Learning by selective plasmid loss for intracellular synthetic classifiers

Oleg Kanakov[a], Shangbin Chen[b], Alexey Zaikin[c,d,a,b,e,f,*]

[a] *Lobachevsky State University of Nizhniy Novgorod, Prospekt Gagarina 23, Nizhniy Novgorod, 603022, Russia*
[b] *Britton Chance Center for Biomedical Photonics, Wuhan National Laboratory for Optoelectronics-Huazhong University of Science and Technology, Wuhan, China*
[c] *Department of Mathematics, University College London, Gower Street, London, WC1E 6BTUK, United Kingdom*
[d] *Institute for Women's Health, University College London, Medical School Building 74 Huntley Street, London, WC1E 6AU, United Kingdom*
[e] *Centre for Analysis of Complex Systems, Sechenov First Moscow State Medical University, Trubetskaya Street 8 bldg. 2, Moscow, 119991, Russia*
[f] *Institute for Cognitive Neuroscience, University Higher School of Economics, 20 Myasnitskaya, Moscow, 101000, Russia*

Abstract

We propose a learning mechanism for intracellular synthetic genetic classifiers based on the selective elimination (curing) of plasmids bearing parts of the classifier circuit. Our focus is on a two-input, two-plasmid classifier scheme designed to solve a simple proof-of-concept learning problem. The problem is formulated in terms of Boolean variables, and the learning process boils down to selecting the classification rule from three options, given a set of training examples. We begin with a Boolean description of the classifier circuit, demonstrating how it implements the required learning algorithm. We then transition to a continuous steady-state model and establish conditions on its parameters to ensure that the learning process and the classifier output correspond to the Boolean description, at least approximately. The approach to intracellular classifier learning presented here essentially relies on two key prerequisites: (i) compatibility among the plasmids constituting the classifier, such that they have independent or weakly interacting copy number control systems, and (ii) conditional elimi-

*Corresponding author
Email address: okanakov@rf.unn.ru (Oleg Kanakov)

nation mechanism in each plasmid triggered by a signal from the gene network. The feasibility of this approach is supported by recent experimental findings on engineering compatible pairs and triplets of plasmids and controlled selective plasmid curing. While learning by plasmid loss has certain limitations in universality, we anticipate that it provides greater persistence of a trained classifier to internal and external fluctuations and to degradation over time, as compared to alternative intracellular learning mechanisms outlined in the literature, such as based on gene network dynamics or on variable copy numbers of plasmids sharing a common copy number control system.

Keywords: classifier, learning, synthetic gene networks, intracellular intelligence

---

1. Introduction

Since the identification of the molecular DNA structure by Crick and Watson in 1953 [1] and the model for the regulation of enzyme synthesis in cells by Jacob and Monod in 1961 [2], considerable scientific breakthroughs have been made in the lapse of a few decades. As the desire to understand and delve deeper into the way our molecular systems work has increased, the need for mathematical tools modeling the way these system function has exponentially grown as well. A specific field where substantial growth has been witnessed is synthetic biology, which is defined as the science of the design and construction of novel functional devices, systems, and organisms by applying mathematical, engineering and computational principles [3, 4]. The principal aim in synthetic biology is to create synthetic genetic designs or construct nanorobots which can perform predictable functionalities [5]. This is achieved by using standardized biological building blocks to design and create functional devices with various purposes. For example, Nissim et al. developed a synthetic dual-promoter circuit that works like a digital AND gate, to effectively recognise and kill cancerous cells. It does so by releasing a toxic derivative only when both cancer-indicating signals are present in the cell, resulting in targeted cancer cell death [6].

The advancements made in synthetic biology provide possible industrial applications and new therapeutic opportunities for future gene and cell based therapies [7], such as designing "smart" cells that can respond to the physiological status of a patient, thus providing tailored treatment for the cellular physiology of each individual [8].

As many essential functions in living cells depend on the interaction of a network of bio-molecules, it is important that genetically programmed cells have sensors that receive information, circuits that process the inputs, and actuators to link the circuit output to a cellular response [9]. Due to the similarities in topology between genetic networks and electronic devices, gene networks are often referred to as "gene circuits" [10, 11]. Numerous gene regulatory circuits have been designed and created to this day. One of the earliest examples of a synthetic gene circuit is the repressilator [12], and the consequent further developments in synthetic biology resulted in the engineering of a vast number of networks such as transcriptional or metabolic oscillators, spatially coupled and synchronised oscillators, calculators, inducers of pattern formation, learning systems, optogenetic devices, memory circuits and logic gates. One of the much awaited kinds of synthetic gene circuits with principally new functionality would work as intelligent biosensors, for example, realized as genetic classifiers able to assign inputs with different classes of outputs.

Implementing a learnable classifier in a synthetic gene circuit remains a challenging problem because of current tight limitations upon synthetic gene network complexity in a single cell. In [13, 14] it was proposed to address this challenge by using an ensemble of cells, each harboring an elementary synthetic gene circuit with non-identical parameters across the ensemble. It was shown in [13, 14] that by combining the outputs of multiple non-identical cells, the entire ensemble may act as a multi-cell "distributed" classifier, which is capable of implementing more complex classification rules than it is possible within a single cell; moreover, this distributed classifier admits learning by adjusting the composition of the cell ensemble, without requiring any intracellular learning mechanisms in the individual cells. Distributed classifier learning can be

achieved by selection of cells using common methods of synthetic biology, e.g. by means of fluorescense-activated cell sorting [15], based on the responses of the individual cells to the training examples; two strategies of such selection were proposed in [13, 14]. Note, however, that this approach in its original multi-cell form does not solve the problem of implementing a learnable classifier at the intracellular level, which becomes necessary e.g. when the classifier is considered as part of a larger gene network within a cell.

The present study aims at filling this gap by adapting the selection-based learning approach to the intracellular gene network level. We note that the key idea behind distributed classifier learning is that parts of the gene network constituting the classifier are selectively eliminated or kept during learning. In order to adapt this approach to the intracellular level, we propose that the classifier gene circuit is distributed across plasmids within a cell, and these plasmids are subject to selective elimination, depending on their responses to the training examples. In this paper we propose a structure of a synthetic gene classifier circuit implementing this principle, we describe a learning strategy based on selective plasmid loss for this classifier, and show by analyzing a mathematical model that this strategy solves a simple classifier learning problem.

A similar principle based on controlling the ratio of copy numbers of two plasmids within a cell (referred to as synthetic plasmid heteroplasmy) was suggested in [16] for implementing intracellular memory; moreover, a learnable multicellular decision-making system (thus more similar to distributed classifiers of [13, 14] than to intracellular classifiers) based on this principle was suggested in [17]. Another approach to learning in synthetic gene networks suggested in the context of associative memory [18, 19] is based on slow decay rates of intermediate transcription factors, which act as learning parameters. The lifetime of the learned state is then limited by the decay time of these transcription factors; this lifetime can be extended using bistable genetic switches as memory elements [20], but it still remains limited due to stochastic transitions in bistable elements, and comes at the cost of increased gene network complexity, including not only the genetic switch itself, but also the circuitry for memory storage and

readout [20]. In contrast, plasmid loss is irreversible, thus the learned state achieved by this mechanism is persistent, while the network complexity is kept at a minimum. Alternative approaches to chemical intelligence in synthetic biology, which are beyond the scope of the present study, are based on protein phosphorylation [18, 21, 22, 23] and DNA strand displacement reactions [24, 25, 26].

Elimination of plasmids from cells (typically referred to as plasmid curing) is a standard operation in genome engineering [27]. However, in order that it can be used to implement a required learning strategy for a gene classifier, the process of plasmid curing must be controlled by the synthetic gene network. Methods of synthetically controlled plasmid curing suitable for this task were developed in recent studies [28, 29], which both use synthetic control of plasmid replication by interfering into the plasmid intrinsic copy number control (CNC) mechanism [30], while [28] additionally uses selective *in-vivo* plasmid digestion by the I-SceI meganuclease [31, 32], which shortens the time scale of plasmid elimination from around 24 hours to minutes.

Importantly, the selection-based learning approach requires that the plasmids constituting the classifier gene circuit must have orthogonal (independent) mechanisms of synthetically controlled selective elimination (curing). The selectivity of *in-vivo* plasmid digestion by a meganuclease is ensured by the presence of a specific base pair sequence (referred to as the recognition sequence) in the target plasmid DNA [28, 31, 32], so that selective curing of different plasmids can be achieved by using different meganucleases along with their specific recognition sequences in the plasmids. In turn, implementing selective plasmid curing by means of plasmid CNC mechanisms requires that these mechanisms are different and independent among the plasmids; taking into account that a number of different CNC mechanisms have been actually used for synthetic control of plasmid replication [33, 34, 35, 36, 37], these may also be considered as candidates for implementing classifier learning by plasmid selection.

Based on the results cited above, we assume that synthetically controlled selective plasmid elimination, which is key in implementing the proposed learn-

ing mechanism, is achievable by currently available means of synthetic biology. We do not make any specific assumptions about the implementation details; for definiteness, we refer to it hereafter as a controlled CNC mechanism.

The paper is organized as follows. In Sec. 2 we introduce basic notations and models. In Sec. 3 we consider a simple classifier model described by a Boolean expression, formulate a classifier learning problem and provide a learning algorithm, which is shown to correctly train the classifier, given a proper training set. In Sec. 4 we describe a synthetic gene circuit implementing this classifier including the learning capability; for this circuit we construct models in terms of Boolean and continuous variables and analyze the correspondence of the gene circuit with the idealized Boolean model from Sec. 3. In Discussion we consider limitations of our problem statement and of our model, from which we derive some open problems for future studies.

2. Notations and basic classifier models

The classification problem is among the central in machine learning and consists in choosing from a predefined finite set of decisions, given an input value from a predefined set of inputs (further referred to as the *input space*) [38]. Typically, inputs are real vectors whose components are observable quantities; we consider here only binary classification, where the output is chosen from two options, which will be denoted as logical (Boolean) zero and one (i.e. "false" and "true"), or equivalently as "negative" and "positive".

Hereafter, Boolean variables will be distinguished by a "hat" symbol over the variable name (e.g. $\widehat{Out}$), and Boolean values denoted as $\hat{0}$ and $\hat{1}$. Moreover, the same variable name used with and without the hat symbol will denote a Boolean variable and a continuous quantity representing this Boolean variable (equivalently, a continuous variable given a Boolean interpretation); in such cases the correspondence between Boolean and continuous values will be specified (see Eqs. (5), (7) and (10) below).

A binary classifier with d real inputs is described by a function

$$\widehat{Out} = f(\mathbf{x}; \mathbf{p}), \tag{1}$$

where the input $\mathbf{x} \in \mathbb{R}^d$ is a d-dimensional real vector, $\mathbf{p} \in P$ is a constant vector of parameters (the parameter space P is arbitrary), and the output $\widehat{Out} \in \{\hat{0}, \hat{1}\}$ is a Boolean variable. The function $f(\mathbf{x})$ is referred to as the *classification rule* and defines a splitting of the input space into regions corresponding to different values of the classifier output.

Hypothesis testing is an important application of classification in decision making, where an observable quantity (or a vector thereof) is assumed to depend upon an underlying, directly unobservable discrete variable representing some choice (often dichotomous, i.e. taking on either of two values), which is to be guessed based upon the observation. Such are the problems of signal detection and pattern recognition (e.g. in classifying electroencephalogram signals [39]), as well as recognition of medical pathologies (e.g. determining whether a cell is cancerous or not [6, 40]). Such problems often do not have a deterministic solution (the hidden variable can not be determined unambiguously based on the observable); in such cases a classifier is characterized by some inevitable error rate (i.e. the proportion of incorrect answers among all trials).

Classifier learning problem consists in fitting the classifier parameters (here, vector $\mathbf{p}$ in Eq. (1)) in order to maximize the performance of the classifier in a specified sense (e.g. minimizing the error rate), given a set $\{\pi_i\}$ of training examples, i.e. input-output pairs

$$\pi_i = \left(\mathbf{x}^{(i)}, \widehat{Out}^{(i)}\right), \tag{2}$$

which are apriori assumed to be correct.

A particular case of the general model (1) is the binary linear classifier

$$\widehat{Out} = 1(p_1 x_1 + p_2 x_2 + \cdots + p_d x_d > 1), \tag{3}$$

where $1(\cdot)$ is the Boolean-valued indicator function defined to be $\hat{1}$ whenever the condition in the parentheses holds, and $\hat{0}$ otherwise. The weights p_j, $j \in \overline{1,d}$, are constant parameters.

Replacing the indicator function $1(\cdot)$ in Eq. (3) with an arbitrary nonlinear output function $g(\cdot)$ leads to a model with a continuous output variable

$$Out = g(p_1 x_1 + p_2 x_2 + \cdots + p_d x_d). \tag{4}$$

In particular, if a classifier is implemented as a real object, e.g. a synthetic gene circuit, then its steady-state model (describing only the functional dependence between the input and the output, while omitting possible dynamical and memory effects [41]) in terms of continuous variables takes the form (4). A continuous output may be given a Boolean interpretation e.g. using a thresholding operation

$$\widehat{Out} = 1(Out > \Theta), \tag{5}$$

where Θ is the threshold value.

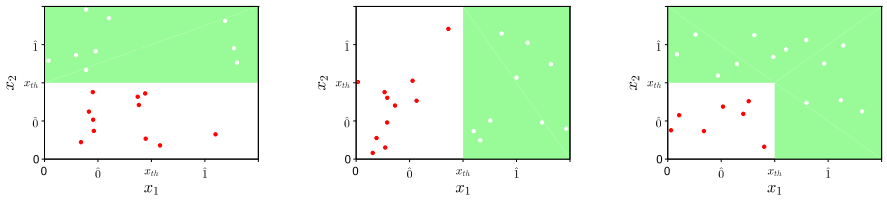
Note that a model of type (4) also describes a basic building block of a multilayer perceptron [42], a paradigmatic model in artificial inteligence, where elementary units of type (4) are arranged into layers, so that the outputs from the previous layer are fed into the inputs of the subsequent. Implementing such systems is a major challenge in synthetic biology of today [21, 22, 23, 43].

3. Boolean model of a learnable classifier

In this Section we formulate a problem statement for classifier learning, which is simple enough to allow implementing a learnable classifier within an intracellular synthetic gene network. For that sake, we limit the number of inputs to $d = 2$ and simplify the linear classifier model (3) by reducing its formulation to an entirely Boolean expression, where all variables and parameters are assumed to be Boolean, while multiplication is replaced with logical AND (conjunction), and summation with logical OR (disjunction). The classifier model then reads

$$\widehat{Out} = \hat{p}_1 \hat{x}_1 \wedge \hat{p}_2 \hat{x}_2, \tag{6}$$

where $\widehat{Out}, \hat{p}_{1,2}, \hat{x}_{1,2}$ are Boolean, multiplication stands for logical AND, and the symbol "$\wedge$" denotes logical OR.



(A) $\hat{\mathbf{p}} = (\hat{0}, \hat{1})$ (B) $\hat{\mathbf{p}} = (\hat{1}, \hat{0})$ (C) $\hat{\mathbf{p}} = (\hat{1}, \hat{1})$

Figure 1: Output of the classifier defined by Eqs. (6), (7) as a function of the continuous inputs x_1, x_2 (white area — $\widehat{Out} = \hat{0}$, pale green area — $\widehat{Out} = \hat{1}$) for the 3 variants of the Boolean parameter vector $\hat{\mathbf{p}}$, as indicated. Threshold x_{th} separates the regions corresponding to Boolean $\hat{0}$ and $\hat{1}$ along the axes. Examples of valid training sets which uniquely identify each variant are shown as scatter plots with filled circles (red — $\widehat{Out}^{(i)} = \hat{0}$, white — $\widehat{Out}^{(i)} = \hat{1}$). Notice that the training set in each panel does not match the classification rules in the other panels.

We assume that the Boolean inputs $\hat{x}_{1,2}$ in the expression (6) correspond to continuous quantities $x_{1,2}$ according to a thresholding rule

$$\hat{x}_j = 1(x_j > x_{th}), \quad j \in \{1, 2\}, \tag{7}$$

where the threshold value x_{th} is an implementation parameter; then Eqs. (6) and (7) together constitute a model of a binary classifier of type (1) operating on continuous inputs $x_{1,2}$.

The Boolean parameter vector $\hat{\mathbf{p}} = (\hat{p}_1, \hat{p}_2)$ can take on the four distinct combinations of $\hat{0}$ and $\hat{1}$ for its components $\hat{p}_1$ and $\hat{p}_2$. In the trivial case of $\hat{\mathbf{p}} = (\hat{0}, \hat{0})$, the result of Eq. (6) is always $\hat{0}$. For each of the three remaining variants, Eq. (6) produces a specific classification rule, all of which are shown schematically in Fig. 1 with values along the axes $x_{1,2}$ taken according to the convention (7) for the correspondence between Boolean and continuous inputs; these three classification rules are hereafter denoted as (A), (B) and (C), as indicated in Fig. 1.

Consider the following problem statement of classifier learning, whereby it is assumed apriori, that the correct classification rule is expressed by a formula of type (6), where the parameter vector $\hat{\mathbf{p}}$ is unknown and has to be determined

by learning (equivalently, one of the three classification rules (A), (B) or (C) has to be chosen) based on a set of training examples $\{\pi_i\}$ of type (2), where

$$\mathbf{x}^{(i)} = \left(x_1^{(i)}, x_2^{(i)}\right). \tag{8}$$

Moreover, all training examples in the set are assumed to comply to exactly one "correct" classification rule out of (A), (B) and (C), meaning that the classification problem does not admit uncertainty of the answer (this assumption corresponds to the "hard" learning problem in terms of [14]), and the training set $\{\pi_i\}$ is assumed to be sufficient to identify this correct rule out of the three options unambiguously; examples of valid training sets satisfying these requirements are shown as scatter plots in the respective panels of Fig. 1. Note that the training examples are assumed to contain values of the continuous inputs $x_{1,2}^{(i)}$ in order to allow their implementation by continuous physical quantities (see Sec. 4.2); the corresponding Boolean values $\hat{x}_{1,2}^{(i)}$ are defined according to Eq. (7).

A classifier learning algorithm for the problem specified above can be obtained by the following reasoning. The disjunction in Eq. (6) produces $\hat{1}$ in the output, whenever at least one of its terms $\hat{p}_1\hat{x}_1$ or $\hat{p}_2\hat{x}_2$ equals $\hat{1}$. Assume that a specific training example π_i provides $\widehat{Out}^{(i)} = \hat{0}$ as the correct answer. If any of the equalities $\hat{p}_1\hat{x}_1^{(i)} = \hat{1}$ or $\hat{p}_2\hat{x}_2^{(i)} = \hat{1}$ holds given the input data $(x_1^{(i)}, x_2^{(i)})$ of this training example, then the classifier output (6) equals $\hat{1}$, which does not match the correct answer from the example. This classification error is to be corrected by resetting the respective coefficient $\hat{p}_1$ or $\hat{p}_2$ to $\hat{0}$. Thus, starting from $\hat{\mathbf{p}} = (\hat{1}, \hat{1})$ and iterating through the entire training set, we find the training example(s) indicating which (or none) of $\hat{p}_1$ or $\hat{p}_2$ has to be reset to $\hat{0}$ in order that the classification rule of the trained classifier matches the training data. Note that assuming $\hat{\mathbf{p}} = (\hat{1}, \hat{1})$ at the start of the training process is equivalent to starting from the classification rule (C) (see Fig. 1); subsequent resetting of either $\hat{p}_1$ or $\hat{p}_2$ to $\hat{0}$ reduces the region of positive answer of the classifier from the variant (C) to either (A) or (B), whenever a training example is encountered that dictates such reduction.

The reasoning above is formalized by the following learning algorithm.

- **Initial data:** set of training examples $\{\pi_i\} = \{((x_1^{(i)}, x_2^{(i)}), \widehat{Out}^{(i)})\}$ complying to exactly one of the classification rules (A), (B) or (C), as shown in Fig. 1.
- Let $\hat{\mathbf{p}} = (\hat{p}_1, \hat{p}_2) = (\hat{1}, \hat{1})$.
- **For each** training example π_i **do**
  - **if** $\widehat{Out}^{(i)} = \hat{0}$ and $\exists j \in \{1, 2\} : \hat{p}_j \hat{x}_j^{(i)} = \hat{1}$, **then** let $\hat{p}_j = \hat{0}$.
- **Result:** trained classifier defined by Eq. (6) implementing the classification rule (A), (B) or (C) that matches the training data.

Consider the execution of the algorithm separately for each of the three variants of the correct classification rule. If the training set follows the classification rule (C), then output $\widehat{Out}^{(i)} = \hat{0}$ occurs among the training examples only simultaneously with input values satisfying $\hat{x}_1^{(i)} = \hat{0}$ and $\hat{x}_2^{(i)} = \hat{0}$; thus, the condition for resetting $\hat{p}_1$ or $\hat{p}_2$ never holds during the training process, and the trained classifier remains with $\hat{\mathbf{p}} = (\hat{1}, \hat{1})$, which corresponds to the classification rule (C). In turn, if the correct classification rule is (A), then some of the training examples contain $\hat{x}_1^{(i)} = \hat{1}$ (but not $\hat{x}_2^{(i)} = \hat{1}$) alongside with $\widehat{Out}^{(i)} = \hat{0}$; once such example is encountered during training, the condition for resetting $\hat{p}_1$ is fulfilled, but the condition for resetting $\hat{p}_2$ never holds, thus the classifier remains with $\hat{\mathbf{p}} = (\hat{0}, \hat{1})$, which corresponds to the classification rule (A). Similarly, if the correct classification rule is (B), then only $\hat{p}_2$ gets reset during training, and the classifier remains with $\hat{\mathbf{p}} = (\hat{1}, \hat{0})$, which corresponds to the classification rule (B). Thus, in all three cases the classification rule which results from the training matches the training data.

4. Gene network implementation of a learnable classifier

In this Section we propose a structure of a synthetic gene network implementing the learnable Boolean classifier model from Sec. 3. A structural scheme

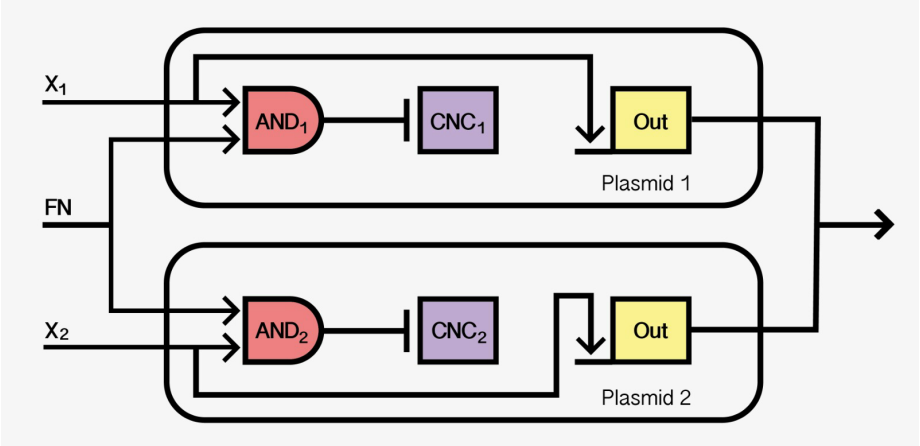


Figure 2: Scheme of the gene network implementing the learnable Boolean classifier described in Sec. 3. The network is composed of two parts located on separate plasmids. The network inputs are x_1, x_2 — the classifier inputs, and FN ("Forced Negative") — the logical negation of the correct answer used in the learning mode only. Output is encoded by the gene denoted as "Out", which is present in both plasmids under different promoters. $CNC_{1,2}$ denote the independent copy number control systems of both plasmids. The outputs of the AND gates $AND_{1,2}$ trigger the elimination (curing) of the respective plasmid from the cell, as necessary for the learning algorithm in Sec. 3.

of the network is shown in Fig. 2. The network is composed of two parts, which are located on separate plasmids with independent CNC mechanisms. We assume that the CNC system of each plasmid is regulated by an "elimination signal" from the gene network (here, from the output of the respective AND gate), so that when this signal is present, the plasmid gets eliminated ("cured") from the cell. Further we show that this controllable plasmid elimination allows to implement the learning mechanism described in Sec. 3.

The network can function both in the "learning mode", when the classifier is trained by being sequentially provided with the training examples, and in the "normal classification mode", when the classifier produces a response to input data according to a specific classification rule, which is expected to agree with the training data after learning.

We assume that the inputs and the output of the network (as well as interme-

diate signals within the circuit) are represented by intracellular concentrations of the respective proteins (transcription factors). This way, instead of describing a cell as a whole (which would require complementing the model with transmembrane signaling or sensory pathways for the inputs and a reporter protein for the output), we focus on a fragment or a building block of a larger intracellular synthetic network, whose functionality may generally reach beyond the classification task per se.

We start with describing the network in terms of Boolean variables (assuming that Boolean values $\hat{0}$ and $\hat{1}$ correspond to "low" and "high" concentrations of proteins) in order to establish its correspondence with the Boolean classifier model from Sec. 3, then we proceed to a quantitative model with concentrations represented by continuous quantities, and establish its correspondence with the Boolean formulation.

4.1. Boolean description

The combined circuit has 3 inputs: x_1, x_2 and FN ("Forced Negative"). The inputs x_1 and x_2 in the normal classification mode are used as the classifier inputs, and in the learning mode receive the input data of training examples. The input FN is always zero in the normal classification mode, and is set to Boolean $\hat{1}$ in the learning mode on condition that the correct answer in the currently presented training example π_i is negative (i.e. $\widehat{Out}^{(i)} = \hat{0}$), while the inputs $x_1 = x_1^{(i)}$ and $x_2 = x_1^{(i)}$ receive the input data from this training example.

The circuit implements the Boolean classifier (6) and its learning algorithm in the following way. The parameters $\hat{p}_j$ ($j \in \{1, 2\}$) in Eq. (6) are interpreted as indicating the presence ($\hat{p}_j = \hat{1}$) or the absence ($\hat{p}_j = \hat{0}$) of the respective plasmid. Both plasmids contain the same output gene (denoted as "Out" in the scheme) with a promoter directly activated by the input x_1 in the 1st plasmid, and by x_2 in the 2nd, so that each term $\hat{p}_j \hat{x}_j$ in Eq. (6) indicates the activation of the output gene expression on the respective plasmid. In particular, if $\hat{p}_j = \hat{0}$ (the jth plasmid is absent), then $\hat{p}_j \hat{x}_j = \hat{0}$ regardless of the input x_j (an absent plasmid produces no output); if $\hat{p}_j = \hat{1}$ (the jth plasmid is present), then $\hat{p}_j \hat{x}_j =$

$\hat{x}_j$ (the output gene of the jth plasmid is expressed, only when it is activated by the respective input x_j).

We assume that logical disjunction in Eq. (6) is implemented by the natural summation of the expression rates of the output gene in both plasmids (further in Sec. 4.2 we analyze this summation quantitatively, in terms of continuous variables). Precisely, we base the Boolean description upon the assumption that the combined network output is interpreted as Boolean $\hat{1}$, whenever the Out gene is expressed in at least one of the two plasmids. Taking into account the above interpretation of the terms $\hat{p}_j\hat{x}_j$ as the expression of the output gene in each plasmid, we conclude that the Boolean description of the circuit output under the assumptions above coincides with the expression (6).

Classifier learning is implemented as follows. Initially, both plasmids are present, so that $\hat{p}_1 = \hat{p}_2 = \hat{1}$. During learning, the training examples are presented to the classifier sequentially, by supplying the example input data to the classifier inputs ($x_1 = x_1^{(i)}$, $x_2 = x_1^{(i)}$) simultaneously with the logical negation of the correct answer supplied to the input FN ($\widehat{FN}^{(i)} = \neg\,\widehat{Out}^{(i)}$, which is defined to be $\hat{1}$ whenever $\widehat{Out}^{(i)} = \hat{0}$ and vice versa).

The CNC systems of both plasmids ($CNC_{1,2}$ in the scheme) are regulated by the plasmid elimination signals obtained from the respective genetic AND gates ($AND_{1,2}$ in the scheme); we assume that the positive output of AND_1 (AND_2) leads to eliminating the plasmid 1 (plasmid 2), whenever $\widehat{FN} = \hat{1}$ and $\hat{x}_1 = \hat{1}$ ($\hat{x}_2 = \hat{1}$) hold simultaneously. The time duration of each training example presentation must be sufficient for the plasmid elimination to take effect, as discussed below in Sec. 4.2. Thus, once an example with $\widehat{Out}^{(i)} = \hat{0}$ and $\hat{x}_1^{(i)} = \hat{1}$ ($\hat{x}_2^{(i)} = \hat{1}$) has been presented to the classifier during learning, the plasmid 1 (plasmid 2) gets eliminated, which corresponds to resetting the respective parameter $\hat{p}_1$ ($\hat{p}_2$) to $\hat{0}$. The above procedure exactly reproduces the learning algorithm described in Sec. 3.

4.2. Quantitative description

It is convenient to consider the circuit as two subsystems, which can be modeled separately: namely, the learning subsystem, which consists of the AND gates and the CNC mechanisms of both plasmids, and the output subsystem consisting of the output genes with their different promoters activated by the respective classifier inputs. In this Section we construct quantitative models of the mentioned network components and analyze the network functioning in both learning and normal classification modes. Note that the parameters $\hat{p}_{1,2}$ are inherently Boolean as indicating the presence of the respective plasmid. All continuous variables are assumed to take on non-negative real values, since they represent concentrations of proteins.

A logical AND gate for synthetic gene networks [44, 45, 46, 47] consists of the gate output gene (here, encoding the intermediate transcription factor which controls the dependent CNC mechanism) equipped with a hybrid promoter activated simultaneously by two transcription factors, which are the gate inputs (here, x_1 and FN in the plasmid 1; x_2 and FN in the plasmid 2). We adopt the AND gate model from [45, Eq. (2)], which can be obtained as follows. The stationary equilibrium concentration of the output protein can be found as the stable equilibrium state of the kinetic equations for the output mRNA and protein concentrations, see e.g. [48, Eq. (1)], where the protein concentrations are taken proportional to that of mRNA. Assuming linear decay of mRNA [48], the stationary protein concentration is then proportional to the mRNA synthesis rate, as given e.g. in [48, Eq. (3)], where the basal (leakage) gene expression in the absence of induction is neglected. The resulting AND gate model [45, Eq. (2)] is finally expressed as a product of two Hill functions [49, 50, 51], which we write down using dimensionless (normalized) variables (as in [50]) in the form

$$e_j = \frac{1}{1+x_j^{-\mu}} \frac{1}{1+f^{-\nu}} \quad \text{for} \quad x_j > 0, f > 0, \quad j \in \{1,2\}, \tag{9}$$

additionally defined to be 0, when $x_j = 0$ and/or $f = 0$. Here f is a continuous variable representing the FN input (according to Eq. (10) below), μ and ν are

the cooperativity coefficients (Hill coefficients; see e.g. [51] for interpretation), which are assumed here to be identical for the classifier inputs x_1, x_2, because the classifier model in Sec. 3 does not impose any apriori asymmetry between the inputs. The dimensionless gate outputs e_j are normalized by (equivalently, measured in the units of) the saturated stationary concentrations of the output proteins (so that $e_j \to 1$ when fully activated, i.e. when both gate inputs x_j and f receive high values); the gate inputs $x_{1,2}$ and f are normalized by the values at which the activation of the respective promoter is half-saturated (so that $e_j \to 1/2$ when $x_j = 1$ and $f \gg 1$, or when $f = 1$ and $x_j \gg 1$); for details, see reasoning behind [48, Eqs. (1),(3)] and [50, Eq. (5)].

The classifier circuit is supposed to be part of a larger gene network intended for some applied task, hence the classifier inputs $x_{1,2}$ can actually take on arbitrary values determined by the outer network, without any additional apriori assumptions. In order to establish a correspondence between the continuous model of the network and its Boolean counterpart from Sec. 4.1, we assume the correspondence rule (7), where the threshold value x_{th} is a design parameter of the overall network; the analysis below leads to recommendations for the optimal choice of this parameter.

In turn, the signal FN is used in the learning mode only, and thus can be seen as an externally controllable parameter in the sense that its values can be predetermined by the experiment design. In this view, we assume that the continuous variable f takes on exactly two predefined real values corresponding to either Boolean value of the input FN:

$$f = \begin{cases} f_0, & \text{if} \quad \widehat{FN} = \hat{0}; \\ f_1, & \text{if} \quad \widehat{FN} = \hat{1}. \end{cases} \tag{10}$$

Detailed quantitative modeling of the plasmid curing process is a complicated problem, because plasmid copy number (PCN) in an individual cell is a time-dependent stochastic quantity, which increases due to plasmid replication, and decreases due to plasmid decay (e.g. by selective meganuclease-mediated digestion [28]) and due to cell division. Normally, a plasmid CNC mechanism

stabilizes PCN so that it fluctuates around some stable level [30]; in synthetic controllable CNC mechanisms, this stable level depends on a controlling signal [33, 34, 35, 36, 37]. In particular, synthetically controlled plasmid curing mechanisms [28, 29] provide that the stable PCN level drops to zero, when plasmid curing is induced by a controlling signal. However, the time evolution of PCN in a specific cell remains stochastic until it actually becomes zero (i.e. until the plasmid is completely eliminated from the cell). It is therefore common to characterize the process of plasmid curing in a population of cells by the time dynamics of the proportion of cells containing (or devoid of) the plasmid, such as the empirical dependencies in [28, Fig. 1(C)], [36, Fig. 4b]. To the best of our knowledge, no mathematical models are currently available to describe this process in time. We note that it is possible to specify a time scale and an allowable percentage of cells harboring the target plasmid, so that after the plasmid curing system has been induced by the controlling signal for the specified or greater time, the remaining percentage of cells still harboring the plasmid is no greater than the specified value. For example, in an experimental study of selective plasmid curing by *in vivo* meganuclease digestion [28, Sec. 3.1] it was found that 20% of cells in the population retain the target plasmid after less than 1 minute of plasmid curing being induced, and less than 10% after 20 minutes. Without the meganuclease digestion mechanism, plasmid curing process becomes orders of magnitude slower, with characteristic time scales achieving the order of 10 to 24 hours [28], [36, Fig. 4b].

The percentage of cells which lose (or retain) the target plasmid is expected to depend also upon the magnitude of the signal inducing the plasmid curing system, but this dependence for the meganuclease digestion mechanism [28] has not been studied. In the context of implementing the Boolean learning rule, it is essential that this dependence is of the threshold type, so that depending on whether the controlling signal is below (above) a threshold, the vast majority of cells in the population retain (lose) the plasmid. Although the actual dependence may differ from this type (see e.g. [36, Fig. 4b], where the empirical dependence of plasmid survival upon the inducer concentration is found to be

quite smooth), thresholding can be introduced on top of the plasmid curing system e.g. by means of protein sequestration [52]. In the following, we assume that the threshold dependence as indicated above holds true, and the time duration of each training example presentation suffices to ensure that the percentage of cells harboring a target plasmid after curing is negligible.

From a perspective of intracellular classifier learning, if a cell fails to lose the target plasmid while the selective plasmid curing system was active, it means essentially that the cell fails to correctly process the particular training example in the course of learning. As soon as the present study focuses on intracellular learning (in contrast to [13, 14], where the entire cell population was considered as a single classifier), we abstract here from the problem of improperly trained cells which may remain in the ensemble due to various reasons, including the one indicated above. The consideration below pertains to the cells which behave during training as expected; this implies a threshold dependence of plasmid elimination or retaining upon the control signal, which is here the output e_j of the respective AND gate (see Fig. 2).

Summarizing the above, we define the model of the selective plasmid curing mechanism as follows: when e_j exceeds a specific threshold e_{th}, the plasmid j is eliminated; until this condition holds, the plasmid is retained.

The Boolean model of the network in Sec. 4.1 assumes that the jth plasmid must be eliminated (i.e. $\hat{p}_j$ reset to $\hat{0}$), when and only when the inputs x_j and FN are set simultaneously to Boolean $\hat{1}$. Taking into account the correspondence between the Boolean and continuous descriptions as defined by Eqs. (7) and (10), the assumption above is equivalent to the requirements that (i) while $f = f_0$ (i.e. $\widehat{FN} = \hat{0}$), both plasmids must be retained, meaning that $e_j < e_{th}$ must hold regardless of x_j:

$$e_j|_{f=f_0} = \frac{1}{1 + x_j^{-\mu}} \frac{1}{1 + f_0^{-\nu}} < e_{th} \quad \text{for any} \quad x_j \geq 0, \quad j \in \{1, 2\}; \tag{11a}$$

and (ii) when $f = f_1$ (i.e. $\widehat{FN} = \hat{1}$), the condition for the jth plasmid removal $e_j > e_{th}$ must be fulfilled if and only if the Boolean value of $\hat{x}_j$ is $\hat{1}$, i.e. if

$x_j > x_{th}$:

$$e_j|_{f=f_1} = \frac{1}{1+x_j^{-\mu}} \frac{1}{1+f_1^{-\nu}} > e_{th} \quad \text{if and only if} \quad x_j > x_{th}. \tag{11b}$$

Taking into account that the right-hand part of Eq. (9) is strictly increasing in both x_j and f, both requirements (11a,b) are fulfilled, provided the values of f_0, f_1 and e_{th} are chosen according to

$$e_{th} = \frac{1}{1+x_{th}^{-\mu}} \frac{1}{1+f_1^{-\nu}}, \tag{12a}$$

and

$$\frac{1}{1+f_0^{-\nu}} < e_{th}. \tag{12b}$$

In particular, it means that for any given x_{th} (which remains a free parameter allowing further optimization, see below), other parameters of the system can always be chosen so that the learning subsystem (i.e. conditional plasmid elimination) works exactly in accordance with the Boolean model of the circuit from Sec. 4.1, and thus implements the Boolean classifier learning algorithm from Sec. 3.

In the normal classification mode the input FN always remains at Boolean $\hat{0}$ (i.e. $f = f_0$), hence we assume that the CNC mechanisms receive "low" input from the AND gates, and thus the copy number of each plasmid is stabilized by its CNC mechanism at the "normal" level [30] determined by the network design, provided that the plasmid has not been eliminated during learning. Once eliminated, the plasmid does not re-appear. Thus, the normal classification mode admits only two options for the copy number of each plasmid — namely, either zero (if $\hat{p}_j = \hat{0}$), or the stable normal level (if $\hat{p}_j = \hat{1}$). The stationary concentration of the output protein due to the summary expression of the Out genes (on both plasmids or on a single one, depending on the learning outcome) is then expressed as a weighted sum of Hill functions

$$Out = p_1 \frac{1}{1+x_1^{-m}} + p_2 \frac{1}{1+x_2^{-m}}, \tag{13}$$

where the numerical parameter values $p_{1,2} \in \{0, 1\}$ are hereinafter assumed to be equivalent to the respective Boolean values of $\hat{p}_{1,2} \in \{\hat{0}, \hat{1}\}$, m is the

cooperativity (Hill) coefficient for the output gene promoters; see comments to Eq. (9) above. In general, m is not equal to μ from Eq. (9), because different promoters may be used, but the Hill coefficients are assumed to be identical for both inputs $x_{1,2}$ due to the same reasons as in Eq. (9). The dimensionless variable Out is normalized by the saturated stationary concentration of the output protein obtained from each single plasmid; hereby we assume that these saturated concentrations are identical for both plasmids. This assumption can be fulfilled only approximately, because the plasmids use different promoters and different CNC mechanisms, thus excluding their perfect symmetry; however, the expression rates of the output genes on different plasmids may be balanced using available techniques of synthetic gene network adjustment, e.g. by choosing proper ribosome binding sites for these genes [53, 54].

We also assume for conciseness that the normalization of the input variables $x_{1,2}$ is the same in the expressions describing the AND gates (9) and the output genes (13). This assumption is not essential; if it does not hold, then in the expressions (9)–(12a) every occurrence of $x_j^{-\mu}$ should be replaced with $a_j x_j^{-\mu}$ (likewise, $x_{th}^{-\mu}$ with $a_j x_{th}^{-\mu}$), where a_j, $j \in \{1, 2\}$, are constant scaling factors which account for the normalization difference. The plasmid elimination threshold e_{th} in Eqs. (12a,b) must be replaced with e_{thj}, which generally may differ for $j = 1, 2$; the inequality (12b) then must hold for both e_{thj}. No changes follow in the further outcomes.

In order to speak of the gene circuit as implementing the Boolean classifier model from Sec. 3, we must ensure that the circuit output given by Eq. (13) implements the classification rules of the Boolean classifier (as given by Eqs. (6), (7) and shown in Fig. 1), at least approximately, for all variants of the Boolean parameter vector $\hat{\mathbf{p}} = (\hat{p}_1, \hat{p}_2)$. We assume that the Boolean classifier output $\widehat{Out}$ is obtained from the continuous output given by Eq. (13) according to the thresholding rule (5), where the value of the threshold Θ (similarly to x_{th}) is a design parameter of the gene circuit.

The cases $\hat{\mathbf{p}} = (\hat{0}, \hat{1})$ and $\hat{\mathbf{p}} = (\hat{1}, \hat{0})$ (respectively, panels (A) and (B) in Fig. 1) are equivalent up to renumbering the inputs, thus it suffices to consider

one of them, e.g. the case (B), $\hat{\mathbf{p}} = (\hat{1}, \hat{0})$; Eq. (13) then yields

$$Out|_{\hat{\mathbf{p}}=(\hat{1},\hat{0})} = \frac{1}{1 + x_1^{-m}}. \tag{14}$$

Taking into account the conventions for Boolean interpretation of the classifier inputs and output according to Eqs. (7) and (5), and defining the threshold value for the output in Eq. (5) as

$$\Theta = \frac{1}{1 + x_{th}^{-m}}, \tag{15}$$

we obtain the positive (negative) decision of the classifier, when $x_1 > x_{th}$ ($x_1 < x_{th}$); this coincides exactly with the classification rule (B), as required.

In turn, the case $\hat{\mathbf{p}} = (\hat{1}, \hat{1})$ in Eq. (13) produces the output

$$Out|_{\hat{\mathbf{p}}=(\hat{1},\hat{1})} = \frac{1}{1 + x_1^{-m}} + \frac{1}{1 + x_2^{-m}}. \tag{16}$$

The region in the input space (x_1, x_2) corresponding to the positive answer of the classifier is expressed by the inequality $Out > \Theta$, which translates into

$$\frac{1}{1 + x_1^{-m}} + \frac{1}{1 + x_2^{-m}} > \frac{1}{1 + x_{th}^{-m}}, \tag{17}$$

where the output threshold Θ is taken according to the expression (15). The solution to the inequality (17) is required to approximate the region of positive decisions of the Boolean classifier (pale green area in panel (C) of Fig. 1); the accuracy of this approximation determines the overall accuracy of the classifier implementation.

In Fig. 3 we plot the classification boundary $Out = \Theta$ (obtained by substituting the equality sign into Eq. (17)) separating the positive and negative classifier responses in the input space (x_1, x_2) for a range of integer values of the cooperativity coefficient m from 1 to 6, for two values of the input threshold $x_{th} = 0.5$ and $x_{th} = 1$. The positive answer is above and to the right of the boundary. We find that all inputs corresponding to the positive answer of the Boolean classifier (pale green area in Fig. 3, same as in panel (C) of Fig. 1) produce also the positive answer of the gene classifier (according to Eq. (17)); in turn, all inputs producing the negative answer of the gene classifier (area

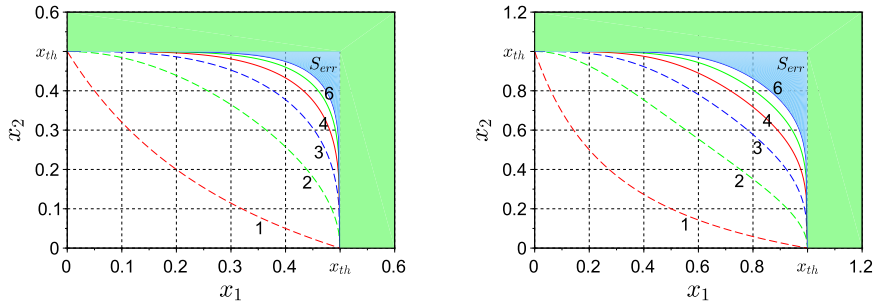


Figure 3: Lines — the boundary separating the gene classifier responses according to Eq. (17), corresponding to $\hat{\mathbf{p}} = (\hat{1}, \hat{1})$, for the integer values of the cooperativity coefficient $m = \overline{1,6}$ (indicated next to the curves; the unlabeled curve is for $m = 5$), with $x_{th} = 0.5$ (left panel) and $x_{th} = 1.0$ (right panel). Pale green area — region of the positive responses of the Boolean classifier (6), same as in panel (C) of Fig. 1, which is to be approximated by the gene circuit. Pale blue area S_{err} — region of false positive responses of the gene classifier for $m = 6$. Notice that higher cooperativity m brings the classification boundary closer to the desired boundary, and choosing the threshold value $x_{th} = 0.5$ leads to a slight improvement over $x_{th} = 1.0$.

below any classification boundary in Fig. 3) correspond to the negative answer of the Boolean classifier. At the same time, however, the gene classifier may also produce positive answers even when the Boolean classifier response is negative (shown as pale blue area for $m = 6$ in both panels of Fig. 3); considering the gene classifier as an implementation of the Boolean, these answers must be interpreted as false positive. As it follows from the consideration above, this is the only unavoidable type of classification error, while other error types vanish under the proper parameter choice; the applicability of this statement is limited by the considered model of the classifier implementation (see Discussion for other possible sources of errors in real gene circuits).

A common quality measure for classifiers is the error rate (proportion of classification errors among all trials). It, however, can not be calculated in the context of the present study, because we do not specify the apriori probability distributions for the classifier inputs neither answers. In order to characterize the performance of the gene classifier without requiring any apriori knowledge

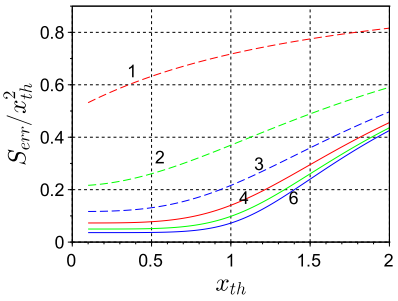


Figure 4: Classification error measure S_{err}/x_{th}^2 (characterizing the accuracy to which the gene classifier approximates the Boolean model (6)) as a function of the threshold x_{th} for the integer values of the cooperativity coefficient $m = \overline{1,6}$ (indicated next to the curves; the unlabeled curve is for $m = 5$). Notice that higher cooperativity m and lower threshold x_{th} improve the accuracy.

about the classification problem other than stated in Sec. 3, we consider an indicative accuracy measure for the approximation of the reference (Boolean) classification rule by the gene circuit. We define this measure as the ratio of the area corresponding to false positive answers of the gene classifier (which we denote as S_{err}, see Fig. 3) to the total area x_{th}^2 of negative answers of the Boolean classifier (white region in panel (C) of Fig. 1). We plot the ratio S_{err}/x_{th}^2 as a function of the threshold x_{th} for cooperativity coefficient values $m = \overline{1,6}$ in Fig. 4.

We observe that higher cooperativity of the output gene promoters improves the approximation accuracy, which is further improved by lowering the threshold x_{th}. For example, accuracy $S_{err}/x_{th}^2 \lesssim 0.2$ is achieved for $m \geq 3$ along with $x_{th} \leq 1$, while $S_{err}/x_{th}^2 < 0.1$ requires $m \geq 5$ if $x_{th} = 1$, or $m \geq 4$ if $x_{th} = 0.5$. Further lowering the threshold below $x_{th} \approx 0.5$ leads only to a minor accuracy improvement. Hence, our indicative recommendation is to use highly cooperative promoters (with m ranging from 3 to 5) for classifier outputs, along with threshold values x_{th} in the range $[0.5, 1]$; recall that the normalization of the variables x_j implies that $x_{th} = 1$ corresponds to half-activation of the promoters. Values of the Hill coefficient in natural promoters with multiple

DNA-binding sites typically range from 2 to 5, as indicated in [52] (see references therein). Moreover, a method of engineered modification of transcription factors and promoters with low natural cooperativity was suggested in [55], which was shown to increase the Hill coefficient e.g. from 0.72 to 4.40. Preference for higher Hill coefficients is a common requirement when digital logic is implemented in gene circuits due to a steeper shape of the promoter response curve [56]; lowering the threshold x_{th} shifts the working range of the transcription factor concentrations towards the steeper part of the curve.

5. Discussion

In this study we propose an approach to implementing a learning strategy in an intracellular synthetic classifier. A similar "hard" strategy consisting in permanently removing parts of the network responsible for incorrect answers was proposed in [14] for multicellular "distributed" classifiers, which learn by selection of cells. The network design proposed in the present study follows the same basic principle and adapts it to the intracellular level, where the object of selection are plasmids, and selection is achieved by synthetically controlled plasmid elimination (curing).

We considered a simple, proof-of-concept classifier learning problem, which consists in choosing the correct classification rule out of the three available options, given a set of training examples that are assumed to determine the correct choice unambigously. We show that this learning task can be solved by an intracellular classifier consisting of only two plasmids, hence we expect that this problem is the simplest to address in a possible experimental implementation.

At the same time, the hard learning strategy per se is applicable to more complicated learning problems; for instance, a variant of this strategy was used in [14] to approximate a convex classification border in the two-dimensional input space. Increasing the complexity of learning problems within reach of synthetic intracellular classifiers based on this strategy is a possible direction of future studies, which would inevitably require increasing the number of plasmids

involved, together with the number of independently controlled selective plasmid curing subsystems.

Our model is agnostic of the particular mechanism of selective plasmid curing, but in our opinion the meganuclease-mediated selective *in vivo* digestion method [28] presents a very promising approach to this end. The method is fast, providing elimination of the target plasmid from 80% of cells in the population within 1 minute [28]. The target plasmid is destroyed due to a double-strand break induced in the plasmid DNA by a meganuclease protein, which targets a specific DNA recognition sequence of 14 to 22 base pairs in length [31, 32], thus providing high selectivity. The recognition sequence has to be synthetically introduced in the target plasmid. Note that the gene encoding a meganuclease can not be located on the plasmid that is cured by this meganuclease (in other words, a plasmid can not produce a meganuclease for its own elimination) [28]; this requirement can be easily fulfilled in a multi-plasmid system, where the plasmids are assigned to eliminate one another. Independently controlled selective curing can be organized for a number of plasmids using the respective number of meganucleases with different recognition sequences. A list of available meganucleases along with their recognition sequences can be found e.g. in [57, Fig. 2a] and contains as many as 11 entries; this number leaves enough headroom for increasing the number of selectively cured plasmids and currently is not the main factor to limit the classifier complexity.

Our model essentially assumes mutual independence of the CNC systems, which in reality can be fulfilled only approximately. Minimizing the dependence among a number of different CNC systems co-existing in a single cell is a topical problem in synthetic biology of today. For instance, pairs and triplets of CNC systems were studied for independence in [58], and the triplet with best mutual compatibility (i.e. the least mutual dependence) was identified. Based on these results, we expect that the currently attainable number of plasmids in a multi-plasmid classifier amounts to 3. Further progress in increasing this number depends upon advancements in studies of plasmid CNC systems interaction, and also on the overall progress in increasing complexity of synthetic gene networks.

The main limitations of the treatment in the present study are due to using steady-state deterministic models for all network components, thereby leaving the dynamical and noise effects out of the scope of the study. Our results apply to the stable state of the system, which takes time to establish after any change in the inputs. In particular, the time scale of the plasmid curing system (of the order of minutes in case of meganuclease digestion) is essential in determining the minimal duration of training example presentation during learning, as discussed in Sec. 4.2. The time scale of signal propagation through a gene network, which depends upon the degradation rates of the transcription factors, also typically in the range of a few to tens of minutes [59], determines the classifier response time.

Stochasticity in the system arises from transcriptional noise in the gene network due to the molecular discreteness of matter [60], and as well from PCN fluctuations (see Sec. 4.2). Although the interplay of dynamics and noise in synthetic classifiers is a subject of great interest and may manifest in a nontrivial way [41, 61, 62], incorporating these effects into our model of classifier learning is currently impeded by the lack of an accepted model of PCN dynamics in multi-plasmid systems. While a stochastic description of PCN dynamics is available for a single plasmid (i.e. for a single CNC system) [63], only a conceptual deterministic model of the Lotka-Volterra type has been suggested so far for multiple, possibly interacting CNC systems [58]. This model was shown to reproduce certain qualitative patterns of multi-plasmid copy number dynamics [58], but accurate fitting of the model parameters to describe quantitatively a particular combination of plasmids remains a major challenge.

Despite leaving out stochasticity and dynamics effects from the model, we expect that learning by removal of classifier fragments (here, plasmids), while limited in universality due to its discrete nature, has a distinctive competitive advantage through its inherently greater robustness to noise and interference, as compared to available alternative intracellular synthetic learning mechanisms based on quantitative changes in network variables. The latter mechanisms suffer from susceptibility to deterioration of the memory state due to dynamics

and noise; in particular, the ratio of copy numbers of two plasmids sharing a common CNC mechanism is unstable against PCN fluctuations in the individual plasmids [16]; concentrations of transcription factors decay on certain time scales even without noise [18]; in turn, genetic switches, although having two stable states, are still subject to stochastic transitions due to noise [20]. In view of the mentioned deterioration effects, these mechanisms are considered in the literature mainly in the context of associative learning, which is supposed to persist on limited time scales. In contrast, a multi-plasmid classifier trained by selective plasmid curing with independent CNC systems, as we propose here, is highly persistent due to the stability of plasmid copy numbers. Namely, if a plasmid survives the training procedure, then its copy number, although subject to inevitable fluctuations, is stabilized by the CNC system around a normal level (the attainable relative precision of PCN stabilization in three-plasmid configurations amounts to about 10%, as found in [64, Fig. 4] for specific plasmids in *Escherichia coli*); in turn, if a plasmid gets eliminated (cured), then it has no way to re-appear in a cell except by being acquired from another cell (or from the environment). This may generally occur through the three mechanisms of horizontal gene transfer: conjugation, transduction, and natural transformation [65]. Plasmid transmission by bacterial conjugation requires the presence of a dedicated DNA sequence (so-called origin of transmission) in the plasmid; although this mechanism has been considered in the literature as a means for multicellular computing [66] and plasmid selection [67], in the context of the present study it is unwanted and easily avoided due to the absence of the origin of transmission in the plasmids. The transduction mechanism is mediated by phages, which can be excluded in an experiment. In turn, natural transformation means acquiring a plasmid from the environment, and the frequency of such events is negligibly low, estimated as ranging under different conditions in *E. coli* from 10^{-10} to 10^{-7} in 1 day [68], and in *Bacillus subtilis* (alone and in co-cultures with *E.coli*, in an experiment aimed at achieving efficient plasmid transfer) from 10^{-6} to 10^{-3} in 6 hours [69].

We anticipate that learnable artificial intracellular systems open up per-

spectives for applications in biotechnology and medicine, such as in creating intelligent biosensors and targeted drug delivery, where intracellular classifiers will control the synthesis of a drug or a signaling molecule depending on the local conditions in an organism or in a bioreactor.

Acknowledgements

We thank Arthiga Vadivelu for discussions and usefull comments.

Funding

This work was funded by Russian Science Foundation grant No. 22-12-00216. AZ acknowledges Medical Research Council grant (MR/R02524X/1).

References

[1] J. D. Watson, F. H. Crick, Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid, Nature 171 (4356) (1953) 737–738.

[2] F. Jacob, J. Monod, Genetic regulatory mechanisms in the synthesis of proteins, Journal of molecular biology 3 (3) (1961) 318–356.

[3] A. S. Khalil, J. J. Collins, Synthetic biology: applications come of age, Nature Reviews Genetics 11 (5) (2010) 367–379.

[4] S. A. Benner, A. M. Sismour, Synthetic biology, Nature reviews genetics 6 (7) (2005) 533–543.

[5] Y. Borg, E. Ullner, A. Alagha, A. Alsaedi, D. Nesbeth, A. Zaikin, Complex and unexpected dynamics in simple genetic regulatory networks, International Journal of Modern Physics B 28 (14) (2014) 1430006.

[6] L. Nissim, R. H. Bar-Ziv, A tunable dual-promoter integrator for targeting of cancer cells, Molecular systems biology 6 (1) (2010) 444.

[7] H. Ye, M. Fussenegger, Synthetic therapeutic gene circuits in mammalian cells, FEBS letters 588 (15) (2014) 2537–2544.

[8] D. N. Nesbeth, A. Zaikin, Y. Saka, M. C. Romano, C. V. Giuraniuc, O. Kanakov, T. Laptyeva, Synthetic biology routes to bio-artificial intelligence, Essays in biochemistry 60 (4) (2016) 381–391.

[9] E. Andrianantoandro, S. Basu, D. K. Karig, R. Weiss, Synthetic biology: new engineering rules for an emerging discipline, Molecular systems biology 2 (1) (2006) 2006–0028.

[10] H. H. McAdams, L. Shapiro, Circuit simulation of genetic networks, Science 269 (5224) (1995) 650–656.

[11] J. Hasty, D. McMillen, J. J. Collins, Engineered gene circuits, Nature 420 (6912) (2002) 224–230.

[12] M. B. Elowitz, S. Leibler, A synthetic oscillatory network of transcriptional regulators, Nature 403 (6767) (2000) 335–338.

[13] A. Didovyk, O. I. Kanakov, M. V. Ivanchenko, J. Hasty, R. Huerta, L. Tsimring, Distributed classifier based on genetically engineered bacterial cell cultures, ACS synthetic biology 4 (1) (2015) 72–82.

[14] O. Kanakov, R. Kotelnikov, A. Alsaedi, L. Tsimring, R. Huerta, A. Zaikin, M. Ivanchenko, Multi-input distributed classifiers for synthetic genetic circuits, PLoS One 10 (5) (2015) e0125144.

[15] Y. Schaerli, M. Isalan, Building synthetic gene circuits from combinatorial libraries: screening and selection strategies, Molecular BioSystems 9 (7) (2013) 1559. doi:10.1039/c2mb25483b.

[16] W. Tang, D. R. Liu, Rewritable multi-event analog recording in bacterial and mammalian cells, Science 360 (6385) (2018) eaap8992. arXiv:https://www.science.org/doi/pdf/10.1126/science.aap8992,

doi:10.1126/science.aap8992.
URL https://www.science.org/doi/abs/10.1126/science.aap8992

[17] S. Prakash, A. Racovita, C. Varela, M. Walsh, R. Galizi, M. Isalan, A. Jaramillo, Engineered sensor bacteria evolve master-level gameplay through accelerated adaptation, bioRxivHttps://www.biorxiv.org/content/early/2023/07/11/2022.04.22.489191.full.pdf. doi:10.1101/2022.04.22.489191.

[18] C. T. Fernando, A. M. Liekens, L. E. Bingle, C. Beck, T. Lenser, D. J. Stekel, J. E. Rowe, Molecular circuits for associative learning in single-celled organisms, Journal of the Royal Society Interface 6 (34) (2009) 463–469.

[19] Z. Li, A. Fattah, P. Timashev, A. Zaikin, An account of models of molecular circuits for associative learning with reinforcement effect and forced dissociation, Sensors 22 (15). doi:10.3390/s22155907.
URL https://www.mdpi.com/1424-8220/22/15/5907

[20] M. Sorek, N. Q. Balaban, Y. Loewenstein, Stochasticity, bistability and the wisdom of crowds: A model for associative learning in genetic regulatory networks, PLOS Computational Biology 9 (8) (2013) 1–15. doi:10.1371/journal.pcbi.1003179.
URL https://doi.org/10.1371/journal.pcbi.1003179

[21] C. C. Samaniego, A. Moorman, G. Giordano, E. Franco, Signaling-based neural networks for cellular computation, in: 2021 American Control Conference (ACC), 2021, pp. 1883–1890. doi:10.23919/ACC50511.2021.9482800.

[22] P. L. Gentili, P. Stano, Chemical neural networks inside synthetic cells? A proposal for their realization and modeling, Frontiers in Bioengineering and Biotechnology 10. doi:10.3389/fbioe.2022.927110.
URL https://www.frontiersin.org/articles/10.3389/fbioe.2022.927110

[23] P. Stano, P. L. Gentili, G. Rampioni, A. Roli, L. Damiano, En route for implanting a minimal chemical perceptron into artificial cells, in: ALIFE 2022: The 2022 Conference on Artificial Life, MIT Press, 2022.

[24] L. Qian, E. Winfree, J. Bruck, Neural network computation with DNA strand displacement cascades, Nature 475 (7356) (2011) 368–372.

[25] K. M. Cherry, L. Qian, Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks, Nature 559 (7714) (2018) 370–376.

[26] X. Xiong, T. Zhu, Y. Zhu, M. Cao, J. Xiao, L. Li, F. Wang, C. Fan, H. Pei, Molecular convolutional neural networks with DNA regulatory circuits, Nature Machine Intelligence 4 (7) (2022) 625–635.

[27] J. Trevors, Plasmid curing in bacteria, FEMS microbiology reviews 1 (3-4) (1986) 149–157.

[28] D. C. Volke, L. Friis, N. T. Wirth, J. Turlin, P. I. Nikel, Synthetic control of plasmid replication enables target- and self-curing of vectors and expedites genome engineering of *Pseudomonas putida*, Metabolic engineering communications 10 (2020) e00126.

[29] A. Riedl, S. Gruber, Z. Ruzsics, Novel conditional plasmids regulated by chemical switches provide versatile tools for genetic engineering in *Escherichia coli*, Plasmid 111 (2020) 102531.

[30] K. Friehs, Plasmid copy number and plasmid stability, in: T. Scheper (Ed.), New Trends and Developments in Biochemical Engineering, Vol. 86, Springer, Berlin, Heidelberg, 2004, pp. 47–82. doi:10.1007/b12440.
URL `https://doi.org/10.1007/b12440`

[31] B. S. Chevalier, B. L. Stoddard, Homing endonucleases: structural and functional insight into the catalysts of intron/intein mobility, Nucleic acids research 29 (18) (2001) 3757–3774.

[32] M. Hafez, G. Hausner, Homing endonucleases: DNA scissors on a mission, Genome 55 (8) (2012) 553–569.

[33] M. Sektas, W. Szybalski, Novel single-copy pETcoco vector with dual controls for amplification and expression, InNovations 14 (2002) 10.

[34] R. P. Shetty, D. Endy, T. F. Knight, Engineering BioBrick vectors from BioBrick parts, Journal of biological engineering 2 (1) (2008) 1–12.

[35] M. V. Rouches, Y. Xu, L. B. G. Cortes, G. Lambert, A plasmid system with tunable copy number, Nature communications 13 (1) (2022) 3908.

[36] S. H.-N. Joshi, C. Yong, A. Gyorgy, Inducible plasmid copy number control for synthetic biology in commonly used *E. coli* strains, Nature communications 13 (1) (2022) 6691.

[37] C. Li, Y. Zou, T. Jiang, J. Zhang, Y. Yan, Harnessing plasmid replication mechanism to enable dynamic control of gene copy in bacteria, Metabolic engineering 70 (2022) 67–78.

[38] E. Alpaydin, Introduction to machine learning, 4th Edition, MIT press, 2020.

[39] M. Degirmenci, Y. K. Yuce, M. Perc, Y. Isler, Statistically significant features improve binary and multiple motor imagery task predictions from EEGs, Frontiers in Human Neuroscience 17. doi:10.3389/fnhum.2023.1223307.

[40] Z. Xie, L. Wroblewska, L. Prochazka, R. Weiss, Y. Benenson, Multi-input RNAi-based logic circuit for identification of specific cancer cells, Science 333 (6047) (2011) 1307–1311.

[41] S. Filicheva, A. Zaikin, O. Kanakov, Dynamical decision making in a genetic perceptron, Physica D: Nonlinear Phenomena 318 (2016) 112–115.

[42] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.

[43] M. Braccini, E. Collinson, A. Roli, H. Fellermann, P. Stano, Recurrent neural networks in synthetic cells: a route to autonomous molecular agents?, Frontiers in Bioengineering and Biotechnology 11. doi:10.3389/fbioe.2023.1210334.
URL https://www.frontiersin.org/articles/10.3389/fbioe.2023.1210334

[44] S. Ayukawa, A. Kobayashi, Y. Nakashima, H. Takagi, S. Hamada, M. Uchiyama, K. Yugi, S. Murata, Y. Sakakibara, M. Hagiya, et al., Construction of a genetic AND gate under a new standard for assembly of genetic parts, BMC genomics 11 (Suppl. 4) (2010) S16.

[45] B. Wang, R. I. Kitney, N. Joly, M. Buck, Engineering modular and orthogonal genetic logic gates for robust digital-like synthetic biology, Nature communications 2 (2011) 508.

[46] T. S. Moon, C. Lou, A. Tamsir, B. C. Stanton, C. A. Voigt, Genetic programs constructed from layered logic gates in single cells, Nature 491 (2012) 249–253.

[47] D. L. Shis, M. R. Bennett, Library of synthetic transcriptional AND gates built with split T7 RNA polymerase mutants, Proceedings of the National Academy of Sciences 110 (13) (2013) 5028–5033.

[48] P. Mendes, W. Sha, K. Ye, Artificial gene networks for objective comparison of analysis algorithms, Bioinformatics 19 (Suppl. 2) (2003) ii122–ii129.

[49] A. V. Hill, The possible effects of the aggregation of the molecules of hemoglobin on its dissociation curves, J. Physiol. 40 (1910) iv–vii.

[50] J.-H. S. Hofmeyr, A. Cornish-Bowden, The reversible Hill equation: how to incorporate cooperative enzymes into metabolic models, Bioinformatics 13 (4) (1997) 377–385.

[51] J. N. Weiss, The Hill equation revisited: uses and misuses, The FASEB Journal 11 (1997) 835–841.

[52] N. E. Buchler, F. R. Cross, Protein sequestration generates a flexible ultrasensitive response in a genetic network, Molecular Systems Biology 5 (1) (2009) 272. arXiv:https://www.embopress.org/doi/pdf/10.1038/msb.2009.30, doi:https://doi.org/10.1038/msb.2009.30.
URL https://www.embopress.org/doi/abs/10.1038/msb.2009.30

[53] H. M. Salis, E. A. Mirsky, C. A. Voigt, Automated design of synthetic ribosome binding sites to control protein expression, Nature biotechnology 27 (10) (2009) 946–950.

[54] G. Kudla, A. W. Murray, D. Tollervey, J. B. Plotkin, Coding-sequence determinants of gene expression in *Escherichia coli*, science 324 (5924) (2009) 255–258.

[55] J. Hou, W. Zeng, Y. Zong, Z. Chen, C. Miao, B. Wang, C. Lou, Engineering the ultrasensitive transcription factors by fusing a modular oligomerization domain, ACS Synthetic Biology 7 (5) (2018) 1188–1194. arXiv:https://doi.org/10.1021/acssynbio.7b00414, doi:10.1021/acssynbio.7b00414.
URL https://doi.org/10.1021/acssynbio.7b00414

[56] R. W. Bradley, M. Buck, B. Wang, Recognizing and engineering digital-like logic gates and switches in gene regulatory networks, Current Opinion in Microbiology 33 (2016) 74–82. doi:10.1016/j.mib.2016.07.004.
URL https://www.sciencedirect.com/science/article/pii/S1369527416300923

[57] S. Suzuki, K.-i. Ohta, Y. Nakajima, H. Shigeto, H. Abe, A. Kawai, R. Miura, Y. Kazuki, M. Oshimura, T. Miki, Meganuclease-based artificial transcription factors, ACS Synthetic Biology 9 (10) (2020) 2679–2691.

[58] S. Chaillou, P.-E. Stamou, L. L. Torres, A. B. Riesco, W. Hazelton, V. B. Pinheiro, Directed evolution of colE1 plasmid replication compatibility: a fast tractable tunable model for investigating biological orthogonality, Nucleic Acids Research 50 (16)

(2022) 9568–9579. arXiv:https://academic.oup.com/nar/article-pdf/50/16/9568/45735536/gkac682_supplemental_file.pdf, doi:10.1093/nar/gkac682.
URL https://doi.org/10.1093/nar/gkac682

[59] Y.-Y. Cheng, A. J. Hirning, K. Josić, M. R. Bennett, The timing of transcriptional regulation in synthetic gene circuits, ACS Synthetic Biology 6 (11) (2017) 1996–2002. doi:10.1021/acssynbio.7b00118.

[60] H. H. McAdams, A. Arkin, Its a noisy business! genetic regulation at the nanomolar scale, Trends in Genetics 15 (2) (1999) 65–69. doi:https://doi.org/10.1016/S0168-9525(98)01659-X.
URL https://www.sciencedirect.com/science/article/pii/S016895259801659X

[61] N. R. Nené, J. Garca-Ojalvo, A. Zaikin, Speed-dependent cellular decision making in nonequilibrium genetic circuits, PLOS ONE 7 (3) (2012) 1–7. doi:10.1371/journal.pone.0032779.
URL https://doi.org/10.1371/journal.pone.0032779

[62] R. Bates, O. Blyuss, A. Alsaedi, A. Zaikin, Effect of noise in intelligent cellular decision making, PLOS ONE 10 (2015) 1–16. doi:10.1371/journal.pone.0125079.
URL https://doi.org/10.1371/journal.pone.0125079

[63] E. Seneta, S. Tavar?, Some stochastic models for plasmid copy number, Theoretical Population Biology 23 (2) (1983) 241–256. doi:https://doi.org/10.1016/0040-5809(83)90016-3.
URL https://www.sciencedirect.com/science/article/pii/0040580983900163

[64] T. S. Lee, R. A. Krupa, F. Zhang, M. Hajimorad, W. J. Holtz, N. Prasad, S. K. Lee, J. D. Keasling, BglBrick vectors and datasheets: a synthetic biology platform for gene expression, Journal of biological engineering 5 (2011) 1–14.

[65] C. Smillie, M. P. Garcillán-Barcia, M. V. Francia, E. P. C. Rocha, F. de la Cruz, Mobility of plasmids, Microbiology and Molecular Biology Reviews 74 (3) (2010) 434–452.

[66] A. Goñi Moreno, M. Amos, F. de la Cruz, Multicellular computing using conjugation for wiring, PLOS ONE 8 (6) (2013) 1–11. doi:10.1371/journal.pone.0065986.
URL https://doi.org/10.1371/journal.pone.0065986

[67] D. Beneš, P. Sosík, A. Rodríguez-Patón, An autonomous in vivo dual selection protocol for boolean genetic circuits, Artificial life 21 (2) (2015) 247–260.

[68] F. Riva, V. Riva, E. M. Eckert, N. Colinas, A. Di Cesare, S. Borin, F. Mapelli, E. Crotti, An environmental *Escherichia coli* strain is naturally competent to acquire exogenous DNA, Frontiers in Microbiology 11. doi:10.3389/fmicb.2020.574301.
URL https://www.frontiersin.org/articles/10.3389/fmicb.2020.574301

[69] Y.-Y. Cheng, Z. Zhou, J. M. Papadopoulos, J. D. Zuke, T. G. Falbel, K. Anantharaman, B. M. Burton, O. S. Venturelli, Efficient plasmid transfer via natural competence in a microbial co-culture, Molecular Systems Biology 19 (3) (2023) e11406.