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. pp 563-582. ISBN: 978-1-4020-5130-2

INDUCTIVE MACHINE LEARNING IN MICROSTRUCTURES

Estimating a finite element optimization using support vector machines

SEAN HANNA, SIAVASH HAROUN MAHDAVI
University College London, United Kingdom

Abstract. A support vector machine is trained to produce optimal structures. The problem of structural optimization is typically solved by a search procedure that samples and repeatedly evaluates a physics-based model, but this process is computationally demanding. Instead, the use of a learning algorithm to generate new structures based on previously optimized examples is described that provides enormous computational saving. The results have shown that the predicted structures are accurate, and the process is highly efficient for cases in which similar optimizations must be performed repeatedly, especially as the number of such optimisations grows.

1. Introduction

Nature builds by trial and error, via the very effective but slow and costly process of evolution. As humans, our capacity to learn from experience has given us the ability to engineer and build based on our knowledge, and while our designs may not outdo nature in all her complexity, they can excel when the problem is simple and well defined. An engineer can design a building or bridge that is structurally sound without the need for failed attempts. Although for several centuries the mathematical tools for explicit analysis have been dominant, the vast majority of design decisions throughout history have been based on experience of precedents: a practiced builder would know what would stand or fall without having to test it. In a similar fashion, this paper demonstrates that nearly optimal solutions to a well defined structural design problem can be found by training a machine learning algorithm on examples of other solutions found by a traditional optimization procedure.

Once trained, the advantage of such a machine is the same advantage that the human builder's training and experience give: the ability to build quickly and without failed attempts. A structural problem is chosen that involves the repeated optimization of many interconnected modules, and thus takes full advantage of this increase in speed. It is also a problem of sufficient complexity that the solution can not be calculated directly, but must be found by slow simulation and testing. An algorithm capable of arriving at a general solution by inductive learning on presented examples is thus highly beneficial.

Given a parameterized structure and a set of loading conditions, it has been shown that various optimization algorithms can be used to design an effective shape to counter the given load. Search procedures including gradient descent (GD) and genetic algorithms (GA) make repeated evaluations of the strength of different structures to do this (Schoenhauer 1996; Von Buelow 2002; Chen 2002; Hanna and Mahdavi 2004). If the load conditions change, the optimal structure will also be different and the optimization can be rerun to find the new shape.

This process is time consuming however, requiring repeated iteration for each new design, and is subject to error due to local optima in the search space. This paper uses inductive learning to eliminate the need for this iterative step once sufficient examples have been generated to save processing time and achieve more constant fitness of solutions. If the optimisation is repeated many times for many sets of loading conditions, the optimal shape of the structure can be considered a function of the load. The work in this paper uses a support vector machine to learn this function of optimal structures given the tensile or compressive loads in each axis, and results in a very efficient and accurate alternative to iterative optimisation.

2. Background

The work presented here draws on previous structural optimisation research by the authors, but extends this by replacing the optimisation step with learning. Before addressing the problem, this section provides a background of related research. First, the particular structural problem is defined, followed by a review of relevant structural optimization and machine learning methods.

2.1 BACKGROUND: THE STRUCTURE

Space frame structures are investigated in this work: 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 specific problem addressed is that of small scale microstructures, an example of which is shown in the photograph (Figure 1) below. The overall dimensions of this object as fabricated are 1cm × 1cm × 2cm, and the individual struts within it are less than 1mm in length. Relevant aspects of the design method will be briefly reviewed in this section.

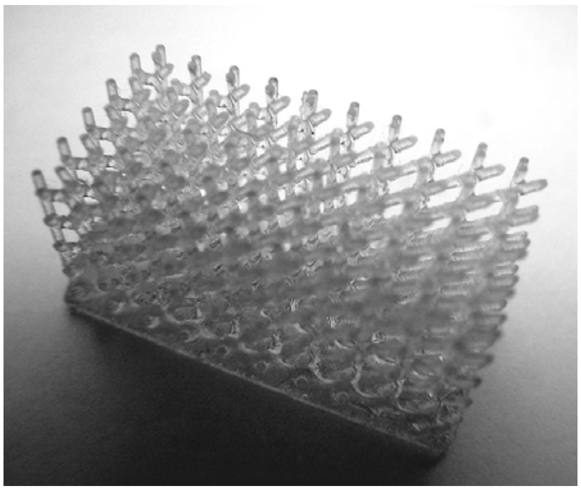

Figure 1. A modular structure fabricated by stereolithography.

2.1.1 Defining the structures

To define a particular space frame one must specify both the members themselves, and the locations and orientations of the nodes in 3-dimensional space. We refer to these as the topology and geometry of the structure respectively.

The distinction between geometry and topology can be described by an example 2-dimensional illustration. Geometry refers specifically to the positions in space of the node points joining the structural members. The following diagrams are of two structures with the same topology but different geometries. As can be observed, the connections and number of members are the same, but the coordinates and orientations of these members differ, (Figure 2 a & b). 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 2 a & c).

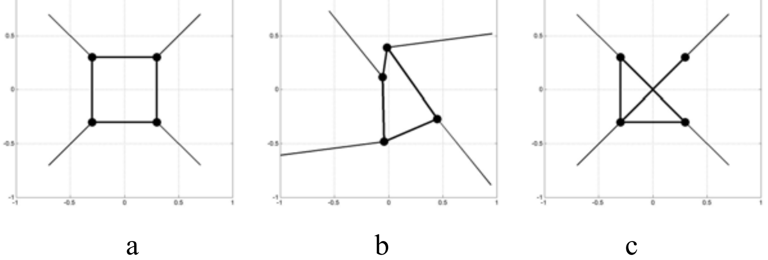

Figure 2. An illustration of a change in the geometry and topology of a structure.

2.1.2 Intended structure and fabrication

The structural problem considered in this work is one based on a modular topology, so that it can be scaled to volumes of any size. A large volume of structure can be subdivided into a grid of cubes, to which we refer as 'unit cubes', each containing a portion of structure with identical topology such that each is connected to its neighbours to form a continuous structure. (Figure 3)

Previous work by the authors has resulted in a method for optimising large and complex structures very efficiently using this modular 'unit cube' approach. (Hanna and Haroun Mahdavi, 2004) An object under a complex loading condition exhibits differing stresses at various points in its volume. 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 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. Both genetic algorithms and gradient descent (Haroun Mahdavi and Hanna, 2004) have been used to find this optimal, using the finite element method to simulate the effects of loading.

The ideal result is a modular structure as displayed in Figure 3 (bottom), with gradual changes in the geometry of the structure as the stresses change continuously across the volume of the object. It is very efficient, with material concentrated in high stress zones and internal struts aligned to counter the changing direction of the stress vectors. To arrive at this, the optimization of structural units must be made repeatedly, once for each unit cube of differing stress. The structural problem is therefore similar to time series problems of control and dynamic systems, but static: instead of changing in time, the geometry morphs in space. Because a similar type of optimization must be performed many times, this paper proposes the method of learning the function of optimal structures from a training set of previously optimized geometries.

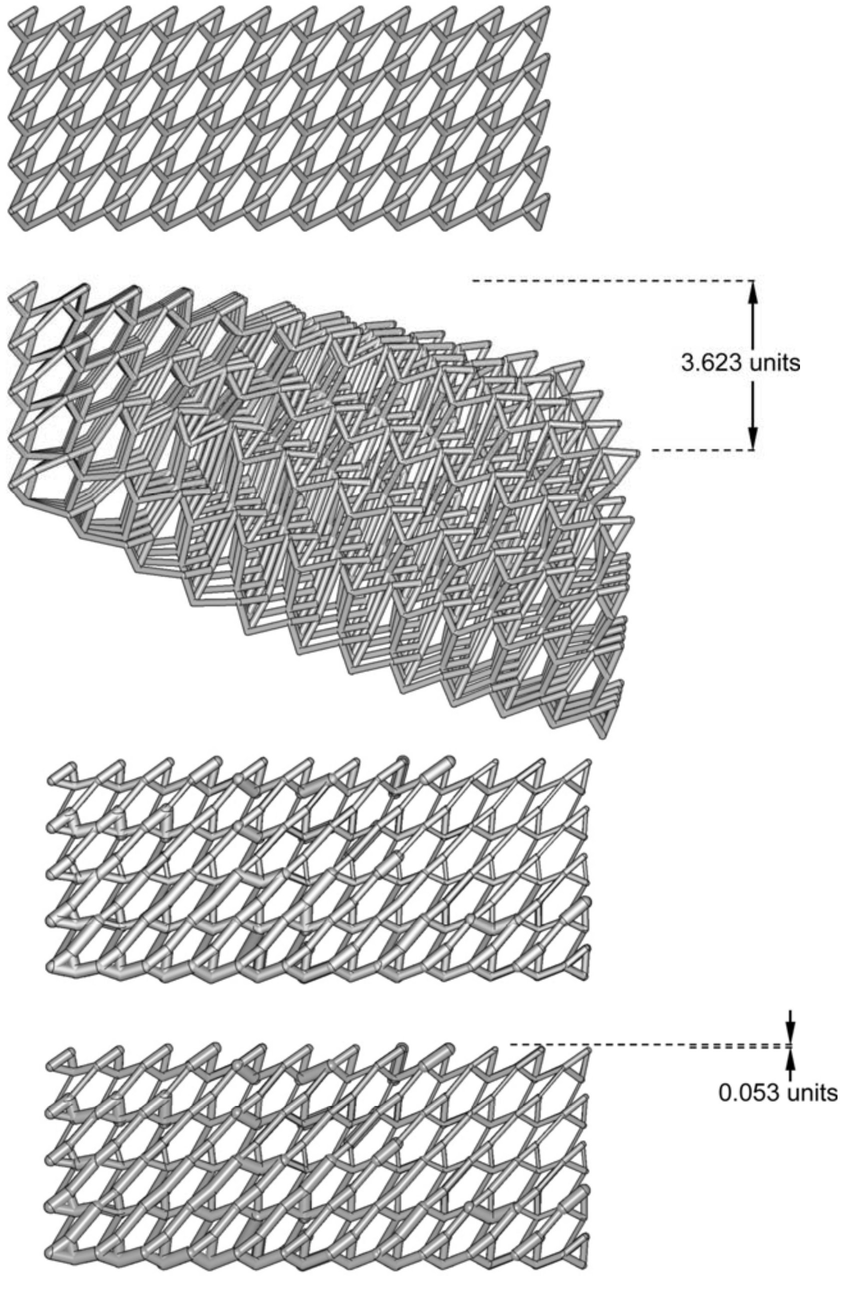

Figure 3. A modular space frame forming a cantilever beam. Both have the same overall mass and topology, but identical modules (top) deflect far more under loading than do the individually optimised ones (bottom).

2.2 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 by the authors 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 (Haroun Mahdavi and Hanna, 2004). These were measured (Haroun Mahdavi and Hanna, 2003) and factored in to the examples presented to the machine for learning.

2.3 OPTIMISATION OF STRUCTURES

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). Because of the repetitive nature of the problem and the well defined behaviour of structures, simulation by the Finite Element Method (FEM) is both the most efficient and accurate. In the design task under consideration here it is a set of optimal solutions that is required.

Several techniques have been devised for generating the topology of continuous solids analysed by FEM. Both GA and non-random iterative methods have been used. Marc Schoenhauer (1996) reviews a number of GA methods for generating topology in 2D or 3D space to optimise structural problems involving continuous shapes, in which the genetic representation can determine a configuration of holes and solid using Voronoï diagrams or a list of hole shapes. Yu-Ming Chen (2002) uses a non-random iterative process of shifting node points in the FEM representation toward high stress zones to examine similar problems. 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 has been in refining only the shape or member sizes, rather than the topology (in terms of members connecting the node points of the structure). Adeli and Cheng (1993) use a GA 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). Yang Jia Ping (1996) has developed a GA that determines both shape and topology, which must begin with an acceptable unoptimised solution and refine the topology by removing connections. Peter von Buelow (2002) used a two stage algorithm nesting one GA within another. 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. Previous work by the authors has also used GA for both topology (Haroun Mahdavi and Hanna 2003) and geometry, but it has been found that gradient descent is more efficient for shape optimization (Haroun Mahdavi and Hanna, 2004).

2.3.1 Optimisation by gradient descent

The optimisation performed is gradient descent to minimise the total deflection in a structure under the specified load, as applied to a unit cube. Simulation of this deflection is performed using the finite element method.

2.4 MACHINE LEARNING FOR BEHAVIOUR AND STRUCTURE

Machine learning has long been applied to structures and in the domain of civil engineering, most commonly as an enhancement of the optimisation process. A recurring bottleneck in optimisation 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).

Some of the most recent and complex problems involve structural prediction in the field of bioinformatics, in which the molecular composition of proteins can 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 been used to find a function to predict movement in time of a dynamic system, which is in some ways similar to structural problems. In both cases 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.

Regardless of the method used in simulation, the repeated iteration of generating and evaluating solutions is the other major hurdle in optimisation. 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.

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.

While much provious research has concentrated on inferring rules to guide a design (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), we use induction to derive a function that directly maps a given load condition to an optimal solution.

2.4.1 Algorithm selection

The choice of algorithm is dependent on the learning problem at hand, including the form and availability of data, and the goal of learning (Reich 1997). The goal, in this case, is induction: to derive a generalisation based on previous evidence of optimal structural solutions. Duffy (1997) lists six major machine learning techniques, of which three potentially apply.

- *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.
- 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) above, yield explicit classification or parameter ranges, and have therefore been used to estimate behaviour or recommend design decisions in symbolic or labelled form.
- *Artificial neural networks* are part of a class of sub-symbolic algorithms (including, more recently, support vector machines) 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.

The data form is most suited to the third category. The solution to structural shape is naturally a continuous function, and it has been noted that discretisation is detrimental to optimisation performance (in tests by the authors), or can lead to large learning error rates (Reich, 1997). 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.

3. Learning methodology

3.1 THE ALGORITHM

Support vector machines (SVM) (Vapnik 1995) are chosen to perform the learning described in this paper. They 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

$$\mathbf{y} = f(\mathbf{x}) \tag{1}$$

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

$$\mathbf{y}' = f'(\mathbf{x}). \tag{2}$$

The type of SVM used by the authors is a Least Squared SVM, in which the solution follows from solving a set of linear equations, instead of quadratic programming for classical SVMs (Suykens et al. 2002). The kernel is the commonly used Gaussian radial basis function.

3.1.1 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 node points that represent the optimal structure (Figure 4). As a function (1), 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:

$$\mathbf{x} = (x_{(\mathrm{x})}, x_{(\mathrm{y})}, x_{(\mathrm{z})}). \tag{3}$$

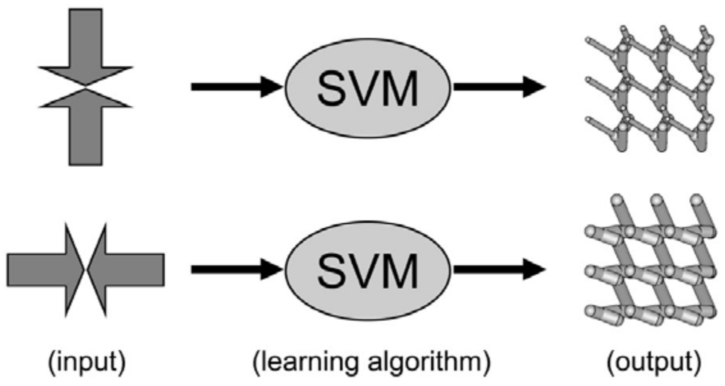

Figure 4. Different force inputs result in ideal geometry outputs.

The output structure **y** consists of the node point positions for the optimal structure as found by prior optimisation: gradient descent as described in section 2.3. This is the set of (x, y, z) coordinates for each of the node points y_i:

$$\mathbf{y} = (y_{1(x)}, y_{1(y)}, y_{1(z)}, y_{2(x)}, y_{2(y)}, y_{2(z)}, \ldots, y_{n(x)}, y_{n(y)}, y_{n(z)}), \tag{4}$$

The nodes are also located in three-dimensional space, so for a topology of n points the output **y** is a $3n$-dimensional vector.

3.2 THE DATA SET

A single topology was used consisting of four node points per unit cube, resulting in a 12-dimensional output **y**. 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 $\mu = 0$ and standard deviation $\sigma = 1$, 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. The node point outputs **y** are found by the gradient descent method described in section 2.3, and result in asymmetrical distributions of node positions throughout the space of the unit cube. The distributions of each of the four node points in the three axes of space are shown in Figure 6. 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.

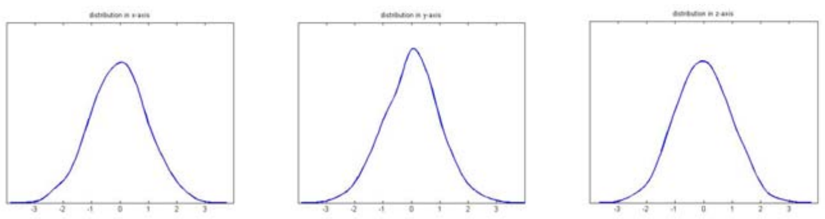

Figure 5. Probability distributions of the x-axis, y-axis and z-axis components of input force vector **x** are based on a normal distribution with mean zero.

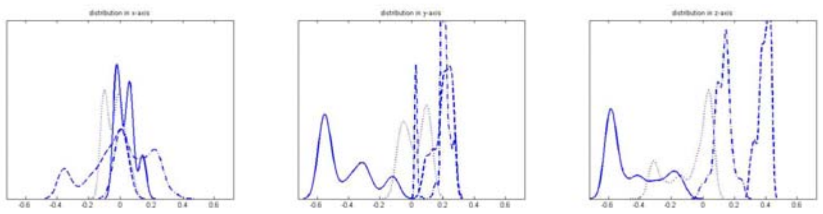

Figure 6. Probability distributions of the x-axis, y-axis and z-axis components of the node points **y** are asymmetrical in physical space, the result of optimization.

3.3 TRAINING

This work investigates whether the SVM can be trained 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 is online, with a gradually increasing data set rather than as a batch process.

Training of the SVM was performed with the gradually increasing set of stress vectors **x** and node points **y** until the accuracy of the learned function no longer increased. A radial basis function kernel with variance σ2 = 0.2. was used to map all dimensions in the SVM.

3.3.1 Error estimation

Methods of error estimation have been systematically evaluated in (Reich and Barai 1999). We have used the most common method, hold-out, which is also 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 $0.2n$ to $0.4n$. 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).

Our 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 the same, randomly selected set of size 300. The size of D for which the SVM will be considered in our tests to be fully trained occurs at $n > 950$, which is approximately equal to the recommended size for 0.95 confidence, and errors for even smaller training sets have the most pessimistic bias of any estimation method. Our results therefore display the worst case estimation of errors, and the true accuracy of the algorithm is likely to be no less than is reported in the following sections.

4. The trained algorithm: results and analysis

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 three graphs below, 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 slightly less than 650 samples were given, at which point the performance plateaued at a very high level.

4.1 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

$$\theta = 1/n \left[\Sigma_{i=1:n} \; (y_i - f'(\mathbf{x}_i))^2 \right] \tag{5}$$

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

$$\theta = 1/n \left[\Sigma_{i=1:n} \; \left(\Sigma_{j=1:d} \; | y_{ij} - f'(\mathbf{x}_i) |^k \right)^{1/k} \right] \tag{6}$$

where d is the dimensionality of the output vector **y** and k is the exponent of the metric. The choice of k=2 (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 Manhattan metric of k=1 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 output by the SVM $\mathbf{y}' = f'(\mathbf{x})$. Distance here is measured by the Manhattan metric (or sum of the difference in each dimension of the 12-dimensional output vectors). The graph below displays the accuracy of the predicted nodes during training with an increasing set *T* of examples and a separate validation set *V* of 300 examples (Figure 7). It indicates a steadily decreasing error for training sets *T* up to approximately 650 (indicated by '○'), at which point there is little further perceptible change.

The number of 650 training examples appears to be a result of the particular data set, rather than inherent in the algorithm, and it is likely the required size of training set *T* would fluctuate 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 at *T* size 650 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 size 650 is likely simply to be the limit of learning for this problem.

The average accuracy of the function at this point is within 0.005 units to the validation set, or 1/5 the 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.

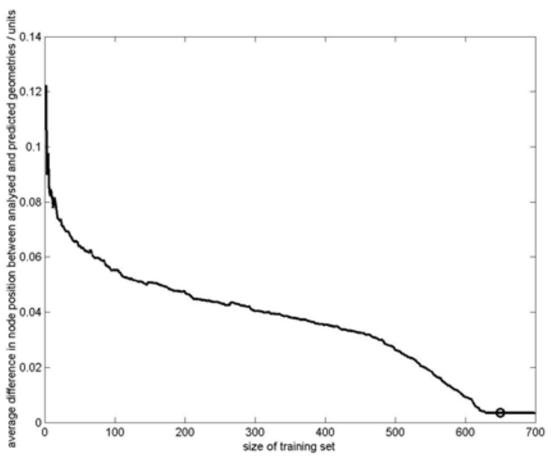

Figure 7. Accuracy of learning increases with increased training

4.2 PERFORMANCE OF THE PREDICTED GEOMETRIES

While the above graph indicates 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 $\mathbf{y}' = f'(\mathbf{x})$ as found by the SVM. Both are loaded with the same input vector of stresses, and their strengths under this load condition are measured as a total displacement of nodes when the load is applied.

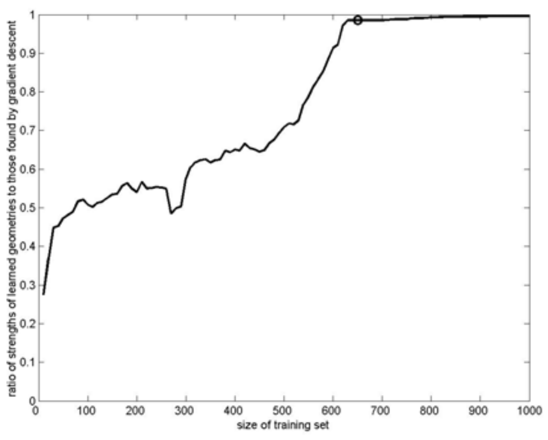

Figure 8. Performance of the structure increases with increased training.

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

$$disp(\mathbf{y}, \hat{\mathbf{y}}) = \Sigma_{i=1:m}\ [(y_{i(x)} - \hat{y}_{i(x)})^2 + (y_{i(y)} - \hat{y}_{i(y)})^2 + (y_{i(z)} - \hat{y}_{i(z)})^2)]^{1/2} \quad (7)$$

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

$$\delta = 1/n\ [\ \Sigma_{i=1:n}\ (\ disp(\mathbf{y}, \hat{\mathbf{y}})\ /\ disp(\mathbf{y}', \hat{\mathbf{y}}')\ )\ ] \quad (8)$$

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

Figure 8 plots this performance *δ* of the predicted structures **y'** against the same validation set as in Figure 7. A ratio of 1.0 would indicate the predicted structures perform (on average) as well as those found by GD. Again the improvement with an increasing training set is evident over the same range, with a ratio nearly approaching 1.0. The percentage difference between the resulting displacement of the original samples and the predicted geometries at a training set size of 650 had dropped to 1.51%. Again this occurred slightly at slightly less than 650 samples.

4.3 LEARNED IMPROVEMENTS OVER THE TRAINING SET

As the average performance of structures over the entire prediction set approaches that of the validation set, it can be seen that some predicted structures actually perform better than their equivalents as found by GD. 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. Figure 9

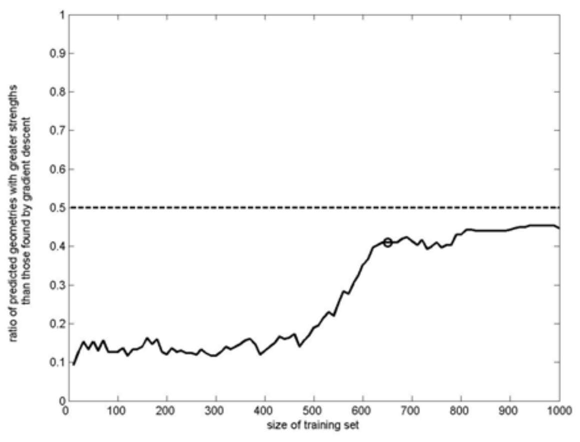

Figure 9. Learned improvements over the training set.

indicates the number of structures of greater strength found by learning for increasing training set sizes. Where 50% would represent the maximum expected value of a perfectly learned function on data with noise, we approach this value at the training set size of 650 with 42% of structures having greater strength 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 was generated. Gradient descent 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. Although it has been shown to be an appropriate method for solving the structural shape optimization problem, it can only do so within an acceptable variance in node positions (Haroun Mahdavi and Hanna, 2004). 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 $f'(\mathbf{x})$.

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 2.1) 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 optimization 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 strength.

5. Conclusions

The aim of this work is principally to investigate whether machine learning algorithms, in particular SVMs, could accurately predict the optimal geometries of structures, and thus be used as a substitute for a traditional optimization algorithm. An SVM was trained on example structures that had been optimized for strength using gradient descent, and used to predict structures that performed almost as well as an independent validation set optimized by GD. Several conclusions can be drawn from the observations:

- *The accuracy approaches that of the GD optimization*. Although the learned function is not as accurate as GD for optimization, it does come close. The function learned from the training samples is learned with a high level of accuracy, but this can never be perfect for any data set. More importantly, if the potential sub-optimal geometries in the training sat are treated as noise in the data, it is evident that the SVM learns a function that improves on some if the initial data. On average, this produced geometries with a deflection under stress only 1.51% greater than those found by GD with a training set of 650. The variance in performance at this point is also low, representing a high degree confidence in these solutions.
- *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 results in a smoother overall structure*. The avoidance of overfitting by a smoother learned function is beneficial both at the scale of the individual unit, and the whole structure. In the first instance, some predicted structures can actually perform better than what would be found by GD in instances where GD results in sub-optimal local optima. In the second instance, the overall combined structure benefits by a continuous functional estimation by producing a more gradual transition between adjacent unit cubes. This avoids potential weak points caused by recombining individually optimized structures.

- *The learned function is quicker for optimising larger structures*. Finding an optimal structural based on the learned function is far quicker than performing a full optimization 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. Learning the function for optimal structures however is time consuming, as in the example case studied, 650 fully optimized examples were required to learn the function at the outset. Many structural problems require the optimization to be performed only once, but for those in which a similar structural optimization is needed repeatedly, the initial investment in learning the function of optimal geometries can make the overall optimization far more efficient. In the case of an object composed of many units of an arrayed topology as shown, the computation time becomes less for the learned function as the size of the object grows beyond 650 unit cubes. Larger sizes take an even greater advantage in time. As this method of optimization is meant to be scalable to ever-larger objects, the learned function represents a substantial advantage in speed.

When the problem is well defined, i.e. the environment and topology are constant and the loads can be quantified by a continuous valued vector, we have shown that it is possible to learn the function of optimal structures given the specified loading condition. Rather than optimization by repeated sampling and evaluation of a physics-based model, it is thus possible to make design decisions for this structural problem based entirely on learning from previous examples. We have shown a method that uses this technique to predict optimal structures based on this principle, in which the training is performed in advance, and a structure is then produced that rivals the initial training set in strength. For structures of repeated units of the type we are considering, this method is many times more efficient than standard optimization algorithms, and is thus a significant contribution to this problem of structural design.

The problem has been formulated as one of microstructures, comprised of a very large number of units with pre-defined topology but flexible geometry. The units used however, have been defined only relatively, and there is no reason in principle why such a technique could not be applied to structures of a larger size. As the training requires several hundred examples, the practical benefit of this approach in terms of speed is only evident when structures contain a number of units far greater than this, as do the microstructures we have been considering even of several centimetres. The rapid prototyping technologies used, however, are only part of a class of manufacturing methods including CNC cutting and milling that are being used at much larger scales. With recent architectural projects in excess of one kilometre and the enclosure of entire city neighbourhoods with space frame roofs becoming feasible, such an approach to optimisation may be valuable.

Most unexpected of the findings was that in generalising from the examples presented, the learning algorithm was so often able to actually outperform the original optimisations on which it was trained. Once trained on successful precedents, the machine, in a sense, knows intuitively what works based on its prior experience, and can then predict optimal structures that rival or even exceed the initial training set in strength. This is a result not of strict analysis however, but of inductive learning.

Acknowledgements

The authors would like to thank Dr. Joel Ratsaby and Prof. Bernard Buxton for their guidance and helpful suggestions in this work.

References

Adeli, H and Cheng, N: 1993, Integrated Genetic Algorithm for Optimisation of Space Structures, *Journal of Aerospace Engineering*, Vol. 6, No. 4.

Arciszewski T and Ziarko W: 1990, Inductive Learning in Civil Engineering: Rough Sets Approach, *Microcomputers in Civil Engineering* 5 pp. 19-28.

Chen, YM: 2002, *Nodal Based Evolutionary Structural Optimisation Methods*, PhD Thesis, University of Southhampton.

Duda, RO, Hart, PE and Stork DG: 2001, *Pattern classificatio*. 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.

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.

Hanna, S and Haroun Mahdavi, S: 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.

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.

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 2nd International Conference on Soft Computing and Intelligent Systems and 5th International Symposium on Advanced Intelligent Systems*, Yokohama, Japan.

Kicinger R, Arciszewski T and De Jong K: 2005, Parameterized versus Generative Representations in Structural Design: An Emperical Comparison, *Proceedings of GECCO ’05*, pp. 2007-2014.

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.

Molecular Geodesics, Inc.: 1999, Rapid prototyping helps duplicate the structure of life, *April 99 Rapid Prototyping Report*.

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.

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.

Ping, Y: 1996, *Development of Genetic Algorithm Based Approach for Structural Optimisation*, PhD. Thesis, Nanyang Technological University.

Quinlan JR: 1993, *C4.5: Programs for machine learning*. Morgan Kaufmann, San Mateo, CA.

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.

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).

Sischka, J., Hensel, M., Menges, A. and Weinstock, M: 2004, Manufacturing complexity. *Architectural design* 74(3). London: Wiley-Academy.

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.

Vapnik V: 1995, *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

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.

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.