosen (just as they would appear different again in each agent's perceptual space), but the overall characteristics are the same. There is a gradual expansion from a common start, as agents' work explores the space of options.

Branching into cliques can be seen (Figure 10.12), each made up of many units produced by several agents. There is individual movement between them, and no strict definition of membership, but after 30 cycles two major groups appear, typified by agents 3 and 4. The building patterns produced by each of these agents in isolation at cycle 30 are shown in Figure 10.13. All agents in the clique represented by agent 3 display a tendency to build in a similar (but not identical) radial network, and those in the other clique in rough horizontal rows. An examination of the *fields* chosen by agents 3 and 4 also reveals that they are mutually exclusive, sharing no agents in common.

Innovation has occurred in the form of cultural change, in that the preferences dictated by the changing perceptual spaces of the agents cause them to build differently over time. At cycle 1 the building pattern of every agent is identical, but at cycle 30 neither of these building patterns is exactly like the others, or like the initial starting point.

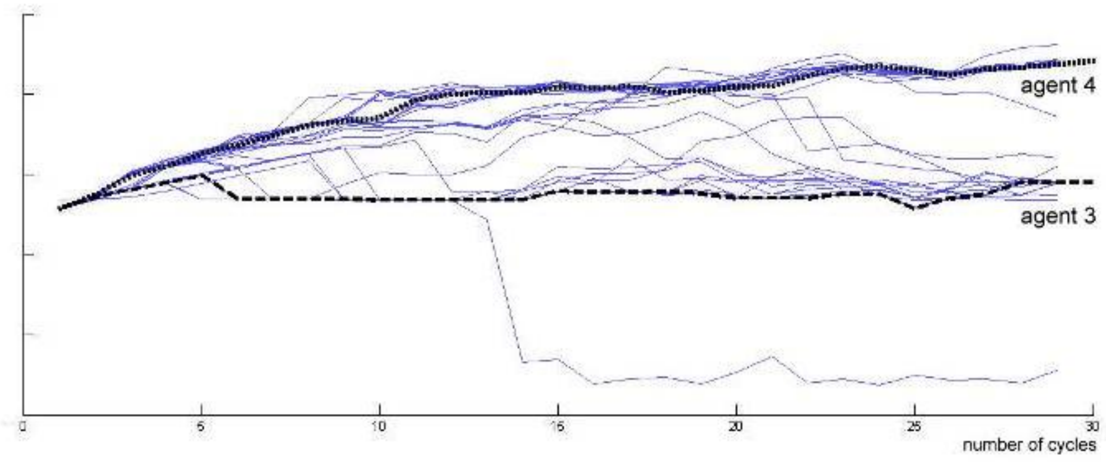


Figure 10.12. A projection of the agents' ideal means in the world space over time.

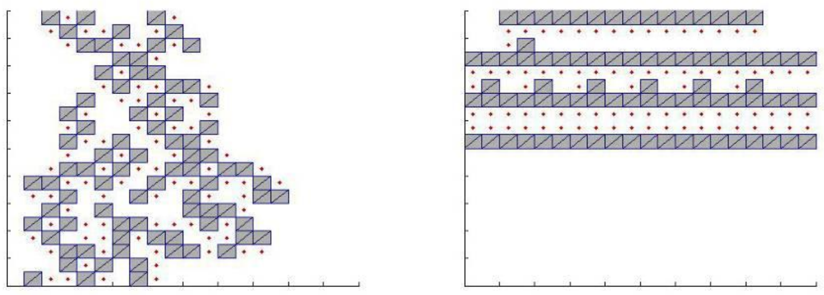


Figure 10.13. Examples of building patterns from two different cliques: agents 3 (left) and 4 (right).

10.3.3 The effect of different points of view

The hypothesis that different perceptual spaces, different ways of seeing the world, are responsible for this innovation was tested in the model in three different runs from the same initial domain. A simplified version was used: the world space was limited to five dimensions so the simulation could be run faster and for a longer time period; and the agents' building affordances were unrestricted (i.e. their output can be any point in this five-dimensional space) to ensure the change was not inherent in the structure of these building affordances.

Figure 10.14(a) shows the result of all agents locked to identical perceptual spaces. There is no difference to the way they see the world, and their ability as a society to explore a broad region of the world space is extremely low. (The ticks on the vertical axis in this case are actually single units rather than hundreds.) All agents have the same internal complexity as one another, and those in the other runs. The only difference in Figure 10.14(b) is the fact that the agents view the world differently from one another, but this produces both overall change and innovation of the group, and far more internal variation and complexity. The individual lines on the plots were found at any given time to contain between one and ten agents. Some can cross, particularly in the dense first 300 cycles. In doing so there is both a branching and merging of a loose collection of individuals. Even when these seem to be in clear fields an apparently interdisciplinary individual can be seen to cut across several branches by being near examples of both in its own perceptual space.

In Figure 10.14(c), the complexity is seen to further increase with the addition of new agents to the culture over time. Every ten time steps a new agent designer was added to the population. To avoid direct introduction of novelty, it was specifically placed in an existing field by being given a perceptual space and a domain of examples identical to an existing agent. Even with this initial similarity, there is a far greater branching and innovation of work than seen in the previous run. Fixed parameters of n_d and n_o ensure that a given field will eventually saturate when the number of examples exceeds their capacity to view, causing a split. This process occurs continuously, then, as long as the population increases.

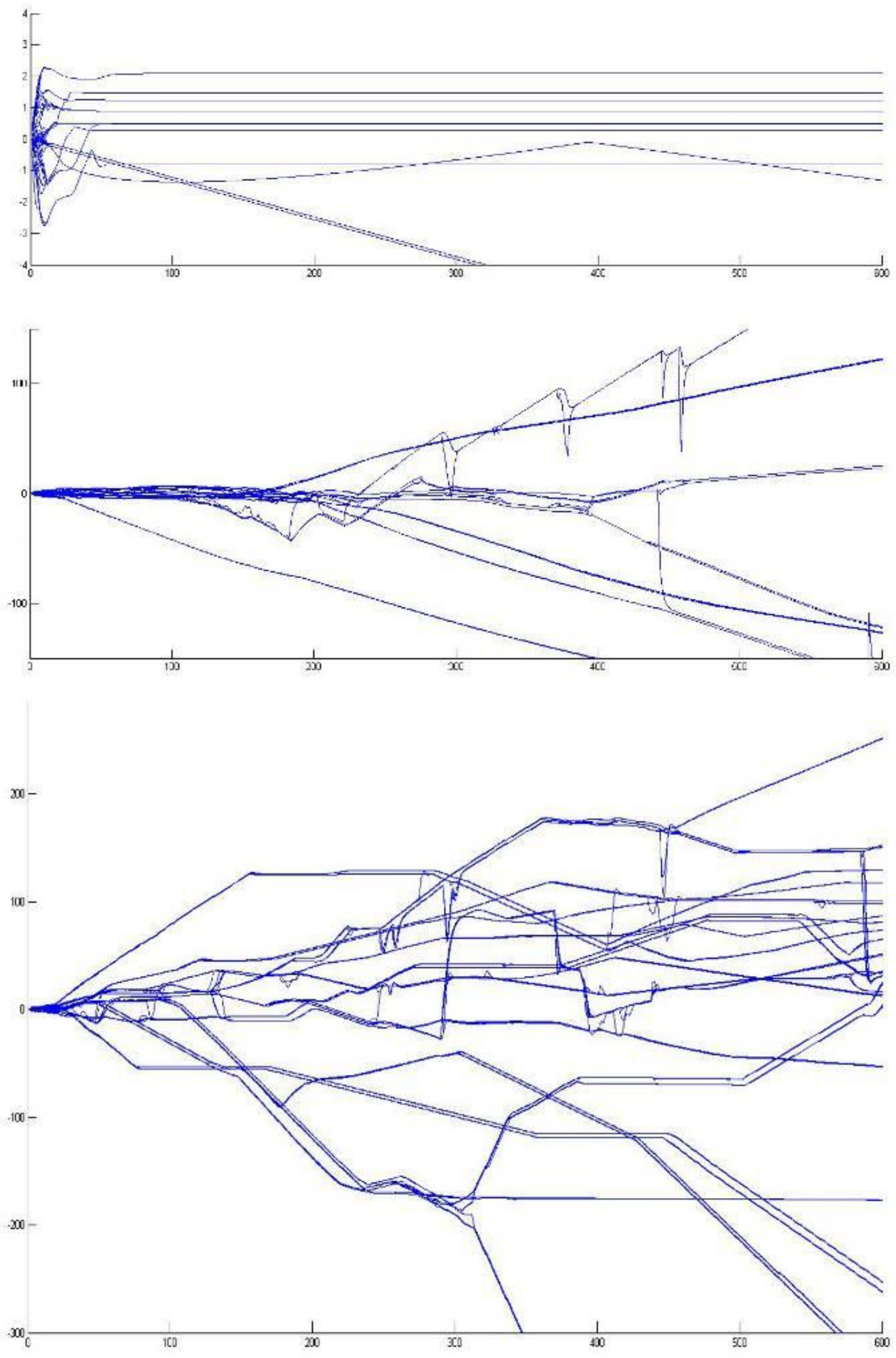


Figure 10.14. Agents' ideals in the real world (vertical) over time (horizontal):
a) (top) Identical perceptual framework in all agents,
b) (centre) Each agent with different perceptual frameworks,
c) (bottom) New agents added over time.

10.3.4 The interdisciplinary individual

An agent seen moving across fields only appears so from outside its own perceptual framework. It does not follow a middle line half-way between the two branches, because in some dimensions these two are very closely aligned. Such agents are found to include examples of both in a single domain, although they may look distinct from other agents' points of view.

This is a plausible description of what we tend to call interdisciplinary work. Different disciplines have their own unique languages and customs that allow for ease of communication and rapid exchange within the community, but misunderstandings and general lack of exchange between groups. The more a discipline becomes 'specialised', the greater is its ability to make progress (i.e. innovation or movement in the space of the world) on the problem at hand, but less its ability to communicate to outsiders. Interdisciplinary work results when individuals from different disciplines approach one another in dimensions that are not part of the general perceptual framework of either group. A multidisciplinary individual can bridge between the two groups not because of a *greater* breadth of understanding, but because of a *different* perceptual framework that includes the dimensions in which the two groups produce work of similar features (see also Kuhn 1962, p. 74 and Gardner 1993 on new arrivals to the field). The work of a particular artist and a particular biologist can seem highly relevant to one another from this new point of view. Interdisciplinary work can itself sometimes develop into a new discipline as more individuals adopt similar views and form a new field.

10.3.5 Non-random sources of innovation

The behaviour of individuals within the creative system modelled appears unpredictable over time, but is not the result of a stochastic process. The outcome is completely deterministic given the initial conditions of domains and perceptual spaces. Instead, the apparent complexity appears to arise from two different processes. The first is the act of design as a process of selection from affordances in the world. The details of the work are input to the agent, rather than output, and a world constantly changing due to new building affords different possibilities at each given point in time (§10.3.2). Thus even an agent that maintains a constant state in terms of domain and perceptual space would produce different output at times t and t+1, and appear to be making unpredictable choices.

Even when the possibilities in the world are unrestricted however (§10.3.3), there is still apparently random behaviour. The other source appears to be the cycle of mapping between the three different spaces of differing dimensionality. This introduces into the overall system a complex non-linearity like that seen in Reynolds' (1987) flocks or Wolfram's (1994) class 3 and 4 cellular automata.

In none of these cases is there any conflict between conservative *utility* and radical *novelty*. Each agent is specifically making a choice that is absolutely conventional from its own point of view, so this is inherently useful, yet possibly perceived as an innovation by the rest of the group.

10.4 Summary: toward a creative system

Hypothesis B was invalidated by showing that a way of seeing an example (via a feature space Φ) could be communicated entirely by other examples. The above model above indicates that this is not only possible, but that it is necessary to communicate in this way, as the predefinition of a single way of seeing would make innovation impossible.

The model is minimal, and omits features often associated with a system of creative agents. The agents are not goal-motivated, novelty-seeking or of varied abilities. They have no intentionality other than the differentiation of perceptions required by neurons or machine classifiers. There is no objective measurement of utility to rate one example of creative output against the next. Also, there are no random processes to introduce innovation internal or external to the agents. These features may exist in real, human designers, but the results suggest that they are not strictly necessary to produce emergent behaviour resembling creativity. The practical implication of this is that even a simple agent can usefully participate in a larger creative process, and can therefore be of use within a creative system of real designers.

In the context of actual design, the question arises of whether a real designer could contribute, or conversely, whether an artificial agent could participate in the loop that occurs in an environment of real designers. As Arthur et al. (1997) suggest with respect to the simulation of a complex system, it is the interaction structure that is crucial, and the system of creative agents, artificial or human, is certainly a complex system. The model introduced illustrated how interaction can occur; this section summarises the nature of this communication and its relevance to useful designer-computer interaction.

10.4.1 Non-representational internal states

The question of what constitutes an agent's internal state is the most basic, as the varied states of agents were found to be valuable in driving innovation. Of greatest relevance is the fact that an internal state is not strictly an internal representation (Clark 1998, p. 168; §3.4.4) and does not necessarily equate to a symbolic representation (Beer 1995). In this case, at the level of the individual agent, no symbolic representation is used.

Instead, an agent's state corresponds to a particular way of seeing, as defined by the entirely private *perceptual space*. The space itself is primarily for taking in an experience of the world. Considered as an 'action oriented' (Clark 1998, p. 149; §3.2.2) representation of this world, it is strictly non-symbolic both in the way it is calculated (by neural-network) and in its output (a numeric, analogue map in which all perceptions are defined in relation to others). The relations between perceptions are real-valued and dynamic, rather than arbitrary mappings of tokens or memory slots to pre-defined semantic meaning.

10.4.2 Communication with a designer

In contrast to this view of a numerical, analogue agent state, Czikszentmihalyi describes communication with the domain as symbolic—the domain itself is considered to be a reservoir of symbolic knowledge. If language and other symbols are necessary for interaction with other agents in the field and for transfer to and from this domain, then communication may be via discrete signs, but how is this reconciled with the numerical perceptual space above?

Clark and Thornton (1997) suggest language and culture are ways of storing and communicating useful high level encodings for dealing with type-2 problems. One of the useful illustrations of the above model is that it shows how this can be done by virtue of the pairing of examples with evaluations. The latter convey cultural context and can be given symbolically, linguistically, as simple classifications, or as more subtle ratings of value. By virtue of what may be very basic messages communicated about examples (e.g. 'good' vs. 'bad'), the more complex, numerical states of perceptual space were seen to converge among communicating agents. Their building output also converges. Moreover, by communicating via the same simple messages, we should be able to influence any agent in exactly the same way.

The fact that our own internal state and faculties of perception differ greatly in terms of complexity from that of an artificial agent does not appear to matter. This is suggested at a lower level by the relative invariance of overall perceptions and outputs to vastly different kinds of network and learning algorithms in section 10.2.4. The key to this is that in addition to the simple, symbolic communication that occurs, the agents are each referring to examples embedded in a context of a shared world. While agents in swarm algorithms are normally identical, differently constructed agents with different ways of learning can also interact meaningfully together, even while each has a different perceptual space. The point of view, sensory experience and even the dimensionality of perception may be different for every agent, but they can still form a common social dynamic by taking advantage of a shared world in which they act by selection of affordances.

10.4.3 Labelling artefacts and context

The non-symbolic approach to design computation (§3.4.2), embodied, embedded approaches to artificial intelligence (Wheeler 2005) and those based on dynamic systems (Elman 1995; Port and Van Gelder 1995) argue for the importance of context in defining the meaning of any communication. Czikszentmihalyi’s description of the creative system stresses the communication of design artefacts, but it appears essential that these are always within a social or cultural context, and in the above simulation this was seen to affect their relevance. The communication of design intent can be considered to consist, therefore, of two parts: artefacts themselves, and some cultural context.

Context might also include the culturally defined activities in which one might engage a design, in which case very practical considerations—utility—determine the labels that structure the feature space. Novelty, by contrast, is a measurable quantity between artefacts in this space.

In considering communication between an agent similar to the above and a human designer, labels determined by the human are this cultural context. In the absence of the machine's ability to completely engage in our social and cultural world, these labels may always be given along with the artefact itself. If the agent is to be used as a design tool, there are two options for deriving these. In the case of ill-defined problems, the source of contextual labels may be based on a designer’s intuition about the nature of a given design—a way of communicating the subtleties of years of design experience to a machine. Alternatively, when the problem is clear, the context could be based on an objective measurement of the utility of a given design.

10.4.4 Innovation in context: the machine can be creative

The simulation demonstrates how the computer can *contribute* novelty, but this in itself is no guarantee of creativity. A straightforward means of producing novelty is a simple search for any solutions that have not yet been found, perhaps differing as much as possible, but with respect to the above discussion of context it is clear that most such solutions would be out of context, lack utility and be uncreative. The Wundt curve (Wundt 1874) assigns the greatest pleasure to moderate amounts of arousal, and versions of this (Berlyne 1971; Saunders and Gero 2001) have been applied to measures of novelty to explain an aversion to solutions that are too different. This may be largely a correct description of the novelty of creative ideas when measured in a single dimension, but the model demonstrated here illustrates at least the possibility of a multitude of dimensions, each constituting a valid point of view. It has demonstrated that it is possible to innovate without intentionally making something very different.

The previous chapters have shown that the computer can make something useful—appropriate to a given type or to set functional goals—by inductive learning. In showing that novelty is also generated *by the same process,* the above model indicates the machine can actually be creative. In agreement with other social models of creativity, both theoretical (Czikszentmihalyi 1988) and computational (Steels 2000), the simulation indicates that creativity at a global level does not always come from an intentional effort on the part of the individual, but that novelty is sometimes a product of the interaction between the different parts of a creative system. Thus it is an emergent phenomenon that can happen at any level, among groups, neurons, agents or us, and a computer with the appropriate interface can participate with designers in this process.

10.4.5 Importance of the interface

Communication between designers and such agents should be possible and quite natural, so long as it takes a familiar form. An agent’s internal state may be qualitatively different from a real designer’s mind, but by communicating by shared examples as above they may be brought into closer alignment for practical purposes. The ability to indicate a set of examples and evaluations of these is all that is required, and in the case of architectural design this mode of communication is via the interpretation of drawings.

The development of a design tool that engages our own creative process in this way should focus on methods by which the machine can interpret and create designs in the same form as those of the designer. The use of the archetype given in chapter 8 allows input to the computer in terms of design examples and context from a *domain*, and the generative and optimisation methods in chapters 8 and 9 provide output to be evaluated by a *field* of designers. An outline of how such a tool would operate is given in the following section.

10.5 Projected learning/design tools

A useful design tool based on the above premises may have one of two possible goals:

- to help make people themselves more intelligent or creative; this is possible if the tool allows a designer to see ill-defined patterns that would otherwise not have been obvious, and allows these to be communicated among others
- to suggest innovative designs, acting as an individual within a larger system

This tool can therefore be considered in two parts, one to analyse and facilitate communication between members of a creative group of designers, and a second to make design proposals.

10.5.1 Tool 1: feature space for analysis and communication

One tool is for learning, classification and clarification of relationships between plans. The feature spaces Φ from previous chapters 7–9 can numerically quantify relationships between plans even when these are only roughly suggested by provided labels, and provide a means for their communication. Existing plans and their contextual labels (either intuitive or empirically validated) were used to derive the space, and any new plan can be analysed by its precise location within it. By providing an interface to view and manipulate these spaces, this tool would allow the user to precisely define a context in which new designs can be evaluated, and to refine and communicate complex and ill-defined problems.

In using this tool, the designer plays the role of the field in arbitrating the examples that form the domain. The goal of learning is to configure the machine’s perceptual space to match that of the designer’s or design group’s. Implicit in the theory of Czikszentmihalyi and explicit in the implementations of Saunders and Gero (2001), Steels (2000) and this chapter is that this is how design groups work together—agents in the simulation do this among themselves, as do we.

If the labels that define the feature space represent only a vague intuition on the part of a single designer, the tool could be used to quantify this, check it and communicate it to others. In such circumstances it may be used to build a consensus among individual designers as was seen to occur in the cliques in section 10.3. If the labels are based on harder evidence, as was the case of the desk counts and group shapes used as variables in section 9.5, the tool may be used to estimate the degree to which new design proposals measure up to the desired goals.

10.5.2: Tool 2: Optimisation of new designs

Optimisation comes into play as a separate tool that uses this feature space to allow the machine not only to analyse but actually to contribute real designs to the process. The examples in chapter 9 illustrated the process using a GA. In this role the algorithm plays the role of another individual designer. As suggested in section 10.4 this has the same capacity for interface with the group as any other mind: a capacity to evaluate designs in the domain produced by others in the field and a capacity to produce new designs. The general intelligence is certainly not as great as a human designer but it can learn from our examples, and this limitation may be of use precisely because of the focus it provides.

In using this tool, the attributes of a collection of plans may be emulated in another set of suggestions. Constraints on the solution may be set however, for example in the case of an organisation moving to a new space. The learning tool may be used to derive features from a plan of a particular organisation in an existing building, then replicate those features in a new building with different overall plan shape and dimensions. The examples of changing scale in §7.5.5 were examples of this.

The interface is important for both tools. Based on the learning experiments already performed and observations made on the user interface developed in chapter 6, a proposal for each is described in the following sections.

10.5.3 User interface 1: Learning and feature space manipulation

Whether used for optimisation and design generation, or as a stand alone application for analysis only, most interaction with a working learning/design tool would be in manipulating the feature space Φ. The purpose of this interface is to enable the user to set the mapping to coincide with their own judgement of relationships between examples, in that perceived similarities and differences should coincide with the actual metric distance in the space. If the equivalent to the perceptual space of the machine exists for a human designer, it is far from the simple one and two dimensional representations shown here and in previous chapters. This does not appear to matter however, as the similarities between the output of different algorithms (sections 8.3.4 and 10.2.4) suggest that training on shared examples is of primary importance.

Once these examples are input, the user interface to manipulate the feature space must provide a means of manipulating their labels.

The output of feature vectors as labelled points in a one, two or three-dimensional space has been used repeatedly, and would form the graphical display of this part of the tool (figure 10.15). Clusters and classes are immediately visible, and the view is navigable using familiar CAD controls. In most cases, a small diagram of the plan itself aids legibility. In setting labels, pre-defined empirical measurements are easily accommodated via a numerical input, but if the tool is used to express a personal understanding of a particular designer's judgement the interface should allow for more intuitive means of interaction. By taking advantage of the clear visualisation of relationships, the feature space interface may also be used to drag points toward perceived visual targets, and so manipulating example points in the feature space is an intuitive way of labelling. This is essentially the communication of cultural context done between agents in section 10.3, and gives the user the opportunity to influence the domain.

In Chapter 9, binary classification was used to set out the learned function, but regression might be used in this case if the desired labels are real valued. The choice would be determined by the type of initial labels, which may be based on designer's intuition or more objective, quantitative evaluation as to the success of the plans.

All examples and initial labels would first be input as a set. These initial estimations may be based on a simple classification, or if even this is not appropriate, the principal components of the set may be used. Once displayed, an associated target would allow them to be moved by a user to any desired position in the space (figure 10.16). New examples may be added at this point, and targets may be disengaged for some examples while still viewing their vectors. Implemented online, the learning algorithm would attempt simultaneously to learn the function to accommodate these changes, the interaction taking the form of an ongoing dialogue rather than static data entry. Naturally, this update will generally cause the other points to shift in the new feature space—the degree and manner in which this occurs would give the user a sense of how each new classification is affecting the space. In the same manner as agents' spaces were seen to move closer together in section 10.3, the machines' feature space would gradually approximate that of the user's concept.

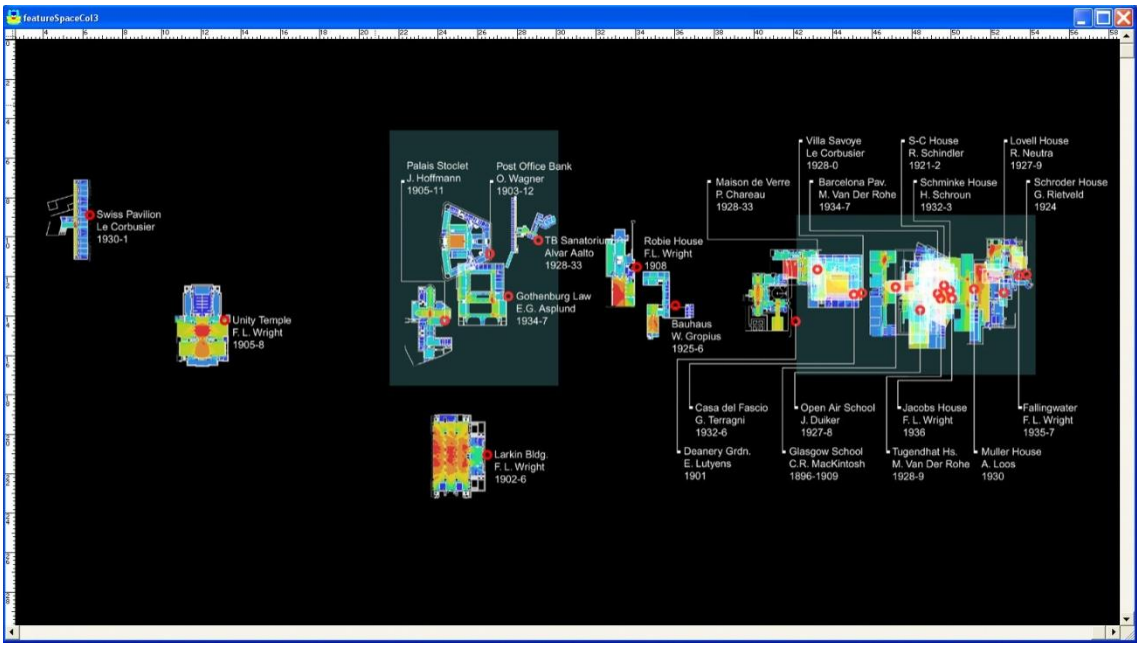


Figure 10.15. A graphical display of the feature space might place thumbnail images of each plan to make clusters or classes visible.

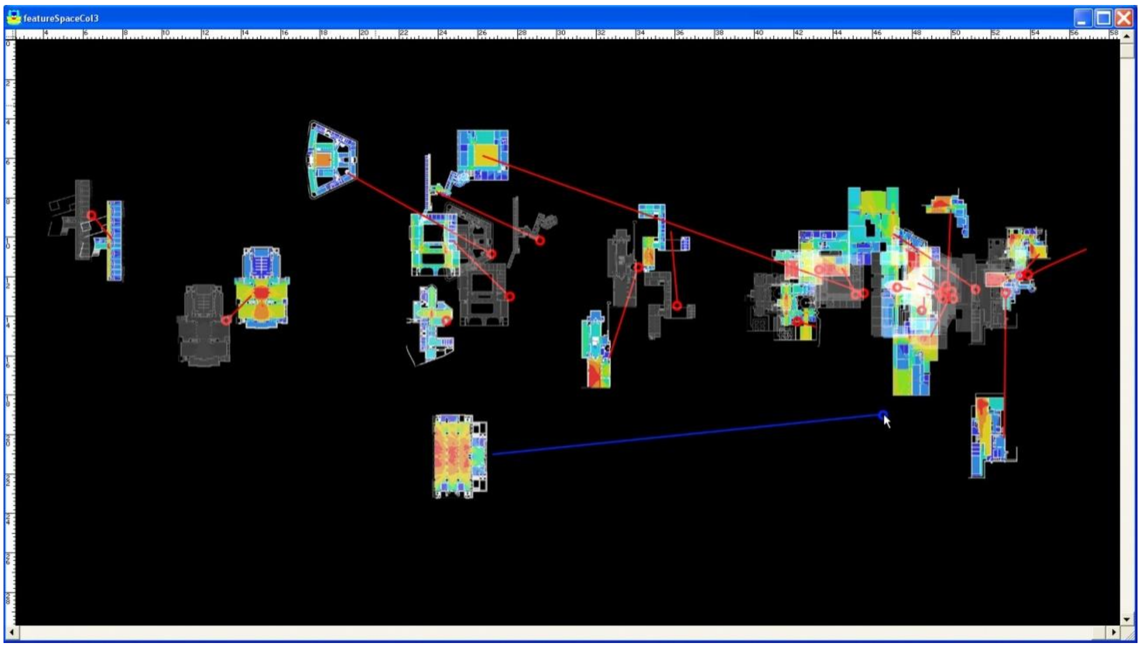


Figure 10.16. Labelling and input could be by means of intuitive 'dragging' of icons. Learning might reconfigure the space in real time so that the effects are immediately visible.

10.5.4 User interface 2: Optimisation

A full design tool would consist of a means to manipulate the feature space in which decisions are made, a means to evaluate new designs (both given by the feature space above) and an algorithm to generate new designs. This generative portion could take many forms, as long as it repeatedly evaluates afforded alternatives against the archetype set in the feature space (chapters 8 and 9). The aggregation model used in chapter 8 and by the agents above is simple, but the optimisation of desks plans by GA in chapter 9 functions under the same principles. This, or a similar optimisation algorithm, forms the third part of the proposed design tool. The three parts (feature space manipulation, evaluation and generation) follow the basic sequential layout as initially described in Chapter 1 (Figure 1.1). In terms of user interface only the first and last are relevant, as once the archetype is set, evaluation of designs is a straightforward distance measurement that requires no interface.

The bulk of the interface discussed in chapter 6 serves as a starting point for the optimisation component (figure 10.17). As a tool this was considered to be too inflexible, but only as a result of the rigid assumptions of the constraints and optimisation algorithm, not the interface. Optimisation would require the setting of overall parameters, possibly a selection of the algorithm itself, the ability to set graphic boundaries for the search. These would be implemented by a two part interface of numerical dialogue box and CAD style graphical interface. As discussed in section 9.5.2, a feature space archetype derived from plans of one overall shape and size is suitable for use on the optimisation of another. The graphic interface of this optimisation tool would be used to set constraints such as overall plan shape and size within which a GA or similar algorithm would function.

The crucial difference at this level would be the replacement of the simple optimisation used in chapter 6 with a more sophisticated algorithm. Due to the increased computation time required, the real-time update that was implemented initially—in which all desk positions could be immediately updated when a boundary was altered—would not be possible. Instead, a live graphic display of population improvements and fitness would allow the user to benefit from the ability to monitor, stop and possibly alter the progress, particularly for very large runs.

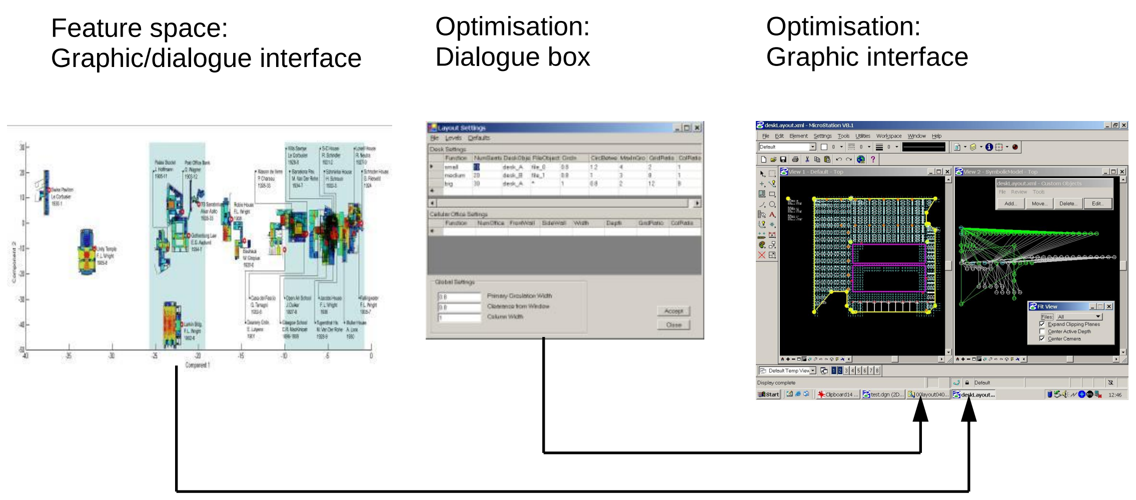


Figure 10.17. An optimisation toolkit would contain several components: the feature space interface as well as the CAD and dialogue based optimisation settings as in the initial tool (Chapter 6).

10.5.5 PRACTICAL CONSIDERATIONS FOR USABILITY

Several observations made in the course of the experiments point to issues that should be taken into account for the usability of such a tool. As discussed in chapter 6, flexibility and control on the part of the user are desired features in a design algorithm. In overall organisation, each part of the tool/interface as presented (initial input, feature space manipulation, optimisation) may stand alone and be replaced or modified as desired.

The labelling of examples may be by discrete binary sets or real-valued. As such, the learning process may correspond to a classification or regression problem. Particularly in the case of classification, the labels themselves may confuse more subtle distinctions within a class, by attempting to set each example to a simple value of –1 or +1. This would potentially eliminate much useful information in a tool for analysis, and produce a feature space that is difficult to search by an optimisation algorithm, as discussed in section 9.4.2. As before, the use of a different neural activation function for training and output may be used to retain these valuable distinctions. The ability to change functions, or even to parametrically adjust a sigmoid, should be given to the user. In the user interface itself, the ability to view the output space generated by a more linear activation while simultaneously setting input targets on a 'tighter' sigmoid would be appropriate.

If the feature space exceeds more than one dimension, the separate dimensions are better learned separately, then brought together in a single MLP (section 9.3). The tool should allow new dimensions to be added to the feature space by the user. This would also allow for more than three dimensions to be used, by displaying a maximum of three at a time. As always, the judgement of the user should be accommodated—if it feels independent intuitively, the user should be able to isolate these dimensions and display them in varied combinations. As discussed with respect to the interface, the actual labelling may be based on the designer's intuition or on a more objective, quantitative evaluation as to the success of the plans.

Chapter 11: Conclusions

Summary: The preceding chapters have demonstrated methods by which the computer is able to make approximations of the behaviour of complex systems in design situations, based only on the observation of precedents. The reflective method of design has been demonstrated as a framework for creativity, and it has been argued that an appropriately trained algorithm may be equipped to participate in this process. The manner in which the computer can take in and make sense of high level observations of the world such as structures and plans (Hypothesis A) and communicate naturally with human designers (Hypothesis B) offers both a potential explanation and aid to design of complexity.

The research presented over the previous chapters has addressed the problem of dealing with design tasks that are complex, perhaps too complex to define explicitly. Both the structural and spatial problems involve the design of a system with complex behaviour, that is, the interaction of the many individual units produces behaviour that cannot be predicted based on the parts taken separately. In the case of structural space frames these parts are individual struts or the small modular groups within unit cubes, and in the case of office planning or its extension to other spaces these parts are the people that use the space.

But coping with complexity is something that as architects and designers we do easily and frequently, and as humans we do largely unconsciously. The task of instructing a computer to do so has not been straightforward when we ourselves are unable to state the behaviour of a system or the desired goals explicitly. Rather than relying on a predetermined model, this work has proposed a methodology by which appropriate responses to a design task learned through induction based on prior examples of exemplary designs.

11.1 Review of basic terms

The understanding of several basic concepts central to the work was initially outlined in Chapters 1–3. Their use with respect to the research should now be clearer, and they can be reviewed in light of contributions made in the subsequent chapters.

11.1.1 INDUCTION

The learning that has been used has been described as induction, with the qualification that what is meant is not Popper's (1979) 'bucket theory' of mind as an empty receptacle for observations. The initial theory of a neural network, for example, consists of a set of random weights, but this can be seen to be progressively refined as new observations are encountered (particularly in §§5.3, 9.3). Experiments to tune initial conditions or the learning process (§9.4) were attempts to aid the algorithm in finding better theories faster.

Nevertheless, the learning is still clearly inductive, as what is learned can be seen to have been derived entirely from those samples themselves. Initial conditions, structure of the algorithm's implementation and even the choice of the algorithm itself—whether support vector machine, multilayer perceptron or self organising map—was seen to matter very little (§8.3.7). This was supported by the social implications shown in Chapter 10, in which nothing more than exposure to shared data was required for convergence of agent behaviour.

11.1.2 ISOMORPHISM AND HOMOMORPHISM

An isomorphism maps a one-to-one relationship between the states of two systems, a homomorphism maps one-to-many. The homomorphism can be at a different level of resolution. A homomorphism was most explicit in the approximation of individual structural units to regions of an optimally graded material (Chapter 4), but each function learned in subsequent chapters was a homomorphism or isomorphism to the particular system that was the subject of the design task.

As the difference between one-to-one and one-to-many is determined by the number of states, the distinction between the two types of map is really relevant only to symbolic and digital systems, and makes little difference to the approximately continuous systems that have been the focus of the research[*]. Where the distinction is much more relevant is in the case of communication, with language and labelling. The clustering required to make sense of feature spaces in Chapters 8–10 was a mapping of an essentially continuous space to discrete labels. The object of learning was to learn an approximation to this particular homomorphism.

[*] Approximately continuous because the desk plans are discrete, but the components of their spectra and relative measurements in Φ are so finely resolved, and the number of possible combinations so great, that they can be treated as continuous. Thus resampling of the spectra in Chapter 9 was possible.

11.1.3 Interface

The interface between elements in a system—whether structural units, an algorithm and the user or an agent and a design problem—has been treated as 'high bandwidth' (Clark 2008) and complex rather than simple and 'nearly decomposable' (Simon 1996). The modular units of the structural problem illustrate this explicitly, in that the connections between units are just as numerous as within, thus the boundaries could be redrawn elsewhere without change in complexity.

When the communication of archetypes as classes was performed (Chapters 8–10), the system had the capacity to take basic input of possible designs that was highly multidimensional, and only then mapped via the reduced feature space to a representation that could be tagged by labels resembling the symbols of classical AI. Communication with the user was also high bandwidth. The interface between a designer and the computer, by utilising real plans of desks as input and output, was thus similar in kind to the high bandwidth interface between human designers.

11.1.4 Engagement with the world: Representation and embodiment

The computer's engagement with prior design precedents and with the output it produces has not taken the classical approach, but an embodied one (§3.4.3) employing 'action oriented' representations (Clark 1998; also §3.4.4). What was learned was a means for acting on the design. In the case of the structural optimisation the function yielding optimal structures was learned directly (§5.3), and for desk arrangement a means for evaluating within another process was learned. The latter task was a matter of perception of the world in that it employed a selection of affordances, either possibilities for aggregation (§8.3) or options in an optimisation algorithm (§9.5). Evidence from working designers suggest that certain favoured rules are learned (Prats et al. 2009; and personal communication) and more expert designers do immediately perceive opportunities as a whole (Lawson 2006) in the way that Gibson (1979) suggests affordances are perceived.

Both design tasks were simple sub-problems compared to what working designers routinely face, but this is not the kind of simplification found in 'toy problems' of classical AI—they scale well, without running into the problems of data bloat, the frame problem, or the difficulty of ever expanding systems of symbols. By actually making real proposals in the world, embodied action is able to create something that is *more complex* than an internal representation can be. The machine need only have a low dimensional archetype (in some cases only a single dimension; §8.3) but can produce something of far higher dimensionality.

11.1.5 Design

Design combines elements of instructive (traditional, habitual; see e.g. Alexander 1964) and elective (e.g. critical, hypothetico-deductive; see Hillier and Leaman 1974) methods, but each of these explanations leaves out the essential point of creative insight. This is not completely inscrutable, as Kuhn (1963) would have it, and a reflective interpretation provides the necessary framework. Examples of this are found in the hermeneutical account of design (Snodgrass and Coyne 1997), the creation of phenomena in science (Hacking 1983) or Schön’s (1983) ‘reflection-in-action’ across professional disciplines. In deriving from existing design examples a means to make decisions about new ones, this research has explained this apparently inscrutable event of inspiration with a mechanism that can be implemented in a machine.

The reflective and the hypothetico-deductive methods differ somewhat with respect to their approach to truth, but this is reflected in fundamental differences in the domains of science and design. The root of these is that science is interested in pure knowledge (Lat. *scientia*), whereas design must act even if that knowledge is incomplete. Science thus has the luxury of clearly defining an object of study (and the tradition of reductionism) as opposed to design’s ill-defined, wicked problems. Science is also concerned with the universal whereas design is concerned only with a particular task, site, client, etc. Similarly, modern engineering, as applied to design, is considered universal, while the trial and error structural experiments of gothic cathedrals were particular instances of design.

This distinction highlights the one between design and research, where the latter is generally concerned with generalisable conclusions. Design as research is somewhat contentious as the contribution a particular design makes to knowledge in general cannot be stated clearly and explicitly—its contribution is only as a precedent. The clear changes measurable in learned functions (Chapter 5) or archetypes (Chapter 8) can make this somewhat clearer, in providing some understanding of how—and how much—a particular example contributes to knowledge.

11.1.6 Creativity

This was not initially introduced as a central topic of investigation, but the prospect of induction as a possible mechanism of the inspiration unexplained by Popper (1959) and Kuhn (1962), as well as the emergence of novelty in Chapter 10, do touch on important aspects of creativity.

Everyday creativity that should be commonplace in design has been explained as the seeing of something from different points of view (Schön 1963; Koestler 1964). The agent simulation in chapter 10 demonstrated this. Likewise, in the explanation of traits in creative genius it has been noted that great figures have experienced a crucial move away or change context (Gardner 1993). The same theories would seem to explain how the creative leap follows quite naturally by the juxtaposition of a new stimulus to an existing way of seeing—it is a more extreme version of the same phenomenon.

11.2 Conclusions on the initial hypotheses

The results of experiments in each of the two design domains indicate that the ability to make relevant design decisions can be learned by a machine, and in doing so the computer can give form to design problems even when these are complex. More specifically, these refute the two hypotheses presented in the introduction:

- Hypothesis A: To effectively predict or design for the behaviour of a complex system we must understand and simulate its underlying structure and causes of its behaviour.
- Hypothesis B: Communication with the machine that runs this approximation must be explicit, using a priori definitions and agreed upon conventions of meaning.

11.2.1 Hypothesis A: High level approximation of complex system behaviour

The first hypothesis was falsified by creating a higher-level approximation of a system's behaviour based only on observations, without knowing underlying causes, and showing this could perform similarly to the system or original simulation. Inductive learning was used in both design domains to allow the computer to take in and make sense of high level observations of the world such as structures and plans.

Models may approximate the systems in several different ways. It is generally acknowledged that if the system structure is known, one can simplify the parts (Simon 1996, Arthur et al

1997). Finite element meshes and VGA raster grids can be scaled to preserve their structure but change their resolution, resulting in a simple homomorphism.

The approximations shown here were of an entirely different structure from the system they modelled. They were necessarily related to the system, but only by the data itself. Model theory describes this relation as a (required) function or set of functions to map from the object language to the model language (Coyne 1999). In this work the learning algorithm has derived these functions, but they are highly complex, and come in a potentially infinite variety, depending on the algorithm, initial conditions, etc. This does not directly contradict the requirement of model theory, but it does contradict the general assumption that these functions are predetermined and relatively special. In linguistic terms, the processing rules may be different, but they produce the same structure rules.

Approximations were made from precedents in both structural and spatial domains. The results confirm the effectiveness of the approximations.

- *Space frame structural optimisation:* In this domain the precedents were structural units that had been previously optimised by a standard algorithm. With the use of the learned function in chapter 5, the approximation was structurally different, as no topology or material properties were explicit in the model. The fact that the output of the SVM approximation so closely resembled these original properties indicates that fine grained structure need not be known to create the model.
- *Desk pattern optimisation*: Precedents were sample plans of desk layouts, either as individual prototypes (Chapter 7) or as groups from which an archetype was derived (Chapters 8 and 9). In the final tests (Chapter 9) very different algorithms were used to generate the original plans (a parametric model) and the final plans (a GA with an intentionally incompatible genotype). An optimisation objective was created to correspond to arbitrary classes of patterns generated by a parametric algorithm, and these were then emulated successfully by a genetic algorithm.

The approximations were shown useful in filling in two types of knowledge that may be lacking in the design process. If a design objective is clear but something about behaviour due to system complexity is not, optimisation based on prior precedents was possible in both structural and spatial design problems:

- In chapter 5 a SVM was able to learn from a set of examples to produce a structure that rivalled, and in some cases exceeded the performance of the originals in terms of stiffness. Also there are advantages in terms of speed. The structural optimisation approximated by the learned function was nearly 50 000 times faster than by gradient descent on the FEM model.

- By labelling examples in chapter 8 and 9, the algorithm produced layouts resembling specific plans and then a set of plans.

Perhaps more relevant to the design of ill-defined problems, by learning a function based on prior examples the algorithm was able to generalise from specific examples, and then act upon this.

- The functions in chapter 5, particularly for model B, were able to outperform the individual optimisations in the combined structures of many units by removing discontinuities, effectively filtering out noise from many gradient based optimisations.
- Chapters 8 and 9 created an archetype of arbitrary features based on a selected set of precedents to be used to replicate plans.

Such approximations of behaviour offer several benefits. The first, speed, is obvious. An approximate model that runs faster but gives results that are accurate within a desired tolerance allows the design process to proceed more quickly and more iterations to be developed. The second is that designers often have knowledge and objectives that are purely at this higher level, and there is a natural barrier in understanding the behaviour of a system between two levels of detail. Because emergent behaviour makes understanding the details in a complex system based on global behaviour just as difficult as predicting the whole from its parts, it is often impossible to explicitly state desirable design moves for the system at the lower level. A desired material property may be more easily specified rather than the microstructure that causes it, or in the case of office interiors, a desired set of social relationships rather than the adjacencies and lines of sight that generate them. For this reason learning methods were investigated whereby design changes were specified implicitly by high level effects in precedents.

11.2.2 Hypothesis B: Communication with the machine

The second hypothesis was falsified by communicating the relevant design conditions to the machine by providing examples only, without a predetermined description or model of what the data means. In all cases, rather than predetermining a representation and using that to construct a model, the example precedents were used by the machine to construct its own ‘action oriented representation’ (Clark 1998; also §3.4.4), and this was used instead for the design task. This was tested by the machine’s ability to produce output satisfying the design requirement—its competence in the design task (§3.4.1). As optimisation goals were well defined for the initial structural optimisation, this hypothesis became more relevant in later chapters.

- *Space frame structural optimisation:* In the first design domain the objectives (stiffness, mass) are well known and easily provided in the context of standard optimisation. However, when the function is learned from examples the computer is not told explicitly about these, and the resulting function is the computer's own 'action oriented' representation. This is of limited use for the known variables, but the value here is that the function can also incorporate unknown factors such as those due to manufacturing processes (§5.1, Eq. 5.2). These were simulated as in §5.2.6.
- *Desk pattern layout:* Here the design objectives were not represented explicitly, but conveyed only by the sets of plans presented to the computer. The machine derived its own representation as an archetype (Chapter 8) from these, and used that to generate demonstrably similar plans. To ensure a valid competence evaluation of competence, set criteria were used to form the sample and test the results (Chapter 9). These were what were effectively communicated to, and represented by, the machine, but they were only done so implicitly via the examples themselves.

This alternative method of communication raises the possibility of a machine communicating more naturally with human designers. Chapter 10 suggested that the concepts implicit in example designs are transferred by designers this way in a social context. The classical, symbolic, GOFAI approach to representation and communication lacks a satisfying explanation of what looks like 'real' understanding. Tokens are essentially arbitrary and meaning is purely formal. For situated, embodied approaches, meaning is possible to the extent that each token has direct relationship to the world. It is this embodied relationship that occurs here. The computer is given the ability to 'make up its own mind' about examples, and thereby 'understand' them. The example set determines how the machine perceives the world by its feature space Φ (§8.2) or perceptual space (§10.2.3), and this in turn determines how it will act on the design. The representations formed by the machine do not have to be identical to be competent (§8.3.7), so effectiveness does not rely on any particular machine or algorithm.

As for Hypothesis A, the computer learns the behaviour of a complex system, only in this case it is a system of meaning and communication—a morphic language. For both design domains, the learned function incorporates a single designer's or a collective community's evaluation of precedents, each of which implies a number of (ill-defined) hidden variables. In the case of the structural problem, these were material and manufacturing variables. In the case of the class plan data set used for desk layouts, the system is the parametric algorithm that creates the data set, which ideally is isomorphic to our own human judgement of the output plans.

11.3 Practical considerations

The two design tasks were chosen because of their differences to represent a range of possible domains—structural versus spatial, known versus ill-defined goals. It is thus intended that the conclusions drawn apply more generally, but new problems will likely differ considerably. In generalising the methods to other design tasks, there are a number of limitations that may arise, and a number of steps that may be taken to mitigate or avoid these.

11.3.1 LIMITED SCOPE OF EACH PROBLEM

Inductive learning was effective in the limited domain of the problems shown, but this does not necessarily imply that the method is applicable to others—a variant of the problem of induction. Are there some functions within the scope of related design problems that may not be learned? In the general case this should depend on the amount of data available. The two structural models learned in §5.3 had quite different levels of accuracy depending on the number of examples on which they were trained, but both could be approximated, just as the varied arbitrary classes in §9.4 could be classified. It may also be dependent on whether the learning task is a simple type-1, or type-2 (Clark and Thornton 1997; also §3.2.6) problem that requires recoding. Effectiveness depends on there being a pattern in the examples that a designer can follow and that this should be discernable by machine, so the use of more data or more carefully structured learning (extending the experiments of §9.4) in the case of more complex problems may be necessary.

11.3.2 SUFFICIENT DIMENSIONALITY OF INPUT FEATURES

Induction should be dependent mainly on the data provided, so changes to the particular algorithm or the method for coding samples should make little difference. This was generally shown to be the case—by comparing different learning algorithms in §8.3.7 and by the use of topological input features (graph spectra) to produce geometrical features in plans in Chapters 7 and 9—except for one exception. In §7.5 it was seen that spectra taken from axial graphs alone were not enough to fully capture the arrangement of spaces. The general approach for overcoming this problem was to give as much initial information as possible, in two ways:

- Use a rich (i.e. high-dimensional) representation: graph spectra contain one eigenvalue for each node in the graph and so convey much more information from which a feature space may be extracted than a limited set of (predetermined) global scalar measures.

- Overlap different graph types (axial, boundary, desk) so that information from one fills in gaps in the other. This was found to be necessary to capture the full geometry of the plans being replicated.

The result was to provide as many possible dimensions from which the machine could extract the relevant features for the task at hand.

11.3.3 Input features and inherent patterns

Conversely, the underlying input features used as the raw data for induction may also contain strong patterns within them that compete with the function that needs to be learned. Using axial graphs as in §7.4, it would appear that the spatial and social factors they are normally used to quantify would be more readily evident than geometrical patterns. The former would be closer to type-1 and the latter closer to type-2 (Clark and Thornton 1997), although the fact that this underlying socio-spatial representation was used to produce geometrical patterns in §9.5 suggests that this distinction may not be absolute. There is a possible continuum between type-1 and 2, and the purpose of labelling samples was to move the problem along this continuum toward type-1 by changing the coding performed by the algorithm. These labels are a simplified version of a social process, filling in for our language and culture as social processes that recode our perception of the world (Clark and Thornton 1997). For some tasks this recoding may be more difficult, and learning may be better accomplished in stages (as Elman 1993) or as an extended dialogue between user and machine. The ability to view a changing feature space as §10.5.2 would aid this.

11.4 A different approach to design computation

At least as far as tools are concerned, there is an approach to design that relies on fixed concepts, ensuring they are clear and rational but resistant to change. The vast majority of computational tools, and all of CAD, shares this with the classical, symbolic branch of AI. Much of the reason may be historical, as these are descended from a common source—Sutherland’s (1963) ‘Sketchpad’ implemented the familiar screen based version of the engineering drawing and many of the underlying representations of the geometry that make it possible. The meaning—line, circle, etc.—is given first, and then determined in the abstract—as centre, plane and radius, for example—before being drawn to the screen. A reliance on clearly defined symbolic representations to create this geometry was necessitated by the limited computational power of the time, but was likely also influenced by the predominance of the classical AI approach at MIT in the 1960s. Chomsky (1957) used formal rules to explain language, Minsky and Papert (1969) appeared to discredit the connectionist alternative, and Stiny (1976) would then explicitly adapt the linguistic rules to design. This has not changed in the intervening decades. While descriptions of contemporary parametric software emphasises the ease with which the model can be changed, it operates on the symbolic approach. This is acknowledged explicitly in the 'Symbolic Model' of Bentley Generative Components.

What has been proposed here is a small part of the alternative view of the role of computation in design, in which real, changing data rather than fixed primitives are the basis for tools. It is supported by the success of similar changes in other fields like embodied robotics, by data intense methods in design such as those in Space Syntax, and by the general increase in computing power. This alternative approach is new, and so some speculation may be necessary, but the following seem to be justifiable. This conclusion extrapolates somewhat from the results of the work to propose several advantages of the approach.

11.4.1 Problems we don't understand

Most of this research has explored the possibility of coping with design problems we can't fully describe or even understand. The two sources of complexity explored yielded design situations in which either the full performance of the system being designed or the objectives that design was to meet could not be explicitly stated. The learned function was able to cope with these.

The improved performance due to the correction of noise in §5.3.2 suggests that such a learned model may also be more accurate for situations we believe we do understand. Small effects on a system that may otherwise be averaged out statistically may display patterns that are discernable in the data, when an appropriate learning technique is used. These may make small but significant improvements over traditional optimisation with an explicit simulation.

11.4.2 Robust and efficient design processes

Because much of the design activity involves more clearly defining the problem, wicked problems are solved by a process that has to cope with frequent and rapid changes. Where the technology for parametric models, optimisation and symbolic methods of CAD has been developed for the highly specified design and production lines of the automotive and aerospace industries, the alternative, data centred approach may suit the changing nature of ill-defined problems because, like the connectionist representation, it is more robust. For an *unexpected change*, a parametric model requires the creation of a new parametric schema (Barrios 2006), but small changes to a learned function can be accommodated easily because the function automatically changes gradually with each new piece of data.

What is learned in each case is a statistical approximation of the complex behaviour of a system, and with this it has been possible to reproduce the effect of that behaviour. In the case of the structural modules (Chapters 4 and 5) the redesign of part of an object would have effects on the distribution of forces, but the whole structure does not have to be optimised again. The function that has already been learned can be reapplied rapidly to vary each of the individual cells appropriately. This function itself is the most complex element of the design and this is robust to change.

At a higher level, it appears that it would be possible to derive a function indicating the patterns for how a localised change to the design would propagate throughout the object as a whole. It is likely that such a change would have an effect only on certain regions of the object, possibly decreasing with the distance from the point of change but possibly more complex than this. If this still higher level pattern is known, most of the structure can be retained while only modifying the necessary region. Here, most of the object is robust to change. This same principle could apply equally to larger scale problems such as truss roofs, or other design domains entirely. Research would be required as to the applicability to each problem, but the design strategy in general would be to use the learned functions to map out and retain generally robust aspects of the design, while allowing others to change.

11.4.3 Robust and sustainable design products

The search for such stable, subtle patterns could also yield final designs that are more robust to changes in their ultimate environment. Many runs of the structural optimisation (Chapter 5) or instances of floor plans (Chapter 9) were used here to yield a function to produce an optimal solution, but the same data could also be used to map out more resilient solutions. A complex simulation of computational fluid dynamics, for example, will yield a precise description of air flow under a particular set of parameters and boundary conditions, but if these later change (even slightly—the system is non-linear and complex) the result may differ catastrophically. Given data from many such simulations, the basins of attraction (stability) or zones of instability may be mapped, and these used to guide the design. Instead of looking for an optimal solution that may be sensitive to change, one can concentrate on the system's more stable attractors—a design placed here would be more robust to change after it is built and used. This would ultimately result in more sustainable designs.

11.4.4 Systems themselves might be creative

It is a grand claim to say that a machine might be creative in anything like the way a human designer is creative, but while this is already suggested by ‘human competitive’ design research, particularly in the evolutionary computation (Spector 2008), these methods lack something that would seem to be essential to creativity—the ability to ‘see-as’ (Schön 1963) or interpret existing designs. The possibility of this has been demonstrated (e.g. Chapter 10).

As humans and designers we also deal with complexities of our world by approximations of how it behaves. These can be anywhere in a continuum of representation from abstract models to embodied methods of coping, and are not always explicit. Tradition, rules of thumb and a craftsman’s ‘feel’ for a material have produced great structures for thousands of years before the development of modern engineering over the past few centuries and the recent analytical possibility of the finite element method. Similarly the collective work of many individuals has resulted in cities that function without central planning. These are certainly creative even without explicit problem statements of goals, and a similar kind of innovation appears to occur with simple, interacting agents (§10.3). Induction and more natural modes of communication (§11.2.2) have been demonstrated in a machine, and these may similarly allow a computer to engage in the collective process of interacting human designers.

The definition of creativity as ‘novelty + utility’ is quite widely accepted, only considered to be difficult to achieve due to the apparent conflict between the two components (§10.1.5). In the process of ‘seeing-as’ (Schön 1963), ‘displacing concepts’ (Koestler 1964) or changing ‘frames of reference’ (Akin and Akin 1996) with respect to an object in the world, this is not a paradox but a natural result. Inductive learning from design precedents would seem to allow the computer to do just that. To the extent that this is the case, it is conceivable that the computer can come up with its own concepts, innovative and useful, and propose them in design.

Appendix A: Genome representations used in office space optimisation

Three methods were devised to represent the layout of desks within a plan, varying in the amount of constraints used and the corresponding size of the search space. All take an underlying orthogonal grid as their structure, assuming a planning grid similar to those used in the planning tool in Chapter 6. As such, the methods only produce orthogonal arrangements as implemented, but could be used with plans that contain several non-orthogonal grids or in principal extended for use with hexagonal or other planning grids if required.

The three methods ultimately represent desk/chair units as a filled/void pair within one grid square. These may be in one of four orientations, as well as completely empty or completely filled.

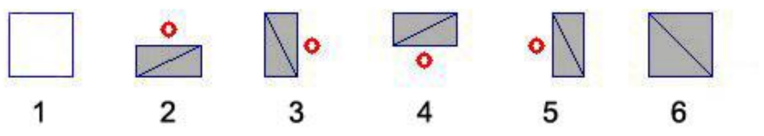


Figure A.1. Possible desk types and positions in the grid.

Representation A

The first representation is the simplest, least constrained and therefore has the largest associated search space. The base plan grid is simply represented as a matrix of the same dimensions as the grid with each entry encoding the position of a desk using one of the above values 1 to 6.

In the genetic algorithm, the matrix is reordered to a single vector and two point crossover is used on this linear arrangement. Mutation changes the value (and therefore the orientation of the desk) if a grid square with equal probability to any of the remaining five values. The search space is large: for an $n \times n$ plan grid the number of possible solutions is 6^{n^2} or roughly 10^{112} for even a small plan of n=12.

Representation B

The above representation can universally represent every arrangement of desks based on the plan grid, including a great deal of nonsensical ones. There is nothing to prevent large numbers of desks clumped together with no access or impassable snakes blocking individual units and in fact most random configurations result in such unusable spaces. A more efficient method takes the basic unit of space as a convex open area, and defines this space instead of the individual desks. To enable access to all seats, it is assumed that desks are placed around the perimeter of a convex space with chairs facing toward the centre. Convex spaces may be connected together by the absence of desks to allow passage from one to the next.

The use of a graph to define such an arrangement of spaces follows from Space Syntax (Hillier and Hanson 1984) and has been shown to be an efficient and universal method for encoding any orthogonal arrangement of rectangles (Flemming 1986). The use of such a graph also parallels the similar representation of space frames used in chapters 4 and 5. The representation of edges as a weighted adjacency matrix can be implemented in the same manner as in the genetic algorithm described earlier (chapter 4), and the position of the nodes is appropriate for the same kind of two stage search procedure.

Plan representation B defines the arrangement of convex spaces in plan as a graph in which each node represents the centre of a space and graph edges indicate a passage from one space to the next. Node points are given coordinates in the space of the plan and the graph is weighted both in its edges to indicate the relative width of clear passage and in its nodes to indicate the relative size of the convex space.

The process for converting the weighted graph to a plan involves the finding of points on a gradient defined by the graph in space. This gradient is continuous but can be approximated on a grid of double the resolution of the desk planning grid. For each node in the graph, the space of the plan is given a gradient in the form of a Gaussian with a mean $\boldsymbol{\mu}$ on the node point position and a variance σ^2 equal to its weight. This is normalised to have a maximum value of 1.0 to increase the effect of the higher weighted nodes: i.e.

$$\mathbf{g} = \exp(-(\mathbf{x} - \boldsymbol{\mu})^2 / 2\sigma^2). \tag{A.1}$$

Lines connecting the nodes along the graph edges are treated the same way, resulting in a continuous gradient that is greatest in the centre of convex spaces and along connecting passages and lowest at the boundaries between convex spaces. Desks are then placed at the perimeter of the resulting spaces by locating the minima of the gradient and where this is below a set threshold a desk is oriented with its seat in the direction of the increase in gradient.

Because of the essentially continuous nature of the initial graph representation and the fact that only the final desk placement is determined by the planning grid, this method is easily extensible to irregular grids. In fact the grid may be removed altogether, or given a weighting to allow dominant planning lines such as structural grids to be incorporated into the desk arrangement as a stronger influence on placement.

The size of the search space cannot be known exactly based on the number of inputs, as this may be affected by precision and in the process of discretisation of the weights to a regular grid many slightly different inputs may give identical solutions. It can be estimated based on the fact that certain desk positions will not occur adjacent to one another (e.g. front corner adjacent desks that therefore isolate passage from one seat to the next) and therefore the number of possibilities of orientation given a single prior adjacent desk drops from 6 to approximately 3.3. The search space may be estimated to be 3.3^{n^2} or roughly 10^{74} for a 12 × 12 plan.

Representation C

The most compact representation adds to the definition of convex spaces the assumption that there is a regularity to their arrangement based on the repetition of desk group configurations. A parametric model is used to determine the shape size and connectivity of spaces, but these can be repeated over any sized plan in a manner similar to the parametric layout of desk rows in chapter 6.

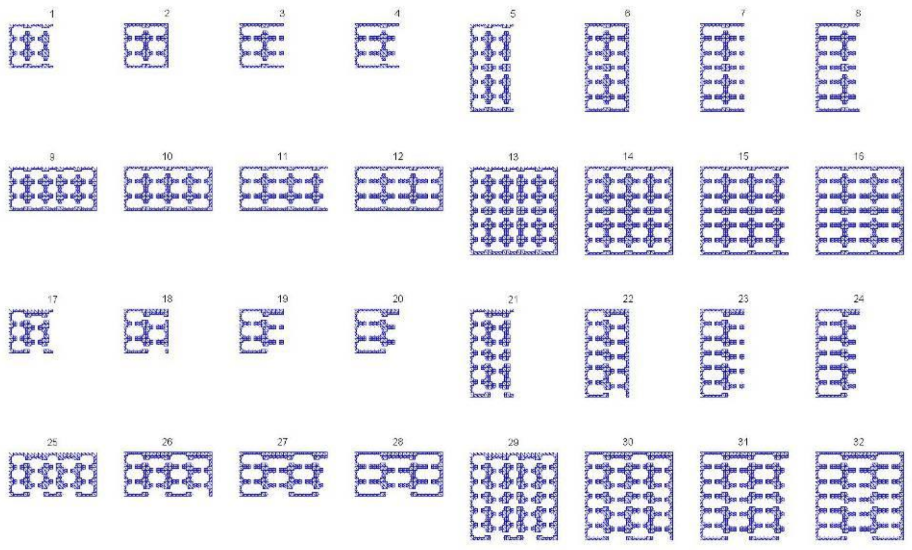
Convex spaces are laid out according to a larger grid, its size and the configuration of desks within spaces determined by five distinct parameters:

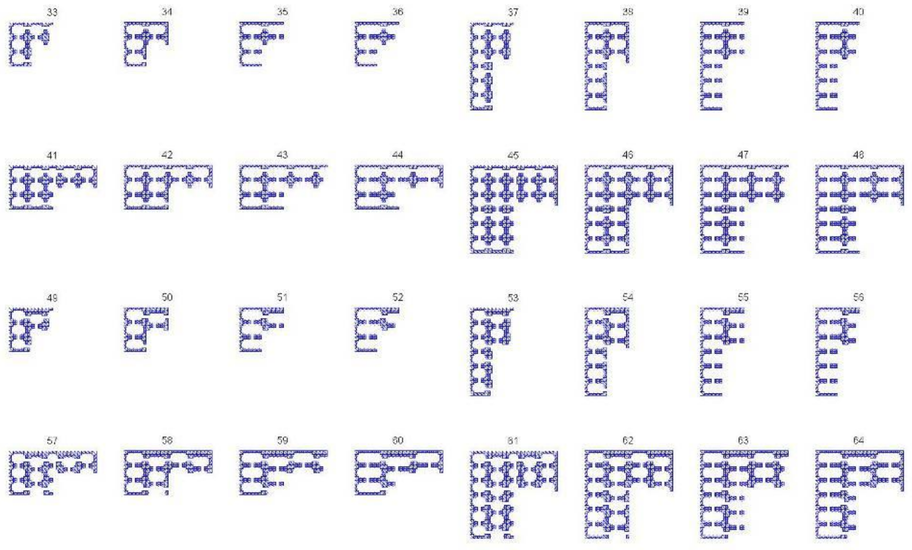
di	the i-axis size of a space unit (integer 2 to 9)
dj	the j-axis size of a space unit (integer 2 to 9)
off	optional offset of every second row (integer 0 to 7)
C	distance along the side of a space for a passage (4 × 4 array of reals –1 to 1)
G	indices indicating which spaces to use from C (4 × 1 array of integers 1 to 4)

The five parameters are converted to binary values and assembled to form a single gene string for use in the GA. The real valued distances of *C* are approximated to five bits to ensure a resolution higher than that of the maximum length of *di* or *dj*. All parameters taken together form a 97 bit string which is operated upon using standard binary mutation and two point crossover. There is some redundancy in the encoding of four desk configurations in *C* when some may not be used, and in the unused, negative portion of the range of *C*, but the total size of the search space is no greater than 2^{97} or roughly 10^{29}. This is far smaller than either representation A or B and is constant regardless of the plan size.

Appendix B: The *ClassPlan* data set

The *ClassPlan* data set was created manually from 128 permutations of five parameters, as described in §9.1.1. The full set of plans is illustrated below.





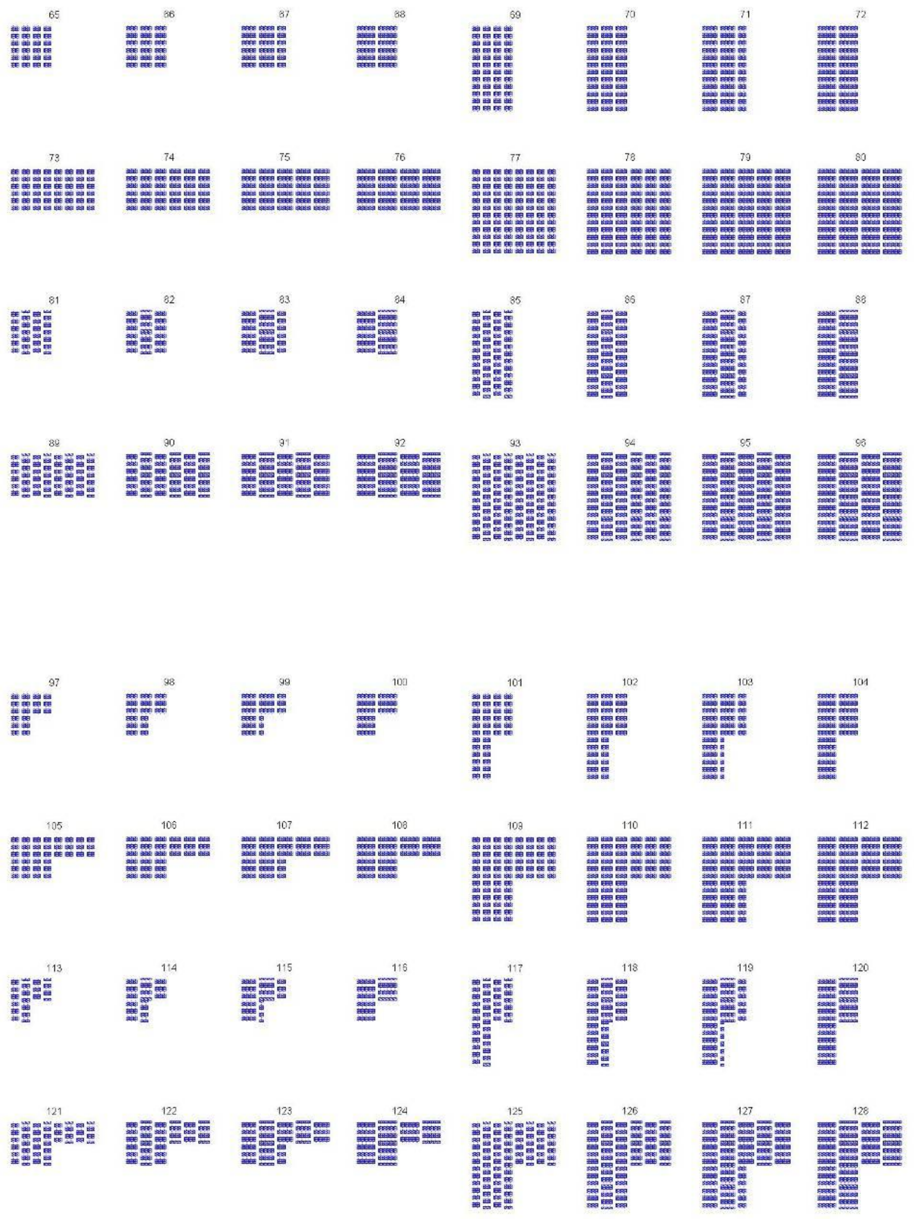


Figure B.1. The *ClassPlan* data set.

Appendix C: Graph spectral and Depthmap measures

Spectral analysis of a graph uses the eigenvalues and eigenvectors of its connectivity matrix, which appear to capture the kinds of graph structure relevant to spatial analyses. The calculation of the convergent solution of an agent simulation, for example, is given by the leading eigenvector of a graph's transition matrix (Turner 2007) *T*, where *T* is derived by normalising the columns of the graph's adjacency matrix *A*

$$T_{ij} = A_{ij} / \deg(j). \tag{C.1}$$

This eigenvector indicates the result at each node of the steady state of a random walk on any graph.

The adjacency matrix itself only directly quantifies local properties of a graph such as node degree, but spectral analysis reveals global properties of the whole graph. The corresponding eigenvector of *A* contains a more global measure of the graph comparable to integration. In the diagrams of three graphs in Figure C.1 the values of this maximum eigenvector are shown at left, compared with the level of integration indicated by 1/(mean depth) for each node. The bracketed figures show the rank of each node when sorted in ascending order.

1. In the first simple graph, the nodes have the same rank when sorted by eigenvector or by integration.
2. The second graph is made asymmetrical by adding links between the nodes on the right hand side. This causes the figures associated with these nodes to increase, but this increase on the more connected nodes is more pronounced in the eigenvector than in the mean depth measure. The two centre nodes are locally identical in that they have the same mean depth and degree, but connect to portions of the graph with very different levels of connectivity. This difference does not appear in the measure of integration but is evident in the eigenvector.
3. In the third graph, the asymmetrical sections are connected not by their central, but by more peripheral nodes. Again the eigenvector reveals the global discrepancy between the two halves at these nodes, whereas the measure of mean depth does not.

This does not suggest that the eigenvector has a higher correlation to the movement of agents or people on a graph, but it does capture some relevant qualities of global structure that integration does not.

The visibility graph of Frank Lloyd Wright’s Fallingwater was used to test the correlation between the spectral analysis and standard space syntax measures. Figure C.2 is a plot of the sum of the eigenvectors of a Depthmap visibility graph scaled by their eigenvalues:

$$\sum |\lambda \boldsymbol{\varphi}|. \tag{C.2}$$

This correlates highly, particularly with local measures such as isovist area and perimeter. Scaling the magnitudes of each node value by the magnitudes of the eigenvalues

$$\sum |\lambda||\boldsymbol{\varphi}|, \tag{C.3}$$

results in values that appear to correlate higher with global measures such as integration. There is no relationship of these to Depthmap agent analysis, as the correlation in both cases is negligible.

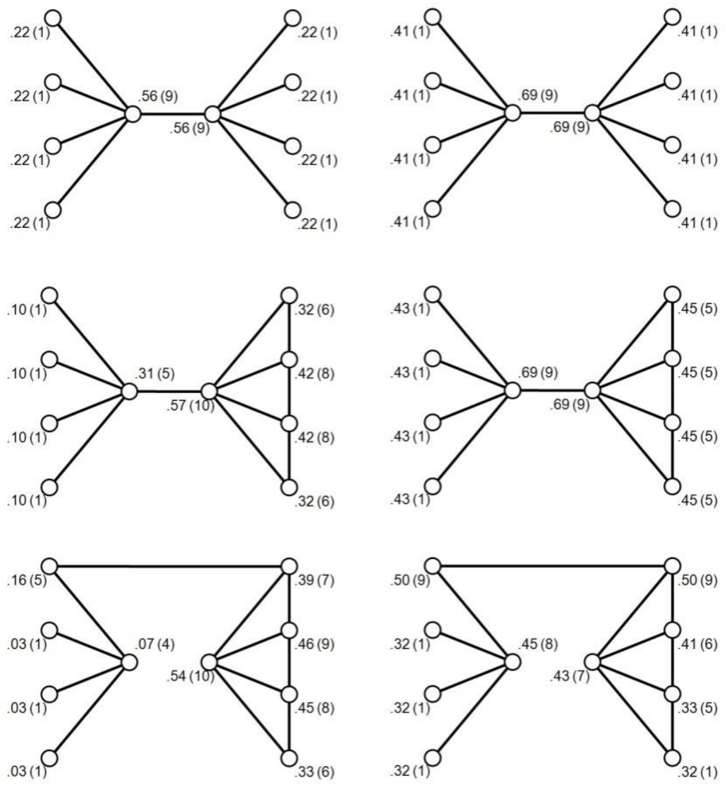
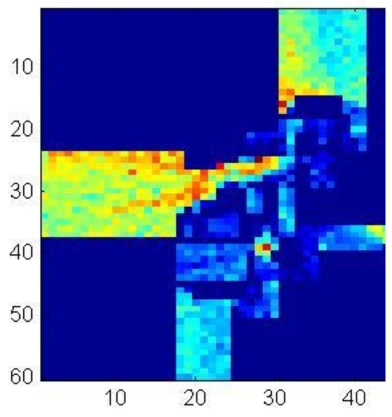
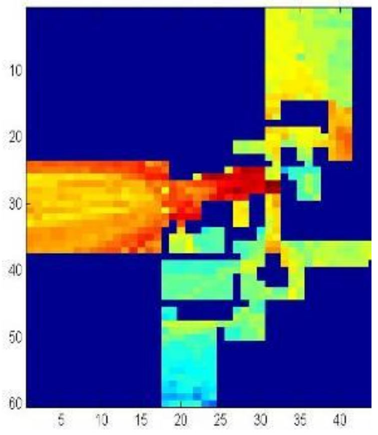


Figure C.1. graphs with nodes labelled by maximum eigenvector (left) and by level of integration expressed as 1/mean depth (right). Numbers in brackets indicate rank of each node when sorted in ascending order.



	Scaled eigenvectors
degree / isovist area	0.80
point first moment	0.73
isovist perimeter	0.72
visual integration [HH]	0.30
agents	−0.07



	Scaled eigenvector magnitudes
visual integration [HH]	0.70
visual integration [Tekl]	0.72
isovist occlusivity	0.75
isovist perimeter	0.77
agents	−0.04

Figure C.2. Scaled adjacency eigenvectors (top) and (absolute value) scaled eigenvector magnitudes (bottom).

The table indicates correlations between various functions of the eigenvalues/eigenvectors of the VGA graph and the Depthmap measures for Fallingwater. High correlations (≥0.7) are highlighted in red.

Table C.1. Table of correlations between graph spectral and Depthmap measures.

	Agent Counts	Connectivity (Degree) / Isovist Area	Isovist Compactness	Iso Drift Angle	Iso Drift Magnitude	Isovist Max Radial	Isovist Min Radial	Isovist Occlusivity	Isovist Perimeter	Point First Moment	Point Second Moment	Visual Entropy	Visual Integration [HH]	Visual Integration [P-value]	Visual Integration [Tekl]	Visual Mean Depth	Visual Relativised Entropy
adjacency (V*d)	-0.02	**0.71**	0.09	0.04	0.33	0.55	0.29	0.30	0.65	0.68	0.58	0.31	0.39	0.39	0.44	-0.50	-0.44
abs(V*d)	-0.07	**0.80**	0.31	-0.02	0.37	0.60	0.46	0.18	**0.72**	**0.73**	0.60	0.36	0.30	0.30	0.30	-0.29	-0.16
abs(V)*abs(d)	-0.04	0.67	-0.21	0.08	0.63	**0.71**	0.07	**0.75**	**0.77**	**0.76**	**0.76**	0.14	**0.70**	**0.70**	**0.72**	**-0.73**	-0.68
transition (Vt*dt)	-0.27	0.19	0.03	-0.10	0.12	0.15	0.02	0.12	0.13	0.23	0.24	-0.19	0.26	0.26	0.26	-0.23	-0.19
abs (Vt*dt)	0.24	-0.51	-0.43	0.12	-0.28	-0.38	-0.31	-0.02	-0.37	-0.49	-0.42	-0.26	-0.13	-0.13	-0.10	0.05	-0.03
abs(Vt)*abs(dt)	0.14	-0.53	**-0.71**	0.13	-0.07	-0.28	-0.52	0.36	-0.29	-0.39	-0.25	-0.68	0.20	0.20	0.21	-0.21	-0.28
Laplacian (Vl*dl)	-0.11	-0.01	0.03	-0.03	0.01	0.00	0.00	-0.03	-0.03	-0.01	-0.01	0.04	-0.06	-0.06	-0.05	0.04	0.06
abs (Vl*dl)	-0.01	0.65	0.13	-0.02	0.34	0.54	0.31	0.26	0.62	0.62	0.55	0.24	0.35	0.35	0.35	-0.34	-0.26
abs(Vl)*abs(dl)	-0.15	**0.92**	0.24	-0.08	0.51	**0.78**	0.42	0.34	**0.84**	**0.90**	**0.80**	0.34	0.47	0.47	0.47	-0.45	-0.33
unsigned L (Vu*du)	-0.05	0.07	-0.04	-0.06	0.06	0.11	0.06	0.09	0.08	0.09	0.09	-0.03	0.11	0.11	0.11	-0.10	-0.10
abs (VU*dU)	-0.01	0.62	0.13	-0.01	0.31	0.51	0.30	0.26	0.59	0.59	0.52	0.23	0.34	0.34	0.34	-0.32	-0.25
abs(VU)*abs(dU)	-0.12	**0.93**	0.20	-0.07	0.47	**0.78**	0.44	0.34	**0.85**	**0.90**	**0.78**	0.30	0.50	0.50	0.50	-0.48	-0.35
normalised L (Vn*dn)	0.02	0.00	0.00	0.00	0.06	-0.04	-0.03	0.00	-0.02	-0.01	-0.01	-0.02	-0.01	0.00	0.00	0.00	0.00
abs (Vl*dl)	0.00	0.02	0.02	0.03	-0.01	0.03	0.02	0.00	0.01	0.02	0.03	0.03	0.03	0.00	0.00	0.00	0.10
abs(Vl)*abs(dl)	0.03	-0.14	-0.14	-0.66	0.11	0.16	0.03	-0.41	0.59	0.05	0.01	0.11	-0.51	0.48	0.48	0.49	-0.52

References

Abu-Mostafa YS (1990) Learning from hints in neural networks. *Journal of Complexity*, 6(2), pp. 192-198.

Addis B (2007) *Building: 3000 Years of Design, Engineering and Construction*. Phaidon, London.

Adeli H and Cheng N (1993) Integrated Genetic Algorithm for Optimisation of Space Structures, *Journal of Aerospace Engineering*, 6(4).

Adrian E D (1928) *The Basis of Sensation*. Christophers, London.

Agre (1997) *Computation and Human Experience*. Cambridge University Press.

Akeike H (1973) Information Theory and an Extension of the Maximum Likelihood Principle, in Klotz and Johnson (1993) *Breakthroughs in Statistics: Foundations and Basic Theory*, Springer.

Akin Ö and Akin C (1996) Frames of reference in architectural design: analysing the hyperacclamation (A-h-a-!), *Design Studies* 17(4), pp. 341–361.

Aleinikov AG (1999) Humane Creativity, in Runco MA and Pritzer SR (eds), *Encyclopedia of Creativity*, Academic Press, San Diego, vol 1. pp. 837–844

Alexander C (1964) *Notes on the synthesis of form*. Harvard University Press.

Alexander C, Ishikawa S, Silverstein M, Jacobsen M, Fiksdahl-King I and Angel S (1977) *A Pattern Language*, Oxford University Press, New York.

Allen T (1977) *Managing the Flow of Technology*, MIT Press, Cambridge,

Arciszewski T and Ziarko W (1990) Inductive Learning in Civil Engineering: Rough Sets Approach, *Microcomputers in Civil Engineering* 5 pp. 19-28.

Argyris JH and Kelsey S (1960) *Energy Theorems and Structural Analysis*, Butterworths, London.

Arthur WB (1994) Inductive Reasoning and Bounded Rationality (The El Farol Problem), In *American Economic Review (Papers and Proceedings)*, **84**, 406.

Arthur WB, Durlauf S and Lane DA (1997) Introduction: Process and Emergence in the Economy, In Arthur WB, Durlauf S and Lane DA *The Economy as an Evolving Complex System II.* Addison-Wesley, Reading, Mass.

Ashby WR (1956) *An Introduction to Cybernetics*. Chapman & Hall Ltd. London.

——— (1962) Principles of the self-organizing system, in Von Foerster H and Zopf GW Jr. (eds.) *Principles of Self-Organization: Transactions of the University of Illinois Symposium*, Pergamon Press, London, pp. 255-278.

Astley RJ, Harrington JJ and Stol KA (1997) Mechanical modelling of wood microstructure, an engineering approach, IPENZ Transactions, 24(1), pp. 21-29.

Auyang SY (1998) *Foundations of Complex-System Theories: in Economics, Evolutionary Biology, and Statistical Physics*. Cambridge University Press, Cambridge.

Axelrod R (1997) The dissemination of culture: a model with local convergence and global polarization. *Journal of conflict resolution*, Vol. 41, No. 2. pp. 203-206.

Backhouse A and Drew P (1991) The design implications of social interaction in a workplace setting. In Environment and Planning B: Planning and Design, Vol 19, pp. 573-584.

Badea L, Ionescu IR and Wolf IS (2004) Domain decomposition method for dynamic faulting under slip-dependent friction. *Journal of Computational Physics*, Elsevier, vol. 201, pp. 487-510.

Ball P (1999) *The Self-Made Tapestry: Pattern formation in nature*. Oxford University Press.

——— (2004). *Critical Mass: How one thing leads to another*. Random House, London.

Barrios C (2006) Thinking parametric design: introducing parametric Gaudi. *Design Studies* **27** (3), May 2006, pp. 309-324.

Beer R (1995) Computational and Dynamical Languages for Autonomous Agents. In Port RF and Van Gelder T (Eds.) *Mind as Motion: Explorations in the Dynamics of Cognition*. MIT Press.

Bendsøe MP and Guedes JM (1994) Some computational aspects of using extremal material properties in the optimal design of shape, topology and material, *Control and Cybernetics* **23**(3) pp. 327–349.

Bendsøe MP and Kikuchi N (1988) Generating optimal topologies in structural design using a homogenization method, *Computer Methods in Applied Mechanics and Engineering*, Vol. 71, No. 2, pp. 197-224.

Bendsøe MP and Sigmund O (1999) Material interpolation schemes in topology optimisation, *Archives of Applied Mechanics*, Vol. 69, pp. 635-654.

——— (2003) *Topology Optimisation: Theory, Methods and Applications.* Springer.

Bentley P (1999) *Evolutionary Design by Computers. Morgan Kaufmann*, San Francisco.

Berlyne DE (1971) *Aesthetics & psychobiology*. Appleton-Century-Crofts, New York.

Bey J (1995) Tetrahedral grid refinement. *Computing*, vol. 55, pp. 355-378.

Bishop C (1995) *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford.

Boden MA (1990) *The creative mind: myths & mechanisms*. Weidenfeld & Nicolson,London.

Bodmer W and Cavalli-Sforza LL (1976) *Genetics, Evolution and Man*. W H Freeman, San Francisco.

Booker, P J (1963) *A history of engineering drawing*. Chatto & Windus, London.

Brockman JB and Director SW (1991) The Hercules Task Management System, *in Proceedings of the International Conference on Computer-Aided design.*

Brooks RA (1991) Intelligence without representation, *Artificial Intelligence* **47**, 139–159.

Burgess C (1998) From simple associations to the building blocks of language: Modelling meaning in memory with the HAL model. *Behavior Research Methods, Instruments, and Computers,* 30, 188-198.

Burgess C and Lund K (1997) Modelling parsing constraints with high-dimensional semantic space. Language and Cognitive Processes, 12, pp. 1-34.

Chan CS (1994) Operational definitions of style, *Environment and Planning B: Planning and Design*, vol. 21 pp. 223-246.

Chen YM (2002) *Nodal Based Evolutionary Structural Optimisation Methods*, PhD Thesis, University of Southampton.

Chomsky N (1957) *Syntactic Structures*. Mouton, The Hague.

——— (1965) *Aspects of the Theory of Syntax*. MIT Press.

Cilliers P (1998) *Complexity and Postmodernism : Understanding Complex Systems*, Routledge, London

Clark A (1990) Connectionism, Competence, and Explanation, *British Journal for the Philosophy of Science.* 41(2), pp. 195–222.

——— (1993) *Associative Engines: Connectionism, Concepts and Representational Change*, MIT Press.

——— (1998) *Being There: Putting Brain, Body and World Together Again*. MIT Press.

——— (2008) *Supersizing the mind*. Oxford University Press.

Clark A and Thornton C (1997) Trading spaces: Computation, representation and the limits of uninformed learning, *Behavioral and Brain Sciences* **20**, pp. 57–90.

Colquhoun A (1967) Typology and Design Method, *Arena*, 83, pp. 11–14. Reprinted in Colquhoun A (1985) *Essays in Architectural Criticism: Modern Architecture and Historical Change*. The MIT Press.

Conroy-Dalton R and Kirsan C (2008) Small graph matching and building genotypes. *Environment and Planning B: Planning and Design*, 35 (5), pp. 810–830.

Cope D (2001) *Virtual Music: Computer Synthesis of Musical Style*. MIT Press, Cambridge, MA.

Courant R (1943) Variational methods for the solution of problems of equilibrium and vibrations, *Bulletin of the American Mathematical Society*., 49 (1943), 1-23.

Cover TM and Tomas JA (1991) *Elements of Information Theory*. John Wiley & Sons.

Coyne R (1999) *Technoromanticism: digital narrative, holism, and the romance of the real*. MIT Press, Cambridge, MA.

——— (2004) Wicked problems revisited, *Design Studies* 26(1), Elsevier.

Cropley AJ (1999) Definitions of creativity, in MA Runco and SR Pritzer (eds.), *Encyclopedia of Creativity*, Academic Press, San Diego, pp. 511–524.

Czikszentmihalyi M (1988) Society, culture, and person: a systems view of creativity, *in* Sternberg, RJ (ed), *The nature of creativity: Contemporary psychological perspectives*, Cambridge University Press, Cambridge, pp. 325-339.

Dalton N, Peponis J and Conroy-Dalton R (2003) To Tame a TIGER one has to Know its Nature: Extending Weighted Angular Integration Analysis to the Description of GIS Road-centreline Data for Large Scale Urban Analysis. In *Hanson J (Ed.) Proceedings, 4th International Space Syntax Symposium*, University College London.

Dawkins R (1976) *The Selfish Gene*. Oxford University Press.

——— (2006) *The God Delusion.* Houghton Mifflin, New York

DeJong E (2000) *Autonomous Formation of Concepts and Communication*, Vrije Universiteit Brussel.

Del Tin L, Gaddi R, Gnudi A, Rudnyi EB, Greiner A, Korvink JG (2006) Efficient pre-stressed harmonic analysis of RF-microresonators by means of model order reduction**.** *EuroSimE 2006, Thermal, Mechanical and Multiphysics Simulation and Experiments in Micro-Electronics and Micro-Systems*, 24-26 April, Como (Milano), Italy,

Derand F (1643) *L'Architecture des Voutes*. Paris.

Derrida L (1976) *Of Grammatology*. Johns Hopkins University Press, Baltimore.

Dennett D (1991) Real Patterns, *Journal of Philosophy*, 88(1) pp. 27–51.

Desargues G (1639) *Brouillon-Project*. Paris.

Descartes, R. (1637) Discourse on the method. In *Philosophical Writings*, The Open University Press, Middlesex, 1954, pp. 3-57.

Desyllas J (2000) *The Relationship between Urban Street Configuration and Office Rent Patterns in Berlin*, PhD thesis, Bartlett School of Graduate Studies, UCL, London.

Desyllas J and Duxbury E (2001) Axial Maps and Visibility Graph Analysis, *Proceedings, 3rd International Space Syntax Symposium*, Georgia Institute of Technology Atlanta.

Devitt M (2006) *Ignorance of Language*. Oxford University Press.

Dorigo M and Gambardella L M (1997) Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1 (1), 53-66.

Dourish P (2001) Where the *Action Is - The Foundations of Embodied Interaction*, MIT Press, Cambridge, MA.

Dreyfus HL (1991) Being-in-the-World. MIT Press, Cambridge, MA.

——— (1992) *What Computers Still Can't Do: A Critique of Artificial Reason*. MIT Press. Cambridge, MA.

——— (2007) Why Heideggerian AI failed and how fixing it would require making it more Heideggerian, *Artificial Intelligence* 171, pp.1137–1160. Elsevier.

Dreyfus HL and Dreyfus SE (1988) Making a Mind Versus Modeling the Brain: Artificial Intelligence Back at a Branchpoint, *Artificial Intelligence* **117**(1), reprinted in Boden MA (ed.) *The Philosophy of Artificial Intelligence*, Oxford, pp. 309-333.

Duarte JP, Ducla-Soares G, Caldas JG and Rocha J (2006) An Urban Grammar for the Medina of Marrakech: Towards a Tool for Urban Design in Islamic Contexts. In Gero JS (ed.) *Design Computing and Cognition '06*, Springer, pp. 483-502.

Dubinski J, Humble R J, Loken C, Pen U-L, Martin PG (2003) Mckenzie: A Teraflops Linux Beowulf Cluster for Computational Astrophysics, in *Proceedings of the 17th Annual International Symposium on High Performance Computing Systems and Applications*, May 11-14, 2003 Sherbrooke, PQ

Duda, RO, Hart, PE and Stork DG (2001) *Pattern classification*. John Wiley & Sons, NY.

Duffy AHB (1997) The "What" and "How" of Learning in Design, *IEEE Expert: Intelligent Systems and Their Applications*, 12(3) pp. 71-76.

Durand JNL (1800) *Recueil et parallèle des édifices en tout genre*. L'École Polytechnique, Paris.

Durrant-Whyte H (2004) Autonomous navigation in unstructured environments, *Proceedings of the 8th International Conference on Control, Automation, Robotics and Vision*.

Elman JL (1990) Finding structure in time. *Cognitive Science*, **14**, pp. 179–211.

——— (1991) Distributed Representations, Simple Recurrent Networks, and Grammatical Structure, *Machine Learning*, **7**, 195-225. Kluwer.

——— (1993) Learning and development in neural networks: The importance of starting small, *Cognition*, **48**, 1993, pp. 71-99.

——— (1995) Language as a dynamical system. In Port, R. F., and van Gelder, T., eds., *Mind as Motion*.

Erdős P and Rényi A (1959) On random graphs. *Publications Mathematicae,* 6, pp. 290-297.

Feller W (1968) *An Introduction to Probability Theory and Its Applications*. Wiley, New York.

Figueiredo L and Amorim L (2005) Continuity Lines in the Axial System. A van Nes (Ed.) *Proceedings 5th International Space Syntax Symposium*, TU Delft, Faculty of Architecture, Delft, pp. 161-174.

Flemming U (1986) On the representation and generation of loosely-packed arrangements of rectangles, *Environment and Planning B. Planning and Design*, **13**, pp. 189–205

Florida R (2002) *The Rise of the Creative Class*. Basic Books, New York.

Fodor J (1975) *The Language of Thought*, Harvard University Press, Cambridge, Ma.

——— (1983) *The modularity of mind*. The MIT Press, Cambridge, Ma.

Freeman WJ (1991) The physiology of perception, *Scientific American*, 242, Feb. 1991, 78.

Fuller RB (1975) *Synergetics: explorations in the geometry of thinking*. Macmillan, New York.

Gadamer H-G (1975) *Truth and Method,* Sheed and Ward, London.

Gamkrelidze TV and Ivanov VV (1990) The Early History of Indo-European Languages. *Scientific American,* March 1990, P.110.

Gardner, H (1993) *Creating Minds: an Anatomy of Creativity seen through the Lives of Freud, Einstein, Picasso, Stravinsky, Eliot, Graham, and Gandhi,* BasicBooks, New York.

Gentner D (1983) Structure-mapping: A theoretical framework for analogy. *Cognitive Science,* **7**, pp. 155–170.

Gero JS (1994) Towards a model of exploration in computer-aided design, *in* J. S. Gero and E. Tyugu (eds) *Formal Design Methods for CAD,* North-Holland, Amsterdam, pp. 315-336.

Gero JS and Kazakov V (2001) Entropic-based Similarity and Complexity Measures of 2D Architectural Drawings, in Gero, JS, Tversky, B and Purcell, T (eds), *Visual and Spatial Reasoning in Design II,* Key Centre of Design Computing and Cognition, Sydney.

——— (2002) Complexity Measures of Design Drawings and their Applications. *The Ninth International Conference on Computing in Civil and Building Engineering,* Taipei, Taiwan.

Gibbs RW (1999) Metaphors, Runco MA and Pritzker SR (eds.), *Encyclopedia of Creativity*, Academic Press, San Diego, vol 2. pp. 209–220.

Gibson JJ (1979) *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin.

Gladwell M (2000) *The Tipping Point: How little things can make a big difference*. Abacus, London.

——— (2000a) Designs for Working, *The New Yorker,* December 11, 2000, p. 60

Gombrich EH (1960) *Art and Illusion*. Phaidon, London.

Goodman N (1955) *Fact, Fiction and Forecast*, 2nd edition (1965), Bobbs-Merrill, Indianapolis.

——— (1975) The Status of Style. *Critical Inquiry*, vol. 1. Reprinted in Goodman N: 1978, *Ways of Worldmaking*. Hackett Publishing Company Inc. Indianapolis.

——— (1976) *Languages of Art*. Second ed. Hackett Publishing Company, Indianapolis.

Granovetter MS (1973) The strength of weak ties. *American Journal of Sociology*, vol. 78, pp. 1360-1380.

Grzeszczuk R, Terzpoulos D and Hinton G (1998) NeuroAnimator: fast neural network emulation and control of physics-based models, *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 9-20.

Habermas J (1974) *Theory and Practice,* trans. Viertel J, Heinemann, London.

Hacking I (1983) *Representing and Intervening: introductory topics in the philosophy of natural science*. Cambridge.

Haugeland J (1978) The Nature and Plausibility of Cognitivism, *Behavioral and Brain Sciences*, 2 pp. 215–60.

——— (1981) Analog and Analog. *Phiosophical Topics* 12, pp. 213–225.

——— (1985) *Artificial intelligence: The very idea.* MIT Press. Cambridge, MA.

——— (1995) Mind Embodied and Embedded, reprinted in Haugeland J (1998) *Having Thought: Essays in the Metaphysics of Mind.* Harvard University Press, Cambridge, MA.

——— (1998) *Having Thought: Essays in the Metaphysics of Mind.* Harvard University Press, Cambridge, MA.

Haken H (2006) Recognition of Natural and Artificial Environments by Computers: Commonalities and Differences. Presented at *Design Computing Cognition '06, Workshop on Models of Machine Learning in Creative Design*, Eindhoven.

Hanna S (2007) Representation and Generation of Plans Using Graph Spectra, in Kubat A,Ertekin O,Guney Y,Eyuboglu E (eds.) *Proceedings, 6th International Space Syntax Symposium,* Istanbul.

——— (2007) Automated Representation of Style by Feature Space Archetypes: Distinguishing Spatial Styles from Generative Rules, *International Journal of Architectural Computing.* 5(1), pp. 2–23.

Hanna S and Haroun Mahdavi S (2006) Inductive machine learning of microstructures: Estimating a finite element optimisation using support vector machines. In Gero, JS (ed) *Design Computing and Cognition '06*. Springer. pp. 563-582.

——— (2004) Modularity and Flexibility at the Small Scale: Evolving Continuous Material Variation with Stereolithography. In Beesley P. Cheng W. and Williamson R Eds.

Fabrication: examining the digital practice of architecture. University of Waterloo School of Architecture Press, Toronto.

Harman G (1992) Induction: enumerative and hypothetical. In Dancy J and Sosa E (eds.) *Blackwell Companions to Philosophy: A Companion to Epistomology*. Blackwell Publishers Ltd. Oxford.

Haroun Mahdavi S and Hanna S (2003) An Evolutionary approach to microstructure optimisation of stereolithographic models, *Proceedings of CEC2003, The Congress on Evolutionary Computation,* Canberra, Australia.

——— (2004) Optimising Continuous Microstructures: A Comparison of Gradient-Based and Stochastic Methods, *Proceedings of SCIS & ISIS 2004, The Joint 2nd International Conference on Soft Computing and Intelligent Systems and 5th International Symposium on Advanced Intelligent Systems*, Yokohama, Japan.

——— (2004a) Blurring the Boundaries between Actuator and Structure: Investigating the use of Stereolithography to build Adaptive Robots, *Proceedings of ICARCV 2004: 8th International Conference on Control, Automation, Robotics and Vision.*

Hayes PJ (1979) The Naïve Physics Manifesto, in Mitchie D (ed.) *Expert Systems in the Micro-Electronic Age*, Edinburgh University Press, pp. 242–70.

Heidegger M (1962) *Being and time*, trans. Macquarrie J and Robinson E, Blackwell, Oxford.

Hersey GL and Freedman R (1992) Possible Palladian villas: (plus a few instructively impossible ones). The MIT Press, Cambridge MA.

Hillier B (1996) *Space is the Machine: A Configurational Theory of Architecture*. Cambridge University Press.

Hillier B and Hanson J (1984) *The Social Logic of Space*. Cambridge University Press.

Hillier B, Hanson J, Peponis J, Hudson J and Burdett R (1983) Space Syntax, *Architects Journal*, 178(48), pp 67-75.

Hillier B and Leaman A (1974) How is design possible? *Journal of Architectural and Planning Research*, **3**(1). pp. 4–11.

Hillier B, Leaman A, Stansall P, and Bedford M (1976) Space syntax, *Environment and Planning B* 3(2), pp. 147–85

Hillier B, Musgrove J, and O'Sullivan P (1972) Knowledge and Design, in J W Mitchell (ed.) *Environmental Design Research and Practice*, University of California Press, Los Angeles.

Hillier B, Penn A, Hanson J, Grajewski T and Xu J (1993) Natural movement: or, configuration and attraction in urban pedestrian movement, *Environment and Planning B: Planning and Design*, vol. 20 pp. 29-66.

Hillier B and Shu S (2001) Crime and urban layout: The need for evidence, in Ballintyne S, Pease, K and McLaren V: *Secure Foundations: Key Issues in Crime Prevention and Community Safety*. IPPR, London.

Hofstadter, D (1979) *Gödel, Escher, Bach: an Eternal Golden Braid.* Basic Books, New York.

——— (1993) How could a copycat ever be creative? *Papers from the 1993 Spring Symposium Technical Report SS-93-01* AAAI Press

——— (2001) Epologue, to Gentner D, Holyoak KJ and Kokinov BN, *The Analogical Mind: Perspectives from Cognitive Science*. MIT Press.

Holland JH (1975) *Adaptation in Natural and Artificial Systems An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press, Ann Arbor.

——— (1998) *Emergence : from chaos to order*. Oxford University Press, Oxford.

Holland JH, Holyoak KJ, Nisbett R and Thagard PR (1986) *Induction*. MIT Press, Cambridge, Mass.

Holyoak KJ and Thagard P (1997) The analogical mind. *American Psychologist*, 52: 35–44.

Hrennikoff A (1941) Solution of problems of elasticity by the frame-work method, *ASME Journal of Applied Mechanics* 8 (1941), A619–715.

Hume D (1740) *A Treatise of Human Nature*. New ed. (2001), Oxford University Press.

Hutchins E (1995) Cognition in the Wild. MIT Press. Cambridge, MA.

Jacobs J (1961) *The Death and Life of Great American Cities*. Random House, New York.

Jeffreys H (1957) *Scientific inference*, Cambridge University Press.

Johnson J (2005) Complexity science in collaborative design, *CoDesign*, 1(4), pp. 223–242.

——— (2006) Can complexity help us better understand risk?, *Risk Management* (2006) 8, pp. 227–267.

Jupp J and Gero JS (2003) Towards computational analysis of style in architectural design. in S Argamon (ed), *IJCAI03 Workshop on Computational Approaches to Style Analysis and Synthesis*, IJCAI, Acapulco, pp 1–10

——— (2004) A Characterisation of 2D Architectural Style, Journal of the American Society of Information Science.

Kant I (1787) *Critique of pure reason*, trans. (1998) Guyer P and Wood AW, Cambridge University Press.

Kauffman S (1995) *At Home in the Universe*, Oxford Univ. Press, New York.

Kemeny JG (1953) The Use of Simplicity in Induction, *The Philosophial Review* 62(3) pp.391–408.

Kennedy J and Eberhart RC (2001) *Swarm Intelligence*. Morgan Kaufmann, San Francisco.

Kicinger R, Arciszewski T and De Jong K (2005) Parameterized versus Generative Representations in Structural Design: An Emperical Comparison, *Proceedings of GECCO ’05*, pp. 2007–14.

Knight TW (1998) Shape Grammars, *Environment and Planning B: Planning and Design,* Anniversary Issue (1998), pp. 86-91.

Knoppers R and Hague R (2005) CAD for Rapid Manufacturing, in Hopkinson N, Hague R and Dickens P (Eds.) *Rapid Manufacturing: An Industrial Revolution for the Digital Age*. Wiley.

Koestler A (1964) *The act of creation*. Hutchinson.

Kohonen T (1982) Self-organized formation of topologically correct feature maps, *Biological Cybernetics*, 43(1) pp. 59-69.

Koile K (1997) Design conversations with your computer: evaluating experiential qualities of physical form, *CAAD futures 1997*, pp. 203-218.

——— (2004) An intelligent assistant for conceptual design, in Gero, JS (ed*), Design Computing and Cognition ’04*. Kluwer, Dordrecht. pp. 3-22.

Koning H and Eizenberg J (1981) The language of the prairie: Frank Lloyd Wright’s prairie houses, *Environment and Planning B: Planning and Design,* vol. 8 pp. 295-323

Krier L (1988) God Save The Prince! *Modern Painters*, No. 2, pp. 23–25. Reprinted in: *Archives d'Architecture Moderne*, vol. 38, pp. 12–21. Reprinted in: Jencks C (1988), *The Prince, the Architects, and New Wave Monarchy*, Rizzoli, New York, pp. 50–51.

Kripke SA (1982) *Wittgenstein on rules and private language: an elementary exposition*, Blackwell, Oxford.

Kuhn, Thomas S (1962) *The Structure of Scientific Revolutions*, 3rd edition, University of Chicago Press.

Kumar S, Ramos F, Douillard B, Ridley M and Durrant-Whyte HF (2006) A Novel Visual Perception Framework. In *ICARCV '06. 9th International Conference on Control, Automation, Robotics and Vision, 2006*.

Kvan T and Gao S (2006) A comparative study of problem framing in multiple settings. In Gero, JS (ed) *Design Computing and Cognition '06*. Springer. pp. 563-582.

Lakes RS (1987) Foam structures with a negative Poisson's ratio. *Science*, **235**, pp.1038-1040.

Lakoff G and Johnson M (1999) *Philosophy in the flesh: the embodied mind and its challenge to Western thought*. Basic Books, New York.

Lakoff G and Núñez RE (2000) *Where mathematics comes from: how the embodied mind brings mathematics into being,* Basic Books

Langendoen D (1998) Linguistic Theory. in *Blackwell Companions to Philosophy: A Companion to Cognitive Science*. Blackwell Publishers Ltd. Oxford. pp.235-244

Lawson B (2006) *How Designers Think: The design process demystified, Fourth Edition*. Elsevier, Oxford.

Lederberg J (1958) Genetic Approaches to Somatic Cell Variation: Summary Comment, *Journal of Cellular and Comparative Physiology,* supplement 1(52), pp. 383–401.

Lenat DB (1996) From *2001* to 2001: Common Sense and the Mind of HAL, in Stork DG (ed.) *HAL's Legacy: 2001's Computer as Dream and Reality,* MIT Press, pp. 193–210.

Lenat DB and Guha RV (1990) *Building Large Knowledge-Based Systems.* Addison-Wesley, Reading, Mass.

Lewes GH (1891) Problems of Life and Mind. J.R. Osgood, London.

Li P, Burgess C and Lund K (2000). The acquisition of word meaning through global lexical co-occurrences. In *Proc. of the 30th Child Language Research Forum*, pp. 167-178. Stanford, CA: CSLI.

Liu YT (2000) Creativity or novelty? *Design Studies* **21**(3) 261–276.

Lorenz EN (1963) Deterministic Nonperiodic Flow, *Journal of the Atmospheric Sciences* **20**(2) pp. 130–141

L'Orme P de (1567) *Le Premier tome de l'architecture de Philibert de l'Orme*. Paris.

Lund K and Burgess C (1996) Producing high dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, and Computers* **28**:203–8.

Luo B, Wilson RC and Hancock ER (2003) Spectral embedding of graphs, *Pattern Recognition*, vol. 36, pp. 2213-2233

Machotka P (1980) Daniel Berlyne's Contributions Empirical Aesthetics, *Motivation and Emotion*, 4(2) pp.113–121.

Maher ML and Poon J (1996) Modelling design exploration as co-evolution, *Microcomputers in Civil Engineering (Special Issues on Evolutionary Systems in Design)*

Marr D (1977) Artificial Intelligence: A Personal View. Reprinted in Boden MA (ed.) *The Philosophy of Artificial Intelligence,* Oxford, pp.133–146.

——— (1982) Vision. *A computational investigation into the human representation and processing of visual information.* W.H. Freeman, New York.

Martin LJ (1906) An experimental study of Fechner's principles of aesthetics. *Psychological Review,* 13, 142–219.

Maturana HR and Varela FJ (1987) *The Tree of Knowledge: the biological roots of human understanding*. Shambhala Publications Inc. Boston.

McClelland DC, Atkinson JW, Clark RA and Lowell EL (1953) *The achievement motive.* Appleton-Century-Crofts, New York.

McGuire W, Gallagher RH and Ziemian RD (2000) *Matrix Structural Analysis, 2nd edition.*

Medawar PB (1960) *The future of man: the B.B.C. lectures 1959,* Methuen.

Mitchell M (1996) *An Introduction to Genetic Algorithms*, MIT Press, Cambridge MA.

Meiler J and Baker D (2003) Coupled prediction of protein secondary and tertiary structure, Proceedings of the National Academy of Sciences of the United States of America, 100(21), pp. 12105-12110.

Minsky M and Papert S (1969) *Perceptrons: an introduction to computational geometry*. MIT Press. Cambridge MA.

Molecular Geodesics Inc. (1999) Rapid prototyping helps duplicate the structure of life, *April 99 Rapid Prototyping Report*.

Monge G (1795) *Géométrie descriptive*. Nouvelle éd. (1811) Klosterman, Paris.

Murawski K, Arciszewski T and De Jong K (2000) Evolutionary Computation in Structural Design, *Engineering with Computers* (2000) 16 pp. 275-286.

Murdoch T and Ball N (1996) Machine learning in configuration design, *AI EDAM* (1996) 10 pp 101-113.

Nash J F (1950) *Non-cooperative Games*. PhD dissertation, Princeton University.

Nehaniv CL and Dautenhahn K (1999) Of hummingbirds and helicopters: An algebraic framework for interdisciplinary studies of imitation and its applications. *In Learning Robots: An Interdisciplinary Approach*, J. Demiris and A. Birk, eds., World Scientific Press, in press.

Neocleous CC and Schizas CN (1995) Artificial neural networks in marine propeller design. *Proceedings of ICNN'95-International Conference on Neural Networks*, New York. IEEE Computer Society Press, Vol. 2 pp 1098-1102.

Newell A and Simon H (1976) Computer Science as Empirical Enquiry: Symbols and Search. *Communications of the Association for Computing Machinery* 19, pp. 105- 32.

Nicolis G and Prigogine I (1989) *Exploring Complexity*. New York: Freeman and Co.

Norman DA (1988) The *psychology of everyday things*. New York: Basic Books.

O'Reilly RC and Munakata Y (2000) *Computational explorations in cognitive neuroscience: Understanding the mind by simulating the brain*. Cambridge, MA: MIT Press.

Oudeyer P-Y (2006) *Self-organization in the evolution of speech*. Hurford JR (Trans.), Oxford University Press.

Pask G (1961) *An Approach to Cybernetics*.

Peponis J, Hadjinikolaov E, Livieratos C and Fatouros DA (1989) The spatial core of urban culture, *Ekistics* 56(334/335), pp. 43-55.

Pezeshk S and Camp CV (2002) State of the Art on the Use of Genetic Algoriths in Design of Steel Structures. *Recent Advances in Optimal Structural Design. ASCE*, 2002

Ping Y (1996) *Development of Genetic Algorithm Based Approach for Structural Optimisation*, PhD. Thesis, Nanyang Technological University.

Plunkett K and Sinha CG (1992) Connectionism and developmental theory. *British Journal of Developmental Psychology*, 10, pp. 209–54.

Polanyi M (1967) *The Tacit Dimension*. Doubleday and Co., New York.

Pollan M (2008) *In Defence of Food: The myth of nutrition and the pleasures of eating.* Allen Lane, London.

Popper KR (1959) *The Logic of Scientific Discovery*. Hutchinson, London.

——— (1973) The Rationality Of Scientific Revolutions, in Notturno MA (ed.) (1994) *The Myth of the Framework: In Defense of Science and Rationality*, Routledge.

——— (1979) *Objective Knowledge: An Evolutionary Approach*, Revised edition. Oxford University Press.

Prats M, Earl C, Garner S and Jowers I (2006) Exploring style through generative shape description, *AIEDAM Journal*, Cambridge University Press, 20(3).

Prats M, Lim S, Jowers I, Garner SW, Chase S (2009) Transforming shape in design: Observations from studies of sketching, *Design Studies* **30**(5) pp. 503–520.

Psarra S and Grajewski T (2001) Describing Shape and Shape Complexity Using Local Properties. In: *Proceedings of Space Syntax International Symposium III*, Georgia Institute of Technology, Atlanta.

Quatremère de Quincy AC (1832) *Dictionnaire historique d'architecture comprenant dan son plan les notions historiques, descriptions, archaeologiques, biographiques, théoriques, didactiques et practiques de cet art.* Excerpt translated in Rossi A (1982) *The Architecture of the City*, MIT Press.

Quick T, Dautenhahn K, Nehaniv CL and Roberts G (1999) The essence of embodiment: a framework for understanding structural coupling between system and environment, *Proc Third International Conference on Computing Anticipatory Systems*, (CASYS, '99).

Quinlan JR (1993) *C4.5: Programs for machine learning*. Morgan Kaufmann, San Mateo, CA.

Rao S S (1996) *Engineering Optimization: Theory and Practice*. Wiley-Interscience.

Ravindran A, Ragsdell K M, and Reklaitis G V (2006) *Engineering Optimization: methods and applications*. Wiley.

Reich Y (1997) Machine Learning Techniques for Civil Engineering Problems, *Microcomputers in Civil Engineering*, 12 pp. 295-310.

Reich Y and Barai SV (1999) Evaluating machine learning models for engineering problems. *Artificial Intelligence in Engineering* 13 (1999) pp. 257-272.

Reiger BB (1983) Clusters in Semantic Space: Analysing natural language texts to model word meaning as a procedural representation, in: Delatte, L. (Ed.): *Actes du Congr`es International Informatique et Science Humaines, Li`ege (Laboratoire d'Analyse Statistique des Langues Anciennes)* 1983, pp. 805-814.

Reynolds CW (1987) Flocks, herds and schools: A distributed behavioural model. *Computer Graphics*, 21, 25-34.

Ringertz U (1993) On finding the optima distribution of material properties, Structural Optimisation 5, 265-267. Springer.

Rittel HWJ and Webber MM (1984) Planning problems are wicked problems. In Cross N (ed.) *Developments in Design Methodology*, John Wiley and Sons.

Robles-Kelly A and Hancock ER (2003) Edit Distance From Graph Spectra. *Proceedings of the ninth IEEE International Conference on Computer Vision* (ICCV 2003)

Rosen DW (2007) Computer-Aided Design for Additive Manufacturing of Cellular Structures. *Computer-Aided Design & Applications*, 4(5), pp 585–594.

Rosenblatt F (1958) The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain, *Psychological Review*, 65(6) pp. 386-408.

Rosenman, M (1997) The Generation of Form Using an Evolutionary Approach, *in* Dasgupta, D and Michalewicz, Z (eds), *Evolutionary Algorithms in Engineering Applications*, Springer-Verlag, pp. 69-86.

Rossi A (1982) *The Architecture of the City*. The MIT Press.

Rudnyi E B, Van Rietbergen and Korvink J G (2005) Efficient Harmonic Simulation of a Trabecular Bone Finite Element Model by means of Model Reduction. 12th Workshop The Finite Element Method in Biomedical Engineering, Biomechanics and Related Fields, 20-21 July 2005, Ulm University, Germany.

Rumelhart D (1980) Schemata: the Building Blocks of Cognition, In Spiro R, Bruce B and Brewer W (eds.), *Theoretical Issues in Reading Comprehension*. Lawrence Erlbaum, Hillsdale, N.J.

Rummelhart DE and McClelland JL (1986) *Parrallel Distributed Processing: Explorations in the Microstructure of Cognition.* MIT Press, Cambridge, Mass.

Runco MA (1999) Contrarianism, in Runco MA and Pritzker SR (eds.), *Encyclopedia of Creativity*, Academic Press, San Diego, pp. 367–371.

Russell S and Norvig P (1995) *Artificial Intelligence: a Modern Approach*. Prentice-Hall, Inc. New Jersey.

Ryle G (1949) *The Concept of Mind*. Hutchinson, London.

Sarkar P and Chakrabarti A (2008) Studying Engineering Design Creativity: Developing a Common Definition and Associated Measures, *Proceedings of NSF International Workshop on Studying Design Creativity'08*, reprinted in Gero JS (ed.) *Studying Design Creativity*, Springer (Forthcoming).

Saunders R and Gero JS (2001) Artificial creativity: A synthetic approach to the study of creative behaviour, in JS Gero and ML Maher (eds), *Computational and Cognitive Models of Creative Design V*, Key Centre of Design Computing and Cognition, University of Sydney, Sydney, pp. 113-139.

Saussure F de (1974) *Course in General Linguistics.* Fontana, London.

Schön D A (1963) *Displacement of Concepts*. Tavistock, London.

——— (1983) *The Reflective Practitioner: How Professionals Think in Action*. Basic Books, New York.

Schoenhauer M (1996) Shape Representations and Evolution Schemes, *Proceedings of the 5th Annual Conference on Evolutionary Programming.*

Schwabacher M, Ellman T and Hirsh H (1998) Learning to Set Up Numerical Optimizations of Engineering Designs, *AI EDAM* 12(2).

Serra R and Zanarini G (1990) *Complex Systems and Cognitive Processes.* Springer-Verlag, Berlin.

Seuren PAM (2004) *Chomsky's Minimalism*. Oxford University Press

Shea K and Cagan J (1998) Topology design of truss structures by shape annealing, in *Proceedings of DETC98: 1998 ASME Design Engineering Technical Conferences, ASME*, New York. pp. 1-11.

Simon HA (1955) A Behavioural Model of Rational Choice. *The Quarterly Journal of Economics*, vol. 69, pp. 99-118.

——— (1996) *The Sciences of the Artificial.* 3rd ed. MIT Press, Cambridge, MA.

Sischka J, Hensel M, Menges A and Weinstock M (2004) Manufacturing complexity. *Architectural design* 74(3). London: Wiley-Academy.

Skaife AM (1967) The role of complexity and deviation in changing taste. *Proceedings of the American Psychological Association,* Vol. 2 (1967) pp. 25-26

Smith A (1776) *An Inquiry into the Nature and Causes of the Wealth of Nations*. Reprinted (1998) *Wealth of Nations*, Oxford.

Smolensky P (1991) The constituent structure of connectionist mental states: A reply to Fodor and Pylyshyn. in Horgan T and Tienson J (eds.) *Connectionism and the philosophy of mind,* Kluwer, pp. 281–308.

Snodgrass AB and Coyne RD (1997) Is designing hermeneutical? *Architectural Theory Review*, 2(1), pp.65–97.

——— (2006) *Interpretation in architecture: design as way of thinking*. Routledge, London.

Sober E (2000) Simplicity, in Newton-Smith WN (ed.) *A Companion to the Philosophy of Science*, Blackwell.

Sosa R and Gero JS (2002) Computational models of creative situations: Towards multiagent modelling of creativity and innovation in design, in Gero JS and Brazier FMT (eds) *Agents* *in Design 2002*, Key Centre of Design Computing and Cognition, University of Sydney, Sydney, pp. 165-180.

——— (2003) Design and change: A model of situated creativity, *in* Bento, C, Cardosa, A and Gero JS (eds) (2003) *Approaches to Creativity in Artificial Intelligence and Cognitive Science*, IJCAI03, Acapulco, pp 25-34.

——— (2005) Social Models of creativity: Integrating the DIFI and FBS frameworks to study creative design, in Gero JS and Maher ML (eds), *Computational and Cognitive models of Creative Design VI*, pp. Key Centre of Design Computing and Cognition, University of Sydney, Sydney, pp. 19-44

Spector L (2008) Introduction to the Special Issue on genetic programming for human-competitive designs, *AIEDAM: Artificial Intelligence for Engineering, Design, and Manufacturing*, 22, pp. 183–184. Cambridge University Press.

Spiliopoulou G and Penn A (1999) Organisations as Multi-Layered Networks: face to face, e-mail and telephone interaction in the workplace, *Proceedings, 2nd Intl. Space Syntax Symposium*, pp. 1-24.

Steadman P (2008) *The Evolution of Designs: Biological Analogy in Architecture and the Applied Arts*. Routledge.

Steels L (2000) The Emergence of Grammar in Communicating Autonomous Robotic Agents, in Horn W (ed), *ECAI2000*. IOS Press, Amsterdam. pp 764-769.

Stiny G (1976) Two exercises in formal composition. *Environment and Planning B: Planning and Design*, vol. 3 pp. 187-210.

——— (1980) Introduction to shape and shape grammars. *Environment and Planning B: Planning and Design*, vol. 7 pp. 343-351.

Stiny G and Mitchell WJ (1978) The Palladian grammar. *Environment and Planning B: Planning and Design*, vol. 5 pp. 5-18.

——— (1980) The grammar of paradise: on the generation of Mughul gardens. *Environment and Planning B: Planning and Design* **7**(2) 209–226.

Sutherland IE (1963) *Sketchpad, a man-machine graphical communication system*. PhD Thesis, MIT. Reprinted: Kuhn M (ed.) *Technical Report UCAM-CL-TR-574,* University of Cambridge Computer Laboratory. ISSN: 1476-2986.

Suykens JAK, Van Gestel T, De Brabanter J, De Moor B and Vandewalle J (2002) *Least Squares Support Vector Machines*, World Scientific, Singapore.

Szczepanik W, Arciszewski T and Wnek J (1996) Emperical Performance Comparison of Selective and Constructive Induction, *Engineering Applications of Artificial Intelligence* 9(6) pp. 627-637.

Tang H-H and Gero J (2001) Sketches as affordances of meanings in the design process. In J. S. Gero, B. Tversky and T. Purcell (eds) (2001) Visual and Spatial Reasoning in Design II, Key Centre of Design Computing and Cognition, University of Sydney, Australia, pp. 271-282.

Thomson D (1917) *On Growth and Form*. Abridged ed. (1961) Cambridge University Press.

Tillmann, B, Abdi, H and Dowling, WJ (2004) Musical style perception by a linear auto-associator model and human listeners, *Proceedings of the 8th International Conference on Music Perception & Cognition*, Evanston, IL.

Turk M and Pentland A (1991) Eigenfaces for Recognition, *Journal of Cognitive Neuroscience*, 3(1) pp. 71-86.

Turner A (2001) Depthmap: a program to perform visibility graph analysis. In *Proceedings 3rd International Symposium on Space Syntax* pp. 31.1–31.9

——— (2005) An Algorithmic Definition of the Axial Map, *Environment and Planning B: Planning and Design*, 32(3) 425-444.

——— (2006) First steps in evolution of animats for pedestrian simulation. Presented at *Design Computing Cognition '06, Workshop on Models of Machine Learning in Creative Design*, Eindhoven.

——— (2007) To move through space: lines of vision and movement. In *6th International Space Syntax Symposium* pp. 037.001–037.012

Turner A, Doxa M, O'Sullivan D, and Penn A (2001) From isovists to visibility graphs: a methodology for the analysis of architectural space. *Environment and Planning B: Planning and Design*, 28(1) 103-121.

Turner A and Penn A (2002) Encoding natural movement as an agent-based system: an investigation into human pedestrian behaviour in the built environment. *Environment and Planning B: Planning and Design*, vol. 29 no. 4, pp. 473-490.

Turner M, Clough RW, Martin H C and Topp L J (1956) Stiffness and Deflection Analysis of Complex Structures, *Journal of Aeronautical Science* 23 (9), pp. 805-823.

Van Dam ER and Haemers WH (2002) Spectral Characterizations of Some Distance-Regular Graphs, *J. Algebraic Combin.* **15**, pp. 189-202.

Van Leusen M (1993) A System of Types in the Domain of Residential Buildings. Delft University of Technology.

Van Gelder T and Port R F (1995) It's about time: An overview of the dynamical approach to cognition. In Port R F and Van Gelder (Eds.) *Mind as Motion*. MIT Press. pp 1–43.

Vapnik V (1995) *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

Varela FJ Thompson E and Rosch E (1991) *The Embodied Mind: Cognitive science and human experience,* MIT Press, Cambridge Ma.

Vitruvius (2001) *Ten Books on Architecture*. Cambridge University Press.

Vitz PC (1972) Preference for tones as a function of frequency (Hz) and intensity (db), *Perception and Psychophysids*, 11, 84-88.

Von Buelow P (2002) Using Evolutionary Algorithms To Aid Designers of Archictural Structures, in Bentley, PJ and Corne DW. (Eds.) *Creative Evolutionary Systems*, Morgan Kaufmann Pub.

Von Neumann J (1958) *The Computer and the Brain*, Yale University Press.

Wang LH, Liu J, Li YF and Zhou HB (2004) Predicting Protein Secondary Structure by a Support Vector Machine Based on a New Coding Scheme, *Genome Informatics*, 15(2) pp. 181-190.

Watts D J (2003) *Six Degrees*. W. W. Norton and Company, Inc. New York.

Weaver W (1948) Science and Complexity, *American Scientist*, 36: 536-544.

de Weck O (2004) Multiobjective Optimization: History and Promise", *The Third China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems*, Kanazawa, Japan.

Westfall CW (1991) Building Types. In van Pelt RJ and Westfall CW, *Architectural Principles in the Age of Historicism*. Yale University Press, New Haven.

Weston R (2004) *Key buildings of the twentieth century: plans, sections, elevations*. W W Norton & Co. Inc., New York.

Wheeler M (1996) From Robots to Rothko: the Bringing Forth of Worlds, in Boden MA (ed.) *The Philosophy of Artificial Life*. Oxford University Press, pp.209–36.

——— (2005) *Reconstructing the Cognitive World: the Next Step*. The MIT Press, Cambridge MA.

Whitehead AN (1941) Mathematics and the Good, in Schilpp PA (ed.) *The philosophy of Alfred North Whitehead*, Northwestern University, Eranston.

Wiener N (1948) *Cybernetics: or Control and Communication in the Animal and the Machine*, MIT Press, Cambridge, MA.

Williams C (2002) The analytic and numerical definition of the geometry of the British Museum Great Court Roof. In Burry M, Datta S, Dawson A, and Rollo A J (eds.) *Mathematics & design 2001*, Geelong: Deakin University, pp. 434-440.

Wilson A (2006) Ecological and urban systems models: some explorations of similarities in the context of complexity theory, *Environment and Planning A*, 38, pp. 633–646.

——— (2010) *Knowledge Power: Interdisciplinary education for a complex world*, Routledge.

Wittgenstein L (1958) *Philosophical investigations*. Translated by G.E.M. Anscombe, 3rd ed. (2001) Blackwell, Oxford.

Wolfram S (1994) *Cellular Automata and Complexity: Collected papers*, Addison-Wesley, Reading MA.

——— (2002) *A New Kind of Science*. Wolfram Media Inc.

Wundt W M (1874) *Grundzüge de physiologischen Psychologie.* W. Engelman, Leipzig.

Yates J F and Estin P A (1998) Decision Making. In Bechtel W and Graham G (Eds.) *Companion to Cognitive Science*. Blackwell Publishers Ltd, Oxford.

Yan W and Kalay Y E (2006) Geometric, cognitive and behavioural modelling of environmental users: Integrating an agent-based model and a statistical model into a user model. In Gero JS (ed.) *Design Computing and Cognition ’06*, Springer, The Netherlands, pp. 61-79.

Zhu P and Wilson RC (2005) A Study of Graph Spectra for Comparing Graphs. *British Machine Vision Conference 2005.*

Zwicky AM and Sadock JM (1975) Anbiguity tests and how to fail them, in Kimball J (ed.) *Syntax and Semantics*, **4**, Academic Press, New York, pp. 1–36.