Compilation and Inference with Chemical Reaction Networks

Thesis by
William Poole

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy in Computation and Neural Systems

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2022
Defended August 24, 2021

William Poole
ORCID: 0000-0002-2958-6776

ACKNOWLEDGEMENTS

This thesis would not have been possible without the help of a great many people. First and foremost, my family. My parents Will and Janet, and my sister Sarah have been incredibly supportive through the years and challenges of a PhD. A few years ago, I met my wonderful wife, Aabha, who has since been a center point of my life. She has humored my complaints and (frequently unilateral) excitement about science, advanced math, and other academic details. She has encouraged me when I struggled and frequently reminded me that my passion and excitement for science would make this long process worthwhile.

Obviously the support of my advisors Erik Winfree and Richard Murray has been crucial. Erik has instilled in me a deep respect and appreciation for rigorous mathematics and computer science—an essential counterbalance to my tendency to rely on intuition and jump prematurely to some conclusions. Richard has taught me to think like an engineer; that the ability to design and build something is a sure road to knowledge and that in this endeavor, powerful tools can be as helpful as theorems and intuition.

One of the greatest joys of my PhD has been the collaborative sharing of knowledge, teaching, and learning from exceptional people. Along the way, many friends, colleagues, peers, and collaborators have played a huge role in helping me hone my ideas and knowledge, provided support in my projects or just kept me sane. In particular, the work in this thesis would not have been possible without Tom Ouldridge, Manoj Gopalkrishnan, Anandh Swaminathan, Andres Ortiz, Vipul Singhal, Ayush Pandey, Andrey Shur, Zoltan Tuza, and Ankita Roychoudhury.[1]

Starting in high school, my physics and biology teachers Jonathan Briggs and Adam Waltzer introduced me to these areas to such great effect that I haven't stopped studying them. In college, my thesis advisor Derek Stein, as well as informal advisors/teachers Tom Powers and Gary Wessel empowered me to dive into the intersection of physics, computation, and biology in a way I would never have been able to on my own. After college, Theo Knijnenburg and Brady Bernard taught me bio-statistics and how to manage big data which have been invaluable skills.

So to everyone who has supported me along the way, thank you.

[1] No doubt I am missing names here, so I should add a few more: Logan Cross, Matt Rosenberg, Anghad Singh, Abhishek Behera, Anish Sarma, Sam Clammons, Stefan Badelt, Miki Yun, and doubtless, I am still missing people.

ABSTRACT

The successful advancement and deployment of technologies in the field of synthetic biology will require sophisticated computational infrastructure coupled with new theoretical ideas in order to more effectively engineer and reverse engineer biochemical networks. This thesis argues that the field of machine learning can inform the development of these underlying principles and techniques. First, software for compiling diverse chemical reaction network models of biological circuits from simple specifications is described. Second, three chemical reaction network implementations of a powerful machine learning model called a Boltzmann machine are analyzed and compared. Third, the class of detailed balanced chemical reaction networks are proven to be capable of probabilistic inference and, when coupled to a driven chemical system, autonomous learning. Finally, the use of machine learning to interpret and understand biological systems is explored in an experimental case study modeling *E. coli* cell extract metabolism.

PUBLISHED CONTENT AND CONTRIBUTIONS

[1] W. Poole, A. Pandey, Z. Tuza, A. Shur, and R. M. Murray, "BioCRNpyler: Compiling Chemical Reaction Networks from Biomolecular Parts in Diverse Contexts," *BioRxiv*, 2020. DOI: https://doi.org/10.1101/2020.08.02.233478,
W.P. lead this project, designed the high level architecture of the software, and participated in all aspects of software development. W.P. also drafted the manuscript.

[2] W. Poole, A. Ortiz-Munoz, A. Behera, N. S. Jones, T. E. Ouldridge, E. Winfree, and M. Gopalkrishnan, "Chemical Boltzmann Machines," in *International Conference on DNA-Based Computers*, Springer, 2017, pp. 210–231. DOI: 10.1007/978-3-319-66799-7_14,
W.P. contributed equally with A.O.M. and A.B. as co-first authors developing the different models and theorems in this paper. W.P. also drafted the manuscript.

TABLE OF CONTENTS

PREFACE

The personal side of science is often omitted from the formal narrative presented at conferences and published in academic journals. But, at least in my experience, the late-night musings, short tangential conversations with friends and colleagues, and otherwise unpublished aspects of science make the work fun and fulfilling. After all, if my intellectual nourishment consisted solely of the papers I published, I would have long since starved myself out of academia. So I have decided to take the opportunity to indulge in some less rigorous discussion in this preface and will return to these ideas briefly at the very end of this thesis. For those of you interested in rigorous scientific content, I encourage you to read this section with an open mind, for it focuses not so much on what we know,[2] nor on what I have added to the body of scientific knowledge,[3] but rather on what we may know in the future. Speculations of future science can accurately be called *science fiction* and, needless to say, I will be flattered if this preface is read again in the distant future so I could be called either a visionary or a naive dreamer.

Before beginning to write this section, I dug up the *Statement of Purpose* I wrote on my application to the Computation and Neural Systems program at Caltech. An excerpt from the first paragraph is included below:

> ... I have been deeply fascinated by the question: what constitutes a "thinking" network capable of processing information and how do these networks operate under different contexts? I believe answering this question in mathematical terms is essential to our ability to understand the complexity found in biological circuitry and to effectively engineer biological systems. [...] With the aid of compute clusters and powerful mathematical abstractions, I believe we can learn a lot about how complex biological systems process information and use models to make testable and/or clinically relevant predictions. Crucial to this endeavor is the use of real biological data.

Considering how much I have learned over the past 6 years, I am surprised at how accurately this paragraph reflects the content of this thesis. But, I also know that

[2] In the introduction of this thesis, you will find a succinct summary of some relevant scientific knowledge which may even be mildly accessible.

[3] Chapters 2-5 of this thesis represent about two thirds of 6 years of work compressed into the scientific fact-telling format we call journal articles.

what I wrote in my application and what I was hoping to accomplish are two very different things. Then, I had the notion that math and computation could uncover *the secrets* that make life special. It is now rather apparent to me that these secrets are not in fact all that secret; they take the form of the laws of physics entangled together in a myriad of complex ways by evolution. And as always, the devil is in the evolutionary details. Biologists spend lifetimes reverse engineering specific organic programs (more frequently called living organisms) and, despite incredible advances in the past century, there exists countless more lifetimes of work before we can hope to understand the full depth of biological complexity. Indeed, Chapter 5 of this thesis focuses on my costly attempt to reverse engineer a relatively tiny and idealized piece of this puzzle: the metabolism of mushed up *E. coli*—one of the most studied organisms in history—made even simpler by being robbed of its genome and cellular membrane. Even in such a simplified biological system, all I can do is point out how little we understand, emphasize the challenges of collecting sufficient data to build quantitative mechanistic models that can make concrete predictions, and fall back to improving our heuristic and phenomenological understanding.

That said, I am confident we will make progress in both synthetic and systems biology, two closely linked endeavors corresponding to engineering and reverse engineering biological systems. In many ways, I believe this technological revolution will be driven out of necessity. Humanity is faced with daunting 21st century challenges: to provide an ever-growing population with nourishment, shelter, and healthcare, and most importantly, to do so in a way that is robust to and does not intensify climate change. Advances in biotechnology may allow us to build a sustainable society where *matter* and *energy* are as available as information has become in the age of the internet. Scientists have already begun engineering genetically modified crops to automatically adapt to droughts and floods. But we can do better—our agriculture could replenish depleted soil and contain nutrients currently only available from animal products. Today, wood is considered one of the more sustainable construction materials—but at best, it is carbon neutral with new timber replacing the old as it is destroyed by time. In the future, I foresee sustainable *living* materials that sequester carbon from the atmosphere to repair themselves. Similarly, many diseases caused by specific genetic mutations may be curable with highly efficient personalized medicine. Maybe we will even learn how to repair or prevent the damage caused by aging. These are just a few of the imaginable *science fiction* technologies I believe will be made possible by synthetic and systems biology.

I have already said that I was only mildly successful at reverse engineering a particular biological system. Nor did I build new synthetic biological parts or circuits. Instead, I attempt to provide a new lens through which we can think about how biochemical circuits function. This lens is deeply inspired by the explosion of a field called machine learning.[4] The central point however is subtle; instead of exploring "how can computers help us understand biochemical circuits," the larger take away from this thesis is exploring how *biological computation can be understood as machine learning*. Just as idealized mathematical representations of biological neurons inspired the deep neural networks being deployed by technology firms around the world, idealized mathematical models of biochemical networks can also map to powerful machine learning methods. In other words, the ideas fundamental in machine learning theory may be a natural language to use to understand biochemical computation. Chapters 3 and 4 of this thesis rigorously defend this statement in a variety of contexts.

Finally, Chapter 2 of this thesis focuses on a software tool—BioCRNpyler—partly inspired by the computational infrastructure that has accelerated the success of machine learning. BioCRNpyler allows synthetic and systems biologists to easily compile diverse and complex biochemical models from simple specifications. Just as a computer programmer can use high level programming languages such as Python instead of writing with assembly code,[5] biologists and bioengineers will need software tools to help them translate high level designs and hypotheses into specific biochemical implementations and models. Indeed, drawing the computer analogy with biology further, I would argue that *molecular programming* is still in its infancy, perhaps equivalent to the early era of vacuum-tube based computers. We are still waiting for a biochemical "transistor" and some biochemical version of Moore's law to take hold. In fact, it may be that this biochemical transistor will not consist of a single physical device which enables seemingly endless biochemical computers to be built, but rather a series of conceptual breakthroughs in design methodology that allow us to seamlessly engineer biological systems in a holistic way. An underlying premise of this thesis is that this methodological breakthrough will likely include many ideas inspired by machine learning. Realizing the potential synthetic biology offers will require new theoretical insights, sophisticated computational infrastructure, and large data sets to validate and test new principles. I certainly do

[4] Briefly, the central idea of machine learning is to automate the construction and optimization of some very specific kinds of mathematical models by computers with access to boatloads of data.

[5] Or the infamous punch cards my father likes to brag about using.

not claim to have accomplished all of these, but I hope that this work provides a step in those directions and will inspire scientists to help grow a sustainable future built on biological technology.

Chapter 1

INTRODUCTION

1.1 Chemical Reaction Networks as a Biochemical Programming Language

Chemical reaction networks (CRNs) are a unifying theoretical foundation for most of this thesis. CRNs are defined mathematically as a set of N species $\{S_i\}$ and reactions of the form:

$$\sum_i I_r^i S_i \xrightarrow{\rho_r(s)} \sum_i O_r^i S_i. \tag{1.1}$$

Here, I_r^i and O_r^i are the number of inputs and outputs of species S_i to the reaction r, and $\rho_r(s) \geq 0$ can, in general, be any function of the amount of species in the system s which goes to 0 when any input (reactant) is not present. Every CRN can be simulated mathematically to produce dynamic trajectories representing how a model of a system behaves in two mutually compatible ways. If the amount of each species is assumed to be a real valued concentration, the system is best modeled by the *chemical rate equation* (commonly called deterministic dynamics) [1]:

$$\frac{d[S_i]}{dt} = \sum_r \Delta_r^i \rho_r([S]). \tag{1.2}$$

Here, $[S] \in \mathbb{R}^N$ denotes a vector of species concentrations and $[S_i]$ the concentration of the specific species i. The index r sums over reactions in the network, $\Delta_r^i = O_r^i - I_r^i$ is a matrix describing the amount of species i produced or consumed when reaction r occurs, and $\rho_r([S])$ is the rate at which reaction r occurs. On the other hand, if each species is assumed to consist of discrete counts of molecules, the same CRN can be simulated as a Markov jump process via the *chemical master equation* (commonly called stochastic dynamics) [2]:

$$\frac{d\mathbb{P}(s,t)}{dt} = \sum_r \mathbb{P}(s - \Delta_r, t)\rho_r(s - \Delta_r) - \mathbb{P}(s,t)\rho_r(s). \tag{1.3}$$

Here $s \in \mathbb{Z}_{\geq 0}^N$ denotes an N-vector of species counts and $\mathbb{P}(s,t)$ is the probability that the system has counts s at time t. It is worth noting that stochastic dynamics can rigorously be derived from the statistical physics of a well-mixed solution (meaning diffusion is so fast that spatial interactions are negligible) [3]. Deterministic dynamics can then be seen as a limit of the stochastic dynamics in an infinite volume with infinite counts but a finite concentration [4].

CRN models have become widely used in synthetic and systems biology[1] [5, 6] because they can succinctly encode detailed molecular interactions at the mechanistic level [7] as well as coarse-grained phenomenological behavior [8, 9]. It is easiest to see the difference between *mechanistic* models and *phenomenological* models via an example. First, consider a gene G which produces a transcript T. This could be written as a single reaction:

$$G \xrightarrow{V_{tx}} G + T. \tag{1.4}$$

When the species G produces the transcript, it is not used up, so it appears on both sides of the equation. A more detailed description of this process might involve a second species, the polymerase P, which first binds to G before transcription is enabled. This could be written as:

$$G + P \underset{k_u}{\overset{k_b}{\rightleftharpoons}} G : P \qquad \text{Polymerase binds or unbinds from the gene,} \tag{1.5}$$

$$G : P \xrightarrow{k_{tx}} G + P + T \qquad \text{Polymerase transcribes the gene.} \tag{1.6}$$

In this example, the process consists of three reactions: P and G can bind together to form a complex $G : P$ (1.5 forward direction) which can also unbind (1.5 reverse direction) or be transcribed to produce T (1.6). The first model could be called *phenomenological* because it describes what might be observed in the lab—production of a transcript from a gene—without describing how that process occurs. Specifically, the constant V_{tx} denotes the transcription rate, and in general may be a complex function of the amount of polymerase P, temperature, and a whole host of other factors [10]. The second model is a more *mechanistic* model because it describes a process by which transcription occurs. However, it is worth noting that the boundary between mechanistic models and phenomenological models is largely one of perspective. For example, the second model could be called phenomenological because the growth of the transcript nucleotide by nucleotide is not modeled; once again, a complex process has been lumped into a rate constant

[1] Simply put, synthetic biology is engineering biology. Systems biology is reverse engineering.

k_{tx}. An even more detailed mechanistic model might look like:

$$G + P \underset{k_u}{\overset{k_b}{\rightleftharpoons}} G : P \qquad \text{Polymerase binds or unbinds from the gene,} \tag{1.7}$$

$$G : P + n \xrightarrow{k_p} G : P : T_0 \qquad \text{Polymerase begins transcribing,} \tag{1.8}$$

$$G : P : T_{i-1} + n \xrightarrow{k_p} G : P : T_i \qquad \text{Polymerase adds nucleotide } i \in 1...M, \tag{1.9}$$

$$G : P : T_M \xrightarrow{k_r} G + P + T \qquad \text{Transcript is released.} \tag{1.10}$$

Here, n denotes nucleotide and k_p is the rate of polymerization. This elongation model is also a simplification of an even more complex process involving conformation changes of the polymerase [11] and potentially multiple polymerases bound to each gene [12]. The flexibility of chemical reaction networks coupled with their well-tailored nature to describe biochemical phenomena has made these models very popular in systems and synthetic biology. For example, CRNs have been used to produce systems level metabolic models [13], detailed mechanistic and phenomenological models of real biochemical circuits [6, 14, 15], and detailed mechanistic and phenomenological models of synthetic biochemical circuits [16, 17].

The utility of CRNs is that they are very expressive and can represent virtually any process at multiple levels of abstraction. This statement has been made rigorous using the branch of computer science called *complexity theory* which asks "What kind of problems is a particular class of systems capable of solving?" It has been proven that finite stochastic CRNs are probabilistically Turing universal (meaning they can perform any computation a Turing machine can, and hence any computer can, with an arbitrarily small chance the output will have an error) [18]. Deterministic CRNs are somewhat less powerful than their stochastic counterparts, but they can are powerful function approximators[2] [19–21]. Subtly, some sub-classes of CRNs, meaning systems where only reactions of a very particular form are allowed to occur such as carefully engineered DNA strand displacement systems, can in fact implement any arbitrary CRN with no restrictions. Put in other words, CRNs can be viewed as a programming language where a phenomenological description (in the form of a simple CRN) is compiled into a detailed mechanistic CRN (in the form of a much larger implementation CRN) [22, 23].

[2]Technically, they can produce any piece-wise linear function, which could be used to approximate any function over a bounded domain.

The idea of biochemical compilation has helped birth a very fruitful field, *molecular programming*, which sits at the intersection of computer science, DNA nanotechnology, and synthetic biology. Molecular programmers have made numerous *in vitro* biochemical computers out of DNA [24–26], RNA [27, 28], enzymatic systems [29, 30], and small molecules [31]. Molecular compilation has also been used to a lesser extent *in vivo*, but significant challenges remain in this area including orthogonality of components [32, 33], toxicity [34, 35], and evolutionary stability [36]. Extending CRN compilation to synthetic and systems biology is a central aspect of this thesis. Towards this goal, Chapter 2 introduces a Python software package called BioCRNpyler[3] designed as a general purpose CRN compiler for synthetic and systems biologists inspired, in part, by DNA strand displacement (DSD) compilers [23, 26].

The ability to rapidly explore the design space of biochemical models naturally leads to the question: what are the most *natural* ways to program CRNs? To date, most *in vivo* implementations of CRN-based programs take the form of logic circuits [33, 37, 38] or feedback controllers [39]. However, it is unclear that these programming paradigms are easily represented by the intrinsic interactions of biomolecules and whether those representations scale to larger systems [40, 41]. One alternative is that biochemistry naturally implements algorithms inspired by machine-learning methodologies.

1.2 Machine Learning as a Programming Methodology

The methods and considerations common in machine learning have been a direct inspiration for many aspects of this thesis. Briefly, machine learning can be characterized as a set of techniques to learn parameters W of a function F given data X [42]. The basic idea is that the function F encodes some kind of computation (such as image classification) and that the parameters can be learned from a representative set of training data. Commonly, this is framed as an optimization problem:

$$W^* = \underset{W}{\operatorname{argmin}}\, \mathcal{L}(F(X, W)). \tag{1.11}$$

Here, $\mathcal{L}$ is some kind of loss function or error. In general, finding the globally optimal value of W^* is infeasible. Instead, stochastic gradient descent is used to search for a local optimum which in practice is often close to a global optimum:

$$\partial_\tau W_i \propto \frac{\partial}{\partial W_i} \mathcal{L}(F(X, W)). \tag{1.12}$$

[3] pronounced "Biocompiler".

Here W_i is a single parameter and $\partial_\tau W_i$ denotes the change in W_i at the training epoch τ. For example, in an image classification problem, F might be a feed forward neural network, X would be images, and $\mathcal{L}$ could be the squared-difference between the true classification and the classification output from F. The key to effective machine learning is in choosing the right function F for a given problem [42] and the correct optimization procedure to learn that function. However, these topics only play a tangential role in this thesis by highlighting the powerful idea of function optimization being a computational tool which can be applied to an incredibly diverse number of problems ranging from image recognition [43] to drug discovery [44] to natural language processing [45] to solving the structures of proteins from sequences [46].

This thesis is primarily focused on a particular subclass of machine learning functions called generative probabilistic models [47] and specifically probabilistic graphical models [48]. Generative probabilistic models consist of functions which produce probability distributions[4] $F : x \to \mathbb{R}$ by assigning each possible data point x a probability $\mathbb{P}(x)$. Probabilistic generative models have a deep history of being used in neuroscience and are hypothesized to underlie many aspects of how the brain functions [49, 50]. It therefore seems plausible that generative models could also play a role in chemical computation.

Probabilistic graphical models are a subclass of generative models which use a graphical approach to write their distribution in a factorized form. For example, a graphical model where y and z depend on x but not on each other would be denoted:[5]

$$\begin{array}{ccc} & X & \\ \swarrow & & \searrow \\ Y & & Z \end{array} \qquad \Rightarrow \qquad \mathbb{P}(x, y, z) = \mathbb{P}(x)\mathbb{P}(y \mid x)\mathbb{P}(z \mid x). \tag{1.13}$$

Furthermore, these distributions are commonly parameterized as exponential functions which makes these models analytically tractable. Continuing with the above example:

$$\mathbb{P}(x, y, z) = \frac{1}{Z} e^{f_x(x)} e^{f_y(y,x)} e^{f_z(z,x)} = \frac{1}{Z} e^{f(x,y,z)} \qquad Z = \sum_{x,y,z} e^{f(x,y,z)}. \tag{1.14}$$

Here, the functions f_x, f_y, and f_z may have many forms, typically chosen based upon prior knowledge of the data or process being modeled. Common choices include

[4] A distribution is a (potentially infinite) vector of values p_i normalized to $\sum_i p_i = 1$.

[5] Note that the arrows used here are different from the reaction arrows of the previous section.

Guassians, Poisson distributions, etc. [48]. Graphical models are well suited for Bayesian inference and computing conditioning distributions such as $\mathbb{P}(y \mid x)$. In practice, these models are commonly trained to learn the distribution of the training data by using the relative entropy D as a cost function:

$$\mathcal{L} = D(Q \mid\mid P) = \sum_x Q(x) \log \frac{Q(x)}{P(x)}. \tag{1.15}$$

The relative entropy is not symmetric. Depending on the specific algorithm and model used, both Q and P can be either the data distribution or the algorithm distribution [51]. When P is the algorithm distribution, the relative entropy has a natural interpretation as the amount of information lost when P is used to approximate Q which makes this cost function a natural choice from an information theoretic perspective.

1.3 Past Work Relating Machine Learning and Chemical Reaction Networks

Given the computational power of deterministic CRNs and the utility of machine learning, it comes as no surprise that a large number of schemes have been developed allowing CRNs to implement common functions (denoted F in the previous section) trained via external (presumably non-chemical) machine learning algorithms. For example, CRN implementations of neural network architectures inspired by the perceptron [52] and Hopfield associative memories [53] have been designed and analyzed at an abstract level [54–59]. Concrete implementations based on enzymatic circuits have been proposed [60–62], and built at a small scale both *in vitro* [30, 63] and *in vivo* [64, 65]. Similarly, concrete implementations based on *in vitro* DNA based circuits have been proposed [66, 67] and built at a much larger scale than their enzymatic counterparts [24, 68, 69].

The learning process has also been investigated in the context of deterministic CRNs beginning with the connection between learning and evolution [70] leading to a number of CRN networks which are evolved *in silico* in order to minimize an error function [58, 71]. Various CRN architectures have also been proposed which internally incorporate the ability to be optimized or tuned via dynamic changes in species' concentrations, such as error signals. This allows for adaptive CRN neural networks which learn via chemical implementations of gradient descent and related algorithms [72–75]. A different approach has been to understand learning from a Bayesian probabilistic inference perspective leading to deterministic CRN models which can infer the state transition matrices of a Markov chain [76] and find the maximum likelihood estimators for log-linear models [77].

Parallels between CRNs and neural networks have also been used as a lens through which to understand and model various biological processes including developmental biology [78, 79], protein-protein interaction networks [70], and combinatorial transcriptional regulation [80]. It has also been argued that bio-molecular computations at the cellular level are fundamentally analog due to effects like resource loading [81] suggesting that a neural perspective is *natural* in molecular biology.

Unlike most of the preceding references, this thesis is primarily focused on the relationship between stochastic CRNs and generative probabilistic models (as opposed to deterministic CRNs and neural networks). At the highest level, this connection seems intuitive because stochastic CRNs and generative models both produce probability distributions. In other words, the noise due to discrete molecular counts in stochastic CRN dynamics might *naturally* give rise to behavior reminiscent of generative models. A few papers make similar points by showing that stochastic CRNs can use their intrinsic noise to implement a version of the message passing algorithm [82] and to optimize probabilistic models directly [83, 84]. In this vein, Chapter 3 of this thesis provides more examples of CRNs which use stochasticity to implement a particular kind of probabilistic graphical model called a Boltzmann machine [85]. Chapter 4 uses tools from information theory and statistical physics to generalize the analogy between specific classes of CRN and graphical models. These insights are then used to develop a *fully autonomous*[6] chemical implementation of a learning (meaning parameter optimization with gradient descent).

1.4 Statistical Physics Connects Chemical Reaction Networks to Machine Learning

Stochastic CRNs are a form of stochastic process which, by the definition of their dynamics (1.3), embody dynamic probability distributions. Stochastic processes, in turn, have been extensively studied in physics resulting in many methods of numerically simulating and analyzing the properties of these systems [86]. However, little can be said about the general overarching behavior of stochastic CRNs because they are Turing universal and in principle capable of just about any kind of behavior. Fortunately, by applying a series of constraints, stochastic CRNs can become more analytically tractable. Crucially, we will apply restrictions with clear physical interpretations with the ultimate goal of arriving at a model superficially similar to probabilistic graphical models. In the process, a number of important considerations related to the physicality of CRNs will be discussed.

[6] By "fully autonomous" I mean that no external signals or interventions are required for learning.

The first step is to focus on the steady-state distribution, $\mathbb{P}_{ss}$, of a CRN. This occurs when the chemical master equation (1.3) is set to 0:

$$\frac{d\mathbb{P}(s,t)}{dt} = 0 \implies \sum_r \mathbb{P}_{ss}(s-\Delta_r)\rho_r(s-\Delta_r) - \mathbb{P}_{ss}(s)\rho_r(s) = 0. \tag{1.16}$$

Notice that there is no time dependence—simplifying the probability distribution—which also seems reasonable in light of the lack of dynamics in the definition of probabilistic graphical models. The steady state distribution can also be thought of as the infinite time-averaged behavior of the CRN. Mandating that a steady state solution exists has the added benefit of ensuring that some CRNs which result in unbounded dynamics like $A \to 2A$ are excluded from consideration.

The second step is to realize that any *irreversible* reaction must be phenomenological because physics is fundamentally reversible[7] [88]. For example, consider the elongation of a transcript by a polymerase, as represented in reaction (1.9). Although this reaction is written irreversibly, it is also possible that the nucleotide will de-polymerize from the transcript as soon as it is added. Rewritten to include this possibility, the reaction becomes:

$$G:P:T_{i-1} + n \underset{k_p^{rev}}{\overset{k_p}{\rightleftharpoons}} G:P:T_i + Ph \qquad \frac{k_p}{k_p^{rev}} = e^{\frac{-\Delta G_p}{k_B T}}. \tag{1.17}$$

Here, the additional species Ph represents the polyphoshate produced during polymerization which is important to account for in the reversible dynamics. In the second equation, k_b is Boltzmann's constant and T is temperature. This equation is the definition of microscopic reversibility: the ratio of the forwards and backwards rates are given by the exponential of the change in free energy of the process divided by k_bT. Intuitively, what this condition implies is that transitions between states are linked to thermal fluctuations. In the case of RNA polymerization, ΔG_p is on the order of -30 kJ/mol.[8] When translated to units of k_BT, the forward rate is hugely faster than the reverse rate. This means, that for all practical purposes the phenomenological description of the reaction being irreversible is correct provided the amount of fuel is abundant. Indeed, any phenomenologically irreversible reaction can be written mechanistically as one or more reversible reactions connected

[7]The reversibility of physics can be seen through classical physics such as Newton's laws as well and the electromagnetic force as well as quantum physics which all have time-reversal symmetry. Informally, this means the laws of physics still hold backwards in time as well as forwards. The only exception to this symmetry occurs in statistical mechanics where entropy always increases in time, but this is fundamentally a probabilistic effect [87].

[8]This is estimated from the standard free energy difference between ATP and AMP noting that during polymerization a polyphosphate is released [89].

to reservoirs which provide fuel species and remove waste species species [90, 91]. In the above example, the triphosphate inside the NTP acts as a fuel source driving polymerization. By explicitly analyzing the production of Ph as a waste product, it becomes clear that this reaction is very energetically favorable. Provided that the amount of NTP remains high and the amount of polyposphate waste remains low, the polymerization process (and hence transcription) is approximately irreversible despite being mechanistically microscopically reversible.

The final step is to mandate that the microscopically reversible CRN *not* be connected to any non-equilibrium reservoirs or chemostats[9]. Then, the steady state distribution will become an equilibrium distribution and the resulting CRN is called *detailed balanced* (db). Formally, the detailed balanced condition states:

$$\text{Every species } S_i \text{ has an energy } G_i. \tag{1.18}$$

$$\text{Reactions are reversible: } \sum_i I_i S_i \xrightarrow{k^+} \sum_i O_i S_i \Leftrightarrow \sum_i O_i S_i \xrightarrow{k^-} \sum_i I_i S_i. \tag{1.19}$$

$$\text{Microscopically reversible rates: } \frac{k^+}{k^-} = e^{-\Delta G} \quad \Delta G = \sum_i (O_i - I_i) G_i. \tag{1.20}$$

Intuitively, this condition means that there are no driving forces powering any reactions in the dbCRN, which ensures that a dbCRN is driven only by thermal fluctuations at steady state.

Equilibrium systems are the basis of thermodynamics and statistical mechanics, and have been studied for around two centuries. One well known result about any system at physical equilibrium is that it will have a probability distribution in the Boltzmann-Gibbs form [92]:

$$\mathbb{P}(x) = \frac{1}{Z} e^{-\frac{E(x)}{k_B T}} \qquad Z = \sum_x e^{-\frac{E(x)}{k_B T}}. \tag{1.21}$$

And indeed, it has been shown that dbCRNs have an equilibrium distribution $\pi(x)$ in the same form [93]:

$$\pi(s) = \frac{1}{Z} e^{-\frac{\mathcal{G}(s)}{k_B T}} \qquad Z = \sum_{s \in \Gamma_{s0}} e^{-\frac{\mathcal{G}(s)}{k_B T}} \qquad \mathcal{G}(s) = \sum_i G_i s_i + \log s_i!. \tag{1.22}$$

Here, s is a vector of species counts and Γ_{s0} is called the reachability class which denotes all the states that can be reached via some sequence of reactions from

[9] A chemostat is effectively a chemical battery that can hold fuel (e.g. ATP) and waste (e.g. ADP) species at constant values to power a system [90]. Note that, technically, a system could be connected to an equilibrium chemostat which is analogous to a drained battery because the equilibrium chemostat will not provide any power to the system.

initial condition s^0. The key thing to notice is that Equations (1.14) and (1.22) are suggestively similar - an analogy which will be elaborated on extensively in Chapter 4 of this thesis and ultimately used to implement a CRN capable of autonomous learning which implements gradient descent (1.12) using the relative entropy as a cost function (1.15).

1.5 Synthetic and Systems Biology: Two Sides of the Same Coin

Broadly, synthetic and systems biology provide the motivation and potential application areas for many of the ideas in this thesis. Briefly, synthetic biology is concerned with repurposing existing biochemical components to genetically engineer biological organisms for new purposes [5]. Synthetic biology has been used to prototype biochemical circuits in single cells [33, 37, 38, 94] and in multi-cellular systems [95–97] with potential applications in bio-manufacturing [98, 99], biosensors[100, 101], and medicine [102–104]. Synthetic biology also emphasizes building mechanistic models of biological parts and components so they can be applied in a modular fashion similar to the parts of a circuit board [105]. However, despite the emphasis on modularity, the context-dependent effects of different biochemical parts both between chassis and due to unintended cross-talk remain major hurdles in scaling synthetic biological circuits [106]. Additionally, synthetic circuits frequently have unintended effects on their host organism which can make deployment of these technologies challenging [107]. For example, a circuit which slows bacteria growth by diverting essential resources to production of a pharmacological product may be quickly evolved away [108].

Systems biology, on the other hand, attempts to bring a high-level holistic and mathematically quantitative view to the understanding of biological organisms at the molecular [109, 110], cellular [14], and multi-cellular levels [111]. One common approach in systems biology is collecting quantitative and often high throughput data sets from the same biological system under different experimental conditions [112–115]. Mathematical models are then made from the data which qualitatively or quantitatively represent the observed behavior [116, 117]. Systems biology is a rapidly developing field with new experimental techniques and mathematical methods constantly being deployed. However, despite an ever increasing amount of data systems biologists are able to collect, it frequently remains challenging to tease causal relationships from biological data [118]. Additionally, as more scientists move towards single cell data, many questions remain about how biochemical noise implicit in small molecule counts [119–122] is used or mitigated by cells [123–125].

Bringing systems and synthetic biology together is both natural and inevitable. Systems biology provides the tools and understanding so that synthetic biochemical circuits can be built and optimized for specific contexts [126, 127] and integrated within an organism in a way that their function remains robust [128]. On the flip side, synthetic biology has shown some of its greatest potential when applied toward understanding systems biology research questions [129]. However, for this synthesis to occur, scientists need to be able to easily share and combine diverse models so that bioengineers can use systems level knowledge in their designs and to enable the rapid redeployment of synthetic biochemical systems to new systems. Recent software advanced in multi-scale modeling [130] and data-driven machine learning methods [131, 132] provide one potential avenue for this to occur.

Chapter 5 of this thesis takes a small step in these directions via an experimental study of *E. coli* cell lysate—a platform used in synthetic biology for bioproduction [133] and circuit prototyping [134]—at a systems level. High throughput untargeted small molecule mass spectrometry (metabolomics) [135] is used to gather time course data of cell extract metabolism during cell-free protein expression. This data coupled with easy-to-automate calibration data and a machine learning technique called Bayesian parameter inference are then used to develop a phenomenological model of extract metabolism designed to be integrated with existing circuit models.

Chapter 2

BIOCRNPYLER: COMPILING CHEMICAL REACTION NETWORKS FROM BIOMOLECULAR PARTS IN DIVERSE CONTEXTS

[1] W. Poole, A. Pandey, Z. Tuza, A. Shur, and R. M. Murray, "BioCRNpyler: Compiling Chemical Reaction Networks from Biomolecular Parts in Diverse Contexts," *BioRxiv*, 2020. DOI: `https://doi.org/10.1101/2020.08.02.233478`,

2.1 Forward

The following chapter is quoted directly from the pre-print journal article by the same name which is currently under review [136]. An early version of the paper was presented at the International Workshop on Biodesign Automation (IWBDA) conference in 2020. This work was conducted jointly Ayush Pandey, Andrey Shur, and Zoltan Tuza under the supervision of Richard Murray.

The BioCRNpyler software package compiles chemical reaction networks from simple specifications with the goal of enabling principled model exploration and reuse in systems and synthetic biology. BioCRNpyler uses a high level abstraction to represent diverse biochemical `Components` (parts) in various `Mixtures` (contexts) which can interact via different `Mechanisms` (reaction schemas). The idea underlying this abstraction is that biochemical models are dependent both on their biological context— e.g. *in vivo* versus *in vitro*—as well as their modeling context—e.g. the assumptions made to derive models used to represent different interactions. BioCRNpyler is also an extensive library of commonly used motifs and components in synthetic and systems biology. By using a flexible object-oriented architecture, BioCRNpyler allows users to easily interchange different contexts and combine different components in a modular way in order to rapidly deploy diverse and sophisticated models.

Importantly, BioCRNpyler is not a simulator — it produces models in the Systems Biology Markdown Language (SBML,) which can be simulated with a diverse set of simulators. During my PhD, I have also contributed heavily to the Bioscrape CRN simulator [137] and the Vivarium Engine [130]. The vision of these packages as a unit is inspired by machine learning libraries such as pytorch [138] and

tensorflow [139], which enable diverse network architectures to be easily deployed and efficiently run by machine learning engineers. Similarly, by having flexible model compilation (BioCRNpyler) automatically connected to powerful simulators (Bioscrape and Vivarium) which can further be coupled to parameter inference capabilities (Bioscrape), biochemical models can be deployed and learned from data in a pipeline similar to those used in the machine learning community. Obviously, there is still work to be done. In particular, parameter inference from biochemical data remains a challenge. Chapter 5 of this thesis will demonstrate how inadequate current state of the art inference techniques are. One potential remedy for this is to develop CRN-specific inference software that takes advantage of the underlying mathematical structure of CRNs in order to be more efficient. The theoretical work in Chapters 3 and 4 may be a first step in this direction.

Contribution: I designed the overall software architecture and CRN compilation framework described in the paper and wrote the initial, very rough, version of the code including many of the core `Mechanisms`, `Components`, and `Mixtures`. Then, over the course of two years, this code was almost universally re-implemented by the BioCRNpyler team. Ayush Pandey took care of most of the SBML compatibility. Zoltan Tuza was instrumental in improving code quality, adding integrated testing, and generally making everything "Pythonic" and developer friendly. Andrey Shur focused on prototyping many of the more complex and specific example applications of the software such as `DNA_construct` and `IntegraseEnumerator` as well as building a number of visualization packages. I was involved in the design of every large feature and improvement. I also helped implement many new features and ensured that changes were propagated throughout the entire code base in a way that maintained the integrity of the software. Finally, I wrote the entire publication included in the thesis (with review from the other coauthors).

2.2 Abstract

Biochemical interactions in systems and synthetic biology are often modeled with chemical reaction networks (CRNs). CRNs provide a principled modeling environment capable of expressing a huge range of biochemical processes. In this paper, we present a software toolbox, written in Python, that compiles high-level design specifications to CRN representations. This compilation process offers four advantages. First, the building of the actual CRN representation is automatic and outputs Systems Biology Markup Language (SBML) models compatible with numerous simulators. Second, a library of modular biochemical components allows for dif-

ferent architectures and implementations of biochemical circuits to be represented succinctly with design choices propagated throughout the underlying CRN automatically. This prevents the often occurring mismatch between high-level designs and model dynamics. Third, high-level design specification can be embedded into diverse biomolecular environments, such as cell-free extracts and *in vivo* milieus. Finally, our software toolbox has a parameter database, which allows users to rapidly prototype large models using very few parameters which can be customized later. By using BioCRNpyler, users can easily build, manage, and explore sophisticated biochemical models using diverse biochemical implementations, environments, and modeling assumptions.

2.3 Introduction

Chemical reaction networks (CRNs) are the workhorse for modeling in systems and synthetic biology [14]. The power of CRNs lies in their expressivity; CRN models can range from physically realistic descriptions of individual molecules to coarse-grained idealizations of complex multi-step processes [1]. However, this expressivity comes at a cost. Choosing the right level of detail in a model is more an art than a science. The modeling process requires careful consideration of the desired use of the model, the available data to parameterize the model, and prioritization of certain aspects of modeling or analysis over others. Additionally, biological CRN models can be incredibly complex including dozens or even hundreds or thousands of species, reactions, and parameters. Maintaining complex hand-built models is challenging and errors can quickly grow out of control for large models. Software tools can answer many of these challenges by automating and streamlining the model construction process.

Due to CRN's rich history and diverse applications, the available tools for a CRN modeler are vast and include: extensive software to generate and simulate CRNs [140, 141], databases of models [142], model analysis tools [143, 144], and many more. However, relatively few tools exist to aid in the automated construction of general CRN models from simple specifications. For example, even though synthetic biologists have adopted a module and part-driven approach to their laboratory work [5], models are still typically built by hand on a case-by-case basis. Recognizing the fragile non-modular nature of hand built models, several synthetic biology design automation tools have been developed for specific purposes such as implementing transcription factor or integrase-based logic [37, 145]. These tools indicate a growing need for design and simulation automation in synthetic biology,

as part and design libraries are expanded.

As the name would suggest, BioCRNpyler (pronounced bio-compiler) is a Python package that compiles CRNs from simple specifications of biological motifs and contexts. This package is inspired by the molecular compilers developed by the DNA-strand displacement community and molecular programming communities which, broadly speaking, aim to compile models of DNA circuit implementations from simpler CRN specifications [22, 24, 146], or rudimentary programming languages [147, 148]. However, BioCRNpyler differs from these tools for three main reasons: first, it is not focused only on DNA implementations of chemical computation; second, it does not take the form of a traditional programming language; and third, modeling assumptions and compilation schemas can be easily redefined by the user. BioCRNpyler combines specifications consisting of synthetic biological parts and systems biology motifs that can be reused and recombined in diverse biochemical contexts at customizable levels of model complexity. In other words, BioCRNpyler compiles detailed CRN models from abstract specifications of a biochemical system. Importantly, BioCRNpyler is not a CRN simulator—models are saved in the Systems Biology Markup Language (SBML) [149] to be compatible with the user's simulator of choice. Figure 2.1 provides motivating examples for the utility of BioCRNpyler by demonstrating the rapid construction of diverse CRNs by reusing common parts and modifying the modeling context.

There are many existing tools that provide some of the features present in BioCRNpyler. Systems Biology Open Language (SBOL) [150] uses similar abstractions to BioCRNpyler but is fundamentally a format for sharing DNA-sequences with assigned functions and does not compile a CRN. The software package iBioSim [151, 152] compiles SBOL specifications into SBML models and performs analysis and simulation. Although BioCRNpyler is capable of similar kinds of compilation into SBML, it is not a simulator. Importantly, BioCRNpyler does not hard-code how models are compiled—instead it should be viewed as a customizable software compilation language that can be applied to compile many kinds of systems beyond genetic networks. The rule-based modeling framework BioNetGen [153] allows for a system to be defined via interaction rules which can then be simulated directly or compiled into a CRN. Internally, BioCRNpyler functions similarly to this rule-based modeling compilation. Similarly to PySB [154], BioCRNpyler provides a library of parts, mechanisms, and biomolecular contexts that allow for models to be succinctly produced without having to manually specify and verify many complex

rules. Finally, the MATLAB TX-TL Toolbox [155] can be seen as a prototype for BioCRNpyler but lacks the object-oriented framework and extendability beyond cell-free extract systems.

BioCRNpyler is purposefully suited to *in silico* workflows because it is an extendable object-oriented framework that integrates existing software development standards and allows complete control over model compilation. Simultaneously, BioCRNpyler accelerates model construction with extensive libraries of biochemical parts, models, and examples relevant to synthetic biologists, bio-engineers, and systems biologists. The BioCRNpyler package is available on GitHub [156] and can be installed via the Python package index (PyPi).

2.4 Motivating Examples

This section highlights the ease-of-use of BioCRNpyler through several well-known synthetic biology examples. As a summary, Figure 2.1 demonstrates the utility of compiling CRNs with BioCRNpyler. The names of Python classes are highlighted `typographically` and are defined more thoroughly in later sections. Time-course simulations in Figure 2.1 were done with Bioscrape [137] and circuit diagrams were created with DNAplotlib [157]. Additional example models of the lac operon and an integrase circuit are depicted in Figures 2.3 and 2.5, respectively.

Inducible Repression, Toggle Switch, and Repressilator

Models A, B, and C show three archetypal motifs from synthetic biology: inducible repression, a bistable toggle switch [158], and the repressilator [159]. All three of these examples are created by reusing the same `Components` wired together in different ways as described in Section 2.8. The ability to reuse `Components` allows for convenient design-space exploration of different circuit architectures. Furthermore, as explained in Section 2.4 and 2.4, examples D, E and F show how these `Components` can be tested in different contexts by changing the `Mechanisms` and `Mixtures` used to compile the `Components` resulting in nuanced implementation-specific and context-specific models.

dCas9 Repressor and Guide RNA Coexpressed with Reporter

Figure 2.1D builds upon the repression Model A by modeling an implementation consisting of a guide-RNA and dCas9 complex that acts as a repressor by inhibiting RNA-polymerase binding to the reporter's promoter [160]. Model A also includes more intra-cellular context such as nucleases and ribosomes. By including cellular

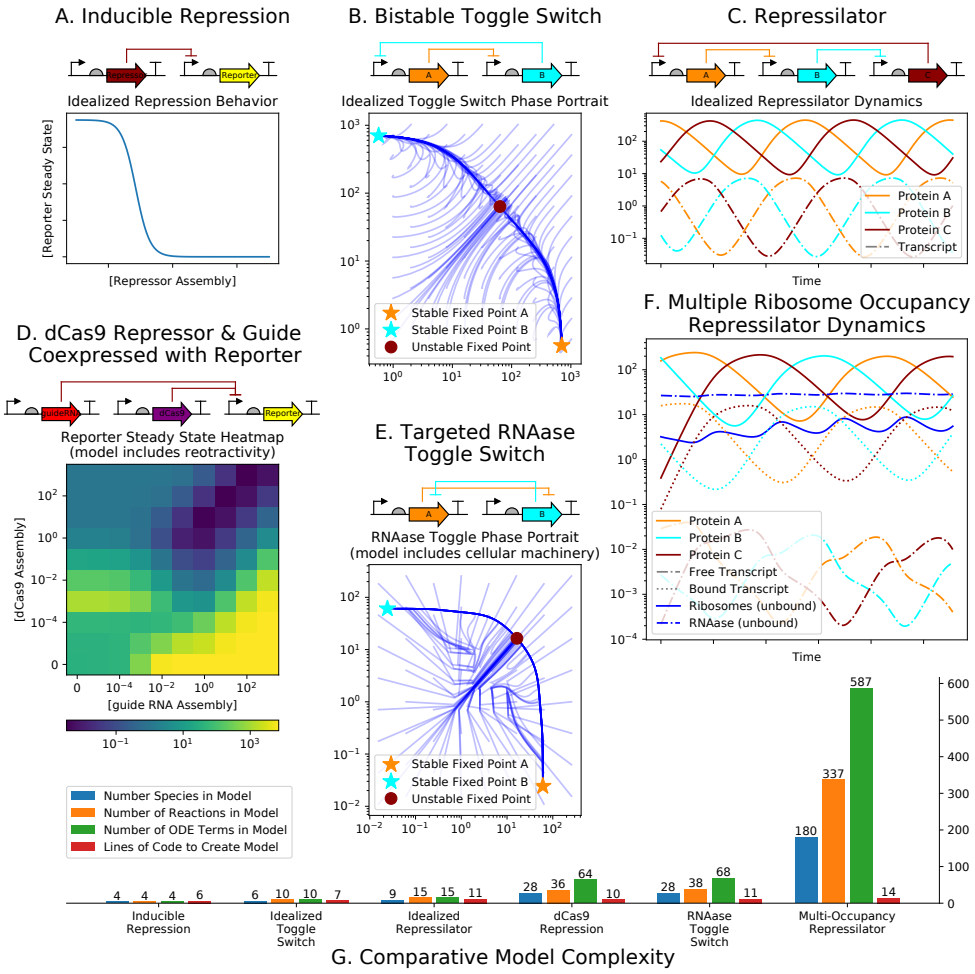


Figure 2.1: Motivating Examples. The idealized models (A, B, and C) do not model the cellular environment; genes and transcripts transcribe and translate catalytically. **A.** Schematic and simulation of a constitutively active repressor gene repressing a reporter. **B.** Schematic and simulations of a toggle switch created by having two genes, *A* and *B*, mutually repress each other. **C.** Schematic and dynamics of a 3-repressor oscillator. The detailed models (D, E, & F) model the cellular environment by including ribosomes, RNases, and background resource competition for cellular resources. **D.** A dCas9-guideRNA complex binds to the promoter of a reporter and inhibiting transcription. Heatmap shows retroactivity caused by varying the amount of dCas9 and guide-RNA expressed. The sharing of transcription and translational resources gives rise to increases and decreases of reporter even when there is very little repressor. **E.** A proposed model for a non-transcriptional toggle switch formed by homodimer-RNase; the homodimer-RNase made from subunit *A* selectively degrades the mRNA producing subunit *B* and visa-versa. **F.** A model of the Repressillator exploring the effects of multiple ribosomes binding to the same mRNA. **G.** Histogram comparing the sizes of models A-F and the amount of BioCRNpyler code needed to generate them.

machinery in the model, the co-expression of dCas9 and the guide RNA is able to influence the reporter via loading effects and retroactivity, which can cause unintended increases and decreases in reporter expression [161] even when only the guide-RNA or dCas9 is present.

Targeted RNase Toggle Switch

Figure 2.1E models a hypothetical toggle switch that functions at the RNA level instead of the transcriptional level. Each DNA assembly expresses a subunits A and B of two homodimer-RNase. The homodimer-RNase made from subunit A selectively degrades the mRNA producing subunit B and visa-versa. Such a system could potentially be engineered via RNA-targeting Cas9 [162] or more complex fusion proteins [163].

Multiple Ribosome Occupancy Repressilator Dynamics

Figure 2.1F illustrates how BioCRNpyler can be used to easily generate more realistic and complex models of biochemical processes in order to validate if model simplifications are accurate. It is common practice in transcription and translation models to use an enzymatic process consisting of a single ribosome (R) to a transcript (T) which then produces a single protein (P). This translation `Mechanism` could be written as: $R + T \leftrightharpoons R : T \rightarrow R + T + P$. Indeed, both example models D and E use such a simplification. However, experiments show that in fact many ribosomes can co-occupy the same mRNA [164]. By changing the underlying translation `Mechanism` to model multiple ribosomal occupancy of a single mRNA, a considerably more complex Repressilator model was created. Importantly, this model exhibits very similar behavior to the simpler model, suggesting that multi-occupancy of ribosomes on mRNA can be neglected in these kinds of genetic regulatory circuits.

Network Complexity

Finally, the bottom bar chart in Figure 2.1G shows that even as the size of the underlying CRN grows, the amount of BioCRNpyler code that is needed to generate the model remains very small. This enables the generation of large and complex models with greater accuracy and lower chance of human error. For example, imagine writing down ODEs with hundreds of terms and then trying to systematically modify the equation: human error is nearly inevitable. By using BioCRNpyler to compile CRNs, models can be easily produced, modified, and maintained.

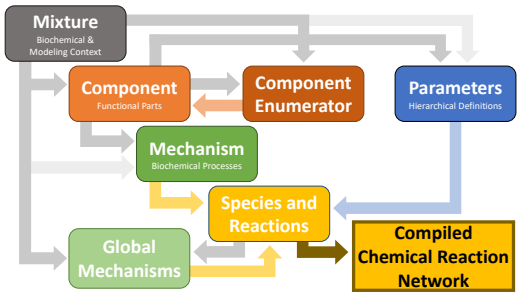


Figure 2.2: The hierarchical organization of Python classes in the BioCRNpyler. Dark gray arrows represent direction of compilation: from high-level design specifications (`Components`) in a modeling context (`Mixtures`) and biochemical processes (`Mechanims`) to a CRN representation. Light gray arrows represent inheritance of default `Parameters` and `Mechanims`. Yellow arrows represent `Species` and `Reaction` generation which are placed in a `ChemicalReactionNetwork` represented by the bottom right gold box. `ComponentEnumerators` are advanced objects used to automate the generation of `Components`. `GlobalMechanisms` are rules used to generate `Species` and `Reactions` at the end of compilation.

Parameter Database

Importantly, all these examples in this section make use of the same underlying set of 10-20 default parameters (estimated from Cell Biology by the Numbers [165]) demonstrating how BioCRNpyler's parameter database makes model construction and simulation possible even before detailed experiments or literature review.

2.5 Framework and Compilation Overview

BioCRNpyler is an open-source Python package that compiles high-level design specifications into detailed CRN models, which then are saved as an SBML files [149]. BioCRNpyler is written in Python with a flexible object-oriented design, extensive documentation, and detailed examples which allow for easy model construction by modelers, customization and extension by developers, and rapid integration into data pipelines. As Figure 2.2 shows, underlying BioCRNpyler is a comprehensive `ChemicalReactionNetwork` class allowing for the direct creation and manipulation of `Reactions` and the participating `Species` to represent molecular interactions at many levels of complexity. For example, an entire gene may be modeled as a single `Species`, as an `OrderedPolymerSpecies` with multiple binding specific sites, or as a `PolymerConformation` which represents the secondary structure of one or more `OrderedPolymerSpecies`.

BioCRNpyler also compiles CRNs objects from high-level specifications defined by modular `Components` combined together in a `Mixture` representing a biochemical

context (e.g. cell lysate extract). Modeling assumptions and specific knowledge of biochemical processes are defined via `Mechanisms` which can be placed inside `Components` and `Mixtures`. This class structure allows for the biochemical parts (e.g. `Components`) to be reused to quickly produce numerous different architectures and implementations, such as those described in the motivating examples. These different architectures and implementations can further be tested in different contexts providing easily customizable levels of biochemical and modeling complexity represented by `Mixtures` and `Mechanisms`.

The `Mixture`, `Component`, and `Mechanism` classes are hierarchical. `Mixtures` represent biological context by containing `Components` to represent the biochemical environment and `Mechanisms` to represent the modeling details. For example, the `TxTlExtract` subclass of `Mixture` represents bacterial cell extract and contains Ribosomes, RNA Polymerase, and RNases `Components` as well as transcription, translation, and RNA-degredation `Mechanisms`. Additionally, `Components` can be added to a `Mixture` to produce a particular biochemical system of interest in a particular context. Figure (2.3) illustrates how a set of BioCRNpyler `Components` and `Mechanisms` can be joined together to produce a systems level model of the lac operon—a highly studied gene regulatory network in *E. coli* which regulates whether glucose or lactose is metabolized [166]. This model consists of around a dozen `Components` and `Mechanisms` which jointly enumerate hundreds of species and reactions representing the combinatorial set of conformations of the lac operon and its associated transcription factors, transcription, translation, transport, mRNA degradation, and dilution.

During compilation, `Components` represent biochemical functionality by calling `Mechanisms` to produce `Species` and `Reactions`. The `Component` class may use the `Mechanisms` contained in the `Mixture` or have their own custom `Mechanisms` to have more differentiated functionality. For example, a `RepressiblePromoter` (a subclass of `Component`) might rely on the `Mixture` for its translation `Mechanism` but use a custom transcription `Mechanism`. BioCRNpyler uses flexible parameter databases contained in both `Mixtures` and `Components` to allow for rapid model prototyping using just a few default parameters, which can later be customized for each `Component` and `Mechanism`. Specifically, `Mechanisms` will first search for parameters in the `Component` that called them before defaulting to the parameters of the `Mixture`. This defaulting behavior is illustrated by the light gray arrows of Figure 2.2. Finally, component enumeration provide a highly flexible framework

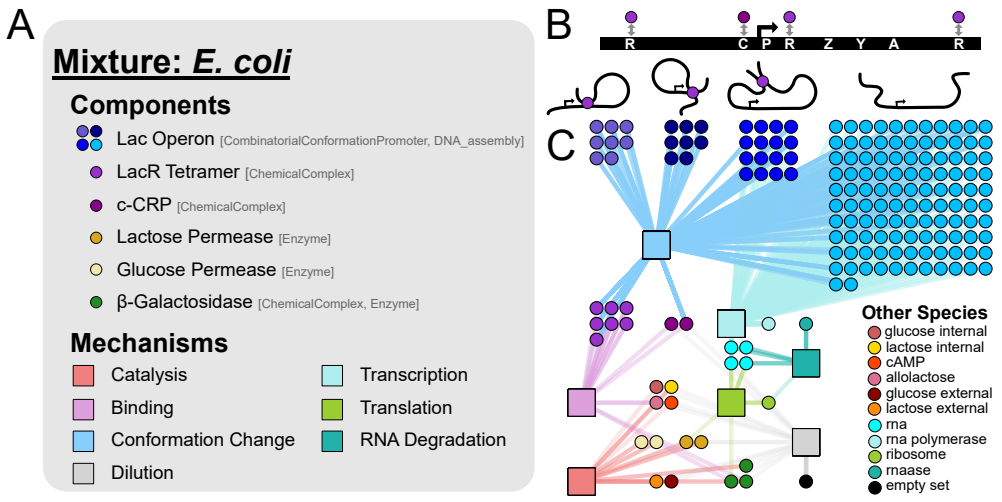


Figure 2.3: Compiling a model of the Lac Operon using BioCRNpyler specifications with 173 species and 343 reactions using ~50 lines of code. **A.** A `Mixture` contains a set of `Components` and `Mechanisms`. The `Component` classes used for each element of the model are shown in brackets. **B.** A schematic of the lac operon and the three looped and one open conformation it can take. Each conformation contains a combinatoric number of states based upon the accessible binding sites: R are lac repressor binding sites; C is the activator c-CRP binding site; P is the promoter; and Z, Y, A are the three lac genes. **C.** A graph representation of the compiled CRN. Each circle is a unique chemical species. Square boxes show how chemical species interact via reactions generated by specific `Mechanisms`.

to automatically generate new `Components` during compilation as illustrated in the integrase example Figure 2.5.

Internal CRN Representation

Formally, a CRN is a set of species $S = \{S_i\}$ and reactions $R : \{I \xrightarrow{\rho(s;\theta)} O\}$ where I and O are multisets of species, ρ is the rate function or propensity, s is a vector of species' concentrations (or counts), and θ are rate parameters. Typically, CRNs are simulated as ordinary differential equations (ODEs) and numerically integrated [1]. A stochastic semantics also allows CRNs to be simulated as continuous-time Markov chains [167]. Besides their prevalence in biological modeling, there is rich theoretical body of work related to CRNs from the mathematical [168], computer science [169], and physics communities [170]. Despite these theoretical foundations, many models are phenomenological in nature and lack mechanistic details of various biological processes. The challenge of constructing correct models is compounded by the difficulty in differentiating between correct and incorrect models based upon experimental data [16, 171, 172].

BioCRNpyler allows users to easily build diverse CRNs with flexible species and reaction representations which are then saved as SBML [149] for simulation with many different simulators. The CRN classes inside BioCRNpyler provide useful functionality so that users can easily modify CRNs produced via compilation, produce entire CRNs by hand, or interface hand-produced CRNs with compiled CRNs. These functionalities include classes to represent `Species` bound together as `ComplexSpecies`, lists of species organized as `OrderedPolymerSpecies`, multi-polymer secondary structures called a `PolymerConformation`, and many diverse propensity function types including mass-action, Hill functions, and general user-specified propensities. Additionally, user-friendly printing functionality allows for the easy visualization of CRNs in multiple text formats or as interactive reaction graphs formatted and drawn using Bokeh and ForceAtlas2 [173, 174].

Mechanisms are Reaction Schemas

When modeling biological systems, modelers frequently make use of mass-action CRN kinetics which ensure that parameters and states have clear underlying mechanistic meanings. However, for the design of synthetic biological circuits and analysis using experimental data, phenomenological or reduced-order models are commonly utilized as well [1]. Empirical phenomenological models have been successful in predicting and analyzing complex circuit behavior using simple models with only a few lumped parameters [17, 175, 176]. Bridging the connections between the different modeling abstractions is a challenging research problem. This has been explored in the literature using various approaches such as by direct mathematical comparison of mechanistic and phenomenological models [7–9] or by studying particular examples of reduced models [1]. BioCRNpyler provides a computational approach using reaction schemas to easily change the mechanisms used in compilation from detailed mass-action to coarse-grained at various level of complexity.

Reaction schemas refer to BioCRNpyler's generalization of switching between different mechanistic models: a single process can be modeled using multiple underlying motifs to generate a class of models. `Mechanisms` are the BioCRNpyler objects responsible for defining reaction schemas. In other words, various levels of abstractions and model reductions can all be represented easily by using built-in and custom `Mechanisms` in BioCRNpyler. For example, to model the process of transcription (as shown in Figure 2.4), BioCRNpyler allows the use of various phenomenological and mass-action kinetic models by simply changing the choice of reaction schema. Notably, this provides a unique capability to quickly compare system models across

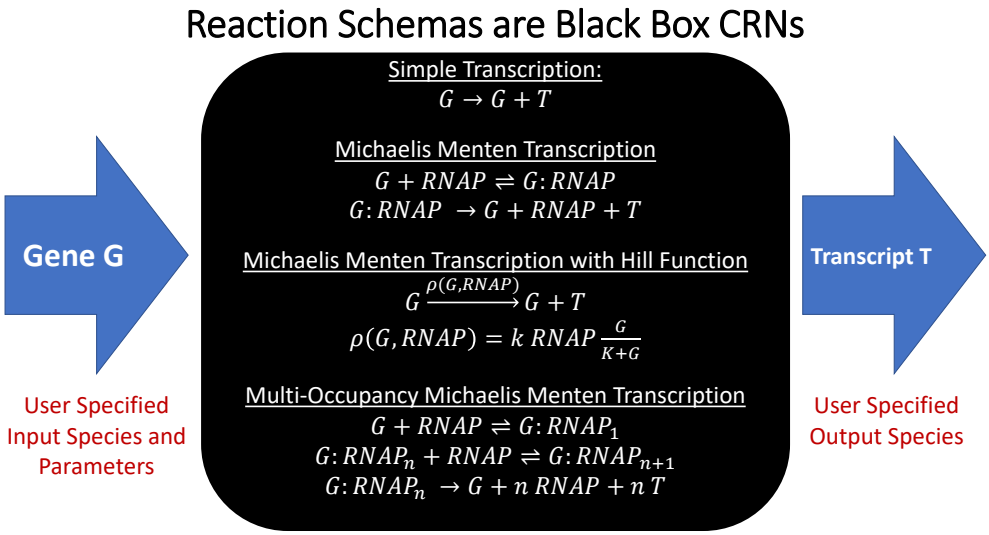


Figure 2.4: Mechanisms (Reaction Schemas) representing transcription.

various levels of abstraction enabling a more nuanced approach to circuit design and exploring system parameter regimes.

Formally, reaction schemas are functions that produce CRN species and reactions from a set of input species and parameters: $f : (S', \theta) \rightarrow (S, R)$. Here the inputs S' are chemical species and θ are rate constants. The outputs $S \supseteq S'$ are an increased set of species and R is a set of reactions. Figure 2.4 gives different examples of reaction schemas representing transcription. This functionality allows modelers to generate CRNs at different levels of complexity and reuse CRN motifs for some `Components` while customizing `Mechanisms` for others. Importantly, BioCRNpyler contains a large and growing library of existing `Mechanisms` extensively documented via examples making them easy to use and re-purpose without extensive coding. Internally, each `Mechanism` class has a type (e.g. transcription) which defines the input and output species it requires. Global mechanisms are a special subclass of `Mechanism` called at the end of compilation to represent processes which act on large subsets of CRN species such as dilution in cellular models. The ability to generate chemical `Species` and `Reactions` via customized `Mechanisms` is one of the key features making BioCRNpyler distinct from other frameworks. Hierarchical SBML and supporting software provide [177] a notable exception—however BioCRNpyler contains a library of reusable chemical reaction motifs, while Hierarchical SBML is a standard for describing embedded CRN models.

Components Represent Functionality

In BioCRNpyler, `Components` are biochemical parts or motifs, such as promoters, enzymes, and chemical complexes. `Components` represent biomolecular functionality; a promoter enables transcription, enzymes perform catalysis, and chemical complexes bind together. `Components` express their functionality by calling particular `Mechanism` types during compilation. Importantly, `Components` are not the same as CRN `Species`; one `Species` might be represented by multiple `Components` and a `Component` might produce multiple `Species`! For example, the single `CombinatorialConformationPromoter Component` used in the lac operon model (shown in Figure 2.3) produces hundreds of unique `Species`. Conversely, the chemical species β-Galactosidase is modeled using two components: an `Enzyme` to model the metabolism of lactose and a `ChemicalComplex` to model the fact that β-Galactosidase is a homeotetramer. `Components` are flexible and can behave differently in different contexts or behave context-independently. To define dynamic-context behavior, `Components` may use mechanisms and parameters provided by the `Mixture`. To define context-independent behavior, `Components` may have their own internal `Mechanisms` and parameter databases. The BioCRNpyler library includes many `Component` subclasses to model enzymes (`Enzyme`), chemical complexes (`ChemicalComplexes`) formed by molecular binding, Promoters (`Promoter`), Ribosome Binding Sites (`RBS`), complex genetic architectures (such as `DNA_construct` illustrated in Figure 2.5), and more.

Mixtures Represent Context

`Mixtures` are collections of default `Components`, default `Mechanisms`, and user-added `Components`. `Mixtures` can represent chemical context (e.g. cell extract vs. *in vivo*), as well as modeling resolution (e.g. what level of detail to model transcription or translation at) by containing different internal `Components` and `Mechanisms`. `Mixtures` also control CRN compilation by requesting `Species` and `Reactions` for each of their `Components`. After receiving all these `Species` and `Reactions`, Mixtures then apply global mechanisms which act on all the Species produced by `Components`. BioCRNpyler comes with a variety of `Mixtures` to represent cell-extracts and cell-like systems with multiple levels of modeling complexity.

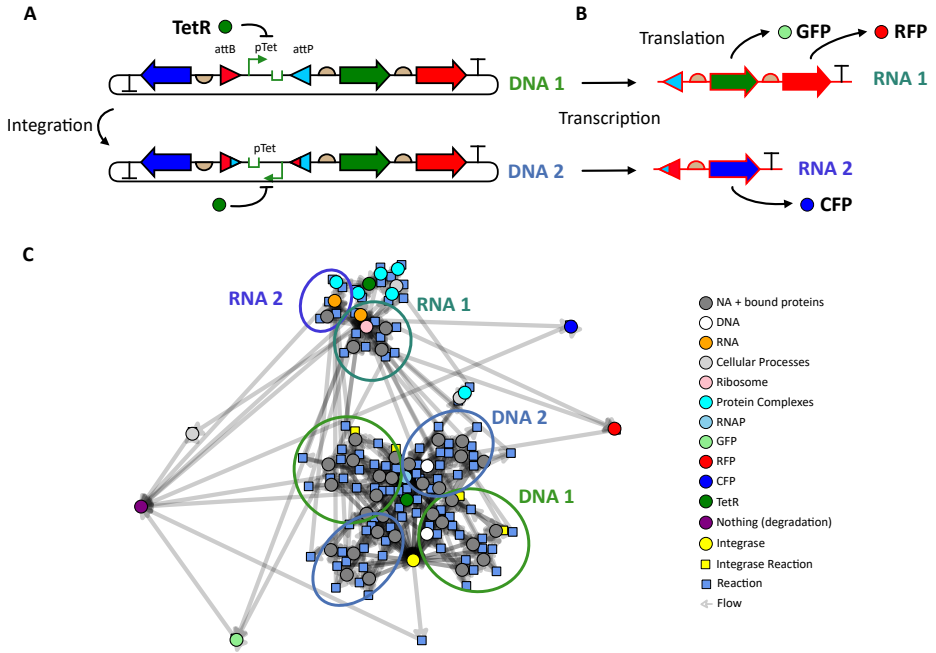


Figure 2.5: Using the DNA construct class, DNA parts (a subclass of `Component`) can be arranged in the same order as they would be in a DNA molecule and compiled into a CRN. **A.** A complicated DNA construct contains three coding sequences and a single promoter which can be flipped by an integrase. Global component enumeration allows an integrase enumerator to generate "DNA 2" from "DNA 1." **B.** Local component enumeration generates an RNA construct from each DNA construct. **C.** A directed graph representation of the compiled CRN from the DNA construct. `Species` represented by circles participate in `Reactions` represented by squares. Circled groups of `Species` and `Reactions` involve the DNA construct labeled "DNA 1," or the RNA constructs labeled "RNA 1" or "RNA 2." The component enumeration process creates all the necessary `Species` and `Reactions` to simulate integration, transcription, and translation from linear and circular DNA, taking into account the compositional context of DNA parts.

Component Enumeration Allows for Arbitrary Complexity

Component enumeration allows additional `Components` to be generated dynamically during the compilation process. Local component enumeration occurs when one `Component` compiles itself into a set of many sub-components. For example, the DNA construct `Component`, made out of an ordered list of DNA parts, such as `Promoters`, `RBSs`, and `CDSs`, uses component enumeration to enumerate all the possible mRNAs which could be transcribed as new RNA construct `Components`. When the CRN is compiled, many RNA constructs may be generated from a single DNA construct. All the objects generated this way can then be coupled to `Mechanisms` automatically to compile a complex CRN.

Similarly, global component enumeration generates a set of `Components` based upon all the `Components` in the `Mixture`. For example, serine integrases are enzymes which are capable of recombining strands of DNA at specific integration sites [178]. Integration events can happen within a single piece of DNA or between multiple DNA species. The integrase enumeration looks at all the DNA constructs present in a `Mixture` and enumerates all possible integration events to generate new DNA constructs which are then fed back into the enumeration recursively. Figure 2.5 illustrates both local and global component enumeration involving a `DNA_construct` with serine integrase attachment sites. Other integrase types (for example homotypic sites) are also supported. Due to the potential of an unbounded number of `Components` being produced by such a process, component enumeration can be called to a user-specified recursion depth in order to compile arbitrarily large chemical reaction networks.

Flexible Parameter Databases

Developing models is a process that involves defining and then parameterizing interactions. Often, at the early stage of model construction, exact parameter values will be unavailable. BioCRNpyler has a sophisticated parameter framework which allows for the software to search user-populated parameter databases for the parameter that closest matches a specific `Mechanism`, `Component`, and parameter name. This allows for models to be rapidly constructed and simulated with "ball-park" parameters and then later refined with specific parameters derived from literature or experiments. This framework also makes it easy to incorporate diverse parameter files together and share parameters between many chemical reactions.

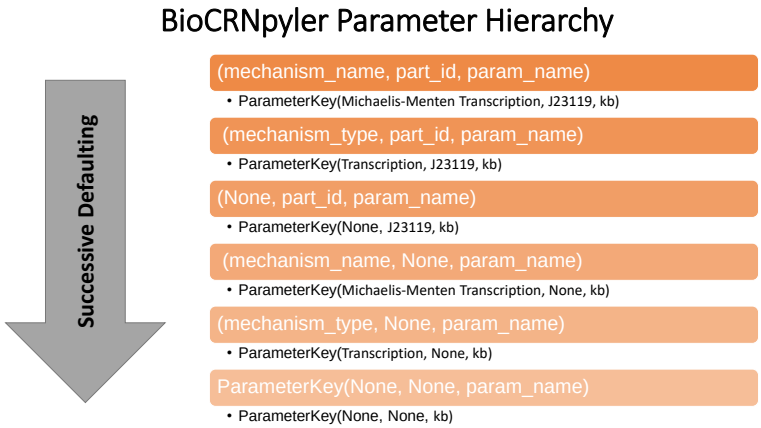


Figure 2.6: BioCRNpyler Parameter Defaulting Hierarchy. If a specific parameter key (orange boxes) cannot be found, the parameter database automatically defaults to other parameter keys. This allows for parameter sharing and rapid construction of complex models from relatively few non-specific (e.g. lower in the hierarchy) parameters.

2.6 Building an Open-Source Community

BioCRNpyler aims to be a piece of open-source community driven software that is easily accessible to biologists and bioengineers with varying levels of programming experience as well as easily customizable by computational biologists and more advanced developers. Towards these ends, the software package is available via GitHub and PyPi, requires very minimal software dependencies, contains extensive examples and documentation in the form of interactive Jupyter notebooks [156], YouTube tutorials [179], and automated testing to ensure stability. Furthermore this software has been extensively tested via inclusion in a bio-modeling course and bootcamps with dozens of users ranging from college freshmen and sophomores with minimal coding experience to advanced computational biologists. BioCRNpyler has already been deployed to build diverse models in systems and synthetic biology [130, 180–182]. Developing new software functionality is also a simple process documented on the GitHub contributions page.

Integrated Testing

BioCRNpyler uses Github Actions and Codecov [183] to automate testing on GitHub. Whenever the software is updated, a suite of tests is run including extensive unit tests and functional testing of tutorial and documentation notebooks. Automated testing ensures that changes to the core BioCRNpyler code preserve the functionality of the package. The integration of Jupyter notebooks into testing

allows users to easily define new functionality for the software and document that functionality with detailed explanations which are simultaneously tests cases.

Documentation and Tutorials

The BioCRNpyler GitHub page contains over a dozen tutorial Jupyter notebooks [156] and presentations explaining everything from the fundamental features of the code to specialized functionality for advanced models to how to add to the BioCRNpyler code-base [179]. This documentation has been used successfully in multiple academic courses and is guaranteed to be up-to-date and functional due to automatic testing.

2.7 Future Directions

BioCRNpyler is an ongoing effort which will grow and change with the needs of its community. Extending this community via outreach, documentation, and an ever expanding suite of functionalities is central to the goals of this project. We are particularly interested in facilitating the integration of BioCRNpyler into existing laboratory pipelines in order to make modeling a central part of the design-build-test cycle in synthetic biology. One avenue towards this goal is to add compatibility to existing standards such as SBOL [150] and automation platforms such as DNA-BOT [184] so BioCRNpyler can automatically compile models of circuits as they are being designed and built. This approach will be a generalization and extension of Roehner et al. [185]. In particular, due to the modular BioCRNpyler compilation process, it will be possible to have programmatic control over the SBML model produced from BioCRNpyler.

We also plan on extending the library to include more realistic and diverse `Mixtures`, `Mechanisms`, and `Components` (particularly experimentally validated models of circuits in *E. coli* and in cell extracts). We hope that these models will serve as examples and inspiration for other scientists to add their own model systems in other organisms to the software library.

Finally, we believe that the Context-Part-Mechanism abstraction of model compilation used in BioCRNpyler is fundamental and could be extended to other non-CRN based modeling approaches. Advanced simulation techniques beyond chemical reaction networks will be required to accurately model the diversity and complexity of biological systems. New software frameworks such as Vivarium [130] have the potential to generate models which couple many simulation modalities. The abstractions used in BioCRNpyler could be extended to compile models beyond

chemical reaction networks such as mechanical models, flux balance models, and statistical models derived from data. The integration of these models together will naturally depend on both detailed mechanistic descriptions as well as overarching system context. We emphasize that building extendable and reusable frameworks to enable quantitative modeling in biology will become increasingly necessary to understand and design ever more complex biochemical systems.

Acknowledgement

We would like to thank the Caltech BE 240 class and the Murray Biocircuits lab for extensive testing of this software and discussions of relevant models, library of parts, and parameters. In particular, we would like to thank Matthieu Kratz, Liana Merk, and Ankita Roychoudhury for contributing to the software library. The authors W.P. and A.P. are partially supported by US National Science Foundation (CBET-1903477). A.P. is also supported by the Defense Advanced Research Projects Agency (Agreement HR0011-17-2-0008). The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

2.8 Supplemental: Code for Examples

This section provides code from the examples Figure 2.1. The first three models are idealized in the sense that they are represented by Hill functions and include no cellular machinery such as ribosomes or polymerases. Producing these models in BioCRNpyler is easy and just requires the reuse of a few parts: `DNAassembly` represents a simple transcriptional unit with a promoter, transcript, and optionally an ribosome binding site (RBS) and protein product. `RepressiblePromoter` creates a promoter modeled by a Hill function. `Species` creates CRN species used in the models. Notice that only `Species` which are shared between different `Components` need created by hand—BioCRNpyler takes care of the rest. For example, in inducible repression example the repressor R is created by hand because it is placed inside the `RepressiblePromoter`. However the DNA, transcript, and protein are automatically generated from the name of the `DNAassmebly`. Finally, everything is added together into a subclass of `Mixture` and compiled into a `ChemicalReactionNetwork`. The second three models build off the general architectures of the first three, but add in more complicated context and implementation details. These models use the considerably more complex context `TxTlDilutionMixture` which includes molecular machinery such as RNAP, ri-

bosomes, RNases, and background cellular processes. Additional implementation details in the form of `Components` and `Mechanisms` are also added to these models.

Inducible Repression

Here a repressor is constituitively produced from a `DNAassembly`. This repressor is then linked to a `RepressiblePromoter` which models repression using a Hill function.

```
from biocrnpyler import *
# Models a piece of DNA that constitutively produces the species R
repressor = Species("R")
const_rep = DNAassembly(name="const_rep", promoter="medium", rbs="medium",
    protein=repressor)
# R represses RepressiblePromoter which is placed into another DNAassembly
    reporter
prom = RepressiblePromoter(name="pR", repressor=repressor)
reporter = DNAassembly(name="Reporter", promoter=prom, rbs="strong",
    initial_concentration=1)
# ExpressionDilutionMixture models gene expression without
    transcription/translation
mixture = ExpressionDilutionMixture(components=[reporter, const_rep],
    parameter_file="params.txt")
CRN = mixture.compile_crn()
```

Toggle Switch

In the following example, a toggle switch is created by connecting two instances of `RepressiblePromoter`. Notice that string names passed to promoter and RBS are used to help find parameters. BioCRNpyler comes with many default parameters to enable rapid model prototyping.

```
from biocrnpyler import *
# Creates A and is repressed by B
repA = Species("A")
promA = RepressiblePromoter(name="pA", repressor=repB)
assemblyA = DNAassembly(name="A", promoter=promA, rbs="medium",
    protein=repA, initial_concentration=1)
# Creates B and is repressed by A
repB = Species("B")
promB = RepressiblePromoter(name="pB", repressor=repA)
assemblyB = DNAassembly(name="B", promoter=promB, rbs="medium",
    protein=repB, initial_concentration=1)
```

```
# SimpleTxTlDilutionMixture includes transcription and translation but no
    machinery
mixture = SimpleTxTlDilutionMixture(components=[assemblyA, assemblyB],
    parameter_file= "params.txt")
CRN = mixture.compile_crn()
```

Repressilator

The code to create a 3-node repression oscillator is really just adding one more unit and rewiring the toggle switch example.

```
from biocrnpyler import *
# Create Repressors
repA = Species("A")
repB = Species("B")
repC = Species("C")
# Create Promoters
promA = RepressiblePromoter(name="pA", repressor=repC)
promB = RepressiblePromoter(name="pB", repressor=repA)
promC = RepressiblePromoter(name="pC", repressor=repB)
#Create DNAassemblies
assemblyA = DNAassembly(name="A", promoter=promA, rbs="medium",
    protein=repA, initial_concentration=1)
assemblyB = DNAassembly(name="B", promoter=promB, rbs="medium",
    protein=repB, initial_concentration=1)
assemblyC = DNAassembly(name="C", promoter=promC, rbs="medium",
    protein=repC, initial_concentration=1)
# Place it all in a Mixture & Compile
mixture = SimpleTxTlDilutionMixture(components=[assemblyA, assemblyB,
    assemblyC], parameter_file="params.txt")
crn = mixture.compile_crn()
```

Cas9 Repressor and Guide RNA Coexpressed with Reporter

Modeling a dCas9-guideRNA repressor in bioCRNpyler requires that the dCAs9 and guide RNA know to bind together. This is accomplished via the `Component` subclass `ChemicalComplex` which models binding between multiple species. The resulting dCas9-guideRNA `ComplexSpecies` is used as a repressor.

```
from biocrnpyler import *
# parameter syntax: (mechanism_name, part_id, parameter_name) : value
# Only one dCas9-guideRNA complex binds to the promoter at once
params = {
```

```
    ("negativehill_transcription", None, "n"):1
}
# Create guide RNA and dCas9 Species
guide = Species("guide", material_type="rna")
dcas = Species("dCas9")
# These species will bind together by placing them in the ChemicalComplex
    Component
# "notdegradable" ensures that RNases do not degrade gRNA-dCas9 complexes.
repressor = ChemicalComplex([dcas, guide], attributes=["notdegradable"])
reporter = Species("reporter")
# Constuitive Assemblies to produce dCas9 and the guideNRA
assembly_dcas = DNAassembly(name="dcas", promoter="medium", rbs="medium",
    protein=dcas)
assembly_guide = DNAassembly(name="guide", promoter="strong", rbs=None,
    transcript=guide)
# Create a repessible promoter
pReg = RepressiblePromoter(name="pA", repressor=repressor,
    parameters=params)
assembly_rep = DNAassembly(name="reporter", promoter=pReg, rbs="strong",
    protein=reporter, initial_concentration=1)
# Place the Components in a Mixture
extract = TxTlDilutionMixture("e coli", components=[assembly_rep,
    assembly_dcas, assembly_guide, repressor], parameter_file="params.txt")
#Compile the CRN
crn = extract.compile_crn()
```

Targeted RNase Toggle Switch

The targeted RNase toggle switch model is a hypothetical model similar to a normal toggle switch, but with regulation at the RNA level instead of the transcriptional level. This is accomplished by creating two constitutively expressed RNases (which are `ChemicalComplexes` made up of two subunits) and adding custom `Mechanisms` to the `Mixture` modeling the degradation of any species with the attribute "tagA" and "tagB" by RNase A and RNase B, respectively.

```
#Create an RNA species with degradation tag sequence tagB
TA = Species("A", attributes=["tagB"], material_type="rna")
#Create homodimer subunit A
A = Species("A", material_type="protein")
#RNase A is a homodimer made up of two identical subunits
RNaseA = ChemicalComplex([A]*2)
#create a DNAassembly that produces A
```

```
assemblyA = DNAassembly(name="A", promoter="strong", transcript=TA,
    rbs="medium", protein=A, initial_concentration=1)

#Same as above but for Species B
TB = Species("B", attributes=["tagA"], material_type="rna")
B = Species("B", material_type="protein")
RNaseB = ChemicalComplex([B]*2)
assemblyB = DNAassembly(name="B", promoter="strong", transcript=TB,
    rbs="medium", protein=B, initial_concentration=1)
#add all the Components to a Mixture
mixture = TxTlDilutionMixture("e coli", components=[assemblyA, assemblyB,
    RNaseA, RNaseB], parameter_file="default_parameters.txt")
# Deg_Tagged_Degredation Mechanism makes RNaseA degrade anything with
    attribute "tagA"
mixture.add_mechanism(Deg_Tagged_Degredation(mechanism_type="tagA_degredation",
    deg_tag="tagA", protease=RNaseA.get_species()))
# Deg_Tagged_Degredation Mechanism makes RNaseB degrade anything with
    attribute "tagB"
mixture.add_mechanism(Deg_Tagged_Degredation(mechanism_type="tagB_degredation",
    deg_tag="tagB", protease=RNaseB.get_species()))
#Compile the CRN
CRN = mixture.compile_crn()
```

Multiple Ribosome Occupancy Repressilator Dynamics

Simulating multiple-ribosome occupancy in the Repressilator mostly reuses the code from Section 2.8 with the main addition of a new `Mechanism` to model transcription being placed into the `Mixture`.

```
#Create Repressors
repA = Species("A")
repB = Species("B")
repC = Species("C")
#Create Promoters
promA = RepressiblePromoter(name="pA", repressor=repC)
promB = RepressiblePromoter(name="pB", repressor=repA)
promC = RepressiblePromoter(name="pC", repressor=repB)
#Create DNAassemblies
assemblyA = DNAassembly(name="A", promoter=promA, rbs="strong",
    protein=repA, initial_concentration=1)
assemblyB = DNAassembly(name="B", promoter=promB, rbs="strong",
    protein=repB, initial_concentration=1)
assemblyC = DNAassembly(name="C", promoter=promC, rbs="strong",
```

```
    protein=repC, initial_concentration=1)
#Extra parameters for the Multi_tx Mechanism
extra_params = {"max_occ":10, ("multi_tx", None, "k_iso"):50, ("multi_tl",
    None, "k_iso"):50, "cooperativity":2}
#Add Everything to a Mixture
mixture = TxTlDilutionMixture("e coli", components=[assemblyA, assemblyB,
    assemblyC],
                          parameter_file="default_parameters.txt",
                              parameters=extra_params,
                              overwrite_parameters=True)
#Add the multi_tl translation mechanism to the mixture, overwriting the
    old one.
mixture.add_mechanism(multi_tl(name="multi_tl",
    ribosome=mixture.ribosome.get_species()), overwrite=True)
#Compile the CRN
crn = mixture.compile_crn()
```

2.9 Supplemental: Tables of Features

This section lists many of the different `Mixture`, `Component`, and `Mechanism` classes available in BioCRNpyler. For more details about these classes and examples using many of them, check out the Examples folder on GitHub.

Mixtures

Mixture Name	Description
ExpressionExtract	A model for gene expression without machinery such as ribosomes, polymerases, etc. Here transcription and translation are lumped into one reaction: expression.
SimpleTxTlExtract	A model for transcription and translation in a cell-free extract without machinery such as ribosomes, polymerases, etc. RNA is degraded via a global mechanism.
TxTlExtract	A model for transcription and translation in a cell-free extract with machinery for ribosomes, polymerases, and endonucleases action. This model does not include any energy buffer.
EnergyTxTlExtract	Transcription and translation with ribosomes, polymerases, and endonucleases labelled as cellular machinery. Also includes a simple model of biochemical energy utilization involving NTPs, amino acids, and energy regeneration from a food source. Adapted from the model in [186].
ExpressionDilutionMixture	A model for *in-vivo* gene expression without any machinery such as ribosomes, polymerases, etc. Transcription and translation are lumped into one reaction and a global mechanism is used to dilute all non-DNA species.
SimpleTxTlDilutionMixture	Mixture with continuous dilution for non-DNA species. Transcription (TX) and Translation (TL) are both modeled as catalytic with no cellular machinery. mRNA is also degraded via a separate reaction to represent endonucleases.
TxTlDilutionMixture	Transcription and translation with ribosomes, polymerases, and endonucleases labelled as cellular machinery. Also includes a background load which represents innate loading effects in the cell. Effects of loading on cell growth are not modeled. It has global dilution for non-DNA and non-machinery species. This model does not include any energy.

Components

Component Type	Component Name	Description
Chemical Complex	ChemicalComplex	A complex that represents the combination of several `Species`. Takes care of binding and unbinding reactions needed to form the complex.
Chemical Complex	CombinatorialComplex	A complex that represents the combination of several `Species`. Binding occurs combinatorially in many possible orders. Allowed and disallowed intermediate complexes can be specified.
Enzyme	Enzyme	An enzyme that converts substrates to products.
Protein	Protein	Basic component that represents a protein.
DNA	DNA	Basic component that represents a DNA sequence.
DNA	DNAassembly	A relatively simple DNA sequence containing one promoter, RBS, and a product.
DNA	DNA_construct	A more complex DNA sequence that can have any number of `Components` in any order.
Promoter	Promoter	Constitutive $\sigma 70$ promoter.
Promoter	RegulatedPromoter	Repressible or activatable promoter such as P_{tet}.
Promoter	ActivatablePromoter	Activatable promoter using a positive Hill function.
Promoter	RepressiblePromoter	Repressible promoter using a negative Hill function.
Promoter	CombinatorialPromoter	Flexible promoter mechanism allowing various transcription factor binding configurations to allow or prevent transcription.
Promoter	CombinatorialConformationPromoter	Enumerates the binding and unbinding events to represent a promoter with many regulators that can also form various secondary structures such as loops.
Ribosome Binding Site	RBS	Simple RBS using a translation mechanism.
Coding Sequence	CDS	Protein coding part used for `DNA_construct`. Doesn't affect CRN.
Terminator	Terminator	Transcriptional terminator used for `DNA_construct`. Doesn't affect CRN.
RNA	RNA	Basic component that represents an RNA sequence.
RNA	RNA_construct	A more complex RNA sequence that can have any number of `Components` in any order. Usually automatically generated by `DNA_construct`.
Polymer Secondary Structure	CombinatorialConformation	Enumerates the binding and unbinding events to produce a `PolymerConformation` with different bound complexes and secondary structure.

Mechanisms

Mechanisms Type	Mechanism Name	Description
binding	Reversible_Bimolecular_Binding	$S_1 + S_2 \leftrightharpoons (S_1 : S_2)$
cooperative_binding	One_Step_Cooperative_Binding	$n\,S_1 + S_2 \leftrightharpoons (nS_1 : S_2)$
cooperative_binding	Two_Step_Cooperative_Binding	$n\,S_1 \leftrightharpoons (nS_1)$, $(nS_1) + S_2 \leftrightharpoons (nS_1 : S_2)$
cooperative_binding	Combinatorial_Cooperative_Binding	Allows a set of species S_i and cooperativities n_i to bind to a target T in any order to form $(n_1S_1 : ... : n_kS_k : T$ along with all combinatorial intermediaries.
binding	One_Step_Binding	$S1 + S2...SN \leftrightharpoons S1 : S2 : ... : SN$
catalysis	BasicCatalysis	$S + C \rightarrow P + C$
catalysis	BasicProduction	$C \rightarrow P + C$
catalysis	MichaelisMenten	$Sub + Enz \leftrightharpoons Sub : Enz \rightarrow Enz + Prod$
catalysis	MichaelisMentenReversible	$Sub + Enz \leftrightharpoons Sub : Enz \leftrightharpoons Enz : Prod \leftrightharpoons Enz + Prod$
copy	MichaelisMentenCopy	$Sub + Enz \leftrightharpoons Sub : Enz \rightarrow Sub + Enz + Prod$
transcription	OneStepGeneExpression	$G \rightarrow G + P$
transcription	SimpleTranscription	$G \rightarrow G + T$
translation	SimpleTranslation	$T \rightarrow T + P$
transcription	PositiveHillTranscription	$G \rightarrow [r]G + P\ r = kG(R^n)/(K + R^n)$
transcription	NegativeHillTranscription	$G \rightarrow [r]G + P\ r = kG/(K + R^n)$
transcription	Transcription_MM	$G + RNAP \leftrightharpoons G : RNAP \rightarrow G + RNAP + mRNA$
translation	Translation_MM	$mRNA + Rib \leftrightharpoons mRNA : Rib \rightarrow mRNA + Rib + Protein$
transcription	multi_tx	$DNA : RNAp_n + RNAp \leftrightharpoons DNA : RNAp_n^{closed} \rightarrow DNA : RNAp_{n+1} DNA : RNAp_n \rightarrow DNA : RNAp_0 + nRNAp + nmRNA DNA : RNAp_n^{closed} \rightarrow DNA : RNAp_0^{closed} + nRNAp + nmRNA$ for $n = \{0, \max_{occ}\}$
translation	multi_tl	$mRNA : RBZ_n + RBZ \leftrightharpoons mRNA : RBZ_n^{closed} \rightarrow mRNA : RBZ_{n+1} mRNA : RBZ_n \rightarrow mRNA : RBZ_0 + nRBZ + nProtein mRNA : RBZ_n^{closed} \rightarrow mRNA : RBZ_0^{closed} + nRBZ + nProtein$ for $n = \{0, \max_{occ}\}$
dilution	Dilution	$s \rightarrow \emptyset$
rna_degredation_mm	Degredation_mRNA_MM	$T + Nuclease \leftrightharpoons T : Nuclease \rightarrow Nuclease.$ Global mechanism effects all RNA species T. ComplexSpecies containing RNA species are broken apart via the reaction $T : X + Nuclease \leftrightharpoons T : X : Nuclease \rightarrow X + Nuclease.$ for any X.
degredation	Deg_Tagged_Degredation	$X + Protease \leftrightharpoons X : Protease \rightarrow Protease.$ Here X is any Species with the deg_tag attribute passed into the constructor of this GlobalMechanism.

2.10 Supplemental: Creating Custom BioCRNpyler Classes

BioCRNpyler is designed to be easily extendable so even non-computer scientists can add their own custom functionality. In this section, we briefly show how to subclass core BioCRNpyler classes. For more details and examples, interested readers should look at the Developer Overview on our Github.

Mechanisms

Developing custom `Mechanisms` is also as easy as making a subclass of `Mechanism` and defining three functions to produce the desired CRN:

```
class CustomMechanism(Mechanism):
    def __init__(self, args, **kwargs ):
        Mechanism.__init__(self, name="name", mechanism_type="type",
            **kwargs)
        # python code to set internal variables

    def update_species(self, ... ):
        # python code to create Species objects
        return species_list

    def update_reactions(self, ... ):
        # python code to create Reaction objects
        return reaction_list
```

Components

It is also straightforward to make custom `Components`: simply subclass `Component` and define three functions:

```
class CustomComponent(Component):
    def __init__(self, args, **kwargs ):
        Component.__init__(self, ... , **kwargs)
        # python code to set internal variables

    def update_species(self):
        # python code calls mechanism.update_species( ... )
        return species_list

    def update_reactions(self):
        # python code calls mechanism.update_reaction( ... )
        return reaction_list
```

Mixtures

Making custom `Mixtures` is also easy—it can be done via simple scripts by adding `Components` and `Mechanisms` to a `Mixture` object:

```
MyMixture=Mixture("customized mixture",
                  components=[List Components],
```

```
                mechanisms=dict("mechanism_type":Mechanism))
```

The `Mixture` class can also be easily subclassed by rewriting the constructor:

```
class CustomMixture(Mixture):
    def __init__(self, args, **kwargs ):
        #python code to set up internal variables,
        # create Components, and default Mechanisms
        Mixture.__init__(self, mechanisms=dict("mechanism_type": Mechanism),
                        components=[List of Components], **kwargs)
```

Chapter 3

CHEMICAL BOLTZMANN MACHINES

[1] W. Poole, A. Ortiz-Munoz, A. Behera, N. S. Jones, T. E. Ouldridge, E. Winfree, and M. Gopalkrishnan, "Chemical Boltzmann Machines," in *International Conference on DNA-Based Computers*, Springer, 2017, pp. 210–231. DOI: 10.1007/978-3-319-66799-7_14,

3.1 Forward

The following chapter is quoted directly from the conference paper *Chemical Boltzmann Machines* presented at the International Conference on DNA Computing and Molecular Programming in 2017 [187]. This was joint work with Andrés Ortiz-Muñoz, Abhishek Behera, Nick S. Jones, Thomas E. Ouldridge, Erik Winfree, and Manoj Gopalkrishnan.

Chemical Boltzmann Machines provides three CRN implementations of a particularly famous probabilistic graphical model called a Boltzmann Machine (BM). The *Direct Chemical Boltzmann Machine* (DCBM) is a non-detailed balanced CRN so-named because it directly recreates the Markov chain underlying a BM and can be easily generalized to implement any Markov chain on the integer lattice. The detailed balanced *Edge-Species Chemical Boltzmann Machine* (ECBM) produces an equilibrium distribution identical to a BM, but may have different dynamics. Similar models and their generalizations are a major element of Chapter 4. Finally, the *Taylor Chemical Boltzmann Machine* (TCBM) approximates the dynamics of a BM using reactions inspired by gene regulatory networks and protein phosphorylation networks. Unlike the DCBM and ECBM, the TCBM is very compact requiring relatively few reactions, which makes it easier to simulate.

These models show how diverse CRNs are capable of implementing probabilistic algorithms and suggest that these kinds of programs are *natural* to encode as CRNs. The paper goes on to derive an *in silico* learning algorithm for the ECBM. Similarly, the DCBM and TCBM can be seen as exact and approximate molecular implementations of BMs which could be trained *in-silico* before being translated into these CRNs. These results are a step in the direction of automated design of probabilistic chemical programs and rely on methods borrowed directly from machine learning.

Contribution: Andres Ortiz-Muñoz, Abhishek Behera, and I are all listed as co-first authors on this work and indeed we each contributed one of the three models presented. Andres Andrés Ortiz-Muño was responsible for the ECBM, Abhishek Behera and Manoj Gopalkrishnan developed DCBM, and I developed the TCBM and carried out the extensive simulations using this model. All theorems and proofs were a collaborative effort over approximately a year of meetings. Although I wrote the initial draft of the paper, it was heavily revised and improved by all co-authors.

3.2 Abstract

How smart can a micron-sized bag of chemicals be? How can an artificial or real cell make inferences about its environment? From which kinds of probability distributions can chemical reaction networks sample? We begin tackling these questions by showing three ways in which a stochastic chemical reaction network can implement a Boltzmann machine, a stochastic neural network model that can generate a wide range of probability distributions and compute conditional probabilities. The resulting models, and the associated theorems, provide a road map for constructing chemical reaction networks that exploit their native stochasticity as a computational resource. Finally, to show the potential of our models, we simulate a chemical Boltzmann machine to classify and generate MNIST digits in-silico.

3.3 Introduction

To carry out complex tasks such as finding and exploiting food sources, avoiding toxins and predators, and transitioning through critical life-cycle stages, single-celled organisms and future cell-like artificial systems must make sensible decisions based on information about their environment [188, 189]. The small volumes of cells makes this enterprise inherently probabilistic: environmental signals and the biochemical networks within the cell are noisy, due to the stochasticity inherent in the interactions of small, diffusing molecules [119, 190, 191]. The small volumes of cells also raises questions not only about how stochasticity influences circuit function, but also about how much computational sophistication can be packed into the limited available space.

Perhaps surprisingly, neural network models provide an attractive architecture for the types of computation, inference, and information processing that cells must do. Neural networks can perform deterministic computation using circuits that are smaller and faster than boolean circuits composed of AND, OR, and NOT gates [192], can robustly perform tasks such as associative recall [53], and can naturally perform

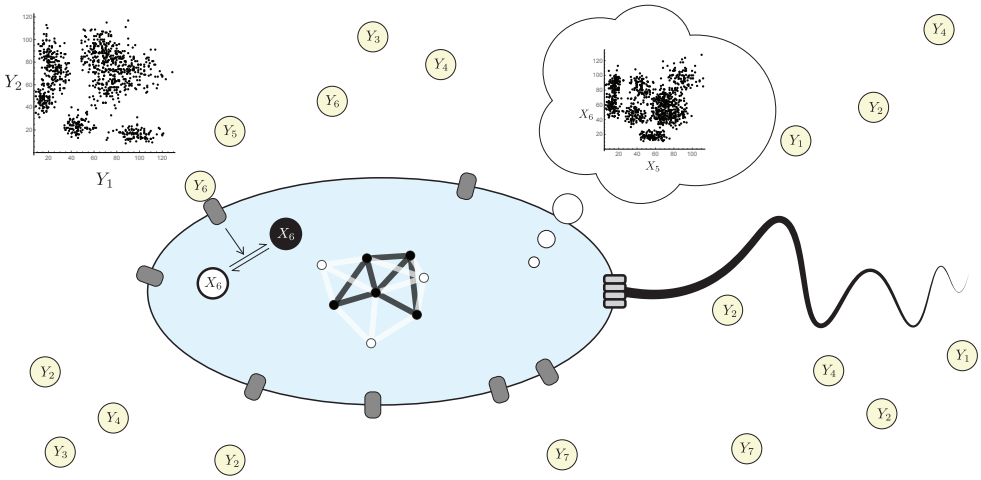


Figure 3.1: In a micron-scale environment, molecular counts are low and a real (or synthetic) cell will have to respond to internal and environmental cues. Probabilistic inference using chemical Boltzmann machines provides a framework for how this may be achieved.

Bayesian inference [85]. Furthermore, the structure of biochemical networks, such as signal transduction cascades [70, 188, 193] and genetic regulatory networks [78, 80, 194–196], can map surprisingly well onto neural network architectures. Chemical implementations of neural networks and related machine learning models have also been proposed [55, 56, 60, 77, 82], and limited examples demonstrated [30, 57, 63, 197], for synthetic biochemical systems.

Most previous work on biochemical neural networks and biochemical inference invoked models based on continuous concentrations of species representing neural activities. Such models are limited in their ability to address questions of biochemical computation in small volumes, where discrete stochastic chemical reaction network models must be used to account for the low molecular counts. The nature of biochemical computation changes qualitatively in this context. In particular, stochasticity has been widely studied in genetic regulatory networks [198], signaling cascades [199], population level bet hedging in bacteria [200], and other areas [201, 202]—where the stochasticity is usually seen as a challenge limiting correct function, but is occasionally also viewed as a useful resource [125]. Our work falls squarely in the latter camp: we attempt to exploit the intrinsic stochastic fluctuations of a formal chemical reaction network (CRN) to build natively stochastic samplers by implementing a stochastic neural network. This links to efforts to build natively stochastic hardware for Bayesian inference [203, 204] and to the substantial litera-

ture attempting to model, and find evidence for, stochastic neural systems capable of Bayesian inference [205, 206].

Specifically, we propose CRNs that implement Boltzmann machines (BMs), a flexible class of Markov random fields capable of generating diverse distributions and for which conditioning on data has straightforward physical interpretations [85, 207]. BMs are an established model of probabilistic neural networks due to their analytic tractability and connections to spin systems in statistical physics [208] and Hopfield networks in computer science [53]. These networks have been studied extensively and used in a wide range of applications including image classification [209] and video generation [210]. We prove that CRNs can implement BMs and that this is possible using detailed balanced CRNs. Moreover, we show that many of the attractive features of BMs can be applied to our CRN constructions such as inference, a straightforward learning rule and scalability to real-world data sets. We thereby introduce the idea of a chemical Boltzmann machine (CBM), a chemical system capable of exactly or approximately performing inference using a stochastically sampled high-dimensional state space, and explore some of its possible forms.

3.4 Relevant Background

Boltzmann Machines (BMs):

Boltzmann machines are a class of binary stochastic neural networks, meaning that each node randomly switches between the values 0 and 1 according to a specified distribution. They are widely used for unsupervised machine learning because they can compactly represent and manipulate high-dimensional probability distributions. Boltzmann machines provide a flexible machine learning architecture because, as generative models, they can be used for a diverse set of tasks including data classification, data generation, and data reconstruction. Additionally, the simplicity of the model makes them analytically tractable. The use of hidden units (described below) allows Boltzmann machines to represent high order correlations in data. Together, these features make Boltzmann machines an excellent starting point for implementing stochastic chemical computers.

Fix a positive integer $N \in \mathbb{Z}_{>0}$. An N-node **Boltzmann machine** (BM) is specified by a quadratic **energy** function $E : \{0,1\}^N \to \mathbb{R}$

$$E(x_1, x_2, \ldots, x_N) = -\sum_{i<j} w_{ij} x_i x_j - \sum_i \theta_i x_i \tag{3.1}$$

where $\theta_i \in \mathbb{R}$ is the **bias** of node i, and $w_{ij} = w_{ji} \in \mathbb{R}$ is the **weight** of the unordered pair (i, j) of nodes, with $w_{ii} = 0$. One may specify a BM **architecture**, or graph

topology, by choosing additional weights w_{ij} that are to be set to 0. In this paper, we will use $\mathcal{N}(i) = \{j \text{ s.t. } w_{ij} \neq 0\}$ to denote the neighborhood of i. From a physical point of view, we are implicitly using *temperature units* $k_B T$ for energy, which we will continue to do throughout this paper. A BM describes a distribution $P(x)$ over **state vectors** $x = (x_1, \ldots, x_N) \in \{0,1\}^N$,

$$P(x) = \frac{1}{Z} e^{-E(x)} \quad \text{with} \quad Z = \sum_{x' \in \{0,1\}^N} e^{-E(x')}. \tag{3.2}$$

Nodes of a BM are often partitioned into sets V and H of **visible** and **hidden**, respectively. Nodes in V represent data, and auxiliary nodes in H allow more complex distributions to be represented in the visible nodes. An **implementation** of a BM is a stationary stochastic process that samples from this distribution in the steady state. A BM can be implemented *in silico* using the **Gibbs sampling** algorithm [211], which induces a discrete time Markov chain (DTMC) on the state space $\{0,1\}^N$ in such a way that the stationary distribution of this Markov chain corresponds to the distribution $P(x)$. In each round, one node $i \in \{1, \ldots, N\}$ is chosen at random for update. For any two adjacent configurations x and x' which differ only at node i — i.e., $x_i = 1 - x'_i$ and $x_j = x'_j$ for all $j \neq i$ — we set the transition probabilities $T_{x \to x'}$ of the DTMC so that

$$\frac{T_{x' \to x}}{T_{x \to x'}} = \frac{P(x)}{P(x')} = \frac{\mathrm{e}^{-E(x)}}{\mathrm{e}^{-E(x')}} = \mathrm{e}^{\left(\theta_i + \sum_{j \in \mathcal{N}(i)} w_{ij} x_j\right)(x_i - x'_i)}. \tag{3.3}$$

Any function $T_{x \to x'}$ can be chosen so long as (3.3) is satisfied. One common choice is $T_{x \to x'} = 1/(N(1 + \mathrm{e}^{E(x') - E(x)}))$, where the factor $1/N$ represents the probability of choosing node i.

A Boltzmann machine is also an inference engine. One can do inference on $P(x)$ by conditioning on the values of a subset of the nodes. Suppose nonempty node subsets U and Y form a partition of the nodes $\{1, 2, \ldots, N\}$, and fix $u \in \{0,1\}^U$. To obtain samples from $P(y \mid u)$ where $y \in \{0,1\}^Y$, one **clamps** every node $i \in U$ to the state u_i while running Gibbs sampling, i.e., one does not allow these nodes to update. Clamping nodes to an input state is the same as specifying the input data for a statistical model. Steady state samples $y \in \{0,1\}^Y$ of this procedure are draws from the distribution $P(y \mid u)$.

Boltzmann machines can be used to learn a generative model from unlabeled data. After specifying the architecture, one then proceeds to find the weights, w_{ij}, and biases, θ_i, that maximize the likelihood of the observed data according to the model,

using gradient descent from a random initial parameterization. This reduces to a very simple two-phase Hebbian learning rule where weights on active edges are strengthened in a "wake phase" during which the network is clamped to observed data and are weakened in a "sleep phase" during which the network runs free [85, 207]. Given a target distribution $Q(x)$, this gradient descent corresponds to calculating the gradient of the Kullback-Leibler divergence from P to Q, $D_{KL}(Q \,||\, P) = \sum_x Q(x) \log \frac{Q(x)}{P(x)}$, with respect to the parameters θ_i and w_{ij}:

$$\frac{d\theta_i}{dt} = -\frac{\partial D_{KL}}{\partial \theta_i} = \langle x_i \rangle_Q - \langle x_i \rangle_P \quad \text{and} \quad \frac{dw_{ij}}{dt} = -\frac{\partial D_{KL}}{\partial w_{ij}} = \langle x_i x_j \rangle_Q - \langle x_i x_j \rangle_P \tag{3.4}$$

where $\langle \cdot \rangle_P$ and $\langle \cdot \rangle_Q$ denote expected values with respect to the distributions P and Q respectively. When hidden units are present, the distribution Q (which is defined on visible units only) is extended to hidden units based on clamping the visible units according to Q and using the conditional distribution $P(y|u)$ to determine the hidden units.

Chemical Reaction Networks (CRNs):
Fix a finite set $\mathcal{S} = (S_1, S_2, \ldots, S_M)$ of M **species**. A **reaction** r is a formal chemical equation

$$\sum_{i=1}^{M} \mu_r^i S_i \to \sum_{i=1}^{M} \nu_r^i S_i, \tag{3.5}$$

abbreviated as $r = \mu_r \to \nu_r$ where $\mu_r, \nu_r \in \mathbb{N}^{\mathcal{S}}$ are the stoichiometric coefficient vectors for the reactant and product species respectively, and $\mathbb{N} = \mathbb{Z}_{\geq 0}$. A **reaction rate constant**, $k_r \in \mathbb{R}_{>0}$, is associated with each reaction. In this paper, we define a **chemical reaction network** (CRN) as a triple $C = (\mathcal{S}, \mathcal{R}, k)$ where $\mathcal{S}$ is a finite set of species, and $\mathcal{R}$ is a set of reactions, and k is the associated set of reaction rate constants.

We will denote chemical species by capital letters, and their counts by lower case letters; e.g., s_1 denotes the number of species S_1. Thus the state of a stochastic CRN is described by a vector on a discrete lattice, $s = (s_1, s_2 \ldots s_M) \in \mathbb{N}^{\mathcal{S}}$. The dynamics of a stochastic CRN are as follows [167]. The probability that a given reaction occurs per unit time, called its **propensity**, is given by

$$\rho_r(s) = k_r \prod_{i=1}^{M} \frac{s_i!}{(s_i - \mu_r^i)!} \quad \text{if} \quad s_i \geq \mu_r^i \quad \text{and 0 otherwise.} \tag{3.6}$$

Each time a reaction fires, state s changes to state $s + \Delta_r$, where $\Delta_r = \nu_r - \mu_r$ is called the reaction vector, and the propensity of each reaction may change. Viewed

from a state space perspective, stochastic CRNs are continuous time Markov chains (CTMCs) with transition rates

$$R_{s\to s'} = \sum_{r \text{ s.t. } s'=s+\Delta_r} \rho_r(s) \tag{3.7}$$

and thus their dynamics follow

$$\frac{dP(s,t)}{dt} = \sum_{s'\neq s} R_{s'\to s}P(s',t) - R_{s\to s'}P(s,t) \ , \tag{3.8}$$

where $P(s,t)$ is the probability of a state with counts s at time t. Equivalently, they are governed by the **chemical master equation**,

$$\frac{dP(s,t)}{dt} = \sum_{r\in\mathcal{R}} P(s-\Delta_r,t)\rho_r(s-\Delta_r) - P(s,t)\rho_r(s) \ . \tag{3.9}$$

A stationary distribution $\pi(s)$ may be found by solving $\frac{dP(s,t)}{dt} = 0$ simultaneously for all s; in general, it need not be unique, and even may not exist. Given an initial state s_0, $\pi(s) = P(s,\infty)$ is unique if it exists. For that initial state, the **reachability class** $\Omega_{s_0} \subseteq \mathbb{N}^M$ is the maximal subset of the integer lattice accessible to the CRN via some sequence of reactions in $\mathcal{R}$. We will specify a CRN and a reachability class given an initial state as a shorthand for specifying a CRN and a set of initial states with identical reachability classes.

Detailed Balanced Chemical Reaction Networks:
A CTMC is said to satisfy **detailed balance** if there exists a well-defined function of state s, $E(s) \in \mathbb{R}$, such that for every pair of states s and s', the transition rates $R_{s\to s'}$ and $R_{s'\to s}$ are either both zero or

$$\frac{R_{s\to s'}}{R_{s'\to s}} = \mathrm{e}^{E(s)-E(s')} \ . \tag{3.10}$$

If the state space Ω is connected and the partition function $Z = \sum_{s\in\Omega} \mathrm{e}^{-E(s)}$ is finite, then the steady state distribution $\pi(s) = \frac{1}{Z}\mathrm{e}^{-E(s)}$ is unique, and the net flux between all states is zero in that steady state.

There is a related but distinct notion of detailed balance for a CRN. An equilibrium chemical system is constrained by physics to obey detailed balance at the level of each reaction. In particular, for a dilute equilibrium system, each species $S_i \in \mathcal{S}$ has an energy $G[S_i] \in \mathbb{R}$ that relates to its intrinsic stability, and

$$\frac{k_{r^+}}{k_{r^-}} = \mathrm{e}^{-\sum_{i=1}^{M}\Delta^i_{r^+}G[S_i]} = \mathrm{e}^{-\Delta G_{r^+}}, \tag{3.11}$$

where $\Delta^i_{r^+}$ is the ith component of $\Delta_{r^+} = \nu_{r^+} - \mu_{r^+}$, and we have defined $\Delta G_{r^+} = \sum_{i=1}^N \Delta^i_{r^+} G[S_i]$. Any CRN for which there exists a function G satisfying (3.11) is called a detailed balanced CRN. To see that the CTMC for a detailed balanced CRN also itself satisfies detailed balance in the sense of (3.10), let $s' = s + \Delta_{r+}$ and note that (3.6) and (3.11) imply that

$$\frac{\rho_{r^+}(s)}{\rho_{r^-}(s')} = \mathrm{e}^{\mathcal{G}(s)-\mathcal{G}(s')} \quad \text{with} \quad \mathcal{G}(s) = \sum_{i=1}^{M} s_i G[S_i] + \log(s_i!), \tag{3.12}$$

for all reactions r^+. Here, $\mathcal{G}(s)$ is a well-defined function of state s (the free energy) that can play the role of E in (3.10). If there are multiple reactions that bring s to s', they all satisfy (3.12), and therefore the ratio $R_{s\to s'}/R_{s'\to s}$ satisfies (3.10) and the CTMC satisfies detailed balance.

It is possible to consider non-equilibrium CRNs that violate (3.11). Such systems must be coupled to implicit reservoirs of fuel molecules that can drive the species of interest into a non-equilibrium steady state [212–214]. Usually – but not always [215, 216] – the resultant Markov chain violates detailed balance. In Section 3.5, we shall consider a system that exhibits detailed balance at the level of the Markov chain, but is necessarily non-equilibrium and violates detailed balance at the detailed chemical level.

Given an initial condition s_0, a detailed balanced CRN will be confined to a single reachability class Ω_{s_0}. Moreover, from the form of $\mathcal{G}(s)$, the stationary distribution $\pi(s)$ on Ω_{s_0} of any detailed balanced CRN exists, is unique, and is a product of Poisson distributions restricted to the reachability class [93],

$$\pi(s) = \frac{1}{Z}\mathrm{e}^{-\mathcal{G}(s)} = \frac{1}{Z}\prod_{i=1}^{M} \frac{\mathrm{e}^{-s_i G[S_i]}}{s_i!}, \tag{3.13}$$

with the partition function $Z = \sum_{s'\in\Omega_{s_0}} \mathrm{e}^{-\mathcal{G}(s')}$ dependent on the reachability class. Note that this implies that the partition function is always finite, even for an infinite reachability class.

3.5 Exact Constructions and Theorems

Clamping and Conditioning with Detailed Balanced CRNs:

In a Boltzmann machine that has been trained to generate a desired probability distribution when run, inference can be performed by freezing, also known as clamping, the values of known variables, and running the rest of the network to obtain a sample; this turns out to exactly generate the conditional probability.

A similar result holds for a subclass of detailed balanced CRNs that generate a distribution, for an appropriate notion of clamping in a CRN. Imagine a "demon" that, whenever a reaction results in a change in the counts of one of the clamped species, will instantaneously change it back to its previous value. If every reaction is such that either no clamped species change, or else every species that changes is clamped, then the demon is effectively simply "turning off" those reactions. More precisely, consider a CRN, $C = (\mathcal{S}, \mathcal{R}, k)$. We will partition the species into two disjoint groups $Y = \mathcal{S}_{free}$ and $U = \mathcal{S}_{clamped}$, where $\mathcal{S}_{free}$ will be allowed to vary and $\mathcal{S}_{clamped}$ will be held fixed. We will define free reactions, $\mathcal{R}_{free}$, as reactions which result in neither a net production nor a net consumption of any clamped species. Similarly, clamped reactions, $\mathcal{R}_{clamped}$ are reactions which change the counts of any clamped species. The clamped CRN will be denoted $C|_{U=u}$ to indicate the the species $U_i \in U$ have been clamped to the values u_i. The clamped CRN is defined by $C|_{U=u} = (\mathcal{S}, \mathcal{R}_{free}, k_{free})$, that is, the entire set of species along with the reduced set of reactions and their rate constants. In the clamped CRN it is apparent that the clamped species will not change from their initial conditions because no reaction in $\mathcal{R}_{free}$ can change their count. However, these clamped species may still affect the free species catalytically. If the removed reactions, $\mathcal{R}_{clamped}$, never change counts of non-clamped species, then $C|_{U=u}$ is equivalent to the action of the "demon" imagined above.

We use equation 3.13 to prove that clamping a detailed balanced CRN is equivalent to calculating a conditional distribution, and to show when the conditional distributions of a detailed balanced CRN will be independent. Together, these theorems provide guidelines for devising detailed balanced CRNs with interesting (non-independent) conditional distributions and for obtaining samples from these distributions via clamping.

We will need one more definition. Let C be a detailed balanced CRN with reachability class Ω_{s_0} for some initial condition $s_0 = (u_0, y_0)$. Let Γ_{s_0} be the reachability class of the clamped CRN $C|_{U=u_0}$ with species U clamped to u_0 and species Y free. We say clamping **preserves reachability** if $\Omega^Y_{s_0|U=u_0} = \Gamma^Y_{s_0}$ where $\Omega^Y_{s_0|U=u_0} = \{y \text{ s.t. } (u_0, y) \in \Omega_{s_0}\}$ and $\Gamma^Y_{s_0} = \{y \text{ s.t. } (u_0, y) \in \Gamma_{s_0}\}$. In other words, clamping preserves reachability if, whenever a state $s = (u_0, y)$ is reachable from s_0 by any path in C, then it is also reachable from s_0 by some path in $C|_{U=u_0}$ that never changes u.

Theorem: Consider a detailed balanced CRN $C = (\mathcal{S}, \mathcal{R}, k)$ with reachability

class Ω_{s_0} from initial state s_0. Partition the species into two disjoint sets $U = \{U_1, \ldots, U_{M_u}\} \subset \mathcal{S}$ and $Y = \{Y_1, \ldots, Y_{M_y}\} \subset \mathcal{S}$ with $M_u + M_y = M = |\mathcal{S}|$. Let the projection of s_0 onto U and Y be u_0 and y_0. The conditional distribution $P(y \mid u)$ implied by the stationary distribution π of C is equivalent to the stationary distribution of a clamped CRN, $C|_{U=u}$ starting from initial state s_0 with $u_0 = u$, provided that clamping preserves reachability.

Proof: We have $\mathcal{G}(u, y) = \sum_{i=1}^{M_u} u_i G[U_i] + \log(u_i!) + \sum_{i=1}^{M_y} y_i G[Y_i] + \log(y_i!)$. Let the reachability class of $C|_{U=u}$ be Γ_{s_0}, its projection onto Y be $\Gamma^Y_{s_0}$, and $\Omega^Y_{s_0|U=u_0} = \{y \text{ s.t. } (u_0, y) \in \Omega_{s_0}\}$ with $\Omega^Y_{s_0|U=u_0} = \Gamma^Y_{s_0}$. Then, the conditional probability distribution of the unclamped CRN is given by

$$P(y \mid u) = \frac{\pi(u, y)}{\sum_{y' \in \Gamma^Y_{s_0}} \pi(u, y')} = \frac{\mathrm{e}^{-\mathcal{G}(u,y)}}{\sum_{y' \in \Gamma^Y_{s_0}} \mathrm{e}^{-\mathcal{G}(u,y')}} \,. \tag{3.14}$$

Simply removing pairs of forward and backward reactions will preserve detailed balance for unaffected transitions, and hence the clamped system remains a detailed balanced CRN with the same free energy function. We then readily see that the clamped CRN's stationary distribution, $\pi_c(y|u)$ is given by

$$\pi_c(y|u) = \frac{\mathrm{e}^{-\mathcal{G}(u,y)}}{Z_c(u)} \quad \text{with} \quad Z_c(u) = \sum_{y' \in \Gamma^Y_{s_0}} \mathrm{e}^{-\mathcal{G}(u,y')} \,. \tag{3.15}$$

The original CRN and the clamped CRN do not need to have the same initial conditions as long as the initial conditions have the same reachability classes. However, even if the two CRNs have the same initial conditions, it is possible that the clamping process will make some part of $\Omega^Y_{s_0|U=u_0}$ inaccessible to $C|_{U=u}$, in which case this theorem will not hold.

Theorem: Assume the reachability class of a detailed balanced CRN can be expressed as the product of subspaces, $\Omega_{s_0} = \prod_{j=1}^{L} \Omega^j_{s_0}$. Then the steady-state distributions of each subspace will be independent for each product space: $\pi(s) = \prod_{j=1}^{L} \pi^j(s^j)$, where $s = (s^1, \ldots, s^L)$ and π^j is the distribution over $\Omega^j_{s_0}$.

Proof: If Ω_{s_0} is decomposable into a product of subspaces $\Omega^j_{s_0}$, with $j = 1 \ldots L$, then each subspace involves disjoint sets of species $Y^j = \{S^j_1, \ldots, S^j_{M_j}\}$. In this case the steady-state distribution of a detailed balanced CRN can be factorized due to the simple nature of $\mathcal{G}(s)$ given by eq. (3.12):

$$\pi(s) = \frac{\mathrm{e}^{-\mathcal{G}(s)}}{Z} = \frac{\prod_{j=1}^{L} \mathrm{e}^{-\mathcal{G}(s^j)}}{\prod_{j=1}^{L} \sum_{s^{j\prime} \in \Omega^j_{s_0}} \mathrm{e}^{-\mathcal{G}(s^{j\prime})}} = \prod_{j=1}^{L} \frac{\mathrm{e}^{-\mathcal{G}(s^j)}}{\sum_{s^{j\prime} \in \Omega^j_{s_0}} \mathrm{e}^{-\mathcal{G}(s^{j\prime})}} = \prod_{j=1}^{L} \frac{\mathrm{e}^{-\mathcal{G}(s^j)}}{Z^j} \,, \tag{3.16}$$

where $s^j = (s_1^j, s_2^j, \ldots, s_{M_j}^j)$ is the state of the set of species within subspace j.

The product form $\pi(s)$ means that species from separate subspaces $\Omega_{s_0}^j$ are statistically independent. To develop non-trivial conditional probabilities for the states of different species, therefore, it is necessary either to use a non-detailed balanced CRN by driving the system out of equilibrium, or to generate complex interdependencies through conservation laws that constrain reachability classes and "entangle" the state spaces for different species. We explore both of these possibilities in the following sections.

Direct Implementation of a Chemical Boltzmann Machine (DCBM):
We first consider the most direct way to implement an N-node Boltzmann machine with a chemical system. Recall that a BM has a state space $\Omega_{BM} = \{0,1\}^N$ and an energy function $E(x_1, x_2, \ldots, x_N) = -\sum_{i<j} w_{ij}x_i x_j - \sum_i \theta_i x_i$. We use a dual rail representation of each node i by two CRN species X_i^{ON} and X_i^{OFF} and reactions that respect a conservation law, $x_i^{ON} + x_i^{OFF} = 1$. The species X_i^{ON} and X_i^{OFF} could represent activation states of an enzyme. The CRN has $M = 2N$ species and states $s = (x_1^{ON}, x_1^{OFF}, \ldots, x_N^{ON}, x_N^{OFF})$. Although there are 2^{2N} states in which each species has a count of at most one, only $1/2^N$ of these states are reachable due to the conservation laws. Let Ω_{DCBM} be the states reachable from a valid initial state. There exists a one-to-one invertible mapping $\mathcal{F} : \Omega_{BM} \to \Omega_{DCBM}$ which maps the states $x \in \Omega_{BM}$ of a BM to states $s = \mathcal{F}(x) \in \Omega_{DCBM}$ of the CBM, according to $x_i^{ON} = x_i$ and $x_i^{OFF} = 1 - x_i$.

Reactions are intended to provide a continuous-time analog of the typical BM implementations, such as the Gibbs sampling method discussed in Section 3.4. In each reaction r, only the species X_i^{ON} and X_i^{OFF}, corresponding to a single node i, change ($\nu_r - \mu_r$ has two non-zero components). To reproduce the stationary distribution of a Boltzmann machine with energy function $E(x)$, it is sufficient to require that the CTMC for the CRN satisfies

$$s \rightleftharpoons s' \quad \text{with} \quad \frac{R_{s\to s'}}{R_{s'\to s}} = \frac{\mathrm{e}^{-E(s')}}{\mathrm{e}^{-E(s)}} = \mathrm{e}^{\theta_i + \sum_{j\in\mathcal{N}(i)} w_{ij}x_j^{ON}} \tag{3.17}$$

where s is any reachable state with $x_i^{OFF} = 1$, and s' has $x_i^{ON} = 1$ but is otherwise the same. Such a choice would enforce detailed balance of the CTMC, with the desired steady-state distribution

$$\pi(s) = \frac{1}{Z}\mathrm{e}^{-E(s)} = \frac{1}{Z}\mathrm{e}^{-\sum_{i<j} w_{ij}x_i^{ON}x_j^{ON} - \sum_i \theta_i x_i^{ON}} \,. \tag{3.18}$$

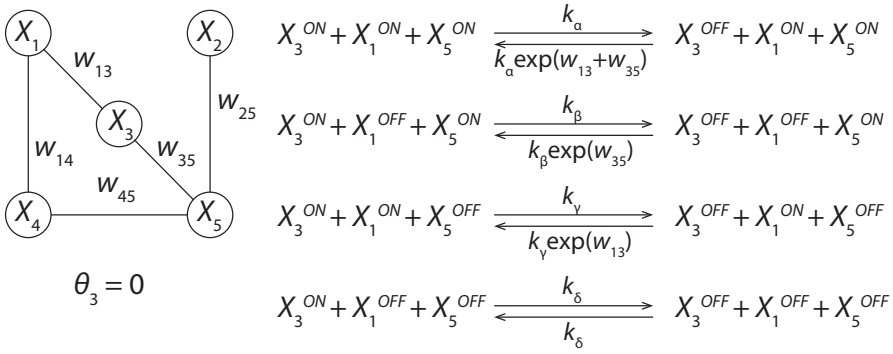


Figure 3.2: The reactions required by the dynamics of a single node using the direct CBM implementation. We consider a simple network with the illustrated topology, and display the required reactions for node 3. Since node 3 has degree 2, there are 4 possible states of its neighbors, and hence four distinct pairs of reactions for the species of node 3. The relative rates of each pair of reactions is set by w_{ij} as indicated (where, for simplicity, we have assumed $\theta_3 = 0$).

To implement such a CRN, we define a reaction set $\mathcal{R}$ that contains a distinct pair of reactions for each possible state of the neighbors of i for which $w_{ij} \neq 0$. Let $\alpha^i \in \{ON, OFF\}^{|\mathcal{N}(i)|}$ denote a state of neighboring species. Then, the necessary reactions and rate constants are

$$X_i^{ON} + \sum_{j \in \mathcal{N}(i)} X_j^{\alpha_j^i} \underset{k_{i^+|\alpha_i}}{\overset{k_{i^-|\alpha_i}}{\rightleftharpoons}} X_i^{OFF} + \sum_{j \in \mathcal{N}(i)} X_j^{\alpha_j^i}, \quad \frac{k_{i^+|\alpha^i}}{k_{i^-|\alpha^i}} = \mathrm{e}^{\theta_i + \sum_{j \in \mathcal{N}(i)} w_{ij} x_j^{ON}}, \quad (3.19)$$

for each i and every possible state α. In physical terms, the species representing the neighbors of node i collectively catalyze $X_i^{OFF} \leftrightharpoons X_i^{ON}$, with a separate pair of reactions for every possible α^i. While this entails a large number of reactions ($2^{|\mathcal{N}(i)|+1}$ for each node i), it allows the rate constants for each configuration of neighbors to be distinct, and thus to satisfy the ratio of rate constants given in (3.19). For CRN states that satisfy the conservation laws $x_i^{ON} + x_i^{OFF} = 1$, there will be a unique reaction that can flip any given bit, and thus the CTMC detailed balance (3.17) also holds, yielding the correct $\pi(s)$. The construction is illustrated by example in Figure 3.2 and compared to other constructions in Figure 3.3.

The distribution $\pi(s)$ is identical to that of the BM, both with and without clamping. Reachability is preserved by clamping, as all states satisfying the conservation laws and clamping can be reached in the clamped CRN. All results derived for traditional BMs therefore apply, including conditional inference and the Hebbian learning rule. The construction can be generalized to any graphical model and indeed to any finite Markov chain defined on a positive integer lattice.

With the DCBM, we have shown that CRNs can express the same distributions as BMs, and are thus very expressive. However, since each possible state α^i of $\mathcal{N}(i)$ is associated with two reactions, the number of reactions of the CRN is exponentially large in the typical node degree d in the original BM. Moreover, the scheme requires high molecularity reactions in which multiple catalysts effect a single transition (the molecularity grows linearly with d). Physical implementations are therefore likely to be challenging. We further note that as a consequence of Theorem 3.5, the DCBM cannot be detailed balanced at the level of the underlying chemistry, due to its simple conservation laws. Physically, this means that the DCBM must use a fuel species to drive each reaction. Details of this argument are given in the appendix (3.9).

The Edge Species CBM Construction (ECBM):
Can a detailed balanced CRN also implement a Boltzmann machine, or is it necessary to break detailed balance at the level of the CRN reactions, as in the DCBM? Here we show that it is not necessary by introducing a detailed balanced CRN that uses species to represent both the nodes and edges of a BM. The N nodes of a BM are converted into N pairs of species, X_i^{ON} and X_i^{OFF}, via a dual rail implementation identical to that used in the DCBM. Similarly, the edges w_{ij} are represented by dual rail edge species W_{ij}^{ON} and W_{ij}^{OFF} with the conservation law $w_{ij}^{ON} + w_{ij}^{OFF} = 1$ for $1 \leq j < i \leq N$. Note that we may slightly abuse notation and let $W_{ij}^{\alpha_{ij}}$ and $W_{ji}^{\alpha_{ij}}$, with $\alpha_{ij} \in \{ON, OFF\}$, represent the same chemical species.

To have detailed balance, we associate energies to each node species determined by the bias in a BM, $G[X_i^{ON}] = -\theta_i$ and $G[X_i^{OFF}] = 0$. Similarly, each edge species has an energy determined by the corresponding edge weight $G[W_{ij}^{ON}] = -w_{ij}$ and $G[W_{ij}^{OFF}] = 0$. Finally, we define a set of catalytic reactions that ensure that the states of edge and node species are consistent, meaning $w_{ij}^{ON} = 1$ if and only if $x_i^{ON} = 1$ and $x_j^{ON} = 1$. To achieve this coupling, the reactions that switch node i are always catalyzed by the species corresponding to the set of neighboring nodes $\mathcal{N}(i)$. Simultaneously, these reactions switch edge ij if $j \in \mathcal{N}(i)$ and $x_j^{ON} = 1$, maintaining $x_i^{ON} x_j^{ON} = w_{ij}^{ON}$. The set of reactions that result from this scheme are

$$X_i^{OFF} + \sum_{j \in \mathcal{N}(i)} X_j^{\alpha_j^i} + \sum_{j \in \mathcal{N}(i), x_j^{ON}=1} W_{ij}^{OFF} \rightleftharpoons X_i^{ON} + \sum_{j \in \mathcal{N}(i)} X_j^{\alpha_j^i} + \sum_{j \in \mathcal{N}(i), x_j^{ON}=1} W_{ij}^{ON}. \quad (3.20)$$

This reaction scheme is visualized in Figure 3.3. Just like in the DCBM, there is a separate pair of reactions for each node i for each state of its neighbors α^i. In this

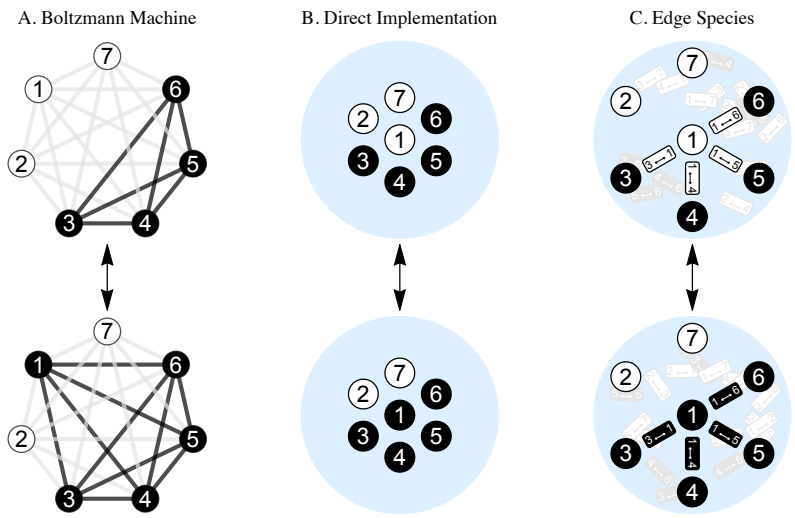


Figure 3.3: Comparison of the switching of a node in exact constructions for fully-connected topologies. Black circles indicate ON species (or nodes), and white circles indicate OFF species. Similarly, black/white rectangles indicate ON/OFF edge species. Species not involved in the reaction have been grayed out. **A.** A Boltzmann machine. Black edges contribute to the energy function. **B.** The direct implementation of a chemical Boltzmann machine. All species jointly catalyze the conversion of X_1^{OFF} to X_1^{ON}. **C.** The edge species chemical Boltzmann machine. X_1^{OFF} is converted to ON simultaneously with W_{14}^{OFF}, W_{15}^{OFF}, W_{16}^{OFF} and W_{17}^{OFF}; all other node species involved act as catalysts.

case, however, the backward reaction in (3.20) does represent a transition that is a true chemical inversion of the forward reaction. So the rate constants can be set to agree with detailed balance (3.11). Further, given a valid initial state, clamping any subset of the $X_i^{ON/OFF}$ species preserves reachability.

Theorem: The stationary distribution $\pi(x^{ON}, x^{OFF}, w^{ON}, w^{OFF})$ of the ECBM is equivalent to the stationary distribution of a Boltzmann machine, $P(x)$, provided that the ECBM begins in a valid state obeying $w_{ij}^{ON} = x_i^{ON} x_j^{ON}$ and one applies a one-to-one invertible mapping $\mathcal{F}$ between BM and ECBM states, as described below.

Proof: If this CRN begins in a consistent state, then every subsequent reaction will conserve this condition. The combined conservation laws $x_i^{ON} + x_i^{OFF} = 1$, $w_{ij}^{ON} + w_{ij}^{OFF} = 1$, and $w_{ij}^{ON} = x_i^{ON} x_j^{ON}$ ensure that the set of values x_i^{ON} uniquely determine the CRN state for the ECBM, and thus — similar to how the BM and DCBM states were related — we can define a one-to-one invertible mapping $\mathcal{F}$ that sets $x_i^{ON} = x_i$ and obeys the conservation laws.

The ECBM is detailed balanced and therefore its stationary distribution has the form (3.13). Substituting the conservation law $w_{ij} = x_i x_j$ and omitting species with 0 energy results in

$$\pi(x^{ON}, x^{OFF}, w^{ON}, w^{OFF}) = \frac{1}{Z_\pi} \mathrm{e}^{-\sum_{i \neq j} G[W_{ij}^{ON}] x_i^{ON} x_j^{ON} - \sum_i G[X_i^{ON}] x_i^{ON}}. \quad (3.21)$$

Comparing this expression to the distribution of a BM, equation (3.2), the above expressions are equivalent provided that their partition functions are equivalent. To see this is the case, notice that: 1) the partition function is just a sum over the Gibbs factors across the entire state space. 2) The Gibbs factors take the same form between the ECBM and BM (as shown above). And 3) the reachable state spaces spaces are equivalent. Thus a sum over all possible Gibbs factors will be equal. Therefore, $Z_{BM} = Z_\pi$ and the theorem is proven.

Via the ECBM, we have shown that even detailed balanced CRNs can represent rich distributions and are able to calculate conditional distributions through clamping as proven in Theorem 3.5. Due to being detailed balanced, this construction requires no fuel molecules and performs sampling via the intrinsic equilibrium fluctuations of the CRN. Moreover, it is only necessary to tune molecular energies in this construction, since appropriate relative rate constants follow by definition. This construction is possible due to the complex set of conservation laws that ensure that the reachability classes of all the $X_i^{ON/OFF}$ species are tightly coupled via the $W_{ij}^{ON/OFF}$ species. One implication is that this construction does not generalize easily to non-binary species counts. Additionally, issues related to high molecularity reactions and large number of reactions remain.

3.6 Approximate Bimolecular Implementations

The DCBM and the ECBM both require reactions of high molecularity. High molecularity reactions and systems involving many species are physically challenging to implement and also potentially suffer from long mixing times. In this section, we discuss an approximation scheme to create CBMs with lower molecularity reactions and thus overcome these issues.

Taylor Series Chemical Boltzmann Machine (TCBM):

Here, we demonstrate a compact CBM that approximates a BM. It is not detailed balanced on either the Markov chain or the CRN level, but uses only $2N$ species and $O(N^2)$ unimolecular and bimolecular reactions. The TCBM is a non-equilibrium CBM of the kind discussed in Section 3.5 that uses a dual-rail representation and single-node transitions to approximately implement a BM. The reactions are given

by:

$$\begin{aligned} X_i^{OFF} &\underset{k}{\overset{k}{\rightleftharpoons}} X_i^{ON} \\ X_j^{ON} + X_i^{OFF} &\xrightarrow{ka_{ij}} X_j^{ON} + X_i^{ON} \\ X_j^{ON} + X_i^{ON} &\xrightarrow{kb_{ij}} X_j^{ON} + X_i^{OFF} \end{aligned} \tag{3.22}$$

which, with appropriate initial conditions, preserve the conservation law that $x_i^{ON} + x_i^{OFF} = 1$.

This model's parameters can be taken directly from the weights of a BM, w_{ij}. First, define a symmetric matrix W. Decompose this matrix into the difference of two positive matrices, $W = A - B$, where $a_{ij} \in A$ are all $w_{ij} > 0$ and $b_{ij} \in B$ are the absolute values of all $w_{ij} < 0$. Finally, k is an arbitrary overall rate. This construction can be understood as an approximation of equation (3.17), which dictates that for two states s and s' that differ only in bit i with $x_i^{ON} = 1$ in state s', the CTMC transition rates must satisfy

$$\frac{R_{s\to s'}}{R_{s'\to s}} = \frac{e^{\sum_{j\neq i} a_{ij}x_j^{ON}}}{e^{\sum_{j\neq i} b_{ij}x_j^{ON}}} = \frac{1 + \sum_{j\neq i} a_{ij}x_j^{ON} + O((\sum_{j\neq i} a_{ij}x_j^{ON}))^2}{1 + \sum_{j\neq i} b_{ij}x_j^{ON} + O((\sum_{j\neq i} b_{ij}x_j^{ON}))^2}\,, \tag{3.23}$$

The bias θ_i has been absorbed into w_{ij} for notational clarity by assuming there is some $x_0^{ON} = 1$ whose weights act as biases. The TCBM is a bimolecular CRN obeying the same conservation laws as the DCBM in which each species j acts as an independent catalyst for transitions in i with reaction rates determined by a_{ij} and b_{ij}. The relative propensities of this network are exactly equal to the linear expansion of the relative propensities shown in the last term in (3.23). Specifically, the numerator is the sum of the reaction propensities for reactions that convert or catalyze $X_i^{OFF} \to X_i^{ON}$ and the denominator is the sum of the reaction propensities for $X_i^{ON} \to X_i^{OFF}$, in each case plus a constant term due to the unimolecular reactions. We thus propose the simple scheme in (3.22) as an approximate CBM; Figure 3.4A depicts this TCBM schematically. This model bears some resemblance to protein phosphorylation networks where adding or removing a phosphate group is analogous to turning a species on or off; both are driven, catalytic processes capable of diverse computation.

Approximate BCRN Inference:
Remarkably, this simple approximate CBM can reasonably approximate the inferential capabilities of a BM. We demonstrate this by using (3.22) to convert a BM

trained on the MNIST dataset [217] to a TCBM (Figure 3.4). We then compare the BM and the TCBM side by side. Digit classification is shown in Figure 3.4 panels C and D for a BM and in Figure 3.4 panels F and G for a CBM as confusion heatmaps. Classification is carried out by clamping the image nodes to MNIST images and averaging the values of the classification nodes. As is apparent from these plots, the BM does a fairly reasonable job classifying these digits, but struggles on the number 5. The CBM functions as a very noisy version of the BM with nodes in general much more likely to be on. The CBM has also faithfully inherited the capabilities and limitations of the BM and similarly struggles to classify the digit 5. Digit generation is shown in Figure 3.4E for a BM and 3.4H for a CBM. Generation was carried out by clamping a single class node to 1 and all other class nodes to 0, then averaging the output of the image nodes after the network had equilibrated. For each generated image, we show the raw output and the top 85th percentile of nodes, a thresholding which helps visualize the noisy output. As is apparent from the raw output, the CBM approximation scheme does not generate images nearly as distinctly as the BM. However, this approximation does faithfully reproduce plausible digits when filtered for the top 85th percentile. Additional training and simulation details can be found in the appendix (3.9).

The overall performance of the CRN is reasonable, given the fact that weights were simply imported from a BM without re-optimization. The TCBM only approximates the distribution implied by these weights and, in the absence of detailed balance, does not have an established formal relationship between clamping and conditioning.

3.7 Detailed Balanced CRN Learning Rule

A broad class of detailed balanced chemical reaction networks can be trained with a Hebbian learning rule between a waking phase (clamped) and sleeping phase (free) that is reminiscent of the classic gradient descent learning algorithm for a BM [85, 207]. We present the CRN learning rule here.

First we state a simple case of Theorem 3.7 where we just want a CRN with stationary distribution π over Ω_{s_0} to learn a target distribution Q also defined on Ω_{s_0}. Then, the learning rule is given by

$$\frac{dg_i}{dt} = -\frac{\partial D_{KL}}{\partial g_i} = \langle s_i \rangle_Q - \langle s_i \rangle_\pi. \tag{3.24}$$

Here, $\langle s_i \rangle_\pi$ and $\langle s_i \rangle_Q$ denote the expected count of the species S_i with respect to the probability distributions π and Q, respectively, and $g_i = G[S_i]$ is the energy of species S_i. Theorem 3.7 generalizes this procedure to cases with hidden species.

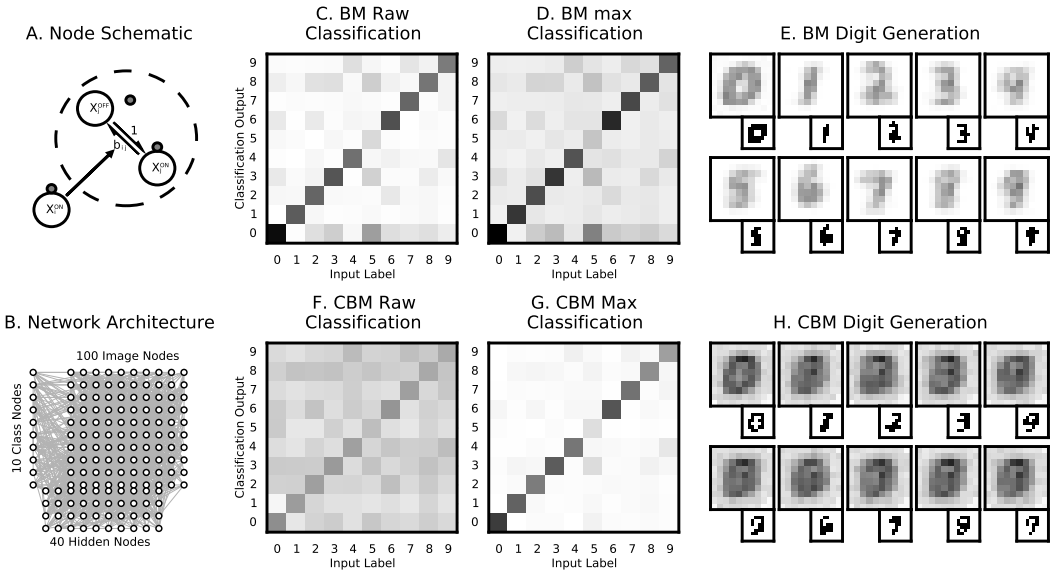


Figure 3.4: **A.** CRN underlying an individual node of the TCBM approximation. In this case a negative weight, $w_{ij} < 0$ is shown because X_i^{ON} catalyzes $X_j^{ON} \rightarrow X_j^{OFF}$. **B.** Network architecture used for simulations is fully connected but only 10 percent of edges are shown for clarity. **C.** Average raw classification output of a BM running with clamped MNIST digits. **D.** Average max classification output of a BM running with clamped MNIST digits. **E.** Digits generated by a BM by clamping individual class nodes. Small sub-boxes in the bottom corners are plots of the top 85th percentile of pixels. **F.** Average raw classification output of a TCBM running with clamped MNIST digits. **G.** Average max classification output of a TCBM running with clamped MNIST digits. **H.** Digits generated by a TCBM by clamping individual class nodes. Small sub-boxes in the bottom corners are plots of the top 85th percentile of pixels.

Theorem: Let $C = (\mathcal{S}, \mathcal{R}, k)$ be a detailed balanced chemical reaction network with stationary distribution $\pi(s)$ on Ω_{s_0}. Consider a partition (V, H) of the set $\mathcal{S}$ of species into visible and hidden species such that $\pi(s) = \pi(v, h)$. Require that for all visible states v, the clamped CRN $C|_{V=v}$ preserves reachability. Let $Q(v) > 0$ for all $v \in \Omega_{s_0}^V = \{v \text{ s.t. } (v, h) \in \Omega_{s_0}\}$ be a target distribution defined on the projection of Ω_{s_0} onto V. Furthermore, let $\pi_Q(v, h) = Q(v)\pi(h \mid v)$ be the weighted mixture of stationary distributions of the clamped CRNs $C|_{V=v}$ with v drawn from the distribution Q. Then, the gradient of the Kullback-Leibler divergence from π_V to Q with respect to the energy, $g_i = G[S_i]$, of the species S_i is given by

$$\frac{\partial D_{KL}(Q||\pi_V)}{\partial g_i} = \langle s_i \rangle_\pi - \langle s_i \rangle_{\pi_Q} \tag{3.25}$$

where $\pi_V(v) = \sum_{h \in \Omega_{s_0}^H} \pi(v, h)$ is the marginal $\pi(v, h)$ over hidden species H.

Proof: Applying Theorem 3.5, the clamped CRN ensemble $\pi_Q(s)$ may be written

$$\pi_Q(s) = \pi_Q(v,h) = Q(v)\pi(h \mid v) = Q(v)\frac{\pi(v,h)}{\sum_{h\in\Omega_{s_0}^H}\pi(v,h)} = Q(v)\frac{\pi(v,h)}{\pi_V(v)} \ . \quad (3.26)$$

Additionally we will need the partial derivative of a Gibbs factor and the partition function with respect to g_i,

$$\frac{\partial \mathrm{e}^{-\mathcal{G}(s)}}{\partial g_i} = -s_i \mathrm{e}^{-\mathcal{G}(s)} \quad \text{and} \quad \frac{\partial Z}{\partial g_i} = -Z\langle s_i\rangle_\pi \ . \quad (3.27)$$

Using these results, the partial derivative of any detailed balanced CRN's distribution at a particular state s, with respect to an energy g_i, is

$$\frac{\partial \pi(s)}{\partial g_i} = \frac{\partial}{\partial g_i}\frac{1}{Z}\mathrm{e}^{-\mathcal{G}(s)} = \langle s_i\rangle_\pi \pi(s) - s_i\pi(s) \ . \quad (3.28)$$

Noting that Q has no dependence on g_i, the gradient of the Kullback-Leibler divergence can then be written,

$$\begin{aligned}
\frac{\partial D_{KL}(Q||\pi_V)}{\partial g_i} &= \frac{\partial}{\partial g_i}\sum_{v\in\Omega_{s_0}^V} Q(v)\log\frac{Q(v)}{\pi_V(v)} = \sum_{v\in\Omega_{s_0}^V} -\frac{Q(v)}{\pi_V(v)}\frac{\partial \pi_V(v)}{\partial g_i} \\
&= -\sum_{v\in\Omega_{s_0}^V}\sum_{h\in\Omega_{s_0}^H}\frac{Q(v)}{\pi_V(v)}\pi(v,h)\langle s_i\rangle_\pi - \frac{Q(v)}{\pi_V(v)}\pi(v,h)s_i \\
&= -\sum_{(v,h)\in\Omega_{s_0}} \pi_Q(v,h)\langle s_i\rangle_\pi - \pi_Q(v,h)s_i = -\langle s_i\rangle_\pi + \langle s_i\rangle_{\pi_Q}.
\end{aligned}$$

In the special case where there are no hidden species, which is to say the target distribution Q is defined over the whole reachability class Ω_{s_0}, then $\pi_V(v) = \pi(s)$ and $\pi_Q(s) = Q(s)$ and the gradient has the simple form shown in equation (3.24).

Applying gradient descent via $\frac{dg_i}{dt} = -\frac{\partial D_{KL}}{\partial g_i}$, we thus have a simple *in silico* training algorithm to train any detailed balanced CRN such that it minimizes the Kullback-Leibler divergence from π_V to Q. If $H = \emptyset$, simulate the CRN freely to estimate the average counts $\langle s_i\rangle$ under $\pi(s)$. Then compare to the average counts under the target $Q(s)$ and update the species' energies accordingly. If $H \neq \emptyset$, clamp the visible species to some $v \in \Omega_{s_0}^V$ with probability $Q(v)$ and simulate the dynamics of the hidden units. Repeat to sample an ensemble of clamped CRNs $C|_{V=q}$. Because clamping v preserves reachability, Gillespie simulations of the CRN with the V species clamped to the data values v will sample appropriately. This gives the average counts under π_Q.

This CBM learning rule is more general than the classical Boltzmann machine learning rule, as it applies to arbitrary detailed balanced CRNs, including those with arbitrary conservation laws and arbitrarily large species counts (but still subject to the constraint that reachability under clamping must be preserved). That said, at first glance the CBM learning rule appears weaker than the classical Boltzmann machine learning rule, as it depends exclusively on mean values $\langle s_i \rangle$, whereas the Boltzmann machine learning rule relies primarily on second-order correlations $\langle x_i x_j \rangle$. In fact, though, conservation laws within the CRN can effectively transform mean values into higher-order correlations. A case in point would be to apply the CBM learning rule to the ECBM network: For $g_i = G[X_i^{ON}] = -\theta_i$, $\frac{d\theta_i}{dt} = -\frac{dg_i}{dt} = \langle x_i^{ON} \rangle_{\pi_Q} - \langle x_i^{ON} \rangle_{\pi}$, and for $g_i = G[W_{ij}^{ON}] = -w_{ij}$, $\frac{dw_{ij}}{dt} = -\frac{dg_i}{dt} = \langle w_{ij}^{ON} \rangle_{\pi_Q} - \langle w_{ij}^{ON} \rangle_{\pi} = \langle x_i^{ON} x_j^{ON} \rangle_{\pi_Q} - \langle x_i^{ON} x_j^{ON} \rangle_{\pi}$, which exactly matches the classical Boltzmann machine learning rule if we assert that the energies of OFF species are fixed at zero.

3.8 Discussion

We have given one approximate and two exact constructions that allow CRNs to function as Boltzmann machines. BMs are a “gold standard" generative model capable of performing numerous computational tasks and approximating a wide range of distributions. Our constructions demonstrate that CRNs have the same computational power as a BM. In particular, CRNs can produce the same class of distributions and can compute conditional probabilities via the clamping process. Moreover, the TCBM construction appears similar in architecture to protein phosphorylation networks. Both models are non-equilibrium (i.e., require a fuel source) and make use of molecules that have an on/off (e.g., phosphorylated/unphosphorylated) state. Additionally, there are clear similarities between our exact schemes and combinatorial regulation of genetic networks by transcription factors. In this case, both models make use of combinatoric networks of detailed-balanced interactions (e.g., binding/unbinding) to catalyze a state change in a molecule (e.g., by turning a gene on/off). We note that our constructions differ from some biological counterparts in requiring binary molecular counts. However, in some cases we believe that biology may make use of conservation laws (such as having only a single copy of a gene) to allow for chemical networks networks to perform low-cost computations. In the future, we plan to examine these cases in a biological setting as well as generalize our models to higher counts.

Developing these CBMs leads us to an important distinction between equilibrium, detailed-balanced CRNs with steady state distributions determined by molecular

Model	Species	Reactions	Molecularity	Detailed Balance
Direct CBM	$2N$	$N2^{d+1}$	$d+1$	CTMC
Edge CBM	$2N+dN$	$N2^{d+1}$	$\leq 2d+1$	CRN and CTMC
Taylor CBM	$2N$	$2N+2dN$	≤ 2	Neither

Table 3.1: The complexity and underlying properties of our constructions for reproducing a BM with N nodes of degree d. Detailed balance describes whether the construction is detailed balanced at the CRN level, at the CTMC level, or neither.

energies, and CRNs that do not obey detailed balance in the underlying chemistry. The second category includes those that nonetheless appear detailed balanced at the Markov chain level. Physically, this distinction is especially important: a non-detailed balanced CRN will always require some kind of implicit fuel molecule (maintained by a chemostat) to run and the steady state will not be an equilibrium steady state due to the continuous driving from the fuel molecules. A detailed balanced CRN (at the chemical level) requires no fuel molecules: and thus *the chemical circuit can act as a sampler without fuel cost*. Despite this advantage, working with detailed balanced CRNs presents additional challenges: to ensure that chemical species do not have independent distributions, species counts must be carefully coupled via conservation laws.

Our constructions also highlight important complexity issues underlying CBM design. The number of species, the number of reactions, and the reaction molecularity needed to implement a particular BM are relevant. Trade-offs appear to arise between these different factors and the thermodynamic requirements of a given design. A breakdown of the main features of each CBM is given in Table 3.1. Summarizing, the TCBM is by far the simplest construction, using $O(N)$ species, at most $O(N^2)$ reactions, with molecularity ≤ 2. However, this happens at the expense of not being an exact recreation of a BM, and the requirement of a continuous consumption of fuel molecules. The DCBM is the next simplest in complexity terms, using $O(N)$ species, $O(N2^N)$ reactions, and molecularity of at most N. Like the TCBM, the DCBM requires fuel molecules because it is not detailed balanced at the CRN level. The ECBM is considerably more complex than the DCBM, using quadratically more species, $O(N^2)$, the same number of reactions, $O(N2^N)$ and double the reaction molecularity. The ECBM makes up for this increased complexity by being detailed balanced at the CRN level, meaning that it functions in equilibrium without implicit fuel species.

Finally, we have shown that a broad class of detailed balanced CRNs can be trained using a Hebbian learning rule between a waking phase (clamped) and sleeping phase (free) reminiscent of the gradient descent algorithm for a BM. This exciting finding allows for straightforward optimization of detailed balanced CRNs' distributions.

This work provides a foundation for future investigations of probabilistic molecular computation. In particular, how more general restrictions on reachability classes can generate other "interesting" distributions in detailed balanced CRNs remains an exciting question. We also wonder if the learning rule algorithm can be generalized to certain classes of non-detailed balanced CRNs, and whether our exact CBM constructions can be generalized to non-binary molecular counts. From a physical standpoint, plausible implementations of the clamping process and the energetic and thermodynamic constraints require investigation. Indeed, a more realistic understanding of how a CBM might be implemented physically will help us identify when these kinds of inferential computations are being performed in real biological systems and could lead to building a synthetic CBM.

Acknowledgements: This work was supported in part by U.S. National Science Foundation (NSF) graduate fellowships to WP and to AOM, by NSF grant CCF-1317694 to EW, by the Gordon and Betty Moore Foundation through Grant GBMF2809 to the Caltech Programmable Molecular Technology Initiative (PMTI), by a Royal Society University Research Fellowship to TEO, and by a Bharti Centre for Communication in IIT Bombay award to AB.

3.9 Appendix

Application of Theorem 3.5: The Direct CBM Must Use Implicit Fuel Species

Here, we use Theorem 3.5 to analyze the direct implementation of a CBM and show that it cannot be detailed balanced and thereby requires implicit fuel molecules. First, notice that the the conservation laws used in this construction are of a simple form. The states accessible by (X_i^{ON}, X_i^{OFF}) are independent of (X_j^{ON}, X_j^{OFF}) for $i \neq j$, and therefore the reachability class is a product over the subspaces of each individual node. As a consequence, by Theorem 3.5, the system must be out of equilibrium and violate detailed balance at the level of the CRN because, by construction, this system is equivalent to a BM and has correlations between nodes i and j whenever $w_{ij} \neq 0$. In physical terms, the presence of catalysts cannot influence the equilibrium yield of a species, and therefore a circuit which uses catalysis to bias distributions of species must be powered by a supply of chemical fuel molecules [212–214]. It is also worth noting that, as a consequence, this scheme cannot be implemented

by tuning of (free) energies; it is fundamentally necessary to carefully tune all of the rate constants individually (via implicit fuel molecules) to ensure that detailed balance is maintained at the level of the Markov chain for the species of interest.

BM Training and TCBM Simulation Details:
We trained a BM using stochastic gradient descent on the MNIST dataset, down sampled to be 10 pixels by 10 pixels [217]. The BM has 100 visible image units (representing a 10 x 10 image), 10 visible class nodes, and 40 hidden nodes as depicted in Figure 3.4B. Our training data consisted of the concatenation of down sampled MNIST images and their classes projected onto the 10 class nodes. The weights and biases of the trained BM were converted to reaction rates for a CBM using the Taylor series approximation. This CBM consists of 300 species, 300 unimolecular reactions and 22350 bimolecular reactions. The resulting CBM was then compared side-by-side with the trained BM on image classification and generation. The BM was simulated using custom Gibbs sampling written in Python. The CRN was simulated on a custom Stochastic Simulation Algorithm (SSA) [167] algorithm written in Cython. All simulations, including network training, were run locally on a notebook or on a single high performance Amazon Cloud server.

Classification was carried out on all 10000 MNIST validation images using both the BM and the CBM. Each 10 by 10 gray-scale image was converted to a binary sample image by comparing the gray-scale image's pixels (which are represented as real numbers between 0 and 1) to a uniform distribution over the same range. The network's image units were then clamped to the binary sample and the hidden units and class units were allowed to reach steady state. This process was carried out 3 times for each MNIST validation image, resulting in 30000 sample images being classified. Raw classification scores were computed by averaging the class nodes' outputs for 20000 simulation steps after 20000 steps of burn-in (Gibbs sampling for the BM, SSA for the CBM). Max classification was computed by taking the most probable class from the raw classification output. Raw classification and max classification confusion heatmaps, showing the average classification across all sample images as a function of the true label are shown in Figure 3.4 panels C and D for a BM and in Figure 3.4 panels F and G for a CBM.

Image generation was carried out by clamping the class nodes with a single class, 0...9, taking the value of 1 and all other classes being 0, and then allowing the network to reach steady state. Generated images were computed by averaging the image nodes over 50000 simulation steps (Gibbs sampling for the BM, SSA for the

CBM) after 25000 steps of burn-in. Generation results are shown in Figure 3.4E for a BM and Figure 3.4H for a CBM.

Chapter 4

DETAILED BALANCED CHEMICAL REACTION NETWORKS AS GENERALIZED BOLTZMANN MACHINES

4.1 Forward

The following chapter has been written as a long-form paper which we aim to submit for review in the near future. A version of this work was presented as a digital poster at *Nucleic Acids, Synthetic Biology and Artificial Life: Engineering and Controlling Out of Equilibrium Systems* workshop hosted (online) at Imperial College London, March 2021. This work has been jointly supervised by Tom Ouldridge, Manoj Gopalkrishnan, and Erik Winfree.

In Chapter 3, we showed that a particular detailed balanced CRN can exactly implement a Boltzmann machine and proved an *in silico* learning rule applicable to all dbCRNs. This chapter's main focus is to elaborate and generalize those results. First, we show that the key requirement for a dbCRN to have a *complex* [1] equilibrium distribution is restricting its reachability class. We then provide a framework by which any dbCRN is capable of probabilistic inference expanding and improving the definition of *clamping* from the previous chapter. We go on to show that this new way of clamping can be implemented via chemical potentials coupled to the dbCRN. These chemical potentials, in turn, can be controlled by the species in the dbCRN producing an autonomous non-detailed balanced CRN which trains itself with a chemical implementation of the *in silico* learning rule from Chapter 3. Finally, we take advantage of having a self-contained CRN implementation of a machine learning algorithm to provide some basic results on the thermodynamic costs of inference and learning.

The significance of this work in the broader scope of this thesis is two-fold. First, it shows that the biochemically relevant class of dbCRNs can be *naturally* interpreted as a kind of probabilistic graphical model capable of inference and learning. However, unlike traditional graphical models which are "programmed" by choosing the form of their energy function, dbCRNs have a preset form for their energy function and are programmed by restricting their reachability class. In some cases, such as the *Edge-Species Chemical Boltzmann Machine* described in Chapter 3,

[1] In this context, we define *complex* as far from a product of independent Poisson distributions.

these reachability class restrictions can result in a new *emergent* energy function. However, understanding how to write dbCRN programs in general using reachability constraints remains an open problem. Second, the autonomous learning CRN described in this chapter points to a new class of CRN-specific machine learning algorithms. Besides being of interest as a self-contained and physically well-defined learning system which could potentially be built in the lab, this result suggests a new class of *in silico* machine learning algorithms based on CRNs. I hypothesize that these algorithms may be particularly useful when applied to learning and inference problems involving CRNs and biological data because these algorithms may be able to better use the underlying mathematical structure of CRNs in order to more effectively optimize them.

Contribution: This work was born out of years of theoretical and computational investigations trying to generalize results from the previous chapter. Different pieces began to fall together in their final form when I traveled to Mumbai to work closely with Manoj Gopalkrishnan for four months in late 2019 and early 2020. Upon returning to the US, I continued formalizing the foundations under Erik Winfree's guidance. I asked Tom Ouldridge to help advise me on the thermodynamic analysis towards the end of the project. I drafted the entire paper and am responsible for the majority of proofs and analyses, with input, guidance, and review from the other co-authors.

4.2 Abstract

Can a micron sized sack of interacting molecules understand, learn, and adapt to a constantly-fluctuating environment? Cellular life provides an existence proof in the affirmative, but the principles that allow for life's existence are far from being proven. One challenge in engineering and understanding biochemical computation is the intrinsic noise due to chemical fluctuations. In this paper, we draw insights from machine learning theory, chemical reaction network theory, and statistical physics to show that the broad and biologically relevant class of detailed balanced chemical reaction networks is capable of representing and conditioning complex distributions. Furthermore, these systems can be augmented via non-equilibrium reactions in order to learn complex distributions copied from an external environment. These results illustrate how a biochemical computer can use intrinsic chemical noise to perform complex computations. Furthermore, by providing an explicit and autonomous physical model of machine learning, we are able to derive basic thermodynamic costs of inference and learning.

4.3 Introduction

Computing with small numbers of molecules at around room temperature presents a unique set of challenges. However, cell and molecular biology demonstrate that every living cell can perform complex information processing using circuitry built out of just a few basic building blocks [218]. More recently, systems biologists have identified a variety of biochemical design principles observed across many organisms and suggesting that unified understanding of biochemical systems may be possible [6, 14]. At the same time, synthetic biologists and engineers, inspired by the sophistication seen in biology, are beginning to build cellular and synthetic cell-like systems [94, 219–221]. Programmed biochemical systems are rapidly expanding into medical diagnostics [222], cancer therapeutics [104], sustainable bioreactors [99], and advanced materials [98]. However, designing and understanding increasingly complex biochemical computation will require new principles and design methodologies. This work adds a new perspective by showing that a wide class of biochemical models can be formally interpreted, designed, and analyzed using machine-learning inspired methods.

Almost all modern computers are built on the digital abstraction of variables having binary values. This allows computer to excel at tasks like Boolean logic and integer arithmetic. However, there is no reason why biochemical computers need be digital or would have evolved that way. Despite early success building synthetic biochemical logic circuits [24, 37], so far these systems have failed to achieve anything close to the complexity a smartphone is capable of, much less a living organism. An alternative is to take an analog approach where biochemical signals are allowed a continuous range of values. In the past decades, analog computing has been reborn under the guise of deep learning and is revolutionizing domains from computer vision [43] to natural language processing [45] to biochemistry [46]. A third approach, and the one we emphasize in this paper, is probabilistic programming [223], a hybrid of the conventional digital abstraction and analog computing where discrete values are assigned probabilities. Indeed, many state of the art machine learning methods involve probabilistic elements [47] and more conventional machine learning methods may be seen as approximations of probabilistic models [224]. It has also been suggested out that analog or hybrid digital-analog approaches to information processing may be more efficient, particularly in the nervous system and biochemical contexts [41, 225]. There is also an increased interest in specialized hardware capable of directly implementing probabilistic computations and machine learning [226]. In this vein, machine learning architectures, such as neural networks, have been theorized [54,

55, 60, 78, 227] and built in a variety of biochemical systems [68], yet are still considerably smaller than their digital relatives.

Inside individual cells, noise also plays an important role in biochemical computation contrasting with the deterministic precision of silicon computers. At the micron to sub-micron scale, the counts of individual (non-solvent) molecules frequently become very small which in turn results in high coefficient of variation [199]. Mathematically, such systems are commonly modeled with stochastic chemical reaction networks (CRNs) [3]. Biochemical noise has been rigorously quantified in systems such as gene expression [119], stochastic partitioning at cell division [122], and neuronal firing [228]. This extensive documentation of noise suggests that chemical computers must function in a a noisy environment, respond to noisy signals, and rely on noisy components. One way to deal with this noise is to mitigate it: methods such as kinetic proofreading [229, 230], low pass filters [231, 232], and fold change detectors [233, 234] have all been observed naturally and engineered in order to mitigate fluctuations. However, an alternate approach is to ask how biochemical systems can use intrinsic noise to their advantage. A number of theoretical biochemical algorithms have been proposed which use intrinsic chemical fluctuations to generate distributions [235], infer parameters of probabilistic models [84], and solve combinatorial constraint satisfaction problems [236].

Probabilistic inference can be seen as the archetypal problem for understanding how to deal with noise and make the *best* decision [237]. In computer science and machine learning, generative probabilistic models have emerged as a powerful framework for inference [48]. Boltzmann machines are one of the most well studied of these models and excel at learning high dimensional probability distribution using hidden (latent) variables and computing conditional distributions [85]. In this paper, we build off previous work which presented a number of CRN implementations of Boltzmann machines [187]. Here, we show how a broad class of stochastic biochemical models called detailed balanced (db) CRNs may be viewed as generalizations of Boltzmann machines: capable of representing complex distributions using hidden variables, computing conditional distributions, and, when coupled to an auxiliary CRN, learning. Due to the explicit and physical nature of these models, we then go on to provide some thermodynamic costs related to inference and learning.

4.4 Background

Inference and Learning

In this work, we are interested in generative models, a broad class of mathematical models which represent probability distributions [238]. Specifically we focus on Boltzmann machines and stochastic chemical reaction networks; when interpreted as generative models, each of these systems probabilistically samples (or explores) a high dimensional state space. In general, we are less concerned with the sampling dynamics than the steady state distribution which occurs in the limit of running a generative model for infinite time. In fact, a low number of dynamic variables coupled together can represent an exponentially larger steady-state distribution with complex correlative structure [187]. Seen in this light, probability distributions present a powerful framework for a system to understand its environment. Within the context of generative models, we equate inference with the computation of conditional steady state distributions. For instance, a cell inferring the state of the environment h given a noisy signal v is equivalent to computing the conditional distribution $\mathbb{P}(h \mid v)$. Similarly, learning, or inferring, the parameters of a model θ given data x can be thought of computing a conditional distribution $\mathbb{P}(\theta \mid x)$. Implicit in our framework is the idea that the variable being conditioned on is held constant while the conditional distribution is being computed—these variables are referred to as *clamped*. Additionally, the un-clamped or *free* variables must be coupled to the clamped variables for inference to be meaningful. In the limit of no coupling between x and y, a distribution can be written in product form, $\mathbb{P}(x, y) = \mathbb{P}(x)\mathbb{P}(y)$, which means x and y are independent and conditioning provides no new information: $\mathbb{P}(x \mid y) = \mathbb{P}(x)$ and $\mathbb{P}(y \mid x) = \mathbb{P}(y)$.

Boltzmann Machines

Boltzmann Machines (BM) are a class of probabilistic graphical model which have been extensively studied theoretically [207, 239] and used for many machine learning applications [44, 240, 241]. For the purposes of this work, BMs serve as a guide for understanding what it means for a system to be capable of inference and learning. Briefly, a BM is a stochastic neural network of binary nodes $x_i \in \{0, 1\}$ which have an equilibrium distribution:

$$\mathbb{P}(x) = \frac{1}{Z} e^{-E(x)} \qquad Z = \sum_{x \in \{0,1\}^N} e^{-E(x)} \qquad E(x) = \sum_{i>j} w_{ij} x_i x_j - \sum_i \theta_i x_i. \quad (4.1)$$

Here, $\theta \in \mathbb{R}$ are bias terms and the weight terms $w_{ij} \in \mathbb{R}$ couple nodes x_i and x_j and induce correlations ensuring x_i and x_j are not independent. Additionally, Boltzmann

machines can use hidden units and marginalization to represent even more complex distributions. Let $X = \{x_i\}$ be partitioned into two disjoint sets V and U of visible and hidden units, respectively. The probability of the visible units is given by the marginalization over the hidden units:

$$\pi(v) = \sum_u \pi(v, u). \tag{4.2}$$

Similarly to the many layers of latent variables in a deep neural network, the hidden units U have been shown to increase the complexity of the distributions the BM can model [207, 239, 242].

BMs are also able to seamlessly compute conditional distributions. Again, partition the nodes X into two sets U and V. Here U will be the free variables and V will be clamped. The conditional distribution $\pi(u \mid v)$ can be exactly sampled by clamping v to a constant value while simulating the BM [207]. In many cases, a combination of clamping and marginalization is used together where both visible and hidden units can be clamped or free. General purpose inference allows Boltzmann Machines to classify data [243] (U are class labels and hidden units, V are data points), generate distributions [210] (U are data and hidden units, V are labels and hidden units), and infer missing data [241] (U are unknown data and hidden units, V are known data).

Finally, the parameters w and θ can be learned to minimize the relative entropy:

$$\mathbb{D}(\psi(v) \mid\mid \pi(v)) = \sum_v \psi(v) \log \frac{\psi(v)}{\pi(v)} \tag{4.3}$$

where $\pi(v)$ is the marginal of $\pi(v, u)$ and the data is distributed according to $\psi(v)$. Note that the relative entropy is not symmetric: $\mathbb{D}(\psi \mid\mid \pi) \neq \mathbb{D}(\pi \mid\mid \psi)$. In the form of Eq. 4.3, we can consider ψ to be the "true" distribution, while π is an approximation; this corresponds to the relative entropy being an average over ψ: $\mathbb{D}(\psi \mid \pi) = \langle \log \frac{1}{\pi} \rangle_\psi - \langle \frac{1}{\psi} \rangle_\psi$ being the excess average code length when using the approximate distribution to design code words instead of the true distribution. In order to explicitly optimize hidden units, this optimization can be equivalently carried out via gradient descent on the relative entropy between the *clamped* distribution $\overline{\pi} = \pi(u \mid v)\psi(v)$ of the BM held to samples from ψ and the *free* distribution $\pi(u, v)$ of BM:

$$\mathbb{D}(\overline{\pi} \mid\mid \pi) = \sum_{v,u} \overline{\pi}(u, v) \log \frac{\overline{\pi}(u, v)}{\pi(v, u)} = \quad \text{where} \quad \overline{\pi}(u, v) = \pi(u \mid v)\psi(v) \tag{4.4}$$

Taking the gradient with respect to w_{ij} and θ_i results in the update rule which looks identical for both hidden and visible units:

$$\frac{\mathrm{d}w_{ij}}{\mathrm{d}t} = -\frac{\partial \mathbb{D}(\overline{\pi} \mid\mid \pi)}{\partial w_{ij}} = \epsilon(\langle x_i x_j \rangle_\pi - \langle x_i x_j \rangle_{\overline{\pi}}) \quad \frac{\mathrm{d}\theta_i}{\mathrm{d}t} = -\frac{\partial \mathbb{D}(\overline{\pi} \mid\mid \pi)}{\partial \theta_i} = \epsilon(\langle x_i \rangle_\pi - \langle x_i \rangle_{\overline{\pi}}). \tag{4.5}$$

Here $\langle \cdot \rangle_\pi$ and $\langle \cdot \rangle_{\overline{\pi}}$ denote the expected value with respect to the free and clamped distributions, respectively, and ϵ is the learning rate.

Chemical Reaction Networks

In this work, we model biochemical fluctuations using stochastic chemical reaction networks (CRNs). This model can be derived from the statistical mechanics of a well-mixed ideal solution [3] and has a long history being applied to biological problems including modeling genetic circuits [1] and understanding noise in biochemical processes [198]. Similarly, stochastic CRNs have been studied as a programming language, shown to be Turing universal [20, 169, 244] and used to guide implementations and analyses of molecular programs in laboratory settings [23, 26, 245]
.

Formally, a CRN $(\mathcal{S}, \mathcal{R}, k)$ is a set of species $\mathcal{S}$, reactions $\mathcal{R}$, and reaction rates k. Reactions convert one multiset of species into another: $I^r \xrightarrow{k_r} O^r$. Here I^r and O^r are the vectors of input and output species for reaction r, respectively. These vectors can be combined into a single matrix called the stoichiometric matrix $M = O - I$. CRNs may have both stochastic and deterministic dynamics. When considered deterministically, reactions cause the concentrations of each species, $[S_i]$, to change according to the differential equation:

$$\frac{\mathrm{d}[S_i]}{\mathrm{d}t} = \sum_r k_r M_i^r \prod_j [S_j]^{I_j^r}. \tag{4.6}$$

When considered stochastically, reactions occur with probability proportional to their mass-action propensity function $\rho_r(s) = k_r \prod_i \frac{s_i!}{(s_i - I_i^r)!}$. Physically, rate constants in deterministic and stochastic equations have different meanings (due to being expressed with different units): the deterministic equations are volume-independent, while the values in the stochastic context must depend on volume (and can be related to the deterministic values given a choice of volume). We use s_i to mean the counts of species S_i and s without a subscript denotes a vector of species' counts. The dynamics of the probability distribution are given by the chemical master equation:

$$\frac{\mathrm{d}\mathbb{P}(s,t)}{\mathrm{d}t} = \sum_r \mathbb{P}(s - M^r, t)\rho_r(s - M^r) - \mathbb{P}(s,t)\rho_r(s). \tag{4.7}$$

Often, we are interested in the steady-state distribution $\mathbb{P}^*(s)$ found by setting (4.7) to 0 or by simulating the CRN dynamics until convergence with exact methods such as the Gillespie algorithm [167]. For numerical results in this paper, CRNs are sampled with the Bioscrape implementation of the Gillespie algorithm [137].

The reachability class of a CRN, $\Gamma_{s^0} \subseteq \mathbb{Z}^{\mathcal{S}}$, is the subset of the integer lattice reachable by a sequence of reactions starting at an initial state s^0. In some cases, Γ_{s^0} may be infinite. We emphasize that this is distinct from the stoichiometric subspace Ω_{s^0} which is given by the kernel of the stoichiometric matrix M. The latter is an affine space of the form $A(s - s^0) = 0$, with A an $|\mathcal{S}|$ by k dimensional matrix representing $k \geq 0$ conserved quantities in the system. Specifically, $\Gamma_{s^0} \subseteq \Omega_{s^0}$ [168]. For example, the CRN $\emptyset \rightleftharpoons 2S$ has an infinite reachability class of either the even or odd positive integers depending on s^0 and a stoichiometric compatibility class that covers *all* the positive integers. Indeed, as we will prove later, it is by restricting the reachability class relative to the stoichiometric subspace that we are able to program detailed balanced CRNs to represent broad classes of distributions. Note that the reachability class may be arbitrarily complex compared to the stoichiometric compatibility class [246].

Detailed Balanced Chemical Reaction Networks

Detailed balanced CRNs (dbCRN) are a subclass of CRNs which represent non-driven chemical systems such as molecular binding. Mathematically, the detailed balanced property requires that:

- Each species $S_i \in \mathcal{S}$ has an energy G_i.
- All reactions are reversible meaning if $I \xrightarrow{k^+} O \in \mathcal{R}$ then $O \xrightarrow{k^-} I \in \mathcal{R}$.
- Reaction rates obey $\frac{k^+}{k^-} = e^{-\Delta G}$, where $\Delta G = \sum_i G_i(O_i - I_i)$.

A detailed balanced reaction network's stationary distribution is an equilibrium distribution meaning there is no net energy flow or entropy production at steady state. These distributions can correspond to both the canonical or grand canonical ensembles of statistical physics, depending on whether only energy is allowed to be exchanged with the heat bath or if certain species within a reservoir can also be exchanged (such as via the reaction $\emptyset \rightleftharpoons 2S$ where $\emptyset$ indicates a molecule moving to or from the reservoir). This equilibrium distribution necessarily has a product-

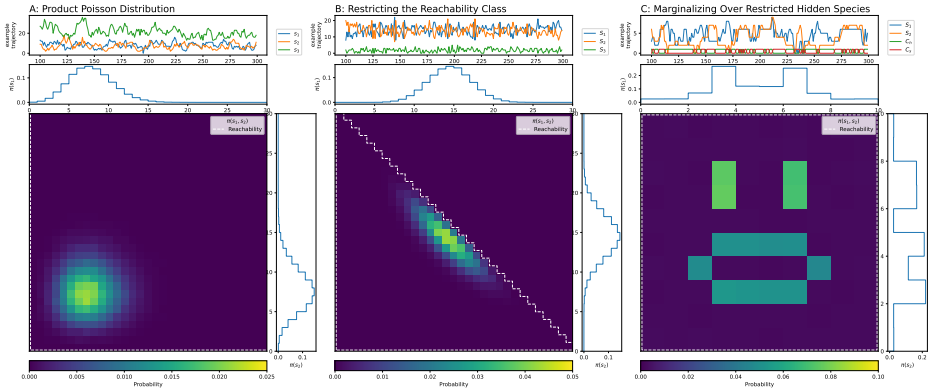


Figure 4.1: Equilibrium distributions of dbCRNs. A. A Product Poisson Distribution of uncorrelated species. B. A dbCRN restricted to an affine subspace. C. A dbCRN marginalized over many binary hidden species. Note: all trajectories shown at the top are just tiny fractions of the total simulated time.

Poisson form written in terms of the free energy function [93]:

$$\pi(s) = \frac{1}{Z}\prod_i \frac{e^{-G_i s_i}}{s_i!} = \frac{1}{Z}e^{-\mathcal{G}(s)} \qquad Z = \sum_{s\in\Gamma_{s0}} e^{-\mathcal{G}(s)} \tag{4.8}$$

$$\mathcal{G}(s) = \sum_i \mathcal{G}_i(s) = \sum_i G_i s_i + \log s_i! \tag{4.9}$$

In this paper, we are primarily concerned with how species' energies affect the equilibrium distribution. If $\mathcal{D} = (\mathcal{S}, \mathcal{R}, k)$ is a detailed balanced CRN, $\mathcal{D}_G$ is the same set of species and reactions with the rate constants k changed to reflect the new energies G.

Detailed Balanced CRNs Can Model Complex Environments

Equation (4.8) seems to suggest that dbCRNs can only produce simple distributions in product-Poisson form which in turn seem to only allow control of the mean, because Poisson distributions are defined by their mean. For example, a product Poisson distribution on the entire integer lattice can be seen in figure 4.1A. Yet, in previous work, we constructed dbCRNs able to produce non-Poisson distributions such as those produced by Boltzmann machines [187]. Indeed, carefully constructed dbCRNs can in fact produce any distribution with finite support [235]. Each of these constructions implicitly restricts the reachability class with specific reactions and initial conditions. In this section, we illustrate two example of dbCRNs producing non-poisson distributions. First, we consider the simple dbCRN used to produce

the distributions in 4.1B and 4.4B:

$$S_1 \rightleftharpoons S_2 \rightleftharpoons S_3 \qquad\qquad s_1 + s_2 + s_3 = c. \tag{4.10}$$

Here a single constraint restricts the reachability class of each species where c is the total number of all the species at the start of the simulation. Even such a simple constraint is enough to break the independence of a product Poisson distribution which can be seen by the obvious anti-correlatiom between S_1 and S_2 in Figure 4.1B.

Detailed balanced CRNs with many more conservation laws can produce arbitrary distributions with finite support. For example, the dbCRN used to produce face in figure 4.1C and the smiley-frowny face in 4.4C is given by a set of reactions based on [235]:

$$\begin{aligned} C_\alpha + P^\alpha_{x,y} &\rightleftharpoons C_\alpha + P^\alpha_{x+1,y} + S_1 \\ C_\alpha + P^\alpha_{x,y} &\rightleftharpoons C_\alpha + P^\alpha_{x,y+1} + S_2 \\ C_\alpha + P^\alpha_{x,y} &\rightleftharpoons C_\beta + P^\beta_{x,y}. \end{aligned}$$

Here, C_α are control species tuning whether the distribution is happy ($\alpha = h$) or sad ($\alpha = s$). There are two sets of pixel species $P^\alpha_{x,y}$, one for the happy face ($\alpha = h$) and one for the sad face ($\alpha = s$) with x and y denoting the pixel locations. The visible species are S_1 and S_2. The energies of the pixel species have been tuned to produce the happy and sad images. This construction requires a very tightly coupled reachability class with initial conditions that satisfy the following constraints:

$$C_s + C_h = 1 \quad \sum_{x,y,\alpha} P^\alpha_{x,y} = 1 \quad \prod_{x,y} P^\alpha_{x,y} = C_\alpha \quad s_1 = \sum_{x,y,\alpha} x P^\alpha_{x,y} \quad s_2 = \sum_{x,y,\alpha} y P^\alpha_{x,y}$$

In words: there is only one C_α species present and one $P^\alpha_{x,y}$ species present at any time and they must both have the same value for α. The counts of S_1 and S_2 must also correspond to the values x and y, respectively, of the single $P^\alpha_{x,y}$ species present.

Chemical Boltzmann Machines

In our previous work on chemical Boltzmann machines [187], we showed that an analog of the BM learning rule (4.5) also works for dbCRNs:

$$\frac{\mathrm{d}\mathbb{D}(\overline{\pi} \,||\, \pi)}{\mathrm{d}G_i} = \epsilon(\langle s_i \rangle_{\overline{\pi}} - \langle s_i \rangle_\pi). \tag{4.11}$$

Naively, unlike the BM learning rule, (4.11) appears to only utilize the means of each species and not second moments. However, we showed that the detailed balanced

Edge-Species Chemical Boltzmann Machine (ECBM) construction constrains the reachability class of certain species with conservation laws. This can be seen by the example of a two-node ECBM (which can be generalized to arbitrary graphs):

$$S_1^0 + S_2^0 \rightleftharpoons S_1^1 + S_2^0 \qquad S_1^1 + S_2^0 + S_W^0 \rightleftharpoons S_1^1 + S_2^1 + S_W^1 \tag{4.12}$$

$$S_1^0 + S_2^0 \rightleftharpoons S_1^0 + S_2^1 \qquad S_1^0 + S_2^1 + S_W^0 \rightleftharpoons S_1^1 + S_2^1 + S_W^1. \tag{4.13}$$

Here, each node $i \in \{1, 2\}$ of a BM is represented by two species S_i^0 and S_i^1 corresponding to the *off* and *on* states. The edge species S_W^0 and S_W^1 relates to the energy term w_{12} in a BM and similarly has *off* and *on* states. This dbCRN has stoichiometric conservation laws $s_i^0 + s_i^1 = N_i \quad i \in \{1, 2, W\}$ as well as the possibility of an emergent conservation law $s_0^1 s_1^1 = s_W^1$ which occurs when $N_1 = N_2 = N_W = 1$ and the CRN starts in a state which respects this law. The key observation is that, due to the this final conservation law, the energy $G_{S_W^1}$ can be updated based upon the mean of $\langle s_W^1 \rangle$ or, equivalently, the second moment $\langle s_1^1 s_2^1 \rangle$. This directly relates the energy w_{ij} to the species energy $G_{W_{ij}^1}$ of an ECBM. Similarly, the bias terms correspond to the species' energies $G_{S_i^0}$ and $G_{S_i^1}$ which are updated based upon the first moments of S_i^0 and S_i^1, respectively. Further, with this constraint, equation (9) takes the exact form of the classical Boltzmann machine learning rule.

Additionally, in our past work, we developed a rudimentary clamping process by which a subset of species in the dbCRN is held constant for conditioning (possibly "turning off" a subset of reactions when some species are clamped to zero). However we observe that our previous clamping construction is fragile because it requires that the reachability class of the unclamped species remains unchanged, which will only be true for very carefully designed dbCRNs. Finally, we note that these methods provided an *in silico* methodology for training some dbCRNs.

In this work, we extend the ideas of clamping and learning to all dbCRNs and show how they can be implemented in a purely chemical setting. First, we will provide a general framework for understanding how restricting the reachability class of a dbCRN enables the production of complex distributions. Then, we will provide a new definition of clamping which is broadly applicable to any dbCRN. Finally, we show how the learning rule (4.11) can be implemented by coupling non-detailed balanced reactions to a dbCRN. This ultimately constructs an autonomous chemical learning system capable of representing complex distributions encountered in its environment.

4.5 Effective Use of Hidden Species Requires Reachability Entanglement

In this section, provide a unifying framework for understanding how some dbCRNs are able to produce equilibrium distributions which are far from independent product Poisson distributions. First, we will show that restricting the reachability class of dbCRNs is essential to producing far-from-Poisson distributions. Furthermore, when using hidden units to increase the representational power of a dbCRN, the reachability class of these units must be "entangled" with the reachability class of the visible units for the hidden units to have any effect on the visible distribution.

As a reference case, consider a dbCRN with reactions such that the entire positive integer lattice is reachable: $\Gamma^{\rho}_{s^0} = \mathbb{Z}^{|S|}_{\geq 0}$ with equilibrium distribution $\rho(s)$. Then all the species are independent of each other because ρ is a product of independent Poisson distributions. *Proof:* Rewrite Equation (4.8) into product form to show independence.

$$\rho(s) = \frac{\prod_i e^{-\mathcal{G}_i(s_i)}}{\sum_{s\in\mathbb{Z}^n_{\geq 0}} \prod_i e^{-\mathcal{G}_i(s_i)}} = \frac{\prod_i e^{-\mathcal{G}_i(s_i)}}{\prod_i \sum_{s_i\in\mathbb{Z}_{\geq 0}} e^{-\mathcal{G}_i(s_i)}} = \prod_i \frac{e^{-\mathcal{G}_i(s_i)}}{\sum_{s_i\in\mathbb{Z}_{\geq 0}} e^{-\mathcal{G}_i(s_i)}} = \prod_i \rho_i(s_i).$$

Notice that switching the order of the sum and product in the denominator is only possible because the entire positive integer lattice is reachable. An example of such a CRN, $\emptyset \rightleftharpoons S_i$ $i \in \{1, 2, 3\}$, is illustrated in Figure 4.1A. In general, the partition function Z cannot necessarily be factored this way.

Now, consider a second dbCRN with the same species and the same species energies but different reactions and a different reachability class $\Gamma^{\pi}_{s^0} \subset \Gamma^{\rho}_{s^0}$ and equilibrium distribution π. How well a product Poisson ρ is approximated by the distribution π can be measured using the relative entropy:

$$\mathbb{D}(\pi \mid\mid \rho) = \sum_{s\in\Gamma^{\rho}_{s^0}} \pi(s) \log \frac{\pi(s)}{\rho(s)} = \sum_{s\in\Gamma^{\rho}_{s^0}} \frac{e^{-\mathcal{G}(s)}}{Z^{\pi}} \log \frac{Z^{\rho}}{Z^{\pi}} = \log(\frac{Z^{\rho}}{Z^{\pi}}) = \log(1 + \frac{Z^{-}}{Z^{\pi}}). \tag{4.14}$$

Here, Z^{ρ} and Z^{π} are the partition functions of ρ and π, respectively, and $Z^{-} = Z^{\rho} - Z^{\pi} > 0$. Note that we are using the convention $\pi(s) = 0 \,\forall\, s \notin \Gamma^{\pi}_{s^0}$ and $0 \log 0 = 0$. Due to the logarithm, for π to be far from product Poisson, a heavily weighted subset of states must be unreachable so that $Z^{-} \gg Z^{\pi}$. Figure 4.1B illustrates how restricting the reachability class with linear conservation laws induces correlations using the dbCRN is $S_1 \rightleftharpoons S_2 \rightleftharpoons S_3$ as an example.

More generally, the combination of marginalization and restricting the reachability class can be employed to increase the relative entropy between the equilibrium

distribution of any dbCRN, $\sigma(v)$, and the equilibrium distribution of another dbCRN with the same species V and G as well as additional hidden species U such that the equilibrium is $\pi(v,u)$. Assume that all the species V have the same energies in both dbCRNs and that the reachable states in the first dbCRN are also reachable in the second dbCRN, for some value of the hidden species u: $\Gamma^{\sigma}_{v(0)} \cap \Gamma^{\pi}_{v(0),u(0)} = \Gamma^{\sigma}_{v(0)}$ where the initial condition is $s^0 = v^0, u^0$. The marginal distribution of the second dbCRN is:

$$\pi(v) = \sum_{u \in \gamma(v)} \pi(v,u) \qquad \gamma(v) = \{u \text{ s.t. } v' = v \;\forall\; (u,v') \in \Gamma^{\pi}_{s^0}\}. \tag{4.15}$$

Here the function γ produces a set of the reachable hidden states u given a visible state v with the dependence on the reachability class and initial condition implied in the function arguments for brevity. The relative entropy can then be written:

$$\mathbb{D}(\pi(v) \,||\, \sigma(v)) = \sum_{v \in \Gamma^{\sigma}} \pi(v) \log \frac{\pi(v)}{\sigma(v)} = \sum_{v} \left(\sum_{u \in \gamma(v)} \pi(v,u) \right) \log \frac{\sum_{u \in \gamma(v)} \pi(v,u)}{\sigma(v)} \tag{4.16}$$

$$= \sum_{v \in \Gamma^{\sigma}} \left(\sum_{u \in \gamma(v)} \frac{e^{-\mathcal{G}(v)-\mathcal{G}(u)}}{Z^{\pi}} \right) \log \frac{Z^{\sigma} \sum_{u \in \gamma(v)} e^{-\mathcal{G}(v)-\mathcal{G}(u)}}{Z^{\pi} e^{-\mathcal{G}(v)}} \tag{4.17}$$

$$= \sum_{v \in \Gamma^{\sigma}} \frac{e^{-\mathcal{G}(v)}}{Z^{\pi}} \left(\sum_{u \in \gamma(v)} e^{-\mathcal{G}(u)} \right) \log \left(\frac{Z^{\sigma}}{Z^{\pi}} \sum_{u \in \gamma(v)} e^{-\mathcal{G}(u)} \right) \tag{4.18}$$

$$= \sum_{(v,u) \in \Gamma^{\pi}} \pi(v,u) \log \left(\frac{Z^{\sigma}}{Z^{\pi}} \sum_{u \in \gamma(v)} e^{-\mathcal{G}(u)} \right). \tag{4.19}$$

If $\gamma(v)$ is constant for all v, the restrictions on the reachability class are independent of the visible species, allowing the partition function to be factored $Z^{\pi} = Z^{\sigma}(\sum_u e^{-\mathcal{G}(u)})$. This simplifies the previous equation to 0 showing that marginalization requires restricted reachability classes to modify distributions:

$$\gamma(v) = \text{const} \implies \mathbb{D}(\pi(v) \,||\, \rho(v)) = \log \frac{Z^{\sigma}(\sum_u e^{-\mathcal{G}(u)})}{Z^{\pi}} = 0. \tag{4.20}$$

A conceptual way to interpret this result is that in dbCRNs, marginalizing over the species U does not effect the species V if the reachable states of V and U are independent. This illustrates that complex reachability restrictions on the hidden species are not enough: for them to provide extra power for representing complex distributions, the reachability of the visible and hidden species must be entangled.

Figure 4.1C depicts the equilibrium distribution of a dbCRN which marginalizes over many hidden species and extensively restricts reachability. Each pixel of the image is represented by a unique binary hidden species $P_{x,y}$. By marginalizing over these hidden species, any finite distribution can be created [235]. More specifically, this CRN can be written compactly as $\{P_{x,y} \rightleftharpoons P_{x+1,y} + S_1, \quad P_{x,y} \rightleftharpoons P_{x,y+1} + S_2\}$ where x and y are indexes of each pixel and the energies of $P_{x,y}$ have been tuned to produce the desired image. This construction works because the pixel species are highly entangled with each other—only one pixel can be present at a time—and also with the visible species—each visible count (s_1, s_2) corresponds to a unique pixel species P_{s_1,s_2}.

In summary, dbCRNs can represent a diverse set of distributions. When a dbCRN is unconstrained with species' counts allowed to take any value, all dbCRNs will have Poisson equilibrium distributions with species' means determined by their energies. Species may then be coupled together via conservation laws which can induce correlations. Furthermore, increasingly complex conservation laws have the potential to dramatically constrain the reachability class to enable the production of distributions with very rich structure. In some cases, the reachability class can be constrained via auxiliary hidden species which have their reachable states entangled with the reachable states of the visible species. In such cases, marginalization over the hidden species may produce even more complex distributions on the visible species. Finally, we comment that marginalization occurs implicitly when two chemical systems interact. Consider subsystem A with species $\mathcal{S}^A = (V, U^A)$ which observes a different subsystem B with species $\mathcal{S}^B = (V, U^B)$. If A observes the species V long enough, it implicitly observes the marginal distribution over the unobserved species U_B.

4.6 Inference with Detailed Balanced CRNs

As referenced in the Background Section 4.4 on Boltzmann machines, general purpose inference is incredibly powerful and can be used for a wide range of computational tasks. In the following sections, we show that dbCRNs are similarly capable of inference. To do this, we define a new notion of clamping which overcomes the limitations of our previous work [187] and is applicable to all dbCRNs. We then show how this clamping can be implemented by an auxilary set of chemical species. In this section, each CRN will be assumed to have its species partitioned into disjoint sets of free and clamped species, $\mathcal{S} = (\mathcal{S}^F, \mathcal{S}^C)$.

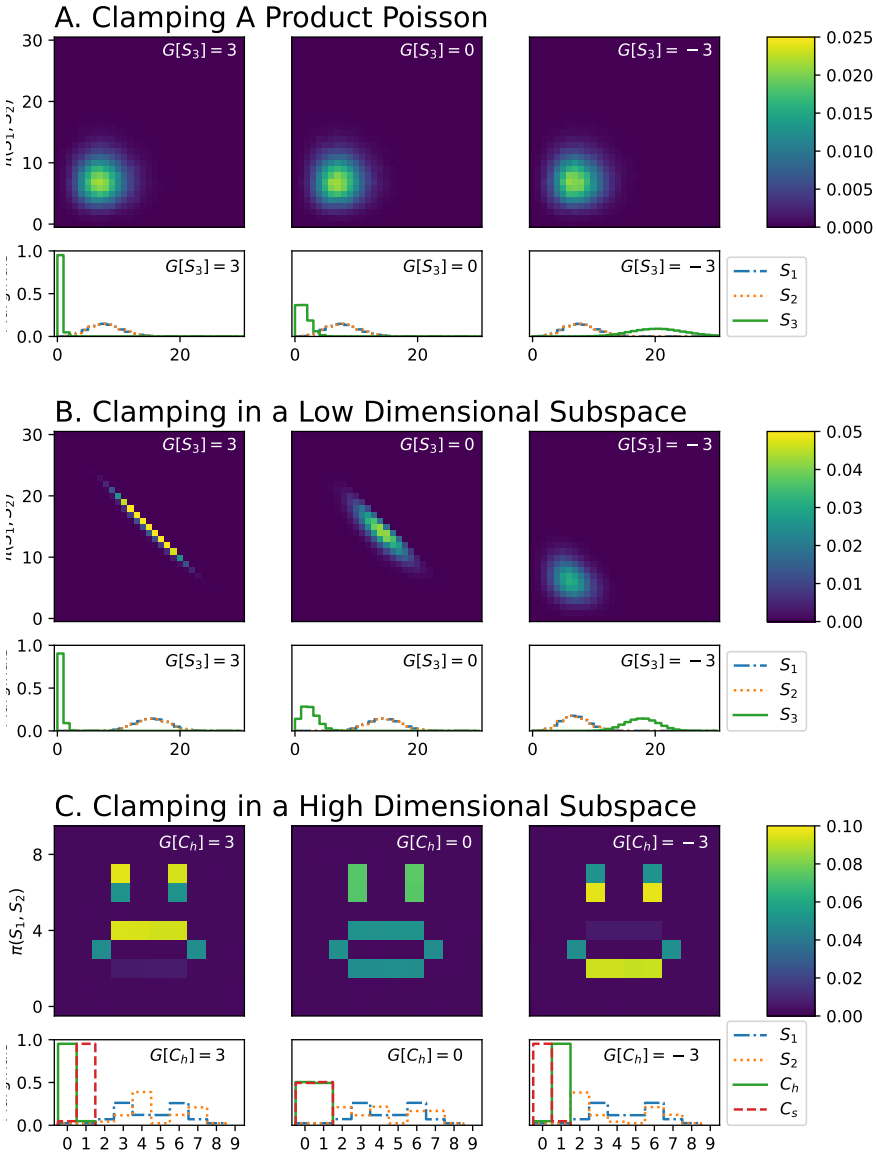


Figure 4.2: The dbCRNs used in panels A and B are shown in Fig 4.1. C. This dbCRN shows how clamping a single species can induce correlated changes in many species. The binary species C_h and C_s which control whether the happy or sad pixels are active. C_h and C_s can also inter-convert.

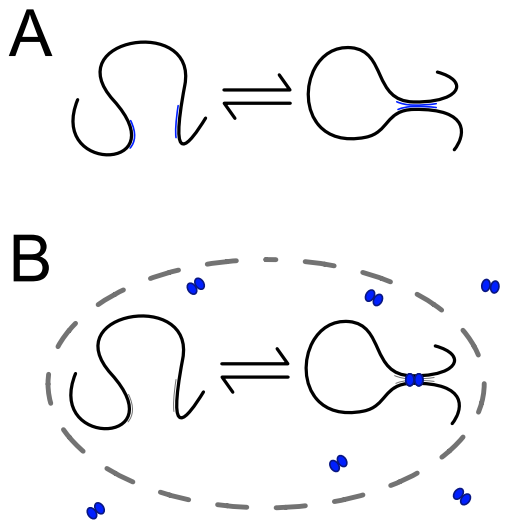


Figure 4.3: **A.** A cartoon of energy clamping the reaction $S_{open} \rightleftharpoons S_{loop}$. The reaction can be pushed in either direction by changing the sequence depicted in blue which can tune the relative energies $\Delta_{S_{open}}$ and $\Delta_{S_{loop}}$. **B.** A cartoon of potential clamping where S_{open} is connected to a potential species P resulting in the reaction: $P + S_{open} \rightleftharpoons S_{loop}$. Changing $[P]$ is the equivalent to changing Δ. The gray lines indicate that S is in a small volume relative to P. P could be many kinds of chemical species such as a transcription factor, small molecule, or even a salt ion such as magnesium.

Energy Clamping

We define a new kind of clamping which is applicable to all dbCRNs. We call this process energy clamping because it works by modulating the energies of chemical species illustrated in Figure 4.3A. The energy clamped dbCRN $\mathcal{D}_{G^C}$ has the same species and reactions as $\mathcal{D}_G$ but different energies and hence different reaction rates:

$$G_i^C = \begin{cases} G_i & S_i \in \mathcal{S}^F \\ G_i + \Delta_i & S_i \in \mathcal{S}^C \end{cases} \tag{4.21}$$

where Δ_i are the changes in energy for each clamped species. The clamped dbCRN has a new equilibrium distribution $\pi_C(s)$:

$$\pi_C(s) = \frac{e^{-\mathcal{G}^C(s)}}{Z^C} \qquad Z^C = \sum_{s \in \Gamma_{s0}} e^{-\mathcal{G}^C(s)} \qquad \mathcal{G}^C(s) = \sum_i G_i^C s_i + \log s_i! \tag{4.22}$$

Here we are treating the energy changes Δ as tunable parameters which control the means of our species. Unlike traditional clamping of a Boltzmann machine or the construction from the CBM paper, energy clamping does not hold the value of a species fixed. Instead, energy clamping can be interpreted as holding the mean

of the clamped species fixed while allowing for fluctuations. These fluctuations are important because they ensure that the reachability class is preserved. This is most properly interpreted as conditioning on the mean via the conditional limit theorem [247]:

Theorem: Energy clamping is equivalent to conditioning upon the mean $\langle s^C \rangle_\pi$ being equal to $\overline{c}$:

$$\pi_C(s^F, s^C) = \pi(s^F, s^C \mid \langle s^C \rangle_\pi = \overline{c}). \tag{4.23}$$

The notation $\pi(s^F, s^C \mid \langle s^C \rangle_\pi = \overline{c})$ indicates that the distribution π sampled to produce an empirical distribution with mean $\langle s^C \rangle_\pi \overline{c}$. In the proof, this is made rigorous as a limiting case of an infinite sequence of samples. Finally, it is important to note that we have implicitly chosen Δ such that $\langle s^C \rangle_{\pi_C} = \overline{c}$.

Proof:

Claim 1: First, we will show that the distribution $P = \pi_C(s^F, s^C)$ minimizes the relative entropy $\mathbb{D}(P \mid\mid \pi)$ subject to the constraint that $\langle s^C \rangle_P = \overline{c}$ using the method of Lagrange multipliers. We first define the functional we wish to minimize

$$J(P) = \mathbb{D}(P \mid\mid \pi) + \sum_s P(s)\Delta \cdot s + \alpha P(s) \tag{4.24}$$

where $\Delta \in \mathbb{R}^N$ will constrain the means and $\alpha \in \mathbb{R}$ will normalize the distribution. Optimize this function with respect to each component of the function space $P_s = P(s)$:

$$\begin{aligned}
\frac{\partial J}{\partial P_s} = \frac{\partial}{\partial P_s} \sum_{s'} P(s') \log \frac{P_{s'}}{\pi_{s'}} + P_{s'} \Delta \cdot s' + \alpha P_{s'} \qquad &= 0 \\
= \frac{\partial}{\partial P_s} [P_s \log \frac{P_s}{\pi(s)} + P_s \Delta \cdot s + \alpha P_s] \qquad &= 0 \\
= \log \frac{P_s}{\pi(s)} + 1 + \Delta \cdot s + \alpha \qquad &= 0 \\
\implies P_s^* = C\pi(s)e^{-\Delta \cdot s} = \frac{1}{Z_C} e^{-\sum_i G_i^C s_i + \log s_i!} = \pi_C(s)
\end{aligned}$$

where the C is a normalizing constant which becomes part of the partition function Z_C and Δ is chosen so that $\langle s^C \rangle_{\pi_C} = \overline{c}$. In practice, the moment learning rule (4.11) can be used to find Δ provided that $\overline{c} \in \overline{\Gamma}_{s^0}$ where $\overline{\Gamma}_{s^0}$ is the convex hull around Γ_{s^0} because the mean does not have to be an integer value. The conceptual meaning of this result is that π_C is optimal choice of distribution, with expected value $\langle s^C \rangle = \overline{c}$

for the clamped species S^C, to encode π.

We are now ready to invoke the conditional limit theorem (See Cover and Thomas Elements of Information Theory theorem 11.6.2 p. 371 [247]). This theorem states that if one were to observe a *very rare* sample of π with mean $\overline{c}$ the distribution of that sample would be given by π_C. Importantly, this provides a rigorous definition of what it means to condition upon a distribution having a mean value which allows us to interpret energy clamping as a form of conditioning and hence inference. We note that this theorem has been applied to understand similar chemical systems previous by Virinchi et al. [83].

Define: $\mathcal{P}$ to be the probability simplex (set of distributions) over the reachability class of the CRN Γ_{s^0}:

$$\mathcal{P} = \left\{ \mu : (s^F, s^C) \to \mathbb{R} \quad \text{s.t.} \sum_{s^F, s^C \in \Gamma_{s^0}} \mu(s^F, s^C) = 1 \right\}.$$

Note that this is a convex subspace of a Banach function space [247].

Define: The space of distributions in the probability simplex, $E \subseteq \mathcal{P}$, where the expected value of the species $s^C = \overline{c}$:

$$E = \{\mu \in \mathcal{P} \text{ s.t. } \langle s^C \rangle_\mu = \overline{c}\}.$$

Here $\langle \cdot \rangle_\mu$ denotes the expected value relative to the distribution μ.

Claim 2: E is convex. *Proof:* the expected value $\langle \cdot \rangle_\mu$ is a linear operator so the constraint $\mathbb{E}_\mu[s^C] = \overline{c}$ defines an affine subspace of P. The intersection of an affine subspace and a convex space is also convex, so E is convex [247].

We now define a set of N samples from the distribution π and the empirical *distribution of types* derived from the sample sequence.
Define: $\mathbb{S}^n = s^1, ..., s^n$ are a vector of n i.i.d. samples from the distribution π.

Define: $\mathbb{P}_{\mathbb{S}^n}$ is the empirical distribution of types derived from the sequence of samples $\mathbb{S}^n$:

$$\mathbb{P}_{\mathbb{S}^n}(s) = \frac{\sum_i \mathbb{I}(s^i, s)}{n}$$

where $\mathbb{I}(s', s)$ denotes the identity operator.

We can now apply the condition limit theorem which states that for any convex set E and empirical distribution of types $\mathbb{P}_{\mathbb{S}^n}$ derived from a distribution π, in the limit $n \to \infty$, the probability of a sample s^i having the value s is given by:

$$\lim_{n\to\infty} \mathbb{P}(s^i = s \mid P_{X^n} \in E) = P^* \qquad P^* = \text{argmin}_{P\in E}\mathbb{D}(P \mid\mid \pi) = \pi_C.$$

In words, consider drawing a set of n samples $\mathbb{S}^n$ from a detailed balanced CRN with equilibrium distribution π where the means of a subset of the species s^C of these samples are given by $\langle s^C \rangle_{\mathbb{P}_{\mathbb{S}^n}} = \overline{c}$. In the limit $n \to \infty$, the samples will be distributed according to a distribution, $P^*(s^F, s^C)$, which can be produced by another detailed balanced CRN with equilibrium distribution $P^* = \pi_C$ where the species' energies $G_i^C = \Delta_i + G_i$ are chosen such that $\langle s^F \rangle_{\pi_C} = \overline{c}$. This technical description allows us to understand the clamped distribution as being a special kind of conditioning on the mean of π:

$$\pi_C(s^F, s^C) = \pi(s^F, s^C \mid \langle s^C \rangle = \overline{c}).$$

Hence energy clamping produces a conditional distribution and therefore can be interpreted as a kind of inference. ■

The idea of conditioning on the mean versus holding the value of a species constant is analogous to the difference between microcanonical ensembles, where free energy is constant, and the canonical ensemble where the mean free energy is fixed, but allowed to fluctuate. Energy clamping modulates the energy of a species which holds the steady-state mean value constant but the actual count of that species is still allowed to fluctuate. The correctness of energy clamping can also be seen more simply through the following theorem on the conditional distributions:

Theorem: Energy clamping produces the same conditional distributions between π and π_C when conditioned on the species S^C taking the exact value c:

$$\pi(s^F \mid s^C = c) = \pi_C(s^F \mid s^C = c). \tag{4.25}$$

Proof:

$$\pi(s^F \mid s^C = c) = \frac{\pi(s^F, c)}{\pi(c)} = \frac{e^{-\mathcal{G}(s^F)-\mathcal{G}(c)}}{\sum_{s^F} e^{-\mathcal{G}(s^F)-\mathcal{G}(c)}} = \frac{e^{-\mathcal{G}(s^F)}}{\sum_{s^F} e^{-\mathcal{G}(s^F)}}$$

$$\pi_C(s^F \mid s^C = c) = \frac{\pi_C(s^F, c)}{\pi_C(\overline{c})} = \frac{e^{-\mathcal{G}^C(s^F)-\mathcal{G}^C(c)}}{\sum_{s^F} e^{-\mathcal{G}^C(s^F)-\mathcal{G}^C(c)}} = \frac{e^{-\mathcal{G}(s^F)}}{\sum_{s^F} e^{-\mathcal{G}(s^F)}}.$$

Here the final step notes that $\mathcal{G}^C(s^F) = \mathcal{G}(s^F)$ by definition. This result shows that the clamped dbCRN has the correct conditional distribution when conditioned on any $s^C = c$. ■

Energy clamping provides a framework to hold the species of an arbitrary detailed balanced CRN around a value by modulating the energy vector $G \Rightarrow G^C$. In the case where S_i can take any value on the integer lattice, energy clamping can be thought of as tuning the mean of S_i directly as seen in the lower plot of Figure 4.4A. In the case where S_i is constrained via a conservation class to some minimum and maximum value, energy clamping may be better thought of as pushing S_i towards one of its extreme values. Furthermore, when reachable states of the clamped species are entangled with the reachable states of free species, energy clamping induce changes in the distribution of free species illustrated clearly in Figure 4.4B. In these cases, energy clamping may perform computationally challenging inferential tasks. For example, in Figure 4.4C, clamping a single species C_h is able to induce dramatic changes in the distribution of the visible species causing the transition between a frown face to a smiley face. Importantly, the energy clamping construction does not change the reachability class of the underlying dbCRN which allows it to apply to any species in any dbCRN, regardless of reactions in the CRN.

Clamping with A Potential Bath

Energy clamping provides a formal framework by which dbCRNs perform inference. However, implementing energy clamping directly would involve carefully modifying the internal energies of different species in a dbCRN. Although theoretically possible, this is not an easy parameter to tune experimentally. In this section, we construct a dbCRN in a small volume v coupled to a large external bath at volume $V \gg v$ to produce a chemical system which implements energy clamping. We call these CRNs *potentiated* dbCRNs. A physical cartoon of a potentiated CRN can be found in Figure 4.3. We will show that the species in the bath, denoted *potential* species, produce a chemical potential equivalent to energy clamping. Notice that this model is a hybrid model in the sense that S_i are measured in counts and vary stochastically but P_i varies continuously and is measured in concentration. This will be derived as the limiting case of a completely stochastic CRN when the count of P_i becomes large. Ultimately, we will couple the potentiated dbCRN back to the

bath through a set of (non-detailed balanced) chemical reactions producing a CRN which will be able to autonomously learn from its environment.

Any dbCRN, $\mathcal{D}_G$, can be converted to a potentiated dbCRN, $\mathcal{D}_G^{\mathcal{P}}$, by adding an additional set of potential species P_i which are coupled to each (or a subset) of the species $S_i \in \mathcal{S}^C$ and held at a constant concentration $[P_i] = \frac{p_i}{V}$ where p_i is a count and V is a volume parameter. Formally, this can be viewed as replacing all instances of S_i in each reaction (both inputs and outputs) with $S_i + P_i$.[2] For example, the dbCRN with reactions $2S_i \rightleftharpoons S_j + S_k$ becomes a potentiated dbCRN with reactions $2S_i + 2P_i \rightleftharpoons S_j + S_k + P_j + P_k$ in the case that all species are clamped. Formally, given a dbCRN $\mathcal{D}_G = (\mathcal{S}, \mathcal{R}, k)$, we will denote the potentiated dbCRN $\mathcal{D}_G^{\mathcal{P}} = (\mathcal{S} \cup \mathcal{P}, \mathcal{R}_{\mathcal{P}}, k)$ where $\mathcal{P}$ are the potential species, $\mathcal{R}_{\mathcal{P}}$ are the reactions from $\mathcal{R}$ modified to include potentials, and the rates k are unchanged.

Theorem: The fully stochastic description of a potentiated dbCRN $\mathcal{D}_G^{\mathcal{P}}$ is detailed balanced with equilibrium distribution given by:

$$\pi^P(s, p) = \frac{1}{Z} \exp -(\sum_i G_i s_i + G_i^P p_i + \log s_i! + \log p_i!). \tag{4.26}$$

Proof:

Consider the reaction $\sum_i I_i S_i \overset{k^+}{\underset{k^-}{\rightleftharpoons}} \sum_i O_i S_i$. After being connected to the potential baths, this reaction becomes:

$$\sum_i I_i(S_i + P_i) \overset{k^+}{\underset{k^-}{\rightleftharpoons}} \sum_i O_i(S_i + P_i) \tag{4.27}$$

$$\frac{k^+}{k^-} = e^{-\sum_i (G_i + G_i^P)(O_i - I_i)} \tag{4.28}$$

$$\rho^+(x) = k^+ \prod_i \frac{s_i! p_i!}{(s_i - I_i)!(p_i - I_i)!} \qquad \rho^-(x) = k^- \prod_i \frac{s_i! p_i!}{(s_i - O_i)!(p_i - O_i)!}. \tag{4.29}$$

I and O are the reactions inputs and outputs which are the same stoichiometry for each species S_i and its potential P_i by construction. G_i and G_i^P are the energies of the species S_i and P_i, respectively. $\rho^{\pm}(x)$ are the propensities of the forward and backward reactions. Note that the rate constants (in units of per second) remain unchanged by the addition of the potential species. Next, we show that

[2] Technically, the potential species P_i could go on either or both sides of the reaction - we consider just one side for simplicity.

$\pi^P(s+O-I)\rho^-(s+O-I) = \pi^P(s)\rho^+(s)$, which is a well known definition of detailed balance:

$$\frac{\rho^+(s)}{\rho^-(s+O-I)} = \frac{k^+}{k^-}\frac{\prod_i \frac{s_i!p_i!}{(s_i-I_i)!(p_i-I_i)!}}{\prod_i \frac{(s_i+O_i-I_i)!(p_i+O_i-I_i)!}{(s_i-I_i)!(p_i-I_i)!}} \tag{4.30}$$

$$= \prod_i \frac{e^{(G_i+G_i^P)(I_i-O_i)}s_i!p_i!}{(s_i+O_i-I_i)!(p_i+O_i-I_i)!} \tag{4.31}$$

$$\frac{\pi^P(s+O-I,p+O-I)}{\pi^P(s)} = \prod_i \frac{s_i!p_i!e^{-G_i(s_i+O_i-I_i)-G_i^P(p_i+O_i-I_i)}}{(s_i+O_i-I_i)!(p_i+O_i-I_i)!e^{-G_is_i-G_i^Pp_i}} \tag{4.32}$$

$$= \prod_i \frac{e^{(G_i+G_i^P)(I_i-O_i)}s_i!p_i!}{(s_i+O_i-I_i)!(p_i+O_i-I_i)!}. \tag{4.33}$$

Because the dbCRN with potential species is detailed balanced, we can simply apply the product Poisson formula (4.8) to get the equilibrium distribution (4.26). ■

Remark: it is not strictly necessary for each S_i to have a unique potential species P_i; many S_i could share the same P_j (a universal potential species) if they are all clamped together simultaneously. Remark: if individual reactions are connected to a potential, instead of each species, the CRN is no longer detailed balanced and most of the results in this paper are not expected to hold. The physics of such systems have been studied elsewhere [90, 170].

Theorem: Let $\mathcal{D}_G^{\mathcal{P}}$ be a potentiated dbCRN with equilibrium distribution $\pi^P(s)$ and $\mathcal{D}_{G^C}$ be an energy clamped dbCRN with equilibrium distribution $\pi^C(s)$. $\pi^P(s) = \pi^C(s)$ provided that:

$$\Delta_i = \mu_i = G_i^P + \log[P_i]. \tag{4.34}$$

Here the energy difference Δ_i is equated to the chemical potential of P_i, commonly denoted μ_i where G_i^P is the energy of P_i.

Proof:

Claim 1: $\pi^P(s,p)$ is just a function of s.

$$\pi^P(s) = \frac{1}{Z}e^{-\sum_i G_is_i - G_i^P(s_i-s_i^0+p_i^0)-\log s_i!-\log(s_i-s_i^0+p_i^0)!}. \tag{4.35}$$

Proof: Given an initial condition (s^0,p^0), it is clear that the change in s_i and p_i are coupled by the construction of the CRN: $p_i = s_i - s_i^0 + p_i^0$. Equation (4.35) results from inserting this conservation law into (4.26). Furthermore, note that the terms $G_i(s_i-s_i^0+p_i^0)-log(s_i-s_i^0+p_i^0)!$ are a kind of stochastic chemical potential. ■

Claim 2: In the limit $p_i \gg 0$ and $p_i \gg s_i - s_i^0$, π^P has the simplified form:

$$\pi^P(s) \approx \frac{1}{Z} e^{-\sum_i G_i s_i - G_i^P s_i - \log s_i! - s_i \log p_i^0} \tag{4.36}$$

Proof: Using Stirling's approximation, $\log((s_i - s_i^0 + p_i^0)!) \approx (s_i - s_i^0 + p_i^0)\log(s_i - s_i^0 + p_i^0)$. If $p_i^0 \gg s_i - s_i^0$ then $\log(s_i - s_i^0 + p_i^0) \approx \log p_i^0$. Finally, the constant terms $G_i^P(p_i^0 - s_i^0)$ and $(p_i^0 - s_i^0)\log p_i^0$ will factor out between the Gibbs factor and the partition function. ■

Claim 3: Equating the exponential terms of (4.36) to the exponential terms of an energy clamped CRN (4.22) term by term results in the relation:

$$G_i^C = G_i + G_i^P + \log p_i^0 = G_i + \mu_i. \tag{4.37}$$

where μ_i is the chemical potential of P_i and is retrieved by changing from units of counts to units of concentration. *Proof:* this follows easily from some simple algebraic manipulation and proves the theorem. ■

Together, these claims show how a potentiated dbCRN, when analyzed as a fully stochastic CRN, is detailed balanced with a product-Poisson equilibrium distribution. In the hybrid model, the potential species are converted to concentrations in a thermodynamic limit and held constant. Then, the concentrations $[P_i]$ can be folded into the energies G_i^C of an energy clamped dbCRN.

Example: Potential Clamping and Polymer Conformations

Consider a piece of DNA which can form two mutually exclusive states where the formation of each state is mediated by some kind of DNA-binding protein. These mutually exclusive states could be caused by DNA loops which occur commonly due to transcription factors in bacteria and eukaryotes [248] or by other kinds of DNA packaging such as histones [249]. Here, we show that, under the assumption of DNA conformations being at quasi-equilibrium, this kind of system can be interpreted as a form of potential clamping where the DNA states are the species S_i and the binding molecules (proteins) are equivalent to potential species P_i. We examine a simple two-state model that could potentially be built synthetically in the lab. This model includes three DNA conformations: S_0 is the free, unbound, molecule. S_1 is a molecule of DNA wrapped around a Histone H such that a binding region a is not exposed. S_2 is the same molecule of DNA where a protein F mediates looping

between the binding site a and another binding site b in such a way that H can no longer bind. This can be written as the following CRN:

$$S_0 + H \rightleftharpoons S_1 \qquad S_0 + F \rightleftharpoons S_2. \tag{4.38}$$

Notice that H and F both play the role of potential species to S_1 and S_2, respectively, in a way slightly different from the previous construction. Instead of being coupled as inputs whenever S_i are input into a reaction, these species are coupled as outputs whenever S_i are coupled to a reaction. However, this is equivalent to our previous construction because the key value of potential baths is to provide a way to tune the energy difference between inputs and outputs. In the limit of many more molecules of F and H than S_i, this system can be written compactly as three unimolecular reactions with two potentiated species:

$$S_1 \rightleftharpoons S_0 \rightleftharpoons S_2. \tag{4.39}$$

The energies of each species are given by:

$$G[S_0] = G_0 \qquad G[S_1] = G_1 - \mu_H \qquad G[S_2] = G_2 - \mu_F. \tag{4.40}$$

By increasing the chemical potentials μ_H and μ_F, the system can be pushed into states S_1 and S_2, which, by virtue of the underlying CRN, will be anti-correlated. We note that it would be possible to generalize this kind of argument to DNA molecules which may take on many conformations by binding to a wide range of factors. The key assumption is that the binding and unbinding reactions occur much more quickly than any change in the amount of the binding proteins. This seems plausible in many biological settings where binding and unbinding of proteins to DNA occur much more quickly than transcription and translation. Additionally, we note that it is very natural to ascribe every DNA conformation S_i a unique energy G_i—although the same underlying molecule forms each of the states, the entropy of a conformation is widely tunable by varying the degrees of freedom of the polymer and mechanical energy could be stored inside bent or wound DNA strands [250].

4.7 Autonomous Learning CRNs

The potentiated dbCRN implementation provides a mechanism to implement energy clamping and hence inference without having to modify the underlying energies of chemical species (which are in a certain sense physical constants). Instead, the concentrations of the potential species can be controlled. In this section, we first derive a non-detailed balanced potential clamping CRN which can tune the

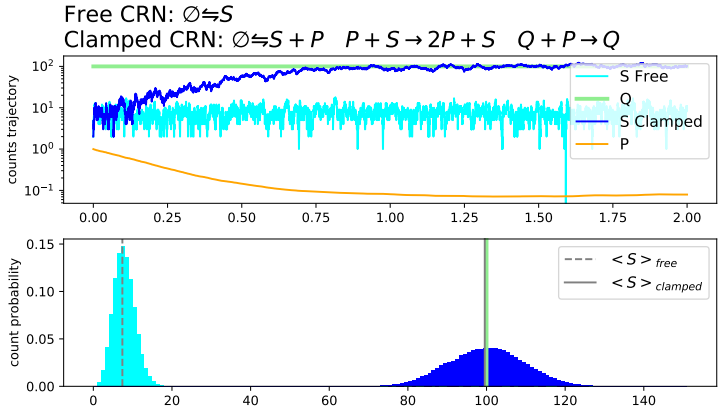


Figure 4.4: Dynamics of the potential clamping CRN C_G^Q derived from the dbCRN $\emptyset \rightleftharpoons S$ where the mean of S is clamped to the value Q.

concentration of the potential species $[P_i]$ such that the mean of S_i matches the mean of a target species Q_i which may be thought of as the environment. This CRN is shown to implement the learning rule (4.5) and provides a chemical mechanism by which clamping may occur dynamically. Then, by combining a number of these clamping CRNs together with potentiated dbCRN modules and an environment representing data, we show how any potentiated dbCRN can be embedded into a learning CRN architecture which is able to learn potentials for both hidden and visible species in order to approximate an environmental distribution. We emphasize that the learning CRN is an autonomous CRN which automatically learns from the environment, or in other words, that we have rewritten a fundamental machine learning algorithm as a CRN. This will allow us to understand some of the underlying physics of learning.

Potential Clamping CRN

First, we consider a potentiated dbCRN, $\mathcal{D}_G^{\mathcal{P}} = (\mathcal{S} \cup \mathcal{P}, \mathcal{R}_{\mathcal{P}}, k)$, as defined in the previous section. We will then add the following species and reactions to $\mathcal{D}_G^{\mathcal{P}}$ and construct a new potential clamping CRN. For each potential species, $P_i \in \mathcal{P}$, add in a target species $Q_i \in \mathcal{Q}$, which are also allowed to fluctuate according to some distribution $\psi(q)$ (we are agnostic about how ψ is generated—it could be detailed balanced or non-equilibrium). Additionally, add the non-detailed balanced reactions

$\mathcal{T}^{P}_{\mathcal{S},\mathcal{Q}}$ which clamp the species $\mathcal{S}$ to the species $\mathcal{Q}$ using the potentials $\mathcal{P}$:

$$P_i + Q_i \xrightarrow{k_i^Q} Q_i \tag{4.41}$$

$$P_i + S_i \xrightarrow{k_i^S} 2P_i + S_i. \tag{4.42}$$

Denote this new CRN $C_G^{\mathcal{Q}} = (\mathcal{S} \cup \mathcal{P} \cup \mathcal{Q}, \mathcal{R}_{\mathcal{P}} \cup \mathcal{T}^{P}_{\mathcal{S},\mathcal{Q}}, k)$ letting the mapping between the $\mathcal{Q}$ species, $\mathcal{P}$, and $\mathcal{S}$ species be implicit. $C_G^{\mathcal{Q}}$ is no longer detailed balanced; instead of an equilibrium distribution, it will in general have a non-equilibrium steady state, $\mathbb{P}(s, q, [P])$. The dynamics of this CRN could be understood stochastically provided that the counts p_i are high enough for (4.36) to be accurate. However, it is more analytically and numerically tractable to approximate the dynamics of P_i as continuous concentrations $[P_i]$ and we provide a more rigorous discussion of when this limit is valid in the supplement. The dynamics of the learning CRN are then a hybrid of stochastic dynamics for the species S_i and Q_i given by (4.7) and deterministic dynamics for P_i given by:

$$\frac{\mathrm{d}[P_i](t)}{\mathrm{d}t} = \epsilon(k_i^S [P_i](t) s_i(t) - k_i^Q [P_i](t) q_i(t)). \tag{4.43}$$

Here, $\epsilon = V^{-1}$ must be small for the continuum approximation of P_i to be valid. Note that S_i and Q_i fluctuate as discrete stochastic counts, so this is a stochastic differential equation. It can be simplified using a quasi-equilibrium approximation $\epsilon k_i^Q, \epsilon k_i^S \ll k_r$. In this limit, the reactions (4.41—4.42) effectively have rate 0, the species P can no longer vary, and the species $\mathcal{S}$ and $\mathcal{Q}$ become independent. By construction, the species $\mathcal{S}$ are governed by only the remaining reactions $\mathcal{R}_{\mathcal{P}}$ and will have an equilibrium distribution $\pi^P(s)$. Similarly, the species $\mathcal{Q}$ vary independently according to ψ. The approximate dynamics are then:

$$\frac{\mathrm{d}[P_i](t)}{\mathrm{d}t} \approx \epsilon [P_i](t) \left(k_i^S \sum_s \pi^P(s) s_i(t) - k_i^Q \sum_q \psi(q) q_i(t) \right) \tag{4.44}$$

$$= \epsilon [P_i](t) \left(\langle s_i(t) \rangle_{\pi^P} - \alpha_i \langle q_i(t) \rangle_{\psi} \right) \tag{4.45}$$

where $\alpha_i = \frac{k_i^Q}{k_i^S}$ and is a parameter to rescale the relative means of S_i and Q_i. If $\alpha_i = 1$, this expression is precisely the learning rule for Boltzmann machines and dbCRNs (4.5) with a dynamic learning rate $\epsilon[P_i]$. This system of equations has a steady state when the right hand side is 0, namely:

$$\frac{\mathrm{d}[P_i]}{\mathrm{d}t} = 0 \implies \langle s_i(t) \rangle_{\pi^P} = \alpha_i \langle q_i(t) \rangle_{\psi}. \tag{4.46}$$

The dynamics of this system can also be solved:

$$\frac{\mathrm{d}[P_i](t)}{[P_i](t)} = \epsilon(\langle s_i(t)\rangle_{\pi^P} - \alpha_i\langle q_i(t)\rangle_{\psi})\mathrm{d}t \implies [P_i](t) = Ce^{\epsilon(\langle s_i(t)\rangle_{\pi^P} - \alpha_i\langle q_i(t)\rangle_{\psi}t} \tag{4.47}$$

where C is a constant of integration. This means that if the rates k are small enough, C_G^Q will vary $[P_i]$, effectively varying the energy parameter $\Delta_i = \mu_i = G_i^P + \log[P_i]$. For some initial conditions, this system will reach a fixed point $[P_i]_{ss}$ such that Equation (4.46) is satisfied—meaning that the mean of S_i is equal to the mean of Q_i multiplied by the scaling factor α_i. The dynamics also illustrate that learning will be fast with exponential convergence to the correct value provided there is no error in the expected value of π^P and ψ. However, as these distributions become noisier (meaning not enough time scale separation), the exponential will amplify fluctuations and learning becomes a biased random walk. Figure 4.4 illustrates the dynamics of the clamping CRN when applied to a birth-death process. We also note that the potential clamping CRN can be viewed as a form of integral feedback control which tracks a reference signal.

Derivation of the Hybrid CRN Model

Consider a potentiated dbCRN with the species S_i confined to a small volume v connected via a semi-permeable membrane to a larger volume, $V \gg v$, containing the potential species P_i. The potential species P can traverse this membrane but the S species cannot. The environmental species Q will be assumed to share the same volume u with all the potential species P. Note that we define V to include both volumes v and u. We assume that the number of P_i species is very large and *instantaneously well mixed* so that fluctuations between the volumes v and u and the volume V can be neglected. This implies that the number of P_i in the volumes v and u is on average proportional to the volume ratios and that fluctuations around this value are averaged over quickly:

$$p_i^v = \frac{v}{V}p_i \quad \text{and} \quad p_i^u = \frac{u}{V}p_i. \tag{4.48}$$

We consider the reactions (4.41) and (4.41) using stochastic mass action kinetics. Rewriting the reactions and their propensities implicitly including the volumes v

and u, we obtain:

$$P_i + Q_i \xrightarrow{k_i^Q} Q_i \qquad \rho_i^Q(q_i, p_i) = \frac{k_i^Q}{u} q_i p_i^u = \frac{k_i^Q}{V} q_i p_i = k_i^Q q_i [P_i] \tag{4.49}$$

$$P_i + S_i \xrightarrow{k_i^S} 2P_i + S_i \qquad \rho_i^S(s_i, p_i) = \frac{k_i^S}{v} s_i p_i^v = \frac{k_i^S}{V} s_i p_i = k_i^S s_i [P_i]. \tag{4.50}$$

These reactions degrade and produce just a single molecule of P_i. We can rewrite this as a change in concentration:

$$\Delta[P_i] = \pm \frac{1}{V}. \tag{4.51}$$

We can informally take a continuum limit by assuming that in some time τ, on average $\rho_i^S(s_i, p_i)$ and $\rho_i^Q(q_i, p_i)$ occur. Each of these reactions will change $[P_i]$ by $\pm\frac{1}{V}$. In the limit $\tau \to 0$, we get the ordinary differential equation:

$$\frac{d[P_i]}{d\tau} = \frac{1}{V}(\rho_i^S(s_i, [P_i]) - \rho_i^Q(q_i, [P_i])) = \frac{1}{V} k_i^S s_i [P_i] - \frac{1}{V} k_i^Q q_i [P_i]. \tag{4.52}$$

Here the parameter $\epsilon = \frac{1}{V}$. Clearly in the limit $V \to \infty$, these reactions will not change $[P_i]$. However, for any finite V, they will produce a change in concentration. In order for the deterministic description to be accurate, we need fluctuations in $[P_i]$ to be small. First we assume that fluctuations in S_i (which will directly cause fluctuations in $[P_i]$) are negligible. This is true if $s_i \ll p_i$. Indeed, if this assumption is broken, the entire learning system may not work. In particular, if the counts of p_i becomes less than the counts of s_i, a potentiated dbCRNs reachability class can change. For example, consider the reaction $S_0 \rightleftharpoons S_1 + P_1$. If $s_1 > p_1 = 0$, this reaction cannot occur preventing the formation of S_0 which would have been possible in the unpotentiated dbCRN. Second, we are relying on previous results showing that the dynamics of sufficient high-count species can be well approximated by deterministic kinetics [251]. This common approximation is valid provided the counts $p_i \gg 0$; in this limit, the standard volume expansion of the chemical master equation predicts that fluctuations around the mean go to 0 at a rate proportional to $V^{-\frac{1}{2}}$ [252]. In practice, satisfying these assumptions may require tuning of the volume V relative to the concentration of the potential species $[P_i]$ and the rates k_i^Q and k_i^S. If V is large, clamping and learning will become infinitely slow unless the rate constants k_i^Q and k_i^S are increased proportionally. However, if V is too small, fluctuations in p_i may begin to play a significant role in the clamping dynamics and could even lead to catastrophic extinction events of p_i. With an eye towards future experimental implementations of such a system, we suggest choosing V as large as possible given the rates k_i^Q and k_i^S such that learning may occur on a reasonable timescale.

Learning CRN Architecture

In this section, we show how to create an autonomous CRN capable of learning by dynamically adjusting potential species so that an internal potentiated dbCRN matches an environmental distribution. To do this, we use potential clamping reactions first to produce a potentiated dbCRN clamped to the environment and then to coupled this clamped potentiated dbCRN to a free potentiated dbCRN. We will argue that this construction is an implementation of the learning rule as a continuous online process.

The full construction is as follows. Let the environmental distribution ψ have visible species Q^V. Let $\overline{C}_G^Q = (Q \cup \overline{S} \cup \overline{\mathcal{P}} \cup \mathcal{P}, \mathcal{R}^{\overline{S}}_{\overline{\mathcal{P}},\mathcal{P}} \cup \mathcal{T}^{\overline{\mathcal{P}}}_{\overline{S},Q}, k)$ be a clamped potentiated dbCRN with a subset of the $\overline{S}$ clamped to environmental species $Q^V \sim \psi$ via the potentials $\overline{\mathcal{P}}$. A second set of potentials $\mathcal{P}$ couple $\overline{C}_G^Q$ with another clamped potentiated dbCRN $C_G^{\overline{S}} = (\overline{S} \cup S \cup \mathcal{P}, \mathcal{R}^S_{\mathcal{P}} \cup \mathcal{T}^{\mathcal{P}}_{S,\overline{S}}, k)$. The species S are clamped to the values of the species $\overline{S}$ using the second set of potentials P. This construction produces one large learning CRN $\mathcal{L}_G^Q = (Q \cup \overline{S} \cup S \cup \overline{\mathcal{P}} \cup \mathcal{P}, \mathcal{R}^{\overline{S}}_{\overline{\mathcal{P}},\mathcal{P}} \cup \mathcal{R}^S_{\mathcal{P}} \cup \mathcal{T}^{\overline{\mathcal{P}}}_{\overline{S},Q} \cup \mathcal{T}^{\mathcal{P}}_{S,\overline{S}}, k)$ illustrated in Figure 4.5A.

Although seemingly complicated, $\mathcal{L}_G^Q$ is actually implementing a version of the moment learning algorithm of Boltzmann machines (Equation 4.5) as a continuous time online process. Recall from Section 4.4 that this learning rule requires sampling the free distribution $\pi(u, v)$ and the clamped distribution $\overline{\pi}(u, v) = \pi(u \mid v)\psi(v)$. The moments of these distributions are then compared and used to update the energies. In the CRN construction, the first set of potential clamping reactions is used to compute the distribution $\overline{\pi}(\overline{s} \mid \langle \overline{s}^V \rangle = q)\psi(q)$ where the visible species S^V are clamped to samples $q \sim \psi$ of the environmental distribution. The clamped CRN is then copied to a free dbCRN with equilibrium $\pi(s \mid \langle s \rangle = \langle \overline{s} \rangle_{\overline{\pi}})$ by the second set of potential clamping reactions which update the potentials of the free CRN in real time according to the learning dynamics (Equation 4.44). An example of this process is illustrated in Figure 4.5B where a three-node chemical Boltzmann machine learns a binary representation of the steady state distribution generated by a bistable genetic toggle switch [158]. Figure 4.5C shows the environmental distribution ψ which is used to clamp the species V_Q producing an XOR distribution (Figure 4.5D). Initially, the free chemical Boltzmann machine produces a uniform distribution (Figure 4.5E). Then, after tuning the potential species of both the visible and hidden units, it ultimately produces an XOR (Figure 4.5F). Note that in these examples the parameter α is used to scale the counts of the toggle switch so they

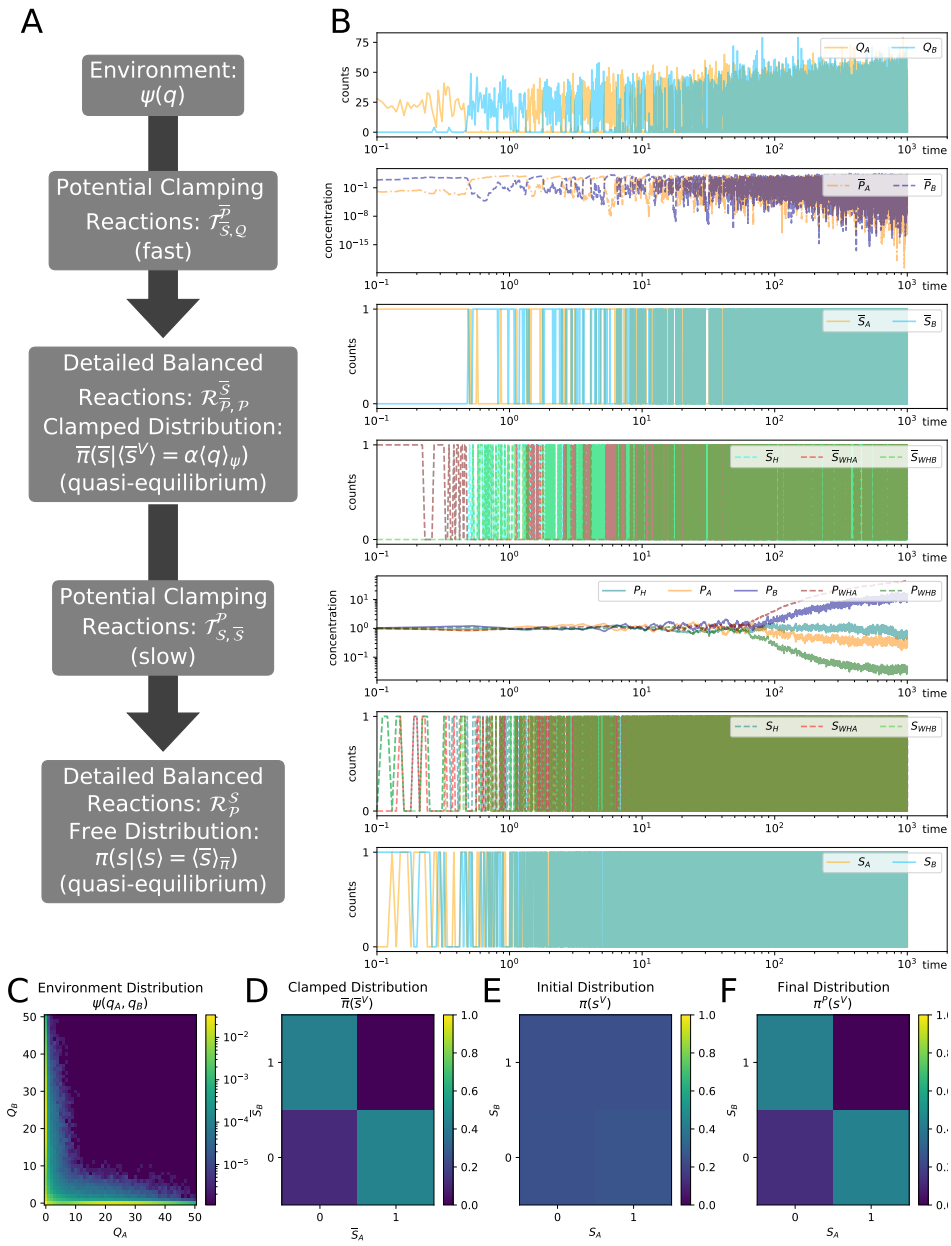


Figure 4.5: **A.** The learning architecture coupling dbCRNs and the environment together with potential clamping reactions. Arrows point from the target species to the clamped species. $\rightleftharpoons$ denote that two sets of species are at quasi-equilibrium. **B.** Trajectories from this architecture applied to learn an XOR distribution with a 3-node chemical Boltzmann machine from a bistable genetic toggle switch. **C.** The steady state distribution of the bistable toggle switch. **D.** The distribution obtained from clamping to the environment. **E.** The initial distribution of the chemical Boltzmann machine before training. F. The final distribution of the chemical Boltzmann machine after training.

can be presented by a binary variable.

Next, we describe the example of the learning system simulated in 4.5B in detail. The free and clamped CRNs, represented by species S and $\overline{S}$, respectively, are both 3-node chemical Boltzmann machines with two visible species A and B and three hidden species including a hidden node H and the weights connecting the hidden unit to the visible units W_{HA} and W_{HB}. This entire model takes the form of one large hybrid CRN with four nominal timescales; ϵ_{fast}, the environmental clamping rate; ϵ_{slow} the hidden unit clamping rate; k_{db} the detailed balanced CRN nominal rate; and k_{env} the environmental rate. These rates are separated into three timescales: $k_{env}c_{ev} \approx k_{db} * c_{db} \gg \epsilon_{fast} * C_p \gg \epsilon_{slow} * C_p$. Here, c_{ev}, c_{db} are the characteristic counts of the environment and dbCRNs, respectively. C_p is the characteristic concentration of the potential species. Due to the fact that the concentrations of the potential species are potentially unbounded, this implies that learning could fail if the potential concentrations begin becoming very high or reach 0. Such a situation could be encountered when the underlying dbCRN is not capable of learning the environmental distribution (e.g. no steady state exists for the potential species).

The environment $\psi(q^V, q^H)$ is generated by a bistable toggle switch consisting of two genes $i \in \{A, B\}$; each gene, G_i, produces a transcript T_i which is translated into a repressor Q_i. These repressors bind cooperatively to genes of the opposite type to form the repressed complex C_{ij}. Finally, a small amount of leak is added even for the repressed genes to help tune switching times and the transcripts and repressors degrade at rate δ. Only the repressors are visible, $Q^V = \{Q_A, Q_B\}$. All other species are hidden: $Q^H = \{G_A, G_B, T_A, T_B, C_{AB}, C_{BA}\}$. We model this process with the stochastic mass action reactions:

$$G_i \xrightarrow{k_{tx}} G_i + T_i \qquad T_i \xrightarrow{k_{tl}} T_i + Q_i \qquad G_i + 2Q_j \underset{k_u}{\overset{k_b}{\rightleftharpoons}} C_{ij}$$

$$C_{ij} \xrightarrow{k_{leak}} C_{ij} + T_i \qquad T_i \xrightarrow{\delta} \emptyset \qquad Q_i \xrightarrow{\delta} \emptyset \qquad (i \neq j).$$

Here $i, j \in \{A, B\}$ and the rate constants $k_{tx}, k_{tl}, k_u, k_b, \delta \geq k_{env}$. Next, the clamped potentiated detailed balanced reactions produce the distribution $\overline{\pi}^{\overline{P},P}(s^V, s^H \mid \langle s^V \rangle = q^V)\psi(q^V)$. Internally, it is a 3-node ECBM with potentiated detailed balanced

reactions:

$$
\begin{aligned}
&\mathcal{R}^{\overline{S}}_{\overline{\mathcal{P}},\mathcal{P}} = \{ \\
&\overline{S}^0_A + \overline{S}^0_H + \overline{P}^0_A + P^0_A \rightleftharpoons \overline{S}^1_A + \overline{S}^0_H + \overline{P}^1_A + P^1_A, \\
&\overline{S}^0_A + \overline{S}^1_H + \overline{S}^0_{W_{HA}} + \overline{P}^0_A + P^0_A + \overline{P}^0_{W_{HA}} \rightleftharpoons \overline{S}^1_A + \overline{S}^1_H + \overline{S}^1_{W_{HA}} + \overline{P}^1_A + P^1_A + P^1_{W_{HA}}, \\
&\overline{S}^0_B + \overline{S}^0_H + \overline{P}^0_B + P^0_B \rightleftharpoons \overline{S}^1_B + \overline{S}^0_H + \overline{P}^1_B + P^1_B, \\
&\overline{S}^0_B + \overline{S}^1_H + \overline{S}^0_{W_{HB}} + \overline{P}^0_B + P^0_B + P^0_{W_{HB}} \rightleftharpoons \overline{S}^1_B + \overline{S}^1_H + \overline{S}^1_{W_{HB}} + P^1_B + P^1_B + P^1_{W_{HB}}, \\
&\overline{S}^0_H + \overline{S}^0_A + \overline{S}^0_B + P^0_H \rightleftharpoons \overline{S}^1_H + \overline{S}^0_A + \overline{S}^0_B + P^1_H, \\
&\overline{S}^0_H + \overline{S}^1_A + \overline{S}^0_B + \overline{S}^0_{W_{HA}} + P^0_H + P^0_{W_{HA}} \rightleftharpoons \overline{S}^1_H + \overline{S}^1_A + \overline{S}^0_B + \overline{S}^1_{W_{HA}} + P^1_H + P^1_{W_{HA}}, \\
&\overline{S}^0_H + \overline{S}^0_A + \overline{S}^1_B + \overline{S}^0_{W_{HB}} + P^0_H + P^0_{W_{HB}} \rightleftharpoons \overline{S}^1_H + \overline{S}^0_A + \overline{S}^1_B + \overline{S}^1_{W_{HB}} + P^1_H + P^1_{W_{HB}}, \\
&\overline{S}^0_H + \overline{S}^1_A + \overline{S}^1_B + \overline{S}^0_{W_{HA}} + \overline{S}^0_{W_{HB}} + P^0_{W_{HA}} + P^0_H + P^0_{W_{HB}} \\
&\rightleftharpoons \\
&\overline{S}^1_H + \overline{S}^1_A + \overline{S}^1_B + \overline{S}^1_{W_{HA}}, + \overline{S}^1_{W_{HB}} + P^1_H + P^1_{W_{HA}} + P^1_{W_{HB}} \\
&\}.
\end{aligned}
$$

Here, all the rate constants are detailed balanced and are scaled by k_{db}. The potential species $\overline{\mathcal{P}}$ and $\mathcal{P}$ are included even though they are effectively held constant by the bath. The potential species $\overline{P}^1_A$ and $\overline{P}^1_B$ are used to clamp the visible species $\overline{S}_A$ and $\overline{S}_B$ to the visible environmental species Q_A and Q_B by the non-detailed balanced potential clamping reactions:

$$
\begin{aligned}
&\mathcal{T}^{\overline{\mathcal{P}}}_{\overline{\mathcal{S}},\mathcal{Q}} = \{ \\
&\quad \overline{P}^1_A + \overline{S}^1_A \xrightarrow{\epsilon_{fast} k^{\overline{S}}_A} 2\overline{P}^1_A + \overline{S}^1_A \qquad\qquad \overline{P}^1_A + Q_A \xrightarrow{\epsilon_{fast} k^{Q}_A} Q_A \\
&\quad \overline{P}^1_B + \overline{S}^1_B \xrightarrow{\epsilon_{fast} k^{\overline{S}}_B} 2\overline{P}^1_B + \overline{S}^1_B \qquad\qquad \overline{P}^1_B + Q_B \xrightarrow{\epsilon_{fast} k^{Q}_B} Q_B \\
&\}.
\end{aligned}
$$

The rates of the above reactions are set to rescale the means of the repressors: $\frac{k^{\overline{S}}_A}{k^{Q}_A} = \frac{k^{\overline{S}}_B}{k^{Q}_B} = \frac{k_{tx} k_{tl}}{\delta^2}$. Next, we describe the free potentiated detailed balanced reactions which produce the distribution $\pi^P(s^v, s^h)$. These reactions are near duplicates of the clamped potentiated detailed balanced reactions; the former lacks the potential

species $\overline{P}_A^1$ and $\overline{P}_B^1$. Specifically, these reactions model a 3-node ECBM:

$$
\begin{aligned}
\mathcal{R}_{\mathcal{P}}^{\mathcal{S}} = \{ & \\
S_A^0 + S_H^0 &\rightleftharpoons S_A^1 + S_H^0 + P_A^1 \\
S_A^0 + S_H^1 + S_{W_{HA}}^0 + P_{W_{HA}}^0 &\rightleftharpoons S_A^1 + S_H^1 + S_{W_{HA}}^1 + P_A^1 + P_{W_{HA}}^1, \\
S_B^0 + S_H^0 &\rightleftharpoons S_B^1 + S_H^0 + P_B^1, \\
S_B^0 + S_H^1 + S_{W_{HB}}^0 + P_{W_{HB}}^0 &\rightleftharpoons S_B^1 + S_H^1 + S_{W_{HB}}^1 + P_B^1 + P_{W_{HB}}^1, \\
S_H^0 + S_A^0 + S_B^0 + P_H^0 &\rightleftharpoons S_H^1 + S_A^0 + S_B^0 + P_H^1, \\
S_H^0 + S_A^1 + S_B^0 + S_{W_{HA}}^0 + P_H^0 + P_{W_{HA}}^0 &\rightleftharpoons S_H^1 + S_A^1 + S_B^0 + S_{W_{HA}}^1 + P_H^1 + P_{W_{HA}}^1, \\
S_H^0 + S_A^0 + S_B^1 + S_{W_{HB}}^0 + P_H^0 + P_{W_{HB}}^0 &\rightleftharpoons S_H^1 + S_A^0 + S_B^1 + S_{W_{HB}}^1 + P_H^1 + P_{W_{HB}}^1, \\
S_H^0 + S_A^1 + S_B^1 + S_{W_{HA}}^0 + S_{W_{HA}}^0 + P_{W_{HB}}^0 + P_H^0 + P_{W_{HB}}^0 & \\
\rightleftharpoons & \\
S_H^1 + S_A^1 + S_B^1 + S_{W_{HA}}^1 + S_{W_{HB}}^1 + P_H^1 + P_{W_{HA}}^1 + P_{W_{HB}}^1 & \\
\}. &
\end{aligned}
$$

The free species $\mathcal{S}$ species are coupled to clamped $\overline{\mathcal{S}}$ species via the potential species $\mathcal{P}$ which are modulated by the potential clamping reactions:

$$
\begin{aligned}
\mathcal{T}_{\mathcal{S},\overline{\mathcal{S}}}^{\mathcal{P}} = \{ & \\
P_A^0 + S_A^0 &\xrightarrow{\epsilon_{slow}} 2P_A^0 + S_A^0 & P_A^0 + \overline{S}_A^0 &\xrightarrow{\epsilon_{slow}} \overline{S}_A^0 \\
P_A^1 + S_A^1 &\xrightarrow{\epsilon_{slow}} 2P_A^1 + S_A^1 & P_A^1 + \overline{S}_A^1 &\xrightarrow{\epsilon_{slow}} \overline{S}_A^1 \\
P_B^0 + S_B^0 &\xrightarrow{\epsilon_{slow}} 2P_B^0 + S_B^0 & P_B^0 + \overline{S}_B^0 &\xrightarrow{\epsilon_{slow}} \overline{S}_B^0 \\
P_B^1 + S_B^1 &\xrightarrow{\epsilon_{slow}} 2P_B^1 + S_B^1 & P_B^1 + \overline{S}_B^1 &\xrightarrow{\epsilon_{slow}} \overline{S}_B^1 \\
P_H^0 + S_H^0 &\xrightarrow{\epsilon_{slow}} 2P_H^0 + S_H^0 & P_H^0 + \overline{S}_H^0 &\xrightarrow{\epsilon_{slow}} \overline{S}_H^0 \\
P_H^1 + S_H^1 &\xrightarrow{\epsilon_{slow}} 2P_H^1 + S_H^1 & P_H^1 + \overline{S}_H^1 &\xrightarrow{\epsilon_{slow}} \overline{S}_H^1 \\
P_{W_{HA}}^0 + S_{W_{HA}}^0 &\xrightarrow{\epsilon_{slow}} 2P_{W_{HA}}^0 + S_{W_{HA}}^0 & P_{W_{HA}}^0 + \overline{S}_{W_{HA}}^0 &\xrightarrow{\epsilon_{slow}} \overline{S}_{W_{HA}}^0 \\
P_{W_{HA}}^1 + S_{W_{HA}}^1 &\xrightarrow{\epsilon_{slow}} 2P_{W_{HA}}^1 + S_{W_{HA}}^1 & P_{W_{HA}}^1 + \overline{S}_{W_{HA}}^1 &\xrightarrow{\epsilon_{slow}} \overline{S}_{W_{HA}}^1 \\
P_{W_{HB}}^0 + S_{W_{HB}}^0 &\xrightarrow{\epsilon_{slow}} 2P_{W_{HB}}^0 + S_{W_{HB}}^0 & P_{W_{HB}}^0 + \overline{S}_{W_{HB}}^0 &\xrightarrow{\epsilon_{slow}} \overline{S}_{W_{HB}}^0 \\
P_{W_{HB}}^1 + S_{W_{HB}}^1 &\xrightarrow{\epsilon_{slow}} 2P_{W_{HB}}^1 + S_{W_{HB}}^1 & P_{W_{HB}}^1 + \overline{S}_{W_{HB}}^1 &\xrightarrow{\epsilon_{slow}} \overline{S}_{W_{HB}}^1. \\
\} &
\end{aligned}
$$

Notice that the visible clamped species $\overline{S}_A$ and $\overline{S}_B$ have their own potentials in $\overline{\mathcal{P}}$ as well as sharing potential species in $\mathcal{P}$ with S_A and S_B. However the hidden clamped species $\overline{S}_H$, $\overline{S}_{W_{HA}}$, and $\overline{S}_{W_{HB}}$ only share potentials $\mathcal{P}$ with the corresponding free species S_H, $S_{W_{HA}}$, and $S_{W_{HB}}$. This is reminiscent of the way the clamped units in a Boltzmann machine use the same energies as the free units during training.

4.8 Thermodynamics of Learning and Inference

This section provides physically motivated energetic and thermodynamic costs of learning and inference with potentiated dbCRNs. First, we note that in the constructions used in this paper, learning and inference are fundamentally the same process. In both cases, we start with a dbCRN C_G with equilibrium distribution π_G and, either via the clamping process or the learning process, end up with a new distribution $\pi_{G'}$. The final distribution $\pi_{G'}$ can be physically realized in a variety of ways: by changing the energies $G' = G + \Delta$; by equipping C_G with potential species so that $G' = G + \mu$; or by using the reactions (4.41 - 4.42) to push π_G from equilibrium. Importantly, in the first two scenarios described, $\pi_{G'}$ remains a dbCRN while in the last scenario, $\pi_{G'}$ is out of equilibrium (at least until the potential clamping reactions are turned off). In the following section, we will investigate reversibly and non-reversibly modulating the potentials of a potentiated dbCRN, the dissipation of the potential clamping reactions, and how these can be used to understand the learning construction of the previous section. Finally, we note that this section tacitly assigns a physical meaning to the species' energies G_i. In the previous sections, these energies could be viewed purely mathematically. However, in this section G_i corresponds to the physical enthalpy[3] of formation of the species S_i which may implicitly depend on factors like solvent conditions and temperature. The arguments used in this section are of a very different flavor from the rest of the Chapter and contain considerably less mathematical rigor. From the perspective of this thesis, these results are meant to be a starting point to illustrate that a fully autonomous learning CRN construction provides a window through which increasingly complex analyses of the physics of learning can be conducted. A reader of this section is cautioned that these arguments are not fully vetted. However, these arguments are meant to be provocative and inspire future study.

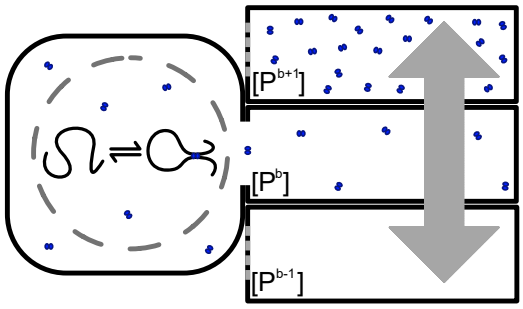


Figure 4.6: A cartoon of inference via potential baths. Large baths containing different concentrations of potential species (shown in blue) can be connected and disconnected to and from a dbCRN in a small volume (dashed circle) for free. Not to scale.

Thermodynamics of Inference via Clamping

To analyze the costs of inference, we consider clamping a potentiated dbCRN by changing the concentrations of the potential species directly by changing the external baths. In the following analysis, the *system* will be a tuple (π, P) where π is an initial distribution of the CRN and P are the potential bath concentrations. The system is always in thermal equilibrium with its environment (the solvent) at a temperature T. By changing the concentration of the external bath of the species P, the chemical potentials μ can be controlled. Similarly to the analysis by Ouldridge and collaborators [253, 254], we will imagine a set of different baths b each with a concentration of potential species $[P^b]$ which can be freely disconnected and reconnected to the reaction volume. The purpose of this analysis is to highlight the cost of inference in certain extreme conditions which can act as benchmarks for future investigations. We begin with some basic thermodynamic definitions for a dbCRN at a (not necessarily equilibrium) distribution ω connected to a potential bath with concentrations $[P]$. Denote the energy function $\mathcal{G}^P$ which includes chemical potential terms. The internal energy, $\mathbb{U}^P_\omega$, entropy, $\mathbb{S}^P_\omega$, and free energy, $\mathbb{G}^P_\omega$, of a

[3]For simplicity, we are assuming that the species energies G_i are in fact enthalpies. In reality, they may contain both enthalpic and entropic components. However, for simplicity of presentation, we ignore that complication.

dbCRN are given by [92]:

$$\mathbb{U}^P_\omega = \langle \mathcal{G}^P(s) \rangle_\omega = \sum_s \omega(s) \mathcal{G}^P(s), \tag{4.53}$$

$$\mathbb{S}^P_\omega = -k_B \langle \log \omega \rangle_\omega = -k_B \sum_s \omega(s) \log \omega(s), \tag{4.54}$$

$$\text{and} \quad \mathbb{F}^P_\omega = \mathbb{U}^P_\omega - T\mathbb{S}^P_\omega. \tag{4.55}$$

If $\omega = \pi^P$, the equilibrium distribution for the volume V^P, then entropy and free energy can also be written:

$$\mathbb{S}^P_{\pi^P} = k_B\left(\frac{\mathbb{U}_{\pi^P}}{T} + \log Z_{\pi^P}\right) \qquad \mathbb{F}^P_{\pi^P} = -\log Z_{\pi^P}.$$

The free energy difference between a potentiated dbCRN with potential concentrations $[P]$ in a non-equilibrium distribution ω and its equilibrium π^P is given by:

$$\Delta\mathbb{F}^P_{\omega \to \pi^P} = \mathbb{F}^P_\omega - \mathbb{F}^P_{\pi^P} = \mathbb{U}^P_\omega - T\mathbb{S}^P_\omega - \mathbb{F}_{\pi^P} \tag{4.56}$$

$$= \sum_s \omega(s)(\mathcal{G}^P(s) + \log \omega(s) + \log(Z_{\pi^P})) \tag{4.57}$$

$$= \sum_s \omega(s)(\log \omega(s) - \log \pi^P(s)) \tag{4.58}$$

$$= \mathbb{D}(\omega \mid\mid \pi^P). \tag{4.59}$$

The Reversible Case: By very slowly connecting and reconnecting the potentiated dbCRN through an infinite series of baths, each of which changes the concentration of $[P]$, by an infinitesimal amount, the potentiated dbCRN can be clamped from $[P^0] \to \ldots \to [P^n] \to \ldots \to [P^N]$ in such a way that it will always be at an equilibrium distribution π^{P^n} as described by Ouldridge and collaborators [253]. This results in a quasistatic reversible process where the baths reversibly do chemical work on the system:

$$(P^0, \pi^{P^0}) \xRightarrow{W_{rev}} (P^N, \pi^{P^N}). \tag{4.60}$$

This same process could be run backwards, through the sequence of potentials $[P^N] \to \ldots \to [P^n] \to \ldots \to [P^0]$ with the dbCRN always at equilibrium. Based on the results from [253], no entropy will be dissipated in either direction by this kind of process. This result indicates that inference, if computed slowly, can be free—something we suspect might be advantageous to cellular life in certain conditions. We note that an explicit calculation of W_{rev} is highly dependent on underlying potentiated dbCRN and is a direction for future investigation.

An Irreversible Case: Next, we consider a non-reversible process. In this process, the potential baths are changed instantaneously from P^0 to P^N. This can be imagined as moving through the same series of baths as above, but infinitely quickly instead of infinitely slowly (or equivalently moving directly to the final buffer). The CRN is initially at equilibrium with distribution π^{P^0}. Following the volume change, the CRN will initially be out of equilibrium at the distribution π^{P^0} but at potential concentration P^N. Then, the CRN relaxes to the equilibrium distribution π^{P^N} based upon the new potential species' concentrations.

$$(P^0, \pi^{P^0}) \xRightarrow{W_{irrev}} (P^N, \pi^{P^0}) \xRightarrow{\mathbb{Q}_{\pi^{P^0} \to \pi^{P^N}}} (P^N, \pi^{P^N}). \tag{4.61}$$

Notice that the system is not interacted with during the second step (and the volume and temperature are held constant) so any changes in energy must be linked to the bath. In this setup, the dissipated entropy Q results from mixing the external baths (when a full cycle is considered). Indeed, π^P is not at equilibrium relative to the concentrations $[P']$ which is reflected in the free energy difference between π^{P^0} and π^{P^N}:

$$\mathbb{Q}_{\pi^P \to \pi^{P'}} = \Delta \mathbb{F}^{P'}_{\pi^P \to \pi^{P'}} = \mathbb{D}(\pi^P \mid\mid \pi^{P'}). \tag{4.62}$$

Finally, noticing that initial and final states of the reversible and irreversible processes are the same, conservation of energy dictates that:

$$W_{irrev} = W_{rev} + \mathbb{D}(\pi^{P^0} \mid\mid \pi^{P^N}). \tag{4.63}$$

In other words, the thermodynamic cost of performing inference quickly is the relative entropy between the initial distribution of the system and the final distribution of the system. Importantly, performing inference this way always has an energetic cost because the relative entropy is always positive. Energy is dissipated by the dbCRN being pushed out of equilibrium and then settling into a new equilibrium state. Additional work may have to be done to create the potentials, but this work is assumed to be recoverable.

Thermodynamics of the Potential Clamping CRN

In the previous section, we analyzed the cost of an external agent with an infinite collection of potential baths performing inference either very slowly or very quickly. Clearly, this situation was idealized. However, the potential clamping reactions provide a mechanism by which the potential baths can be dynamically adjusted by an autonomous CRN. In this section, we analyze the cost of inference incurred by

these reactions and point out some trade-offs between accuracy, reversibility, and dissipation. To begin, we note that the reactions (4.41, 4.42) are not directly amiable to thermodynamic treatment because they are irreversible and therefore infinitely dissipative. However, these equations can be rewritten reversibly and analyzed:

$$P_i + S_i \underset{\delta}{\overset{\epsilon k_i^S}{\rightleftharpoons}} 2P_i + S_i \tag{4.64}$$

$$P_i + Q_i \underset{\delta}{\overset{\epsilon k_i^Q}{\rightleftharpoons}} Q_i, \tag{4.65}$$

where δ is a small reverse rate constant. This CRN can be thought of as driven by a hidden infinite reservoir of fuel molecules [90]. The dynamics can now be written as:

$$\frac{d[P_i]}{dt} = [P_i]\langle s_i\rangle_{\pi^P}(\epsilon - \delta[P_i]) - \langle q_i\rangle_{\psi}(\epsilon[P_i] - \delta) = J_i^S - J_i^Q. \tag{4.66}$$

Here we have assumed $k_i^S = k_i^Q = 1$ for simplicity. In the last step, we have rewritten each term using the reaction fluxes J_i^S and J_i^Q through (4.64) and (4.65), respectively. This ODE cannot be solved analytically, however we can analyze its steady state solutions:

$$\frac{d[P_i]}{dt} = 0 = J_i^S - J_i^Q \qquad \text{Solution 1: } J_i^S = J_i^Q = 0 \qquad \text{Solution 2: } J_i^S = J_i^Q \neq 0.$$

The first solution corresponds to an equilibrium solution when the driving potential goes to 0. In general, we do not expect this case to exhibit accurate learning. The second solution corresponds to a non-equilibrium steady state. Such a solution exists provided that $\psi(q)$ has the same support as $\pi^P(s)$. A little algebraic manipulation reveals that at steady state, the error between the mean of S_i and Q_i is given by:

$$\frac{\langle q_i\rangle_{\psi} - \langle s_i\rangle_{\pi^P}}{\langle q_i\rangle_{\psi}} = 1 - \frac{\langle s_i\rangle_{\pi^P}}{\langle q_i\rangle_{\psi}} = 1 - \frac{(\epsilon[P_i] - \delta)}{[P_i](\epsilon - \delta[P_i])} = \frac{\delta(1 - [P_i]^2)}{[P_i](\epsilon - \delta[P_i])}. \tag{4.67}$$

Notice that when $\delta \to 0$, the error also goes to 0. For non-zero values of δ, the error will also depend non-linearly on the final concentration $[P_i]$. This dependence is illustrated in Figure 4.7. The key insight from figure is that the potential learning CRN can be only weakly driven and still work well provided that the target mean $\langle q\rangle_{\psi}$ of the species $\mathcal{Q}$ is not too far from the unclamped mean $\langle s\rangle_{\pi}$ of the species $\mathcal{S}$. However, as larger clamping potentials need to be applied, δ must become small to keep the error low. The reaction fluxes (4.66) also allow us to calculate the entropy production rate from the learning reactions for potential species i [91]. Notice that

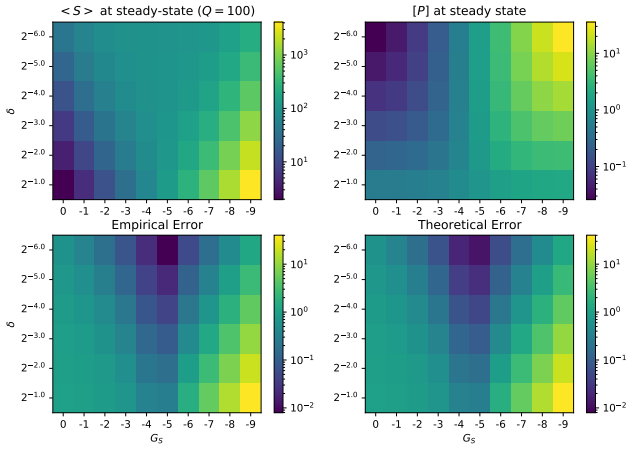


Figure 4.7: Empirical and theoretical error when learning the mean of a product Poisson with the dbCRN $\emptyset \rightleftharpoons S + P$ with potential clamping reactions for P for different initial energies G_S and reverse rate constant δ. Empirical error is defined as $|\frac{\langle s\rangle - \langle q\rangle}{\langle q\rangle}|$ and theoretical error is defined as $|\frac{\delta(1-[P_i]^2)}{[P_i](\epsilon - \delta[P_i])}|$.

all other reactions are at equilibrium, by construction, so these are the only parts of the system which produce entropy:

$$T\frac{d\mathbb{S}_i}{dt} = RT(J_i^S \log\frac{\epsilon}{\delta[P_i]} + J_i^Q \log\frac{\epsilon[P_i]}{\delta}). \tag{4.68}$$

Here, R is the gas constant. At an equilibrium steady state, this simplifies to 0. At a non-equilibrium steady state with $J_{ss} = J_i^S = J_i^Q$, the entropy production is entirely dissipated as heat:

$$\frac{d\mathbb{Q}_i}{dt} = 2RTJ_{ss}\log\frac{\epsilon}{\delta}. \tag{4.69}$$

Thermodynamics of Learning

We can understand the thermodynamics of the learning using the results from the previous section on the reversible and irreversible thermodynamics of clamping a potentiated dbCRN and the thermodynamics of the potential clamping CRN. In these analyses, we will consider the different underlying trajectories the learning CRN must go through during the learning process and examine different limits and mechanisms through which the system can achieve learning.

First, we define a state of the learning system as a tuple $(q, \overline{P}, P, \overline{\pi}^{\overline{P},P}, \pi^P)$ where q is a state of the environment, $\overline{P}$ are potentials connecting a clamped dbCRN to

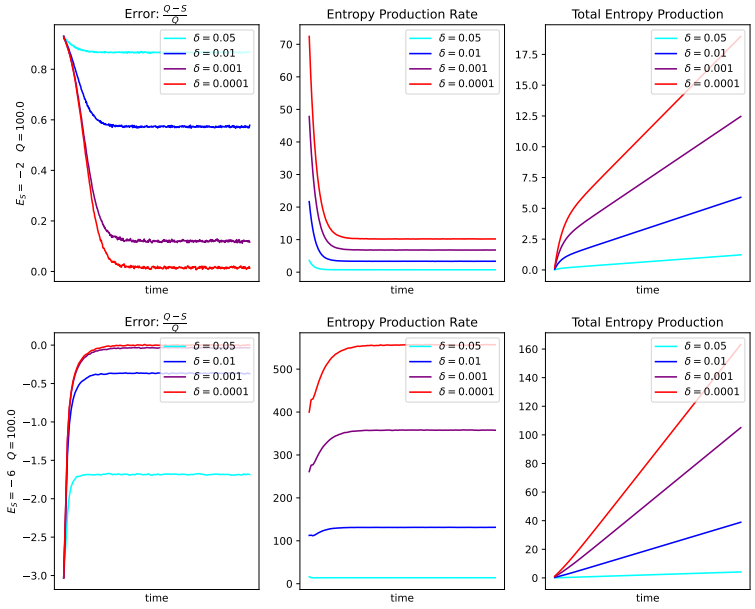


Figure 4.8: Error, entropy production rate, and total entropy production for the potential clamping CRN reactions applied to the dbCRN $\emptyset \rightleftharpoons S$.

the environment, P are potentials connecting the clamped dbCRN to a free dbCRN, $\overline{\pi}^{\overline{P},P}(\overline{s})$ is the clamped distribution, and $\pi^P(s)$ is the free distribution. Throughout training, the environment may be sampled, potentials may be adjusted, and CRNs may be pushed from equilibrium or relax to equilibrium.

Reversible Case: Similar to the reversible clamping analyzed in 4.8, we can imagine that the potentials of the learning CRN are adjusted by an external agent equipped with a set of potential baths which are quasi-statically used to guide the learning process. Formally, such a process could be written as:

$$(q^0, \overline{P}^0, P^0, \overline{\pi}^{\overline{P}^0,P^0}, \pi^{P^0}) \rightarrow \ldots \rightarrow$$
$$(q^n, \overline{P}^n, P^n, \overline{\pi}^{\overline{P}^n,P^n}, \pi^{P^n}) \rightarrow \ldots \rightarrow$$
$$(q^N, \overline{P}^N, P^N, \overline{\pi}^{\overline{P}^N,P^N}, \pi^{P^N}).$$

Lumping all the infinitesimal transitions together, we write:

$$(q^0, \overline{P}^0, P^0, \overline{\pi}^{\overline{P}^0,P^0}, \pi^{P^0}) \xrightarrow{W_{rev}} (q^N, \overline{P}^N, P^N, \overline{\pi}^{\overline{P}^N,P^N}, \pi^{P^N}). \tag{4.70}$$

This limit is not particularly interesting; the system is always at equilibrium so it dissipates no entropy, but requires infinite time and an infinite number of baths.

We note that because this process is reversible, the work done will depend only on the initial and final potentials of the learning system, not on the process by which the system learns. Hence, we do not believe that this process accurately reflects practical considerations of learning.

An Irreversible Case: Once again, the argument used to understand the thermodynamics of irreversible clamping (Section 4.8) can be applied to understand the learning CRN. Here, we imagine an external agent who freely moves the system between a finite number of potential baths. There are a number of different trajectories the learning could take. The simplest irreversible learning trajectory begins at a state $(q^0, \overline{P}^0, P^0, \overline{\pi}^{\overline{P}^0, P^0}, \pi^{P^0})$ and is then immediately connected to potential baths representing the final desired potentials. However, the system will initially be out of equilibrium relative to these potentials and will dissipate entropy as it relaxes to equilibrium:

$$\begin{gathered}
(q^0, \overline{P}^0, P^0, \overline{\pi}^{\overline{P}^0, P^0}, \pi^{P^0}) \\
\Downarrow W_{irrev}^{1 \text{ step}} \\
(q^N, \overline{P}^N, P^N, \overline{\pi}^{\overline{P}^0, P^0}, \pi^{P^0}) \\
\Downarrow \overline{\mathbb{Q}} + \mathbb{Q} \\
(q^N, \overline{P}^N, P^N, \overline{\pi}^{\overline{P}^N, P^N}, \pi^{P^N}).
\end{gathered}$$

Here, $\overline{\mathbb{Q}}$ and $\mathbb{Q}$ are the entropy produced by the clamped and free CRNS, respectively:

$$\overline{\mathbb{Q}}_{1 \text{ step}} = \mathbb{D}(\overline{\pi}^{\overline{P}^0, P^0} \mid\mid \overline{\pi}^{\overline{P}^N, P^N}) \qquad \mathbb{Q}_{1 \text{ step}} = \mathbb{D}(\pi^{P^0} \mid\mid \pi^{P^N}). \tag{4.71}$$

Again, we argue based upon conservation of energy that $W_{irrev} = W_{rev} + \overline{\mathbb{Q}}_{1 \text{ step}} + \mathbb{Q}_{1 \text{ step}}$ which suggests that learning quickly is in some sense the same as clamping a CRN to a new set of potentials. However, in practice we expect the learning process to be even more nuanced. Many samples from the environment will be copied into the clamped CRN which in turn will induce changes to the free CRN. We can model

this process via the transitions:

$$
\begin{aligned}
&(q^{n-1}, \overline{P}^{n-1}, P^{n-1}, \overline{\pi}^{\overline{P}^{n-1}, P^{n-1}}, \pi^{P^{n-1}}) \\
&\qquad \text{Sample } n \Downarrow \text{free} \\
&(q^{n}, \overline{P}^{n-1}, P^{n-1}, \overline{\pi}^{\overline{P}^{n-1}, P^{n-1}}, \pi^{P^{n-1}}) \\
&\qquad \text{Change } \overline{P} \Downarrow \overline{W}_n \\
&(q^{n}, \overline{P}^{n}, P^{n-1}, \overline{\pi}^{\overline{P}^{n-1}, P^{n-1}}, \pi^{P^{n-1}}) \\
&\qquad \text{Equilibrate } \overline{\pi} \Downarrow \overline{\mathbb{Q}}_n \\
&(q^{n}, \overline{P}^{n}, P^{n-1}, \overline{\pi}^{\overline{P}^{n}, P^{n-1}}, \pi^{P^{n-1}}) \\
&\qquad \text{Change } P \Downarrow W_n \\
&(q^{n}, \overline{P}^{n}, P^{n}, \overline{\pi}^{\overline{P}^{n}, P^{n-1}}, \pi^{P^{n-1}}) \\
&\text{Equilibrate } \overline{\pi} \text{ and } \pi \Downarrow \mathbb{Q}_n \\
&(q^{n}, \overline{P}^{n}, P^{n}, \overline{\pi}^{\overline{P}^{n}, P^{n}}, \pi^{P^{n}}).
\end{aligned}
$$

Here, we assume that the system starts at equilibrium relative to a sample q^{i-1}. When a new sample q^i is produced from the environment, it will be copied into the clamped CRN using the potentials $\overline{P}$. The clamped CRN will then equilibrate, then the potentials P will be used to induce a change in the free CRN, and then the free CRN equilibrates. The work $\overline{W}_n$ and W_n will depend on details of the underlying CRNs' reactions and energies. However, the dissipation can be obtained for each step direction from (4.62). Combining these together for N training samples results in a more complicated expression for dissipation from learning from many samples:

$$
\mathbb{Q}_{\text{N steps}} = \sum_{n=1}^{N} [\mathbb{D}(\overline{\pi}^{\overline{P}^{n-1}, P^{n-1}} \;||\; \overline{\pi}^{\overline{P}^{n}, P^{n-1}}) + \mathbb{D}(\overline{\pi}^{\overline{P}^{n}, P^{n-1}} \;||\; \overline{\pi}^{\overline{P}^{n}, P^{n}}) + \mathbb{D}(\pi^{P^{n-1}} \;||\; \pi^{P^{n}})]. \tag{4.72}
$$

When the relative entropy is considered as the cost of copying, this expression has a straightforward interpretation [255]. The first term in the sum is due to copying a sample from the environment onto the clamped distribution. The second term is due to copying any new information in the sample into the clamped CRN model. The third term is due to copying information from the clamped CRN model into the free CRN model. Interestingly, we will show that this value is actually a lower bound of the one-step irreversible learning:

$$
\overline{\mathbb{Q}}_{\text{1 step}} + \mathbb{Q}_{\text{1 step}} \geq \mathbb{Q}_{\text{N steps}}. \tag{4.73}
$$

Proof:

The Pythagorean information inequality states: [247]

$$\mathbb{D}(P \mid\mid Q) \geq \mathbb{D}(P \mid\mid P^*) + \mathbb{D}(P^* \mid\mid Q) \qquad P^* = \underset{P \in E}{\operatorname{argmin}}\, \mathbb{D}(P \mid\mid Q). \tag{4.74}$$

Here, P and Q are any two distributions, $\mathbb{D}$ is the relative entropy, and E is a convex set of distributions. When applied to the learning CRN, energy can be dissipated through both the clamped and free distributions, $\overline{\pi}$ and π. Any samples from these distributions are independent; the coupling via potential learning reactions allows the two dbCRNs to share energy parameters, but does not cause their states to become correlated. Therefore, we will treat the dissipation through each sub-CRN individually. Assume that N data points are sampled from the environmental distribution ψ. Throughout the learning process, the baths $\overline{P}$ and P will update as more data is sampled. The clamped potentiated dbCRN begins with its two potential baths at values $\overline{P}^0$ and P^0 and ends with them at $\overline{P}^N$ and P^N. The amount of entropy produced due to the equilibration of $\overline{\pi}$ as the potentials change is:

$$\begin{aligned}
\overline{\mathbb{Q}} &= \mathbb{D}(\overline{\pi}^{\overline{P}^0, P^0} \mid\mid \overline{\pi}^{\overline{P}^N, P^N}) && (4.75)\\
&\geq \mathbb{D}(\overline{\pi}^{\overline{P}^0, P^0} \mid\mid \overline{\pi}^{\overline{P}^1, P^1}) + \mathbb{D}(\overline{\pi}^{\overline{P}^1, P^1} \mid\mid \overline{\pi}^{\overline{P}^N, P^N}) && (4.76)\\
&\geq \sum_{n=1}^{N} \mathbb{D}(\overline{\pi}^{\overline{P}^{n-1}, P^{n-1}} \mid\mid \overline{\pi}^{\overline{P}^n, P^n}) && (4.77)\\
&\geq \sum_{n=1}^{N} \mathbb{D}(\overline{\pi}^{\overline{P}^{n-1}, P^{n-1}} \mid\mid \overline{\pi}^{\overline{P}^n, P^{n-1}}) + \mathbb{D}(\overline{\pi}^{\overline{P}^n, P^{n-1}} \mid\mid \overline{\pi}^{\overline{P}^n, P^n}). && (4.78)
\end{aligned}$$

Here, we have applied the Pythagorean information inequality iteratively between the second and third steps to expand the total dissipation across each training sample and once in the final step expand the equilibration between the two potential baths. Note that we already proved that the equilibrium distribution of a potentiated dbCRN minimizes (4.74) where E is the space of distributions with a fixed mean and that E is convex in Section 4.6. Similarly, the dissipation through the free dbCRN is given by:

$$\mathbb{Q} = \mathbb{D}(\pi^{P^0} \mid\mid \pi^{P^m}) \geq \sum_{i=1}^{m} \mathbb{D}(\pi^{P^{i-1}} \mid\mid \pi^{P^i}). \tag{4.79}$$

Combining these expressions gives the inequality for total dissipation:

$$\overline{\mathbb{Q}} + \mathbb{Q} \geq \sum_{i=1}^{m} \mathbb{D}(\overline{\pi}^{\overline{P}^{i-1}, P^{i-1}} \mid\mid \overline{\pi}^{\overline{P}^i, P^{i-1}}) + \mathbb{D}(\overline{\pi}^{\overline{P}^i, P^{i-1}} \mid\mid \overline{\pi}^{\overline{P}^i, P^i}) + \mathbb{D}(\pi^{P^{i-1}} \mid\mid \pi^{P^i}).$$

This result makes intuitive sense because the multistep learning process is, informally, slower than the single step process and therefore closer to the completely reversible process. Finally, the irreversible work may then be written as

$$W_{irrev}^{\text{N steps}} = \sum_n (\overline{W}_n + W_n) \geq W_{rev} + \mathbb{Q}_{\text{N steps}}. \tag{4.80}$$

Unlike in the single step process, in the multistep process we know that some amount of work put into the system can theoretically be re-used throughout the learning process which we are not explicitly accounting for in the values of $\overline{W}_n$ and W_n. Therefore, we expect the total irreversible work to be greater than the reversible work and the entropy produced, with the understanding that this could be made an equality if energy stored in the system is repurposed perfectly.

We emphasize that these results are agnostic as to whether the environmental process ψ is dissipative. In the example used in 4.5, ψ is generated by a dissipitive CRN. However, ψ could just as easily been generated by a dbCRN. As far as learning is concerned, the cost comes from copying samples from the environment (data) into the learning CRN, not from how that data is generated.

Dissipation from the Learning CRN: In the previous two examples, we imagined an external agent moving pistons throughout the learning process. However, the full learning CRN construction allows for an autonomous system to learn from its environment. Each set of potential clamping reactions will dissipate energy in this case according to the results in 4.8 which can then be added up throughout the entire learning process. We suspect that there will be similar dissipation-accuracy trade offs for the entire autonomous learning system as there are for a single set of potential clamping reactions. Understanding how these trade offs relate to the specific structure of the underlying dbCRN, the environmental distribution, and the initial energies will prove an interesting avenue for future research.

4.9 Discussion

We have provided a necessary mechanism by which dbCRNs can represent complex distributions far from the canonical product-Poisson form they are most well known for. To do this, dbCRNs must carefully control their species counts and initial conditions in order to restrict their reachability class. Analogously to BMs, marginalization over hidden species can be employed to further increase the complexity of distributions a dbCRN can represent with the important caveat that the reachable states of the visible species must be dependent on the states of the hidden

species. This leads to the question of which dbCRN architectures can produce far-from-Poisson distributions and how the representational power of dbCRNs relates to their underlying structure. So far, we know of two powerful constructions which require many species and/or reactions as well as carefully controlled binary species counts. The pixel CRN construction used to produce the face distribution in Figure 4.1 uses $O(N)$ hidden species where N is the size of the supported visible distribution. Similarly, the chemical Boltzmann machine construction learned in Figure 4.5 uses $O(N \log N)$ reactions where N is the number of binary visible states produced from $\log_2 N$ visible species. This leads us to speculate that restricting the reachability class of a dbCRN is a kind of computational resource analogous to entanglement in quantum computing [256].

Next, we examined how modulating the energies of the species in a dbCRN can be interpreted as computing a conditional distribution and hence a form of inference. Energy clamping, as we named this process, can also be implemented by coupling clamped species in the dbCRN to potential baths. This potentiated dbCRN, in turn, can be controlled by a set of (non-detailed balanced) chemical reactions. These constructions provide many ways for a physical system to perform inference closely related to evolution, adaption, and control. Modulating the energies of chemical species can happen easily by changing the sequences of DNA or other polymers and, therefore, is potentially a form of inference that could occur evolutionary. Using a chemical potential to modulate the mean of a chemical signal could also easily occur in a real or synthetic biochemical network and may be used for an organism to adapt to its environment. We suspect that these kinds of inferential systems could be realised *in vivo* due to the binding and unbinding events between the genome and different proteins or RNAs such as occurs in epigenetic chromatin structures in eukaryotes and in folded or looping conformations caused by many transcription factors in bacteria and eukaryotes [248, 249]. In order to build such a system in the lab, the greatest challenge will be to produce a chemical network with low species counts and dynamically measure the stochastic fluctuations. Advances in droplet-based technologies coupled with microfluidics [257] and positioning of single DNA molecules [258] provide potential avenues for the construction of such a circuit *in vitro*. Finally, we note that an exact implementation of the potential clamping reactions would function as a integral feedback controller capable of reference tracking [220].

Mathematically, the potential clamping reactions correspond to a variation of the

moment learning rule used to train BMs and can be thought of as a chemical implementation of a machine learning algorithm. In our learning CRN construction, we show how these potential clamping reactions can be combined with free and clamped dbCRNs in order to produce a general architecture which learns energies for both hidden and visible species. This construction suggests a wide class of dbCRN-inspired machine learning algorithms. These models would be generative, could utilize both a mixture of discrete and continuous variables, and would inherently learn in a continuous on-line fashion. The outstanding challenge is to find dbCRN architectures which are compact enough for easy simulation yet can also represent complex distributions, or, alternatively, to build design custom tailored software that can quickly simulate very large dbCRNs analogous to GPU based libraries for deep learning [138, 139].

Finally, by defining a machine learning system entirely in terms of chemical reaction networks, we were able to provide some basic thermodynamic costs for the learning process. These costs are intentionally very simple and, we believe, broadly applicable to any physical system because they are rooted in modulating the energies of a detailed balanced distribution. We note that in our construction, inference is closely related to the thermodynamics of copying [255]. Our results suggest that for learning to occur, the system must first read or copy its external environment. Then, the system can compute a distribution conditioned on the environment which it must again copy in order to learn. This closely follows known thermodynamic limits on computing where many computations are *free*, but making use of the result for downstream operations requires it to be copied [255].

To conclude, we wish to emphasize that the mathematical structures described in this paper are in fact quite general, even though they may appear overly specific. At the highest level, detailed balanced CRNs are ubiquitous in biology and engineered bio-molecular systems; interactions such as molecular binding, diffusion, and conformation changes are frequently detailed balanced. Indeed, for our results on energy clamping and inference to hold, a detailed balanced sub-system need only be at a dynamic quasi-equilibrium relative to other subsystems. Similarly, our results on potentiated dbCRNs are also somewhat more general than they may appear: potentials may be shared between multiple species and, as exemplified in the DNA looping model, can occur on either side of a detailed balanced reaction. This flexibility means that dynamic and driven biochemical systems may, in fact, be generalized Boltzmann machines in disguise. Dynamic changes in generalized

potential species coupled to detailed balanced sub-systems could be acting as molecular machine learning models—representing complex distributions and computing inference while simultaneously adapting to and learning from the environment.

Chapter 5

REDUCING THE COMPLEXITY OF *E. COLI* CELL EXTRACT METABOLISM WITH PHENOMENOLOGICAL MODELING

5.1 Forward

The following chapter has been written as a journal article which we aim to post as a preprint in the near future and submit to a journal after a few additional follow-up experiments. This work was conducted in collaboration with Ankita Roychoudhury under the supervision of Richard Murray.

Compared to the preceding chapters, this chapter is very concrete and rooted in practical experiments. Since the start of my PhD, I have dabbled in a variety of *E. coli* cell extract experiments mostly focused around understanding how to best model and characterize these systems. Cell extracts are an interesting middle ground between the biological complexity of a living organism and the simple highly controlled *in vitro* settings commonly used for molecular programming (such as DNA strand displacement systems). They contain much of the complexity of *E. coli*—a great many unknown molecules and interactions—but are relatively high throughput and easier to work with than *living* organisms.

This chapter originally began as an attempt to create a data set "big enough" for machine learning techniques with a goal of parameterizing a systems level model of cell extract metabolism. This model, in turn, could be used to build better synthetic biological circuits and improve cell-free bio-production. Unfortunately, despite acquiring tens of thousands of data points, the complexity of extract metabolism surpassed my expectations. As a result, this chapter is more about reducing complexity into parameterizable heuristics than it is about full on machine learning.

That said, this chapter still includes serious computation. It showcases the model-building to simulation to inference pipeline I described in the preface of Chapter 2. BioCRNpyler was used to produce a phenomenological model of cell extract metabolism. Bayesian parameter inference—a model fitting technique which explores distributions of parameters based on a cost function[1]—run through the

[1] Bayesian parameter inference is related to the generative models discussed in the introduction in the sense that one could frame learning a posterior parameter distribution as a machine learning problem, but in this work I take a purely empirical approach.

Bioscrape software package was then used to understand how well a set of simple experiments could fit a similarly simple model. At a more general level, the idea of fitting phenomenological models to large complex data sets is an example of function approximation in machine learning, and it seems plausible that phenomenological CRN models will do a good job of approximating mechanistic CRN models. Although I do not make these connections rigorous in this chapter, I believe they are an important area of research in order to deal with biological complexity.

Contribution: I wrote the entire paper, prepared the metabolomics samples, analyzed the data, developed the underlying mathematical model, and supervised Ankita's experimental contribution. The metabolomics measurements were conducted by a company Metabolon. Ankita Roychoudhury was responsible for the spiking and ATP timecourse experiments, which were similar to a number of experiments included in her senior thesis.

5.2 Abstract

We present and discuss systems level metabolic measurements of *E. coli* cell extract using high throughput mass spectrometry ("metabolomics"). These measurements conclusively show that the majority of native *E. coli* metabolism is at least partially active in extracts. We use these measurements to inform a phenomenological model of cell extract metabolism which we parameterize using Bayesian inference techniques from cell-free protein synthesis data. This model is designed to work as a module that can be attached to other circuit models in order to better use cell-free systems for circuit prototyping.

5.3 Introduction

Cell extracts, produced by lysing large numbers of cells and removing their DNA and lipid membranes while retaining most constituents of the cytoplasm, have a long history as a powerful tool in the study of biochemistry and molecular biology [259]. More recently, *in vitro* biochemical systems are increasingly important in synthetic biology [133, 260]. Extracts can be used as a flexible bioproduction platform capable of producing compounds which may be toxic *in vivo* [261, 262]. Cell extracts also offer a valuable platform to rapidly test and prototype synthetic biological components and circuits [134, 263]. Finally, cell extracts provide a valuable platform for methodological prototyping and biological discovery because they can be used for high throughput experiments and screens [264–266].

In this paper, we are particularly interested in the metabolism of *E. coli* cell extract.

Understanding extract metabolism will accelerate metabolic engineering of extracts for bio-production purposes. Additionally, when using cell extracts as a genetic circuit prototyping platform, the limited lifetime of cell extracts limits the complexity of circuits that can be prototyped. Increasing the life span of cell-free protein synthesis (CFPS) would allow increasingly complex circuits to be built and analyzed *in vitro*, which can be significantly more efficient than working directly *in vivo* [267, 268].

Most cell-free systems work by combining extract with a fuel source (such as glucose or 3PGA) that can be metabolized by the extract in order to regenerate ATP and other internal fuel carrying molecules and ultimately produce proteins or other molecules of interest [269, 270]. For CFPS, extracts are supplemented with large amounts of NTPs, amino acids, and NAD which power transcription and translation. Additional cofactors are also added to extracts to increase productivity such as crowding agents, antioxidants, cAMP, speridimine to increase ribosome stability, HEPES buffer, and Coenzyme A [271]. Cell extracts prepared with such an energy buffer are in a highly non-equilibrium state. By metabolizing these energy components, cell extract systems relax to equilibrium while expressing proteins, producing biochemical products, or performing computations with biochemical circuits. This process however, is not very efficient with estimates of just at most 65% energy efficiency for bio-production of small molecules [272] and presumably even lower for CFPS [273].

Furthermore, it is widely understood that cell extracts stop functioning because they run out of energy and/or accumulate toxic waste products which inhibit core metabolic processes as opposed to the cellular machinery breaking down. Indirectly, this can be seen by the increased longevity of extracts connected via a semi-permeable membrane to an energy reservoir which acts to provide near additional fuel and sequester toxic waste products [274]. Similar experiments have also been carried out with vesicles showing that extract can function for prolonged times provided its fuel is replenished and waste products removed [275]. However, these kinds of systems are complicated and limit the throughput and scale of cell extract experiments.

Cell extract metabolic engineering and circuit prototyping would greatly benefit from mathematical models describing how cell extract metabolism functions. Systems-level models could be used to identify experimental targets for cell extract metabolic engineering. Understanding how metabolism effects CFPS duration would also en-

able optimization of circuits, energy buffers, and extracts for prototyping purposes. It is also possible that batch-to-batch variability in extracts and differences between preparation methods might have underlying metabolic causes which could be understood through modeling [276]. Unfortunately, most metabolic models of cell extracts currently available are unable to explain all the dynamics of cell free protein expression observed under diverse experimental conditions.

One potential modeling approach would be to use flux balance analysis (FBA), a widely used metabolic modeling framework to understand the metabolism of steady-state-growth of cell cultures [277]. FBA has been so successful, in part, because it does not required detailed enzymatic rate constants and instead uses stoichiometric constraints coupled with optimization in order to predict steady-state metabolic fluxes. Unfortunately, the steady-state assumption of FBA is invalid for the majority of bulk CFPS experiments because extract metabolism is not at steady state.

An alternative modeling approach is to use a chemical reaction network (CRN) model [1] of cell extract metabolism. Briefly, CRNs consist of a set of reactions $I_r \xrightarrow{\rho_r} O_r$ where $I_r, O_r \subseteq \mathcal{S}$ are vectors of chemical species $\mathcal{S}$ and $\rho_r(s) > 0$ is a rate function of the species concentration vector s. Such models result in a system of ODEs: $\frac{ds_i}{dt} = \sum_r (O_r^i - I_r^i)\rho_r(s)$ which can be integrated to understand system dynamics. CRN models have been created to study small metabolic networks such as the lac operon in *E. coli* [15] as well as larger networks like the core glucose metabolism [278]. The challenge with these models is they require detailed biochemical rate parameters and mechanistic information in order to define and parameterize the functions ρ_r. Additionally, parameterized *in vivo* models cannot be directly applied to *in vitro* cell extract systems because enzyme abundances and reaction rates likely vary between the systems. In cell extract, CRN models have been used extensively to understand transcription and translation of genetic circuits [279, 280]. Furthermore, coarse grained phenomelogical models have been used to describe how extract metabolism effect CFPS [186, 281]. More comprehensive models have been built to examine the kinetics of *E. coli*'s central metabolism for the first 3 hours [273]. However these models do not accurately model the many experimental observations regarding extract metabolism, such as the fact that adding more fuel to extracts does not improve yield, or that extracts cease to function after 4-8 hours.

5.4 Results and Discussion

Cell Extract Time Course Metabolomics

We used high throughput untargeted mass spectrometry to broadly identify metabolites [135] which are present and dynamically changing during CFPS in *E. coli* extract. Data was collected across a diverse set of experimental conditions including duration of the cell-free protein synthesis reaction, different cell culture batches, preparation methods, salt concentrations, and additives. Specifically, we used three different cell culture batches, two different magnesium-glutamate concentrations, the addition or absence of 4 nM of a constitutive expressing GFP plasmid, the addition of additional oxygen during the cell-free protein synthesis reactions, and variations in the extract preparation method. The use of many conditions across multiple time points allowed us to statistically find a wide variety of associations in the data. These conditions were distributed in such a way that we could easily pool samples between sets of different conditions and still have enough statistical power to draw meaningful conclusions (see Methods Table 5.1). Details of our statistical methodology can be found in the Methods section.

First, we examined how the distributions of individual metabolites change over time by pooling samples by time point across all the experimental conditions. This analysis showed that 87 of 328 vary statistically significantly over time (Figure 5.1A). These metabolites are spread out over a wide swath of *E. coli* metabolism which will discuss further later. We then examined if different extract conditions gave rise to statistical differences in the measured metabolite dynamics. Surprisingly, the vast majority of conditions we compared produced few statistical differences. Notably, the addition of constitutively active GFP plasmid had no measurable effect on overall extract metabolism, suggesting that cell-free protein synthesis is using relatively little energy compared to other background metabolic processes. The only factor that we found to have statistically significant differences between individual metabolites was the specific extract culture batch (Figure 5.1B) suggesting that variability in growth conditions, when cells are harvested, and other details of the extract preparation protocol can have significant effects on extract metabolism.

Then we simultaneously increased the statistical power of our data and provided a higher level view of *E. coli* extract metabolism by statistically combining the measurements of multiple metabolites involved in the same metabolic pathway [282]. This analysis found that 47 out of 55 analyzed pathways vary significantly in time. Cell culture batch was also revealed to have dramatic changes in 40 pathways

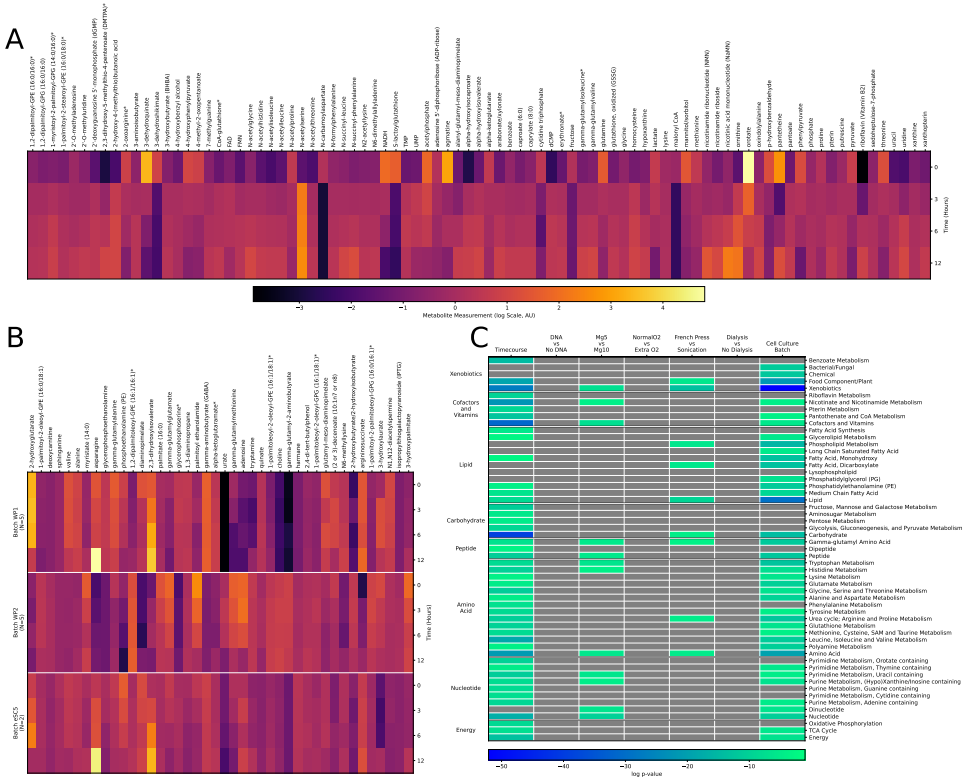


Figure 5.1: An overview of the significant results from the metabolomics statistical analysis. A. Average time course data for all the metabolites which statistically significantly change in time. B. Average data for individual metabolites which vary between extract batches. C. Overview of statistically significantly varying pathways.

emphasizing our previous findings that growth conditions play a major role in extract variability. Lysis method (sonication versus French press) and magnesium-glutamate concentration also resulted in a handful of statistically significant pathway level metabolic changes (Figure 5.1C). Strikingly, even with the increased statistical power afforded by pathway level analysis, cell-free protein synthesis did not result in any statistically significant changes in metabolism.

Fuel and DNA Spiking Experiments

We performed a number of additional experiments to understand how cell extract metabolism effects CFPS and simultaneously understand if CFPS can be used to

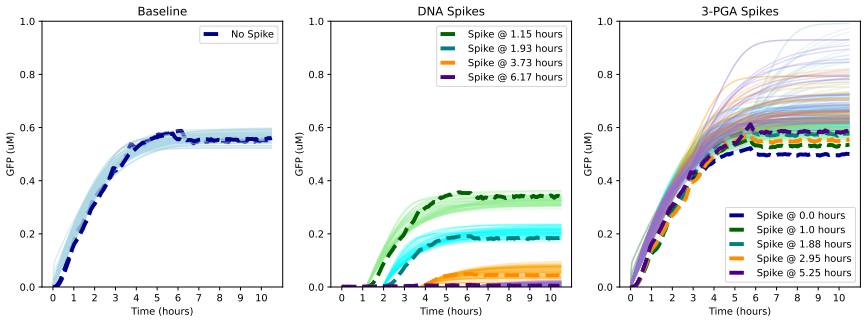


Figure 5.2: A comparison between model simulations for the 100 most likely parameters (light lines) and the experimental spiking and baseline data (dashed lines).

measure high level metabolic effects. First, we added a constitutive expressing GFP construct into extract mixed with energy buffer after various incubation times. These experiments show that extract's innate metabolism diminishes and eventually stops CFPS (as opposed to negative feedback from CFPS itself). Then, we tried adding three additives: additional fuel (3-PGA), HEPES buffer, and water to a CFPS reaction at various times. These results show that adding additional fuel is in fact slightly toxic to extract (Figure 5.2). Decreasing pH had been hypothesized to be responsible for loss of extract function, but HEPES buffer does not appear to dramatically increase CFPS (Figure 5.4). Finally, the water spikes act as a control—it turns out that molecular crowding plays an important role in extract function [283, 284] and therefore relative concentrations and the addition of water can effect CFPS (Figure 5.4). However, we statistically compared the effects of adding water to those of adding HEPES and 3PGA and found that they produced statistically significant differences in the final amount of protein produced. Additionally, when the 3-PGA and HEPES experiments are renormalized to account for the effects of adding water, they still have a statistically significant effect on CFPS. A breakdown of these statistics can be seen in table 5.2. Finally, we measured an ATP time course of extract mixed with buffer which shows that the available amount of ATP diminishes over the course of six hours (Figure 5.3).

A Phenomenological Model of Cell Extract Metabolism

Although the metabolomics measurements produced a significant amount of data, they are not sufficient to fully parameterize a systems level model of *E. coli* extract metabolism. Relatively recent *E. coli* flux balance models (FBA) typically have on the order of over 2000 reactions between over 1000 metabolites involving over

1000 genes [13]. We expect that a systems level extract model would have to be of a similar size considering that most metabolic pathways appear to be active. Furthermore, as discussed in the introduction, the steady state assumption used in flux balance models is not valid for extracts. Therefore, we cannot ignore reaction rate parameters in favor of stoichiometric constraints and optimization used in FBA. Considering that we measured less than 30% of the metabolites likely active in extract metabolism, we chose to take a phenomenological approach to modeling extract metabolism [285, 286]. First, we note that extract is powered by one dominant fuel source (in our experiments 3PGA). This fuel source is fed directly into the citric acid cycle and presumably used for mixed acid fermentation [13]. This allows the extract to regenerate ATP, GTP, NADH, and other energy carrying molecules while simultaneously producing a whole host of other compounds and waste products. Additionally, as shown in the data gathered in this chapter, we know that extract energy buffer is fairly well optimized and that the addition of extra fuel, either initially or later in time, does not improve extract performance and can even be toxic. Finally, if DNA is added to extract after it has been incubated for a few hours, that DNA does not express. Jointly, these observations suggest that waste products build up in extracts which shut down parts of the extract's metabolism.

We require that our phenomenological model matches the following experimental observations:

1. Extract loses the ability to express DNA after 4-6 hours.
2. Adding additional fuel to extract either initially or after a delay does not improve extract performance.
3. Cell-free protein synthesis does not significantly impact extract metabolism.

We then build a simplified metabolic model consisting of three species: fuel F, energy carriers E^* (activated, e.g. ATP) and E (depleted, eg. ADP), and waste W which interact via the following reactions using Hill function kinetics reminiscent

of Michaelis-Menten enzyme kinetics [287]:

$$F + nE \xrightarrow{\rho_{reg}} W + nE^* \quad \rho_{reg}(F, E, W) = V_{reg} \frac{F}{K_F + F} \frac{E}{K_E + E} \frac{1}{1 + (W/K_W)^{n_W}} \tag{5.1}$$

$$F \xrightarrow{\rho_W^{leak}} W \qquad \rho_W^{leak}(F) = V_W^{leak} \frac{(F/K_F^{leak})^{n_F}}{1 + (F/K_F^{leak})^{n_F}} \tag{5.2}$$

$$E^* \xrightarrow{\rho_E^{leak}} E \qquad \rho_E^{leak}(E^*) = V_E^{leak} \frac{E^*/K_E^{leak}}{1 + E^*/K_E^{leak}}. \tag{5.3}$$

The first reaction (5.1) represents energy carrier regeneration from fuel. Here, n is a constant that depends on the particular fuel source (we used $n = 3$ because 1 molecule of 3-PGA produces 3 ATP from 3 ADP, but recognize that this constant is somewhat arbitrary for a non-mechanistic model). The rate function is given by a product of Hill function with a maximum velocity V_{reg}. The first term represents the requirement that F be present for regeneration to occur. The second term represents the requirement that E be present for regeneration to occur. The third term represents the negative feedback due to buildup of waste W which eventually shuts off metabolism.

The second reaction (5.2) represents the degradation of fuel F into waste products W without regeneration of any energy carrier E^*. Because the strength of the negative feedback of W on the rest of the system can be tuned via the constants n_W and K_W, this reaction effectively includes all utilization of the fuel for non-regenerative purposes, only some of which will produce toxic waste. The constants V_W^{leak}, K_F^{leak}, and n_F are the maximum velocity of this reaction, the fuel concentration where this reaction reaches half its maximum velocity, and a Hill coefficient representing how sharp quickly this process turns on as additional fuel is added to the system. Intuitively, we expect this reaction to model the fact that adding additional fuel to extract simply results in the accumulation of additional waste without improving regeneration.

The third reaction (5.3) represents the innate use of energy carriers E^* by the extract. The parameters V_e^{leak} and K_E^{leak} denote the maximum velocity of this reaction and how it depends on the amount of E^* available. Intuitively, we expect this reaction to model the majority of energy utilization in extract.

These three reactions can then be combined with a CFPS module representing gene expression. For the purposes of this work, we coupled the metabolism model to a

one step gene expression model representing constitutive GFP production:

$$G \xrightarrow{\rho_{ex}} G + GFP \qquad \rho_{ex}(G, E^*) = V_{ex} G \frac{(E^*/K_E^{ex})^{n_{ex}}}{1 + (E^*/K_E^{ex})^{n_{ex}}} \tag{5.4}$$

Here, V_{ex} is the maximum rate of gene expression per unit of G. The Hill function term with parameters K_E^{ex} and n_{ex} describe how gene expression depends on the available energy carriers E^*. We note that saturation of gene expression and explicitly modeling transcription and translation could easily be included in this framework, but would require additional data to determine the necessary parameters [288]. Additionally, in a more complex model transcription and translation could consume energy. For example, the methods used in [186] would likely work in conjunction with this metabolism model provided that data were collected to fit parameters for the entire system.

Bayesian Parameter Inference

We fit the phenomenological model to the spiking time course data using Bayesian parameter inference (see Methods). This resulted in a moderately robust fit of our model to the DNA spiking experiments and the baseline measurements. However, the 3PGA spiking experiments were not well fit by the model. We suspect that this is because the effect size of 3PGA spikes is small, resulting in 10% or less reduction on protein expression. We believe that this fit could be improved with more measurements involving larger titrations of 3PGA. A comparison of the mean spiking data with simulations from the top 100 best sets of model parameters can be seen in Figure 5.2.

Model Validation with ATP Time Course Experiments

We did not use the ATP timecourse measurements in order to fit our model. Instead, these were kept separate from the optimization process as a validation data set. Surprisingly, many sets of parameters fit this data better than they fit the 3PGA spiking data. We believe this is evidence that our phenomenological model reasonably approximates a more complex biochemical pathway. In the future, we aim to hone our CFPS measurements in order to more accurately calibrate this model without directly fitting to the ATP timecourse.

Discussion

We set off to build a model of *E. coli* cell extract metabolism. Initially, the hope was that this model could be systems level and at least partially mechanistic. Unfortu-

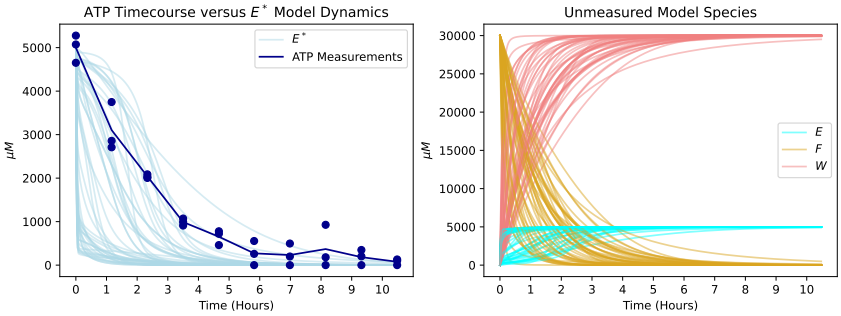


Figure 5.3: 100 best parameter sets simulated from the baseline condition. Left: comparison between model simulations of the species E^* (light lines) and ATP timecourse data (dark dots). Right: simulated dynamics of the other model species E, F, and W.

nately, the metabolomics data we collected highlights the deep biological complexity present in cell extracts and clarifies the scale of experiments and quantitative measurements adequate to rigorously build a systems level extract model. On one hand, this complexity can be seen as a reason to move towards more minimal *in vitro* systems such as PURExpress [289, 290], genelets [291, 292], or PEN-DNA circuits [293]. On the other hand, it demonstrates that cell extract, despite not being alive, has potential as a test bed to develop modeling techniques that work in the face of biological uncertainty and unknown mechanisms at a systems-wide scale. Towards this end, we took steps towards developing a phenomenological model of cell extract metabolism which can be calibrated using simple CFPS experiments. Such a model provides a useful starting point for quantitatively understanding biochemical circuit behavior in cell extracts as fuel supplies dwindle.

Although our model currently only fits the DNA spiking data well, we believe that it can be made more accurate by collecting additional data including more dramatic 3PGA titrations (as a proxy for varying F) and NTP titrations in the extract buffer (as a proxy for varying E^* and E). This model was developed with modularity in mind so it can be coupled to the diverse circuits used in extracts including state-of-the-art vesicle and microfluidic systems. We aim to demonstrate this with future experiments. The use of different fuel sources from 3PGA could also demonstrate if this model is *universal* to a variety of extract conditions.

Indeed, we foresee that quantitative modeling will be necessary to deploy increasingly complex synthetic biological circuits across diverse organisms. Although whole-cell models are beginning to be available [294, 295], they are still difficult to

use for day-to-day engineering applications. Additionally, for non-standard systems detailed mechanistic models will be lacking and instead thoughtful phenomenological models derived from easy-to-gather empirical data will have to suffice. We believe that this phenomenological cell extract metabolism model is a step in that direction.

5.5 Methods

Cell-free Extract Reactions:

Extract Preparation: Crude extracts were all prepared according to the protocol [271]. Briefly, all batches consist of 6 x 0.75L *E. coli* (BL21 Rosetta) cultures grown in 2xYT media supplemented with phosphate buffer to an OD of $\sim$ 3 seeded from 7.5mL of culture grown for 8 hours. Cells are then pelleted and rinsed twice with S30 buffer before being lysed either via sonication [276] or French press [271]. Post lysis, the extract is purified via two rounds of centrifugation at 30000g. All batches were then incubated for 1 hour at 37° C in a runoff reaction. Some batches were dialyzed using 10k MWCO Dialysis Cassettes in S30 buffer. Finally, the extract was flash frozen in liquid nitrogen for later use.

Energy Buffers and Additional Additives: Frozen cell extract samples were thawed and mixed with an energy buffer which uses 3PGA as the main fuel source and also supplements NTPs, amino acids, and a variety of co-factors listed in [271]. Extract (33% by volume) and energy buffer (25% by volume) was mixed together with Mg-Glutamate (to a concentration of 5 or 10 mM), K-glutamate (100mM for the metabolomics measurements, 200mM for spiking experiments), water, and, in some experiments, a constitutively expressing deGFP plasmid [296] MIDI prepped from an overnight cell culture and eluted in water.

Metabolomics

Sample Preparation: The only element of the energy buffer which is varied across these experiments is magnesium glutamate, which is known to strongly effect CFPS [271]. Some samples also received a constitutively expressing GFP plasmid at a final concentration of 4nM. Cell-free reactions (volume = 150 uL) were incubated at 37°C for 0, 3, 6, or 12 hours in a sealed 96-well plate. Some samples received exposure to extra oxygen by periodic unsealing and resealing of the 96-well reaction plate during incubation. Reactions were stopped via the addition of cold methanol (volume = 37.5 uL) [297] and immediately flash frozen in liquid nitrogen.

Condition ID	DNA (nM)	Mg (uM)	Extra O2	Culture Batch ID	Prep Method
U1	0	10	no	WP2	FPD
U2	4	10	no	WP2	FPD
U3	4	10	no	WP2	FPD
U4	4	10	no	WP2	FPD
U5	4	5	no	WP1	SD
U6	4	5	no	WP1	FPD
U7	4	5	no	WP1	FP
U8	0	5	no	WP1	FPD
U9	4	5	no	eSC5	FP
U10	0	5	no	eSC5	FP
T2	4	10	yes	WP2	FPD
T6	4	5	yes	WP1	FPD

Table 5.1: Cell extract conditions. For each sample condition, four time points were collected from independent CFPS experiments which were quenched with cold methanol and flash frozen at 0 hours, 3 hours, 6 hours, and 12 hours. This resulted in a total of 48 different metabolomics samples. Around 328 metabolites were identified across all these samples resulting in a total of 15744 individual small molecule measurements. Prep Methods: FP = French Press; D = Dialysis; S = Sonication.

All the sample conditions are listed in Table 5.1.

Mass Spectrography: Frozen samples were shipped on dry ice to Metabolon for Ultrahigh Performance Liquid Chromatography-Tandem Mass Spectroscopy (UPLC-MS/MS). We quote their technical description of their pipeline: "All methods utilized a Waters ACQUITY ultra-performance liquid chromatography (UPLC) and a Thermo Scientific Q-Exactive high resolution/accurate mass spectrometer interfaced with a heated electrospray ionization (HESI-II) source and Orbitrap mass analyzer operated at 35,000 mass resolution. The sample extract was dried then reconstituted in solvents compatible to each of the four methods. Each reconstitution solvent contained a series of standards at fixed concentrations to ensure injection and chromatographic consistency. One aliquot was analyzed using acidic positive ion conditions, chromatographically optimized for more hydrophilic compounds. In this method, the extract was gradient eluted from a C18 column (Waters UPLC BEH C18-2.1x100 mm, 1.7 µm) using water and methanol, containing 0.05% perfluoropentanoic acid (PFPA) and 0.1% formic acid (FA). Another aliquot was also analyzed using acidic positive ion conditions, however it was chromatographically

optimized for more hydrophobic compounds. In this method, the extract was gradient eluted from the same afore mentioned C18 column using methanol, acetonitrile, water, 0.05% PFPA and 0.01% FA and was operated at an overall higher organic content. Another aliquot was analyzed using basic negative ion optimized conditions using a separate dedicated C18 column. The basic extracts were gradient eluted from the column using methanol and water, however with 6.5mM Ammonium Bicarbonate at pH 8. The fourth aliquot was analyzed via negative ionization following elution from a HILIC column (Waters UPLC BEH Amide 2.1x150 mm, 1.7 μm) using a gradient consisting of water and acetonitrile with 10mM Ammonium Formate, pH 10.8. The MS analysis alternated between MS and data-dependent MSn scans using dynamic exclusion. The scan range varied slighted between methods but covered 70-1000 m/z." Finally, Metabolon used their proprietary software and library of more than 3300 purified samples to identify different metabolites and normalize the resulting data. We emphasize that even with these methods, the metabolite abundances reported are unitless and can only be quantified relatively.

Spiking Experiments

Spiking experiments were conducted with 5 replicates on a 384-well plate using 10uL reaction volumes. Extract and buffer were mixed together and loaded by hand into each well. Then a Labcyte Echo liquid handling robot was used to load DNA and water to a volume of 9uL. The following components were added as 1uL spikes using the Labcyte Echo in different experiments: DNA (final concentration 2.4nM); water; 3PGA (30 mM); and HEPES (50mM). GFP fluorescent data was collected from a Biotek plate reader. Raw data are can be seen in Figure 5.4. The DNA spikes used a different biotek instrument than the other spikes, therefore data was normalized so that the two baseline measurements (shown in red in Figure 5.4) have the same steady final mean (these two conditions contain identical reagents). The other measurements have been calibrated using purified deGFP protein. Additionally, the HEPES and 3PGA spikes were normalized based upon the water spikes to correct for variation due to concentration changes. These different experiments have been statistically compared to each other in Table 5.2 using the Kruskal-Wallis test to analyze variance in the final protein concentrations. Note that the optimal salt concentrations of 10mM Mg and 200mM K were chosen based upon a salt calibration screen for this particular extract batch. The extract used for these experiments was lysed with sonication and was not dialyzed.

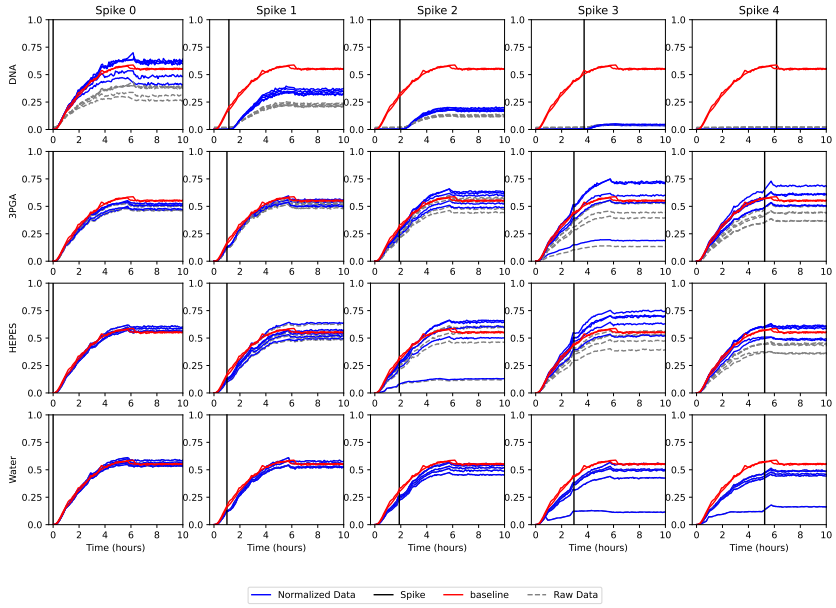


Figure 5.4: Spiking experiment raw data (gray) and renormalized data (blue). Red lines show average baseline measurements from the DNA and Water Spikes $t = 0$.

Comparison	P-value
Baseline: DNA spike 0 vs H_2O spike 0	$p > 0.5$
DNA spikes ($t > 0$) versus baseline	$p < 10^{-7}$
H_2O ($t > 0$) versus baseline	$p < 0.01$ for $t > 1$
3PGA spikes versus baseline	$p < 0.01$
3PGA spikes versus H_2O spikes	$p < 10^{-8}$
H_2O-Renormalized 3PGA spikes versus baseline	$p < 0.01$
HEPES spikes versus baseline	$p < 0.01$ for $t > 1$
HEPES spikes versus H_2O Spikes	$p < 0.001$ for $t < 4$
H_2O-renormalized HEPES spikes versus baseline	$p < 0.01$ for $t > 1$

Table 5.2: Statistics showing that the spiking experiments result in significantly different final protein expression compared to the baseline measurements or the H_2O control spikes. H_2O-renormalized measurements are scaled by the reduction in final output caused by H_2O spikes in order to correct for changes in CFPS due to crowding effects. In some cases, only early $t < n$ or later $t > n$ spikes were found to be significant; if no spike number t is quoted, all spikes are significant.

ATP Time Course Measurements

ATP time course data was collected using the same extract batch and buffer conditions as the spiking experiments. A 384-well plate was loaded with extract and energy buffer and water (30uL total reaction volume per well) and a liquid handling robot (Hamilton) was used to automatically measure the ATP levels using the Cayman Chemical Luciferase based ATP assay every hour for 10 hours (3 replicates per time point).

Statistical Methodology

Statistics comparing measurements between individual metabolites were generated from the Kruskal-Wallis test (a non-parametric ANOVA) [298] between two or more sets of the same metabolite m in difference conditions. Specifically, the time course variation p-value for an individual metabolite m was generated from the Kruskal-Wallis test between four sets of measurements: one for each time point (pooled across all samples). The preparation method p-value for an individual metabolite m was generated from the Kruskal-Wallis test between different sets of preparation methods and/or extract batches (pooled across all timepoints).

Other p-values reported were generated by combining p-values using Empirical Brown's Method [282], an extension of Fisher's method which takes into account covariances in the data used to generate the p-values. Specifically, time course variation p-values between different conditions (e.g. DNA vs No DNA) were generated by computing a p-value with the Kruskal-Wallis test for a metabolite m and timepoint t between the two conditions. The p-values for this metabolite and conditions across all timepoints were then combined with Empirical Brown's Method to get a single time-varying p-value for that condition and metabolite.

Pathway level statistics were computed by combining either time course variation or preparation p-values of many metabolites grouped together based on the pathway annotations provided by Metabolon using Empirical Brown's Method. This methodology significantly enhances the statistical power of the data by allowing less significant variability to contribute to grouped meta-statistical p-values.

Significance testing was conducted at a threshold $p < 0.01$ rescaled for multiple testing using the Holm–Bonferroni correction. A total of 3280 metabolite-specific p-values were computed comparing different conditions of which 171 were found to be significant. Similarly, 550 pathway p-values were computed across different conditions of which 157 were found to be significant.

Simulations and Parameter Inference

Models were produced using BioCRNpyler [136], saved as SBML files [149], and then loaded into Bioscrape [137] which uses the Emcee package [299] for black-box Bayesian parameter inference. The cost function used was the L_2 norm between the mean simulated data for baseline measurements, DNA spike measurements, and 3-PGA Spike measurements, specifically:

$$\mathcal{L}(p) = \sqrt{\sum_{c \in C} \sum_{t} (M_c(t) - X(t, p, X_0(c)))^2}. \tag{5.5}$$

Here, $\mathcal{L}(p)$ is the likelihood of the parameters p. C is a set of all experimental conditions. $M_c(t)$ denotes the measurement of condition c at time t. $X(t, p, X_0(c))$ denotes a simulated trajectory of the CRN evaluated at time t using parameter p with initial condition $X_0(c)$ dependent on the specific measurement. Spiking was modeled by having time dependent rate constants which turn on at the spike times allowing a DNA or Fuel to flow into the system. Inference was split into two runs, the first with 50 walkers for 5000 steps and the second with 100 walkers for 10000 steps. Each step corresponds to simulating each walker with a given set of parameters at each different initial condition. This resulted in over ten million individual simulations with a total of 1250250 parameter combinations sampled. The likelihood of different parameter combinations can be visualized by looking at the marginals and pairwise marginals of the 13 model parameters as shown in 5.5.

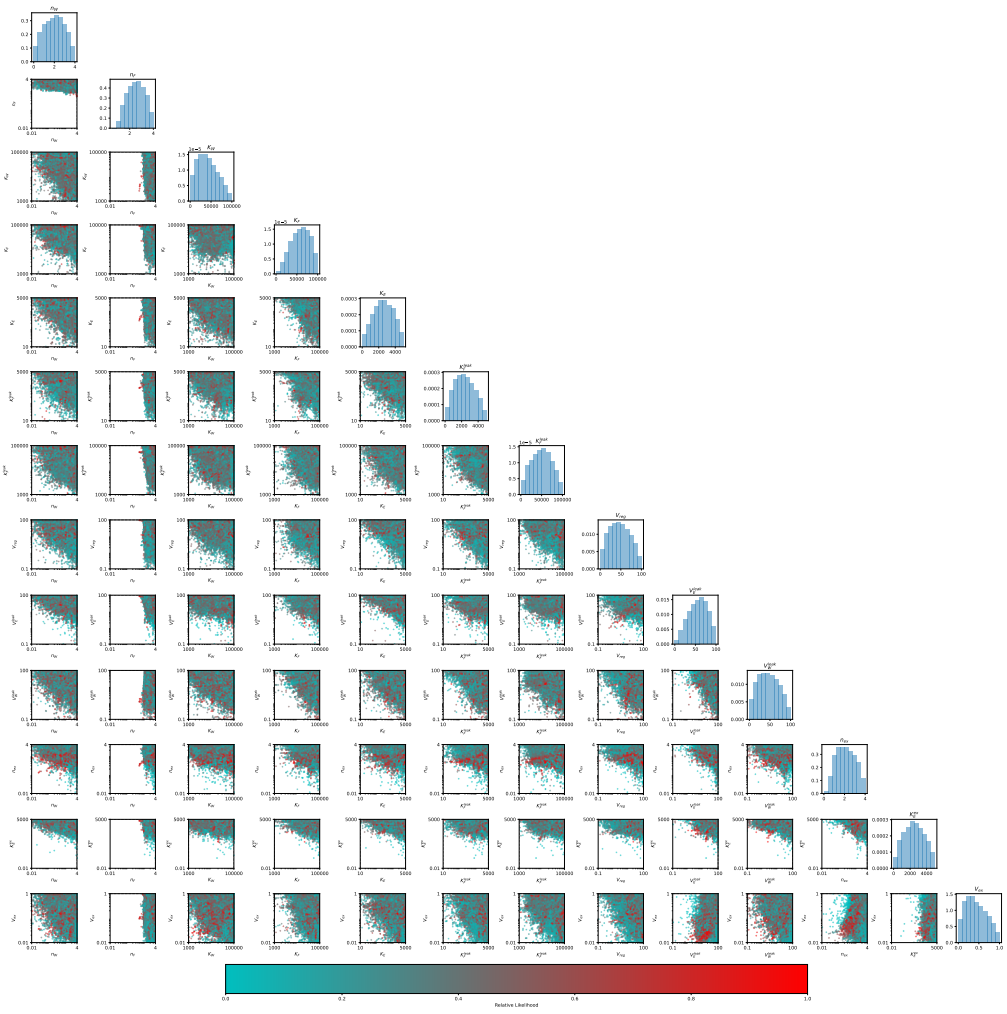
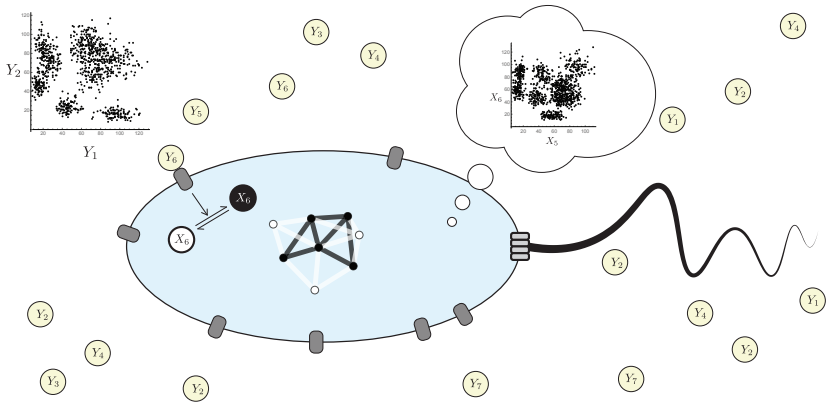


Figure 5.5: Parameter Posterior Visualization. Off-diagonal plots show the pairwise marginals of sampled parameters with the likelihood indicated by the color bar. Diagonal plots show the marginal distributions of each individual parameters.

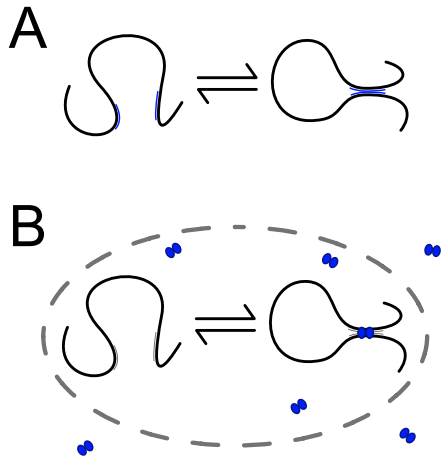
AFTERWORD

I would like to end this thesis by presenting an alternative reading of the preceding chapters as a graphical reordering. In the spirit of the preface, this is *speculative science*, meant to be provocative as opposed to rigorously correct.



Chemical Boltzmann Machines

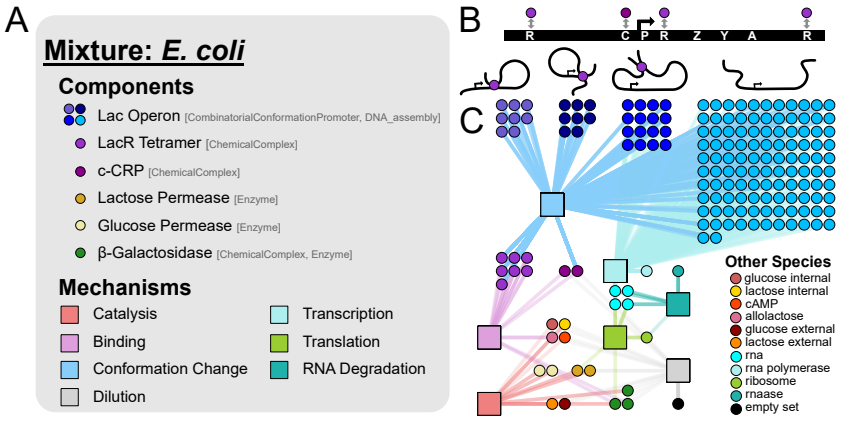
In Chapter 3, we argue that probabilistic inference, as embodied by the machine learning algorithm called a Boltzmann machine, is a plausible model for how a biochemical network, such as a cell, could sense and make decisions based upon a noisy environment. Furthermore, we show that this algorithm can, theoretically, be implemented in multiple ways from purely chemical components.



Detailed Balanced Chemical Reaction Networks
as Generalized Boltzmann Machines

In Chapter 4, we develop a formalism by which the broad class of *detailed balanced chemical reaction networks* can be interpreted as natural probabilistic inference

machines. One example highlighted in this chapter are looping conformations of polymers in small volumes at low copy numbers, such as the genome of a cell. These inferential capabilities can be controlled by the evolution of the DNA's sequence and by auxiliary species (shown in blue) which might favor or inhibit particular conformations.

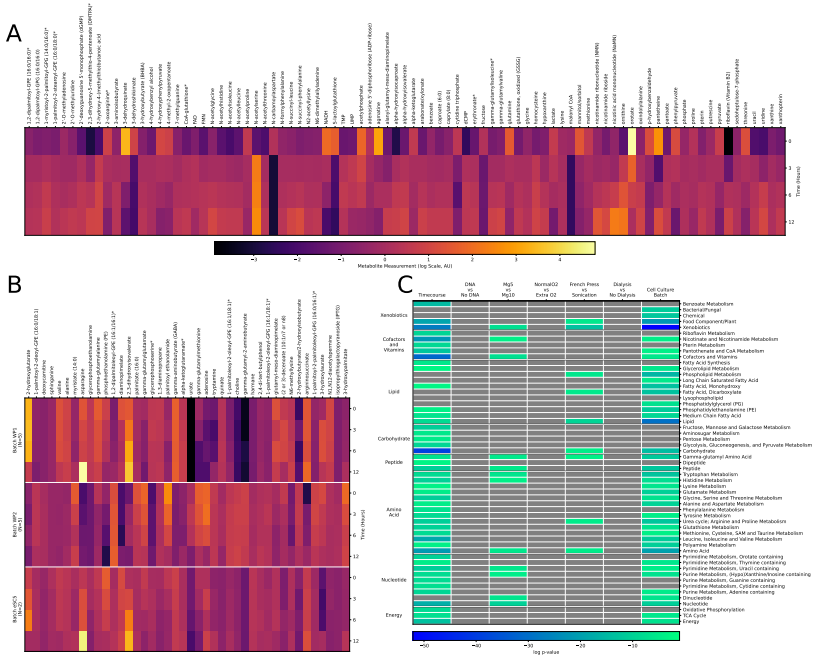


BioCRNpyler: Compiling Chemical Reaction Networks from Biomolecular Parts in Diverse Contexts

In Chapter 2, we describe a BioCRNpyler specification for one of the best studied models in molecular and cell biology, the *lac operon*. This system naturally produces 3 looped and one open DNA conformation with looping induced by the presence of the lac repressor protein (illustrated at the bottom of panel B). The conformation of the DNA allows or prevents the transcription of three genes which play central roles in *E. coli*'s metabolism.

In Chapter 5, we show experimental data suggesting that nearly all of *E. coli's* metabolism is active in cell extract. This suggests that most metabolic enzymes are present in the cells used to produce the extract. However, in a single cell, are all metabolic enzymes present? Our knowledge of the lac operon suggests that that metabolism at the single cell level might be probabilistically controlled in a way analogous to probabilistic graphical models. This would result in a diverse population of single cell state metabolic states.[2] Maybe the global metabolic activity observed in extract is a population level effect, caused by proteins from billions of cells being pooled in a single extract batch. Does this hypothesis help us engineer cell extracts? Not yet, but it suggests that we could improve extracts simply by changing their growth conditions to "probabilistically shift" them into the metabolic space needed for a particular application.

[2] And indeed, some recent experimental work supports this theory [300].



Reducing the Complexity of *E. coli* Cell Extract Metabolism with Phenomenological Modeling

This emergent narrative points to the central theme of this thesis: biochemical computation at the molecular level can *naturally* implement algorithms reminiscent of machine learning algorithms. These parallels between probabilistic graphical models and polymer conformations regulating metabolism are just one of many possible areas where this perspective can bring new interpretations, inspire discoveries, and ultimately enable science to program and control living systems.

BIBLIOGRAPHY

[1] D. Del Vecchio and R. M. Murray, *Biomolecular Feedback Systems*. Princeton University Press, 2014.

[2] J. C. Baez and J. D. Biamonte, *Quantum Techniques in Stochastic Mechanics*. World Scientific, 2018.

[3] D. T. Gillespie, "A Rigorous Derivation of the Chemical Master Equation," *Physica A: Statistical Mechanics and its Applications*, vol. 188, no. 1-3, pp. 404–425, 1992.

[4] T. G. Kurtz, "The Relationship Between Stochastic and Deterministic Models for Chemical Reactions," *The Journal of Chemical Physics*, vol. 57, no. 7, pp. 2976–2978, 1972.

[5] S. A. Benner and A. M. Sismour, "Synthetic Biology," *Nature Reviews Genetics*, vol. 6, no. 7, pp. 533–543, 2005.

[6] D. Del Vecchio, D. Densmore, H. El-Samad, D. Ingber, A. Khalil, S. Kosuri, and C. Tang, "What Have the Principles of Engineering Taught Us about Biological Systems," *Cell Systems*, vol. 2, no. 1, pp. 5–7, 2016.

[7] L. Pasotti, M. Bellato, D. De Marchi, and P. Magni, "Mechanistic Models of Inducible Synthetic Circuits for Joint Description of DNA Copy Number, Regulatory Protein Level, and Cell Load," *Processes*, vol. 7, no. 3, p. 119, 2019.

[8] M. K. Transtrum and P. Qiu, "Bridging Mechanistic and Phenomenological Models of Complex Biological Systems," *PLoS Computational Biology*, vol. 12, no. 5, e1004915, 2016.

[9] A. Pandey and R. M. Murray, "Model Reduction Tools For Phenomenological Modeling of Input-Controlled Biological Circuits," *bioRxiv*, 2020.

[10] J. Gelles and R. Landick, "RNA Polymerase as a Molecular Motor," *Cell*, vol. 93, no. 1, pp. 13–16, 1998.

[11] T. Ghosh, D. Bose, and X. Zhang, "Mechanisms for Activating Bacterial RNA Polymerase," *FEMS Microbiology Reviews*, vol. 34, no. 5, pp. 611–627, 2010.

[12] B. Sendy, D. J. Lee, S. J. Busby, and J. A. Bryant, "RNA Polymerase Supply and Flux Through the Lac Operon in Escherichia Coli," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 371, no. 1707, p. 20 160 080, 2016.

[13] J. D. Orth, T. M. Conrad, J. Na, J. A. Lerman, H. Nam, A. M. Feist, and B. Ø. Palsson, "A Comprehensive Genome-scale Reconstruction of Escherichia Coli Metabolism—2011," *Molecular Systems Biology*, vol. 7, no. 1, p. 535, 2011.

[14] U. Alon, *An Introduction to Systems Biology: Design Principles of Biological Circuits*. CRC press, 2019.

[15] P. Wong, S. Gladney, and J. D. Keasling, "Mathematical Model of the Lac Operon: Inducer Exclusion, Catabolite Repression, and Diauxic growth on Glucose and Lactose," *Biotechnology Progress*, vol. 13, no. 2, pp. 132–143, 1997.

[16] V. Hsiao, A. Swaminathan, and R. M. Murray, "Control Theory for Synthetic Biology: Recent Advances in System Characterization, Control Design, and Controller Implementation for Synthetic Biology," *IEEE Control Systems Magazine*, vol. 38, no. 3, pp. 32–62, 2018.

[17] C. Y. Hu, J. D. Varner, and J. B. Lucks, "Generating Effective Models and Parameters for RNA Genetic Circuits," *ACS Synthetic Biology*, vol. 4, no. 8, pp. 914–926, 2015.

[18] H.-L. Chen, D. Doty, and D. Soloveichik, "Deterministic Function Computation with Chemical Reaction Networks," *Natural Computing*, vol. 13, no. 4, pp. 517–534, 2014.

[19] H.-L. Chen, D. Doty, and D. Soloveichik, "Rate-independent Computation in Continuous Chemical Reaction Networks," in *Proceedings of the 5th Conference on Innovations in Theoretical Computer Science*, 2014, pp. 313–326.

[20] R. Cummings, D. Doty, and D. Soloveichik, "Probability 1 Computation with Chemical Reaction Networks," *Natural Computing*, vol. 15, no. 2, pp. 245–261, 2016.

[21] L. Qian, D. Soloveichik, and E. Winfree, "Efficient Turing-universal Computation with DNA Polymers," in *International Workshop on DNA-Based Computers*, Springer, 2010, pp. 123–140.

[22] D. Soloveichik, G. Seelig, and E. Winfree, "DNA as a Universal Substrate for Chemical Kinetics," *Proceedings of the National Academy of Sciences*, vol. 107, no. 12, pp. 5393–5398, 2010.

[23] S. Badelt, S. W. Shin, R. F. Johnson, Q. Dong, C. Thachuk, and E. Winfree, "A General-purpose CRN-to-DSD Compiler with Formal Verification, Optimization, and Simulation Capabilities," in *International conference on DNA-based computers*, Springer, 2017, pp. 232–248.

[24] L. Qian and E. Winfree, "Scaling Up Digital Circuit Computation with DNA Strand Displacement Cascades," *Science*, vol. 332, no. 6034, pp. 1196–1201, 2011.

[25] A. Padirac, T. Fujii, and Y. Rondelez, "Nucleic Acids for the Rational Design of Reaction Circuits," *Current Opinion in Biotechnology*, vol. 24, no. 4, pp. 575–580, 2013.

[26] N. Srinivas, J. Parkin, G. Seelig, E. Winfree, and D. Soloveichik, "Enzyme-free Nucleic Acid Dynamical Systems," *Science*, vol. 358, no. 6369, 2017.

[27] J. Chappell, K. E. Watters, M. K. Takahashi, and J. B. Lucks, "A Renaissance in RNA Synthetic Biology: New Mechanisms, Applications and Tools for the Future," *Current Opinion in Chemical Biology*, vol. 28, pp. 47–56, 2015.

[28] J. Kim and E. Franco, "RNA Nanotechnology in Synthetic Biology," *Current Opinion in Biotechnology*, vol. 63, pp. 135–141, 2020.

[29] A. Padirac, T. Fujii, and Y. Rondelez, "Bottom-up Construction of In Vitro Switchable Memories," *Proceedings of the National Academy of Sciences*, vol. 109, no. 47, E3212–E3220, 2012.

[30] J. Kim and E. Winfree, "Synthetic in Vitro Transcriptional Oscillators," *Molecular Systems Biology*, vol. 7, no. 1, p. 465, 2011.

[31] J. K. Rosenstein, C. Rose, S. Reda, P. M. Weber, E. Kim, J. Sello, J. Geiser, E. Kennedy, C. Arcadia, A. Dombroski, *et al.*, "Principles of Information Storage in Small-molecule Mixtures," *IEEE Transactions on Nanobioscience*, vol. 19, no. 3, pp. 378–384, 2020.

[32] W. An and J. W. Chin, "Synthesis of Orthogonal Transcription-translation Networks," *Proceedings of the National Academy of Sciences*, vol. 106, no. 21, pp. 8477–8482, 2009.

[33] B. Wang, R. I. Kitney, N. Joly, and M. Buck, "Engineering Modular and Orthogonal Genetic Logic Gates for Robust Digital-like Synthetic Biology," *Nature Communications*, vol. 2, no. 1, pp. 1–9, 2011.

[34] S. Zhang and C. A. Voigt, "Engineered dCas9 with Reduced Toxicity in Bacteria: Implications for Genetic Circuit Design," *Nucleic Acids Research*, vol. 46, no. 20, pp. 11 115–11 125, 2018.

[35] D. Del Vecchio, A. J. Dy, and Y. Qian, "Control Theory Meets Synthetic Biology," *Journal of The Royal Society Interface*, vol. 13, no. 120, p. 20 160 380, 2016.

[36] S. M. Brooks and H. S. Alper, "Applications, Challenges, and Needs for Employing Synthetic Biology Beyond the Lab," *Nature Communications*, vol. 12, no. 1, pp. 1–16, 2021.

[37] A. A. K. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. A. Strychalski, D. Ross, D. Densmore, and C. A. Voigt, "Genetic Circuit Design Automation," *Science*, vol. 352, no. 6281, aac7341–aac7341, Apr. 2016, ISSN: 0036-8075. DOI: 10.1126/science.aac7341.

[38] M. W. Gander, J. D. Vrana, W. E. Voje, J. M. Carothers, and E. Klavins, "Digital Logic Circuits in Yeast with CRISPR-dCas9 NOR Gates," *Nature Communications*, vol. 8, no. 1, pp. 1–11, 2017.

[39] C. Briat, A. Gupta, and M. Khammash, "Antithetic Integral Feedback Ensures Robust Perfect Adaptation in Noisy Biomolecular Networks," *Cell Systems*, vol. 2, no. 1, pp. 15–26, 2016.

[40] Y.-H. Wang, K. Y. Wei, and C. D. Smolke, "Synthetic Biology: Advancing the Design of Diverse Genetic Systems," *Annual Review of Chemical and Biomolecular Engineering*, vol. 4, pp. 69–102, 2013.

[41] R. Sarpeshkar, "Analog Synthetic Biology," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 372, no. 2012, p. 20 130 110, 2014.

[42] D. J. MacKay and D. J. Mac Kay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[44] Y. Wang and J. Zeng, "Predicting Drug-target Interactions Using Restricted Boltzmann Machines," *Bioinformatics*, vol. 29, no. 13, pp. i126–i134, 2013.

[45] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language Models are Few-shot Learners," *arXiv preprint arXiv:2005.14165*, 2020.

[46] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žıdek, A. Potapenko, *et al.*, "Highly Accurate Protein Structure Prediction with AlphaFold," *Nature*, pp. 1–11, 2021.

[47] Z. Ghahramani, "Probabilistic Machine Learning and Artificial Intelligence," *Nature*, vol. 521, no. 7553, pp. 452–459, 2015.

[48] M. I. Jordan and C. Bishop, *An Introduction to Graphical Models*, 2004.

[49] D. C. Knill and W. Richards, *Perception as Bayesian inference*. Cambridge University Press, 1996.

[50] S. J. Gershman, "The Generative Adversarial Brain," *Frontiers in Artificial Intelligence*, vol. 2, p. 18, 2019.

[51] F. Nielsen, "What is an Information Projection," *Notices of the AMS*, vol. 65, no. 3, pp. 321–324, 2018.

[52] F. Rosenblatt, "The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain.," *Psychological Review*, vol. 65, no. 6, p. 386, 1958.

[53] J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982, ISSN: 0027-8424.

[54] O. Rössler, "A Synthetic Approach to Exotic Kinetics (with Examples)," in *Physics and Mathematics of the Nervous System*, Springer, 1974, pp. 546–582.

[55] A. Hjelmfelt, E. Weinberger, and J. Ross, "Chemical Implementation of Neural Networks and Turing Machines," *Proceedings of the National Academy of Sciences*, vol. 88, no. 24, pp. 10 983–10 987, 1991.

[56] A. Hjelmfelt and J. Ross, "Chemical Implementation and Thermodynamics of Collective Neural Networks," *Proceedings of the National Academy of Sciences*, vol. 89, no. 1, pp. 388–391, 1992.

[57] A. Hjelmfelt, F. Schneider, and J. Ross, "Pattern Recognition in Coupled Chemical Kinetic Systems," *Science*, vol. 260, pp. 335–335, 1993.

[58] S. McGregor, V. Vasas, P. Husbands, and C. Fernando, "Evolution of Associative Learning in Chemical Networks," *PLoS Computational Biology*, vol. 8, no. 11, e1002739, 2012.

[59] D. F. Anderson, B. Joshi, and A. Deshpande, "On Reaction Network Implementations of Neural Networks," *Journal of the Royal Society Interface*, vol. 18, no. 177, p. 20 210 031, 2021.

[60] J. Kim, J. J. Hopfield, and E. Winfree, "Neural Network Computation by In Vitro Transcriptional Circuits," *Advances in Neural Information Processing Systems*, vol. 17, pp. 681–688, 2004.

[61] A. Moorman, C. C. Samaniego, C. Maley, and R. Weiss, "A Dynamical Biomolecular Neural Network," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 1797–1802. DOI: 10.1109/CDC40024.2019.9030122.

[62] C. C. Samaniego, A. Moorman, G. Giordano, and 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.

[63] J. Kim, K. White, and E. Winfree, "Construction of an In Vitro Bistable Circuit from Synthetic Transcriptional Switches," *Molecular Systems Biology*, vol. 2, p. 68, 2006.

[64] R. Daniel, S. S. Woo, L. Turicchia, and R. Sarpeshkar, "Analog Transistor Models of Bacterial Genetic Circuits," in *2011 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, IEEE, 2011, pp. 333–336.

[65] R. Daniel, J. R. Rubens, R. Sarpeshkar, and T. K. Lu, "Synthetic Analog Computation in Living Cells," *Nature*, vol. 497, no. 7451, pp. 619–623, 2013.

[66] A. J. Genot, T. Fujii, and Y. Rondelez, "Scaling Down DNA circuits with Competitive Neural Networks," *Journal of The Royal Society Interface*, vol. 10, no. 85, p. 20 130 212, 2013.

[67] X. Liu and K. K. Parhi, "Molecular and DNA Artificial Neural Networks via Fractional Coding," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 3, pp. 490–503, 2020. DOI: 10.1109/TBCAS.2020.2979485.

[68] K. M. Cherry and L. Qian, "Scaling Up Molecular Pattern Recognition with DNA-based Winner-take-all Neural Networks," *Nature*, vol. 559, no. 7714, pp. 370–376, 2018.

[69] H. Pei, X. Xiong, T. Zhu, Y. Zhu, M. Cao, J. Xiao, L. Li, F. Wang, and C. Fan, "Molecular Convolutional Neural Networks with DNA Regulatory Circuits," 2021. DOI: https://doi.org/10.21203/rs.3.rs-926838/v1.

[70] D. Bray, "Intracellular Signalling as a Parallel Distributed Process," *Journal of Theoretical Biology*, vol. 143, no. 2, pp. 215–231, 1990.

[71] J.-C. Chen and M. Conrad, "Pattern Categorization and Generalization with a Virtual Neuromolecular Architecture," *Neural Networks*, vol. 10, no. 1, pp. 111–123, 1997.

[72] P. Banda, C. Teuscher, and M. R. Lakin, "Online Learning in a Chemical Perceptron," *Artificial Life*, vol. 19, no. 2, pp. 195–219, 2013.

[73] P. Banda, C. Teuscher, and D. Stefanovic, "Training an Asymmetric Signal Perceptron Through Reinforcement in an Artificial Chemistry," *Journal of The Royal Society Interface*, vol. 11, no. 93, p. 20 131 100, 2014.

[74] M. R. Lakin and D. Stefanovic, "Supervised Learning in Adaptive DNA Strand Displacement Networks," *ACS Synthetic Biology*, vol. 5, no. 8, pp. 885–897, 2016.

[75] D. Blount, P. Banda, C. Teuscher, and D. Stefanovic, "Feedforward Chemical Neural Network: An In Silico Chemical System that Learns XOR," *Artificial life*, vol. 23, no. 3, pp. 295–317, 2017.

[76] Y. Katz, M. Springer, and W. Fontana, *Embodying Probabilistic Inference in Biochemical Circuits*, 2018. eprint: https://arxiv.org/abs/1806.10161.

[77] M. Gopalkrishnan, "A Scheme for Molecular Computation of Maximum Likelihood Estimators for Log-Linear Models," in *DNA Computing and Molecular Programming*, vol. 9818 of Lecture Notes in Computer Science (LNCS), Springer, 2016, pp. 3–18.

[78] E. Mjolsness, D. H. Sharp, and J. Reinitz, "A Connectionist Model of Development," *Journal of Theoretical Biology*, vol. 152, no. 4, pp. 429–453, 1991.

[79] G. Marnellos and E. D. Mjolsness, "Gene Network Models and Neural Development," *Modeling Neural Development*, pp. 27–48, 2003.

[80] N. Buchler, U. Gerland, and T. Hwa, "On Schemes of Combinatorial Transcription Logic," *Proceedings of the National Academy of Sciences*, vol. 100, no. 9, pp. 5136–5141, 2003, ISSN: 0027-8424.

[81] J. J. Teo, S. S. Woo, and R. Sarpeshkar, "Synthetic Biology: a Unifying Biew and Review Using Analog Circuits," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, no. 4, pp. 453–474, 2015.

[82] N. E. Napp and R. P. Adams, "Message Passing Inference with Chemical Reaction Networks," *Advances in Neural Information Processing Systems*, vol. 26, pp. 2247–2255, 2013.

[83] M. V. Virinchi, A. Behera, and M. Gopalkrishnan, "A Stochastic Molecular Scheme for an Artificial Cell to Infer its Environment from Partial Observations," in *International Conference on DNA-Based Computers*, Springer, 2017, pp. 82–97.

[84] M. V. Virinchi, A. Behera, and M. Gopalkrishnan, "A Reaction Network Scheme which Implements the EM Algorithm," in *International Conference on DNA Computing and Molecular Programming*, Springer, 2018, pp. 189–207.

[85] G. E. Hinton, T. J. Sejnowski, and D. H. Ackley, *Boltzmann Machines: Constraint Satisfaction Networks that Learn*. Carnegie-Mellon University, Department of Computer Science Pittsburgh, PA, 1984.

[86] N. G. Van Kampen, *Stochastic Processes in Physics and Chemistry*. Elsevier, 1992, vol. 1.

[87] C. Maes and K. Netočn, "Time-reversal and Entropy," *Journal of Statistical Physics*, vol. 110, no. 1, pp. 269–310, 2003.

[88] J. S. Lamb and J. A. Roberts, "Time-reversal Symmetry in Dynamical Systems: a Survey," *Physica D: Nonlinear Phenomena*, vol. 112, no. 1-2, pp. 1–39, 1998.

[89] G. G. Hammes and S. Hammes-Schiffer, *Physical Chemistry for the Biological Sciences*. John Wiley & Sons, 2015.

[90] M. Polettini and M. Esposito, "Irreversible Thermodynamics of Open Chemical Networks. I. Emergent Cycles and Broken Conservation Laws," *The Journal of Chemical Physics*, vol. 141, no. 2, 07B610_1, 2014.

[91] R. Rao and M. Esposito, "Nonequilibrium Thermodynamics of Chemical Reaction Networks: Wisdom from Stochastic Thermodynamics," *Physical Review X*, vol. 6, no. 4, p. 041 064, 2016.

[92] T. E. Ouldridge, "The importance of Thermodynamics for Molecular Systems, and the Importance of Molecular Systems for Thermodynamics," *Natural Computing*, vol. 17, no. 1, pp. 3–29, 2018.

[93] D. F. Anderson, G. Craciun, and T. G. Kurtz, "Product-form Stationary Distributions for Deficiency Zero Chemical Reaction Networks," *Bulletin of Mathematical Biology*, vol. 72, no. 8, pp. 1947–1970, 2010.

[94] P. Siuti, J. Yazbek, and T. K. Lu, "Synthetic Circuits Integrating Logic and Memory in Living Cells," *Nature Biotechnology*, vol. 31, no. 5, pp. 448–452, 2013.

[95] S. Regot, J. Macia, N. Conde, K. Furukawa, J. Kjellén, T. Peeters, S. Hohmann, E. De Nadal, F. Posas, and R. Solé, "Distributed Biological Computation with Multicellular Engineered Networks," *Nature*, vol. 469, no. 7329, pp. 207–211, 2011.

[96] Y. Chen, J. K. Kim, A. J. Hirning, K. Josić, and M. R. Bennett, "Emergent Genetic Oscillations in a Synthetic Microbial Consortium," *Science*, vol. 349, no. 6251, pp. 986–989, 2015.

[97] D. M. McCarthy and J. I. Medford, "Quantitative and Predictive Genetic Parts for Plant Synthetic Biology," *Frontiers in Plant Science*, vol. 11, p. 1510, 2020.

[98] R. A. Le Feuvre and N. S. Scrutton, "A Living Foundry for Synthetic Biological Materials: a Synthetic Biology Roadmap to New Advanced Materials," *Synthetic and Systems Biotechnology*, vol. 3, no. 2, pp. 105–112, 2018.

[99] K. French, "Harnessing Synthetic Biology for Sustainable Development," *Nature Sustainability*, vol. 2, no. 4, pp. 250–252, 2019.

[100] L. T. Bereza-Malcolm, G. Mann, and A. E. Franks, "Environmental Sensing of Heavy Metals through Whole Cell Microbial Biosensors: a Synthetic Biology Approach," *ACS Synthetic Biology*, vol. 4, no. 5, pp. 535–546, 2015.

[101] S. Shi, E. L. Ang, and H. Zhao, "In Vivo Biosensors: Mechanisms, Development, and Applications," *Journal of Industrial Microbiology and Biotechnology*, vol. 45, no. 7, pp. 491–516, 2018.

[102] D. Braff, D. Shis, and J. J. Collins, "Synthetic Biology Platform Technologies for Antimicrobial Applications," *Advanced Drug Delivery Reviews*, vol. 105, pp. 35–43, 2016.

[103] J. H. Esensten, J. A. Bluestone, and W. A. Lim, "Engineering Therapeutic T cells: from Synthetic Biology to Clinical Trials," *Annual Review of Pathology: Mechanisms of Disease*, vol. 12, pp. 305–330, 2017.

[104] M.-R. Wu, B. Jusiak, and T. K. Lu, "Engineering Advanced Cancer Therapies with Synthetic Biology," *Nature Reviews Cancer*, vol. 19, no. 4, pp. 187–195, 2019.

[105] C. M. Agapakis and P. A. Silver, "Synthetic Biology: Exploring and Exploiting Genetic Modularity through the Design of Novel Biological Networks," *Molecular BioSystems*, vol. 5, no. 7, pp. 704–713, 2009.

[106] D. Del Vecchio, "Modularity, Context-dependence, and Insulation in Engineered Biological Circuits," *Trends in Biotechnology*, vol. 33, no. 2, pp. 111–119, 2015.

[107] S. Cardinale and A. P. Arkin, "Contextualizing Context for Synthetic Biology–Identifying Causes of Failure of Synthetic Biological Systems," *Biotechnology Journal*, vol. 7, no. 7, pp. 856–866, 2012.

[108] S. C. Sleight and H. M. Sauro, "Visualization of Evolutionary Stability Dynamics and Competitive Fitness of Escherichia Coli Engineered with Randomized Multigene Circuits," *ACS Synthetic Biology*, vol. 2, no. 9, pp. 519–528, 2013.

[109] H. V. Westerhoff and B. O. Palsson, "The Evolution of Molecular Biology into Systems Biology," *Nature Biotechnology*, vol. 22, no. 10, pp. 1249–1252, 2004.

[110] H. Kitano, "Systems Biology: a Brief Overview," *Science*, vol. 295, no. 5560, pp. 1662–1664, 2002.

[111] J. Raes and P. Bork, "Molecular Eco-systems Biology: Towards an Understanding of Community Function," *Nature Reviews Microbiology*, vol. 6, no. 9, pp. 693–699, 2008.

[112] M. Tyers and M. Mann, "From Genomics to Proteomics," *Nature*, vol. 422, no. 6928, pp. 193–197, 2003.

[113] J. R. Idle and F. J. Gonzalez, "Metabolomics," *Cell Metabolism*, vol. 6, no. 5, pp. 348–351, 2007.

[114] Z. Wang, M. Gerstein, and M. Snyder, "RNA-Seq: a Revolutionary Tool for Transcriptomics," *Nature Reviews Genetics*, vol. 10, no. 1, pp. 57–63, 2009.

[115] C. Bock, M. Farlik, and N. C. Sheffield, "Multi-omics of Single Cells: Strategies and Applications," *Trends in Biotechnology*, vol. 34, no. 8, pp. 605–608, 2016.

[116] J. Packer and C. Trapnell, "Single-cell Multi-omics: an Engine for New Quantitative Models of Gene Regulation," *Trends in Genetics*, vol. 34, no. 9, pp. 653–665, 2018.

[117] F. R. Pinu, D. J. Beale, A. M. Paten, K. Kouremenos, S. Swarup, H. J. Schirra, and D. Wishart, "Systems biology and Multi-omics Integration: Viewpoints from the Metabolomics Research Community," *Metabolites*, vol. 9, no. 4, p. 76, 2019.

[118] M. Bizzarri, D. E. Brash, J. Briscoe, V. A. Grieneisen, C. D. Stern, and M. Levin, "A Call for a Better Understanding of Causation in Cell Biology," *Nature Reviews Molecular Cell Biology*, vol. 20, no. 5, pp. 261–262, 2019.

[119] M. B. Elowitz, A. J. Levine, E. D. Siggia, and P. S. Swain, "Stochastic Gene Expression in a Single Cell," *Science*, vol. 297, no. 5584, pp. 1183–1186, 2002.

[120] P. S. Swain, M. B. Elowitz, and E. D. Siggia, "Intrinsic and Extrinsic Contributions to Stochasticity in Gene Expression," *Proceedings of the National Academy of Sciences*, vol. 99, no. 20, pp. 12 795–12 800, 2002.

[121] E. M. Ozbudak, M. Thattai, I. Kurtser, A. D. Grossman, and A. Van Oudenaarden, "Regulation of Noise in the Expression of a Single Gene," *Nature Genetics*, vol. 31, no. 1, pp. 69–73, 2002.

[122] D. Huh and J. Paulsson, "Non-genetic Heterogeneity from Stochastic Partitioning at Cell Division," *Nature Genetics*, vol. 43, no. 2, pp. 95–100, 2011.

[123] J. M. Raser and E. K. O'shea, "Noise in Gene Expression: Origins, Consequences, and Control," *Science*, vol. 309, no. 5743, pp. 2010–2013, 2005.

[124] H. Qian, "Reducing Intrinsic Biochemical Noise in Cells and its Thermodynamic Limit," *Journal of Molecular Biology*, vol. 362, no. 3, pp. 387–392, 2006.

[125] A. Eldar and M. B. Elowitz, "Functional Roles for Noise in Genetic Circuits," *Nature*, vol. 467, no. 7312, pp. 167–173, 2010.

[126] G. M. Church, *From Systems Biology to Synthetic Biology*, 2005.

[127] C. L. Barrett, T. Y. Kim, H. U. Kim, B. Ø. Palsson, and S. Y. Lee, "Systems Biology as a Foundation for Genome-scale Synthetic Biology," *Current Opinion in Biotechnology*, vol. 17, no. 5, pp. 488–492, 2006.

[128] P. Kirk, T. Thorne, and M. P. Stumpf, "Model Selection in Systems and Synthetic Biology," *Current Opinion in Biotechnology*, vol. 24, no. 4, pp. 767–774, 2013.

[129] K. L. Frieda, J. M. Linton, S. Hormoz, J. Choi, K.-H. K. Chow, Z. S. Singer, M. W. Budde, M. B. Elowitz, and L. Cai, "Synthetic Recording and in Situ Readout of Lineage Information in Single Cells," *Nature*, vol. 541, no. 7635, pp. 107–111, 2017.

[130] E. Agmon, R. K. Spangler, C. J. Skalnik, W. Poole, S. M. Peirce, J. H. Morrison, and M. W. Covert, "Vivarium: an Interface and Engine for Integrative Multiscale Modeling in Computational Biology," 2021.

[131] M. Djordjevic, A. Rodic, and S. Graovac, "From Biophysics to 'Omics and Systems Biology," *European Biophysics Journal*, vol. 48, no. 5, pp. 413–424, 2019.

[132] W. Gilpin, Y. Huang, and D. B. Forger, "Learning Dynamics from Large Biological Datasets: Machine Learning Meets Systems Biology," *Current Opinion in Systems Biology*, 2020.

[133] A. D. Silverman, A. S. Karim, and M. C. Jewett, "Cell-free Gene Expression: an Expanded Repertoire of Applications," *Nature Reviews Genetics*, vol. 21, no. 3, pp. 151–170, 2020.

[134] Z. Z. Sun, E. Yeung, C. A. Hayes, V. Noireaux, and R. M. Murray, "Linear DNA for Rapid Prototyping of Synthetic Biological Circuits in an Escherichia Coli Based TX-TL Cell-free System," *ACS Synthetic Biology*, vol. 3, no. 6, pp. 387–397, 2014.

[135] W. B. Dunn and D. I. Ellis, "Metabolomics: Current Analytical Platforms and Methodologies," *TrAC Trends in Analytical Chemistry*, vol. 24, no. 4, pp. 285–294, 2005.

[136] W. Poole, A. Pandey, Z. Tuza, A. Shur, and R. M. Murray, "BioCRNpyler: Compiling Chemical Reaction Networks from Biomolecular Parts in Diverse Contexts," *BioRxiv*, 2020. DOI: https://doi.org/10.1101/2020.08.02.233478,

[137] A. Swaminathan, W. Poole, V. Hsiao, and R. M. Murray, "Fast and Flexible Simulation and Parameter Estimation for Synthetic Biology Using Bioscrape," *bioRxiv*, 2019. DOI: 10.1101/121152. [Online]. Available: https://www.biorxiv.org/content/early/2019/03/25/121152.

[138] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An Imperative Style, High-performance Deep Learning Library," *Advances in Neural Information Processing Systems*, vol. 32, pp. 8026–8037, 2019.

[139] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A System for Large-scale Machine Learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, 2016, pp. 265–283.

[140] The MathWorks, Inc, *MATLAB Simbiology Toolbox*. [Online]. Available: https://www.mathworks.com/help/simbio/.

[141] E. T. Somogyi, J.-M. Bouteiller, J. A. Glazier, M. König, J. K. Medley, M. H. Swat, and H. M. Sauro, "libRoadRunner: a High Performance SBML Simulation and Analysis Library," *Bioinformatics*, vol. 31, no. 20, pp. 3315–3321, 2015.

[142] N. Le Novere, B. Bornstein, A. Broicher, M. Courtot, M. Donizelli, H. Dharuri, L. Li, H. Sauro, M. Schilstra, B. Shapiro, *et al.*, "BioModels Database: a Free, Centralized Database of Curated, Published, Quantitative Kinetic Models of Biochemical and Cellular Systems," *Nucleic Acids Research*, vol. 34, no. suppl_1, pp. D689–D691, 2006.

[143] S. Hoops, S. Sahle, R. Gauges, C. Lee, J. Pahle, N. Simus, M. Singhal, L. Xu, P. Mendes, and U. Kummer, "COPASI—a Complex Pathway Simulator," *Bioinformatics*, vol. 22, no. 24, pp. 3067–3074, 2006.

[144] K. Choi, J. K. Medley, M. König, K. Stocking, L. Smith, S. Gu, and H. M. Sauro, "Tellurium: an Extensible Python-based Modeling Environment for Systems and Synthetic Biology," *Biosystems*, vol. 171, pp. 74–79, 2018.

[145] S. Guiziou, G. Pérution-Kihli, F. Ulliana, M. Leclère, and J. Bonnet, "Exploring the Design Space of Recombinase Logic Circuits," *bioRxiv*, 2019. DOI: 10.1101/711374.

[146] S. Badelt, C. Grun, K. V. Sarma, B. Wolfe, S. W. Shin, and E. Winfree, "A Domain-level DNA Strand displacement Reaction Enumerator Allowing Arbitrary Non-pseudoknotted Secondary Structures," *Journal of the Royal Society Interface*, vol. 17, no. 167, p. 20 190 866, 2020.

[147] M. Vasić, D. Soloveichik, and S. Khurshid, "CRN++: Molecular Programming Language," *Natural Computing*, pp. 1–17, 2020.

[148] C. Spaccasassi, M. R. Lakin, and A. Phillips, "A Logic Programming Language for Computational Nucleic Acid Devices," *ACS Synthetic Biology*, vol. 8, no. 7, pp. 1530–1547, 2018.

[149] M. Hucka, A. Finney, H. M. Sauro, H. Bolouri, J. C. Doyle, H. Kitano, A. P. Arkin, B. J. Bornstein, D. Bray, A. Cornish-Bowden, *et al.*, "The Systems Biology Markup Language (SBML): a Medium for Representation and Exchange of Biochemical Network Models," *Bioinformatics*, vol. 19, no. 4, pp. 524–531, 2003.

[150] M. Galdzicki, K. P. Clancy, E. Oberortner, M. Pocock, J. Y. Quinn, C. A. Rodriguez, N. Roehner, M. L. Wilson, L. Adam, J. C. Anderson, *et al.*, "The Synthetic Biology Open Language (SBOL) Provides a Community Standard for Communicating Designs in Synthetic Biology," *Nature Biotechnology*, vol. 32, no. 6, pp. 545–550, 2014.

[151] C. J. Myers *et al.*, "iBioSim: a Tool for the Analysis and Design of Genetic Circuits," *Bioinformatics*, vol. 25, no. 21, pp. 2848–2849, 2009.

[152] L. Watanabe, T. Nguyen, M. Zhang, Z. Zundel, Z. Zhang, C. Madsen, N. Roehner, and C. Myers, "iBioSim 3: a Tool for Model-based Genetic Circuit Design," *ACS Synthetic Biology*, vol. 8, no. 7, pp. 1560–1563, 2018.

[153] L. A. Harris *et al.*, "BioNetGen 2.2: Advances in Rule-based Modeling," *Bioinformatics*, vol. 32, no. 21, pp. 3366–3368, 2016.

[154] C. F. Lopez, J. L. Muhlich, J. A. Bachman, and P. K. Sorger, "Programming Biological Models in Python using PySB," *Molecular Systems Biology*, vol. 9, no. 1, p. 646, 2013.

[155] Z. A. Tuza *et al.*, "An In Silico Modeling Toolbox for Rapid Prototyping of Circuits in a Biomolecular "Breadboard" System," in *52nd IEEE Conference on Decision and Control*, Dec. 2013, pp. 1404–1410. DOI: 10.1109/CDC.2013.6760079.

[156] *Biocrnpyler*, https://github.com/BuildACell/BioCRNpyler..

[157] B. S. Der, E. Glassey, B. A. Bartley, C. Enghuus, D. B. Goodman, D. B. Gordon, C. A. Voigt, and T. E. Gorochowski, "DNAplotlib: Programmable Visualization of Genetic Designs and Associated Data," *ACS Synthetic Biology*, vol. 6, no. 7, pp. 1115–1119, 2017.

[158] T. S. Gardner, C. R. Cantor, and J. J. Collins, "Construction of a Genetic Toggle Switch in Escherichia coli," *Nature*, vol. 403, no. 6767, pp. 339–342, 2000.

[159] M. B. Elowitz *et al.*, "A Synthetic Oscillatory Network of Transcriptional Regulators," *Nature*, vol. 403, no. 6767, pp. 335–338, 2000.

[160] B. F. Cress, Ö. D. Toparlak, S. Guleria, M. Lebovich, J. T. Stieglitz, J. A. Englaender, J. A. Jones, R. J. Linhardt, and M. A. Koffas, "CRISPathBrick: Modular Combinatorial Assembly of Type II-A CRISPR Arrays for dCas9-mediated Multiplex Transcriptional Repression in E. Coli," *ACS Synthetic Biology*, vol. 4, no. 9, pp. 987–1000, 2015.

[161] S. Jayanthi, K. S. Nilgiriwala, and D. Del Vecchio, "Retroactivity Controls the Temporal Dynamics of Gene Transcription," *ACS Synthetic Biology*, vol. 2, no. 8, pp. 431–441, 2013.

[162] S. C. Strutt, R. M. Torrez, E. Kaya, O. A. Negrete, and J. A. Doudna, "RNA-dependent RNA Targeting by CRISPR-Cas9," *elife*, vol. 7, e32724, 2018.

[163] D. T. Dang and A. T. Phan, "Development of a Ribonuclease Containing a G4-specific Binding Motif for Programmable RNA Cleavage," *Scientific Reports*, vol. 9, no. 1, pp. 1–7, 2019.

[164] X. Yan, T. A. Hoek, R. D. Vale, and M. E. Tanenbaum, "Dynamics of Translation of Single mRNA Molecules In Vivo," *Cell*, vol. 165, no. 4, pp. 976–989, 2016.

[165] R. Milo and R. Phillips, *Cell Biology by the Numbers*. Garland Science, 2015.

[166] M. Santillán and M. C. Mackey, "Quantitative Approaches to the Study of Bistability in the Lac Operon of Escherichia Coli," *Journal of The Royal Society Interface*, vol. 5, no. suppl_1, S29–S39, 2008.

[167] D. T. Gillespie, "Stochastic Simulation of Chemical Kinetics," *Annual Review of Physical Chemistry*, vol. 58, pp. 35–55, 2007.

[168] J. Gunawardena, "Chemical Reaction Network Theory for In-Silico Biologists," *Lecture Notes*, 2003. [Online]. Available: http://vcp.med.harvard.edu/papers/crnt.pdf.

[169] D. Soloveichik, M. Cook, E. Winfree, and J. Bruck, "Computation with Finite Stochastic Chemical Reaction Networks," *natural computing*, vol. 7, no. 4, pp. 615–633, 2008.

[170] T. Schmiedl and U. Seifert, "Stochastic Thermodynamics of Chemical Reaction Networks," *The Journal of Chemical Physics*, vol. 126, no. 4, p. 044 101, 2007.

[171] M. J. Morrison, M. Razo-Mejia, and R. Phillips, "Reconciling Kinetic and Equilibrium Models of Bacterial Transcription," *arXiv preprint arXiv:2006.07772*, 2020.

[172] E. Cinquemani, "Identifiability and Reconstruction of Biochemical Reaction Networks from Population Snapshot Data," *Processes*, vol. 6, no. 9, p. 136, 2018.

[173] Bokeh Development Team, *Bokeh: Python Library for Interactive Visualization*, 2020. [Online]. Available: https://bokeh.org/.

[174] M. Jacomy, T. Venturini, S. Heymann, and M. Bastian, "ForceAtlas2, a Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software," *PLoS ONE*, vol. 9, no. 6, M. R. Muldoon, Ed., e98679, Jun. 2014, ISSN: 1932-6203. DOI: 10.1371/journal.pone.0098679.

[175] S. J. Moore, J. T. MacDonald, S. Wienecke, A. Ishwarbhai, A. Tsipa, R. Aw, N. Kylilis, D. J. Bell, D. W. McClymont, K. Jensen, *et al.*, "Rapid Acquisition and Model-based Analysis of Cell-free Transcription–Translation Reactions from Nonmodel Bacteria," *Proceedings of the National Academy of Sciences*, vol. 115, no. 19, E4340–E4349, 2018.

[176] A. J. Meyer, T. H. Segall-Shapiro, and C. A. Voigt, "Marionette: E. coli Containing 12 Highly-optimized Small Molecule Sensors," *bioRxiv*, p. 285 866, 2018.

[177] L. P. Smith, M. Hucka, S. Hoops, A. Finney, M. Ginkel, C. J. Myers, I. Moraru, and W. Liebermeister, "SBML Level 3 Package: Hierarchical Model Composition, Version 1 Release 3," *Journal of Integrative Bioinformatics*, vol. 12, no. 2, pp. 603–659, 2015.

[178] K. Rutherford, P. Yuan, K. Perry, R. Sharp, and G. D. Van Duyne, "Attachment Site Recognition and Regulation of Directionality by the Serine Integrases," *Nucleic Acids Research*, vol. 41, no. 17, pp. 8341–8356, 2013.

[179] *Buildacell youtube channel*, https://www.youtube.com/watch?v=mu-9MSntd2w&list=PLb2LmjoxZO-g2vbTr3HBcnvVZur8JFiqf.

[180] Z. Karagöz, L. Rijns, P. Y. Dankers, M. van Griensven, and A. Carlier, "Towards Understanding the Messengers of Extracellular Space: Computational Models of Outside-in Integrin Reaction Networks," *Computational and Structural Biotechnology Journal*, 2020.

[181] A. Pandey and R. M. Murray, "A Two-state Ribosome and Protein Model Can Robustly Capture the Chemical Reaction Dynamics of Gene Expression," *bioRxiv*, 2020. DOI: 10.1101/2020.11.25.399287.

[182] L. N. Merk, A. S. Shur, A. Pandey, R. M. Murray, and L. N. Green, "Engineering Logical Inflammation Sensing Circuit for Gut Modulation," *bioRxiv*, 2020. DOI: 10.1101/2020.11.10.377085.

[183] *Codecov*, https://codecov.io/.

[184] M. Storch, M. C. Haines, and G. S. Baldwin, "DNA-BOT: a Low-cost, Automated DNA Assembly Platform for Synthetic Biology," *Synthetic Biology*, vol. 5, no. 1, ysaa010, 2020.

[185] N. Roehner, Z. Zhang, T. Nguyen, and C. J. Myers, "Generating Systems Biology Markup Language Models from the Synthetic Biology Open Language," *ACS Synthetic Biology*, vol. 4, no. 8, pp. 873–879, 2015.

[186] V. Singhal, Z. A. Tuza, Z. Z. Sun, and R. M. Murray, "A MATLAB Toolbox for Modeling Genetic Circuits in Cell-free Systems," *Synthetic Biology*, vol. 6, no. 1, ysab007, 2021.

[187] W. Poole, A. Ortiz-Munoz, A. Behera, N. S. Jones, T. E. Ouldridge, E. Winfree, and M. Gopalkrishnan, "Chemical Boltzmann Machines," in *International Conference on DNA-Based Computers*, Springer, 2017, pp. 210–231. DOI: 10.1007/978-3-319-66799-7_14,

[188] D. Bray, "Protein Molecules as Computational Elements in Living Cells," *Nature*, vol. 376, no. 6538, p. 307, 1995.

[189] D. Bray, *Wetware: a Computer in Every Living Cell*. Yale University Press, 2009.

[190] H. McAdams and A. Arkin, "Stochastic Mechanisms in Gene Expression," *Proceedings of the National Academy of Sciences*, vol. 94, no. 3, pp. 814–819, 1997, ISSN: 0027-8424.

[191] T. Perkins and P. Swain, "Strategies for Cellular Decision-making," *Molecular Systems Biology*, vol. 5, no. 1, p. 326, 2009, ISSN: 1744-4292.

[192] S. Muroga, *Threshold Logic and its Applications*. Wiley Interscience, 1971.

[193] K. Hellingwerf, P. Postma, J. Tommassen, and H. Westerhoff, "Signal Transduction in Bacteria: Phospho-neural Network(s) in Escherichia Coli," *FEMS Microbiology Reviews*, vol. 16, no. 4, pp. 309–321, 1995.

[194] T. Mestl, C. Lemay, and L. Glass, "Chaos in High-dimensional Neural and Gene Networks," *Physica D: Nonlinear Phenomena*, vol. 98, no. 1, pp. 33–52, 1996.

[195] J. Deutsch, "Collective Regulation by Non-coding RNA," *arXiv preprint arXiv:1409.1899*, 2014.

[196] J. Deutsch, "Associative Memory by Collective Regulation of Non-Coding RNA," *arXiv preprint arXiv:1608.05494*, 2016.

[197] L. Qian, E. Winfree, and J. Bruck, "Neural Network Computation with DNA Strand Displacement Cascades," *Nature*, vol. 475, no. 7356, pp. 368–372, 2011, ISSN: 0028-0836.

[198] I. Lestas, J. Paulsson, N. E. Ross, and G. Vinnicombe, "Noise in Gene Regulatory Networks," *IEEE Transactions on Automatic Control*, vol. 53, no. Special Issue, pp. 189–200, 2008.

[199] I. Lestas, G. Vinnicombe, and J. Paulsson, "Fundamental Limits on the Suppression of Molecular Fluctuations," *Nature*, vol. 467, no. 7312, pp. 174–178, 2010.

[200] J. Veening, W. Smits, and O. Kuipers, "Bistability, Epigenetics, and Bet-hedging in Bacteria," *Annual Review of Microbiology*, vol. 62, pp. 193–210, 2008, ISSN: 0066-4227.

[201] G. Balázsi, A. van Oudenaarden, and J. Collins, "Cellular Decision Making and Biological Noise: from Microbes to Mammals," *Cell*, vol. 144, no. 6, pp. 910–925, 2011, ISSN: 0092-8674.

[202] L. Tsimring, "Noise in Biology," *Reports on Progress in Physics*, vol. 77, no. 2, p. 26 601, 2014, ISSN: 0034-4885.

[203] V. Mansinghka, "Natively Probabilistic Computation," Ph.D. dissertation, Massachusetts Institute of Technology, 2009.

[204] S. Wang, X. Zhang, Y. Li, R. Bashizade, S. Yang, C. Dwyer, and A. R. Lebeck, "Accelerating Markov Random Field Inference Using Molecular Optical Gibbs Sampling Units," in *Proceedings of the 43rd International Symposium on Computer Architecture*, IEEE Press, 2016, pp. 558–569, ISBN: 1467389471.

[205] J. Fiser, P. Berkes, G. Orbán, and M. Lengyel, "Statistically Optimal Perception and Learning: from Behavior to Neural Representations," *Trends in Cognitive Sciences*, vol. 14, no. 3, pp. 119–130, 2010, ISSN: 1364-6613.

[206] A. Pouget, J. Beck, W. Ma, and P. Latham, "Probabilistic Brains: Knowns and Unknowns," *Nature Neuroscience*, vol. 16, no. 9, pp. 1170–1178, 2013, ISSN: 1097-6256.

[207] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A Learning Algorithm for Boltzmann Machines," *Cognitive science*, vol. 9, no. 1, pp. 147–169, 1985.

[208] T. Tanaka, "Mean-field Theory of Boltzmann Machine Learning," *Physical Review E*, vol. 58, no. 2, p. 2302, 1998.

[209] Y. Tang and I. Sutskever, "Data Normalization in the Learning of Restricted Boltzmann Machines," *Department of Computer Science, University of Toronto, Technical Report UTML-TR-11-2*, 2011.

[210] G. W. Taylor and G. E. Hinton, "Factored Conditional Restricted Boltzmann Machines for Modeling Motion Style," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009, pp. 1025–1032.

[211] G. Casella and E. George, "Explaining the Gibbs Sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, 1992, ISSN: 0003-1305.

[212] H. Qian, "Phosphorylation Energy Hypothesis: Open Chemical Systems and their Biological Functions," *Annual Review of Physical Chemistry*, vol. 58, pp. 113–142, 2007, ISSN: 0066-426X.

[213] D. Beard and H. Qian, *Chemical Biophysics: Quantitative Analysis of Cellular Systems*. Cambridge University Press, 2008, ISBN: 1139470078.

[214] T. Ouldridge, "The Importance of Thermodynamics for Molecular Systems, and the Importance of Molecular Systems for Thermodynamics," *arXiv preprint arXiv:1702.00360*, 2017.

[215] B. Joshi, "A Detailed Balanced Reaction Network is Sufficient but Not Necessary for its Markov Chain to Be Detailed Balanced," *arXiv preprint arXiv:1312.4196*, 2013.

[216] A. Erez, T. Byrd, R. Vogel, G. Altan-Bonnet, and A. Mugler, "Criticality of Biochemical Feedback," *arXiv preprint arXiv:1703.04194*, 2017.

[217] Y. LeCun, C. Cortes, and C. Burges, *The MNIST Database of Handwritten Digits*, 1998.

[218] B. Alberts, D. Bray, K. Hopkin, A. D. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *Essential Cell Biology*. Garland Science, 2015.

[219] I. Santiago and F. C. Simmel, "Self-Propulsion Strategies for Artificial Cell-Like Compartments," *Nanomaterials*, vol. 9, no. 12, p. 1680, 2019.

[220] S. K. Aoki, G. Lillacci, A. Gupta, A. Baumschlager, D. Schweingruber, and M. Khammash, "A Universal Biomolecular Integral Feedback Controller for Robust Perfect Adaptation," *Nature*, vol. 570, no. 7762, pp. 533–537, 2019.

[221] B.-Y. Xu, J. Xu, and T. Yomo, "A Protocell with Fusion and Division," *Biochemical Society Transactions*, vol. 47, no. 6, pp. 1909–1919, 2019.

[222] S. Slomovic, K. Pardee, and J. J. Collins, "Synthetic Biology Devices for In Vitro and In Vivo Diagnostics," *Proceedings of the National Academy of Sciences*, vol. 112, no. 47, pp. 14 429–14 435, 2015.

[223] A. D. Gordon, T. A. Henzinger, A. V. Nori, and S. K. Rajamani, "Probabilistic Programming," in *Future of Software Engineering Proceedings*, 2014, pp. 167–181.

[224] R. Ranganath, L. Tang, L. Charlin, and D. Blei, "Deep Exponential Families," in *Artificial Intelligence and Statistics*, PMLR, 2015, pp. 762–771.

[225] R. Sarpeshkar, "Analog versus Digital: Extrapolating from Electronics to Neurobiology," *Neural Computation*, vol. 10, no. 7, pp. 1601–1638, 1998.

[226] C. D. James, J. B. Aimone, N. E. Miner, C. M. Vineyard, F. H. Rothganger, K. D. Carlson, S. A. Mulder, T. J. Draelos, A. Faust, M. J. Marinella, *et al.*, "A Historical Survey of Algorithms and Hardware Architectures for Neural-inspired and Neuromorphic Computing Applications," *Biologically Inspired Cognitive Architectures*, vol. 19, pp. 49–64, 2017.

[227] N. E. Buchler, U. Gerland, and T. Hwa, "On schemes of Combinatorial Transcription Logic," *Proceedings of the National Academy of Sciences*, vol. 100, no. 9, pp. 5136–5141, 2003.

[228] A. A. Faisal, L. P. Selen, and D. M. Wolpert, "Noise in the Nervous System," *Nature reviews neuroscience*, vol. 9, no. 4, pp. 292–303, 2008.

[229] J. J. Hopfield, "Kinetic Proofreading: a New Mechanism for Reducing Errors in Biosynthetic Processes Requiring High Specificity," *Proceedings of the National Academy of Sciences*, vol. 71, no. 10, pp. 4135–4139, 1974.

[230] C. G. Evans and E. Winfree, "Physical Principles for DNA Tile Self-assembly," *Chemical Society Reviews*, vol. 46, no. 12, pp. 3808–3829, 2017.

[231] S. Hooshangi, S. Thiberge, and R. Weiss, "Ultrasensitivity and Noise Propagation in a Synthetic Transcriptional Cascade," *Proceedings of the National Academy of Sciences*, vol. 102, no. 10, pp. 3581–3586, 2005.

[232] K. A. Fujita, Y. Toyoshima, S. Uda, Y.-i. Ozaki, H. Kubota, and S. Kuroda, "Decoupling of Receptor and Downstream Signals in the Akt Pathway by its Low-pass Filter Characteristics," *Science Signaling*, vol. 3, no. 132, ra56–ra56, 2010.

[233] L. Goentoro, O. Shoval, M. W. Kirschner, and U. Alon, "The Incoherent Feedforward Loop can Provide Fold-change Detection in Gene Regulation," *Molecular Cell*, vol. 36, no. 5, pp. 894–899, 2009.

[234] J. Kim, I. Khetarpal, S. Sen, and R. M. Murray, "Synthetic Circuit for Exact Adaptation and Fold-change Detection," *Nucleic acids research*, vol. 42, no. 9, pp. 6078–6089, 2014.

[235] D. Cappelletti, A. Ortiz-Muñoz, D. F. Anderson, and E. Winfree, "Stochastic Chemical Reaction Networks for Robustly Approximating Arbitrary Probability Distributions," *Theoretical Computer Science*, vol. 801, pp. 64–95, 2020.

[236] E. Winfree, "Chemical Reaction Networks and Stochastic Local Search," in *International Conference on DNA Computing and Molecular Programming*, Springer, 2019, pp. 1–20.

[237] R. D.Shachter and M. A.Peot, "Decision Making Using Probabilistic Inference Methods," *Eighth Conference on Uncertainty in Artificial Intelligence*, pp. 276–283, 1992.

[238] R. Salakhutdinov, "Learning Deep Generative Models," *Annual Review of Statistics and Its Application*, vol. 2, pp. 361–385, 2015.

[239] T. J. Sejnowski, "Higher-order Boltzmann Machines," in *AIP Conference Proceedings*, American Institute of Physics, vol. 151, 1986, pp. 398–403.

[240] R. Salakhutdinov and G. Hinton, "Deep Boltzmann Machines," in *Artificial Intelligence and Statistics*, PMLR, 2009, pp. 448–455.

[241] Y. Tang, R. Salakhutdinov, and G. Hinton, "Robust Boltzmann Machines for Recognition and Denoising," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2012, pp. 2264–2271.

[242] N. Le Roux and Y. Bengio, "Representational Power of Restricted Boltzmann Machines and Deep Belief Networks," *Neural Computation*, vol. 20, no. 6, pp. 1631–1649, 2008.

[243] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio, "Learning Algorithms for the Classification Restricted Boltzmann Machine," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 643–669, 2012.

[244] M. Cook, D. Soloveichik, E. Winfree, and J. Bruck, "Programmability of Chemical Reaction Networks," in *Algorithmic Bioprocesses*, Springer, 2009, pp. 543–584.

[245] Y.-J. Chen, N. Dalchau, N. Srinivas, A. Phillips, L. Cardelli, D. Soloveichik, and G. Seelig, "Programmable Chemical Controllers made from DNA," *Nature Nanotechnology*, vol. 8, no. 10, pp. 755–762, 2013.

[246] J. Leroux and S. Schmitz, "Reachability in Vector Addition Systems is Primitive-recursive in Fixed Dimension," in *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, IEEE, 2019, pp. 1–13.

[247] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.

[248] N. A. Becker, A. M. Greiner, J. P. Peters, and L. J. Maher III, "Bacterial Promoter Repression by DNA Looping without Protein–protein Binding Competition," *Nucleic Acids Research*, vol. 42, no. 9, pp. 5495–5504, 2014.

[249] A. M. Chiariello, C. Annunziatella, S. Bianco, A. Esposito, and M. Nicodemi, "Polymer Physics of Chromosome Large-scale 3D Organisation," *Scientific reports*, vol. 6, no. 1, pp. 1–8, 2016.

[250] P.-G. De Gennes and P.-G. Gennes, *Scaling Concepts in Polymer Physics*. Cornell University Press, 1979.

[251] E. Wallace, D. Gillespie, K. Sanft, and L. Petzold, "Linear Noise Approximation is Valid over Limited Times for Any Chemical System that is Sufficiently Large," *IET Systems Biology*, vol. 6, no. 4, pp. 102–115, 2012.

[252] N. v. Kampen, "A Power Series Expansion of the Master Equation," *Canadian Journal of Physics*, vol. 39, no. 4, pp. 551–567, 1961.

[253] T. E. Ouldridge, R. A. Brittain, and P. R. t. Wolde, "The Power of Being Explicit: Demystifying Work, Heat, and Free Energy in the Physics of Computation," *arXiv preprint arXiv:1812.09572*, 2018.

[254] R. A. Brittain, N. S. Jones, and T. E. Ouldridge, "What Would it Take to Build a Thermodynamically Reversible Universal Turing Machine? Computational and Thermodynamic Constraints in a Molecular Design," *arXiv preprint arXiv:2102.03388*, 2021.

[255] C. H. Bennett, "The Thermodynamics of Computation—-A Review," *International Journal of Theoretical Physics*, vol. 21, no. 12, pp. 905–940, 1982.

[256] E. Chitambar and G. Gour, "Quantum Resource Theories," *Reviews of Modern Physics*, vol. 91, no. 2, p. 025 001, 2019.

[257] L. Shang, Y. Cheng, and Y. Zhao, "Emerging Droplet Microfluidics," *Chemical reviews*, vol. 117, no. 12, pp. 7964–8040, 2017.

[258] A. Gopinath, E. Miyazono, A. Faraon, and P. W. Rothemund, "Engineering and Mapping Nanocavity Emission Via Precision Placement of DNA Origami," *Nature*, vol. 535, no. 7612, pp. 401–405, 2016.

[259] N. Laohakunakorn, L. Grasemann, B. Lavickova, G. Michielin, A. Shahein, Z. Swank, and S. J. Maerkl, "Bottom-up Construction of Complex Biomolecular Systems with Cell-free Synthetic Biology," *Frontiers in Bioengineering and Biotechnology*, vol. 8, p. 213, 2020.

[260] V. Noireaux and A. P. Liu, "The New Age of Cell-free Biology," *Annual Review of Biomedical Engineering*, vol. 22, pp. 51–77, 2020.

[261] E. G. Worst, M. P. Exner, A. De Simone, M. Schenkelberger, V. Noireaux, N. Budisa, and A. Ott, "Cell-free Expression with the Toxic Amino Acid Canavanine," *Bioorganic & Medicinal Chemistry Letters*, vol. 25, no. 17, pp. 3658–3660, 2015.

[262] J. E. Kay and M. C. Jewett, "A Cell-free System for Production of 2, 3-butanediol is Robust to Growth-toxic Compounds," *Metabolic Engineering Communications*, vol. 10, e00114, 2020.

[263] S. J. Moore, J. T. MacDonald, S. Wienecke, A. Ishwarbhai, A. Tsipa, R. Aw, N. Kylilis, D. J. Bell, D. W. McClymont, K. Jensen, *et al.*, "Rapid Acquisition and Model-based Analysis of Cell-free Transcription–Translation Reactions From Nonmodel Bacteria," *Proceedings of the National Academy of Sciences*, vol. 115, no. 19, E4340–E4349, 2018.

[264] A. S. Karim, J. T. Heggestad, S. A. Crowe, and M. C. Jewett, "Controlling Cell-free Metabolism Through Physiochemical Perturbations," *Metabolic Engineering*, vol. 45, pp. 86–94, 2018.

[265] R. Marshall, C. S. Maxwell, S. P. Collins, T. Jacobsen, M. L. Luo, M. B. Begemann, B. N. Gray, E. January, A. Singer, Y. He, *et al.*, "Rapid and Scalable Characterization of CRISPR Technologies Using an E. coli Cell-free Transcription-Translation System," *Molecular Cell*, vol. 69, no. 1, pp. 146–157, 2018.

[266] L. E. Contreras-Llano and C. Tan, "High-throughput Screening of Biomolecules Using Cell-free Gene Expression Systems," *Synthetic Biology*, vol. 3, no. 1, ysy012, 2018.

[267] H. Niederholtmeyer, Z. Z. Sun, Y. Hori, E. Yeung, A. Verpoorte, R. M. Murray, and S. J. Maerkl, "Rapid Cell-free Forward Engineering of Novel Genetic Ring Oscillators," *elife*, vol. 4, e09771, 2015.

[268] A. S. Karim and M. C. Jewett, "A Cell-free Framework for Rapid Biosynthetic Pathway Prototyping and Enzyme Discovery," *Metabolic Engineering*, vol. 36, pp. 116–126, 2016.

[269] K. A. Calhoun and J. R. Swartz, "Energizing Cell-free Protein Synthesis with Glucose Metabolism," *Biotechnology and Bioengineering*, vol. 90, no. 5, pp. 606–613, 2005.

[270] F. Caschera and V. Noireaux, "Synthesis of 2.3 mg/ml of Protein with an All Escherichia Coli Cell-free Transcription–translation System," *Biochimie*, vol. 99, pp. 162–168, 2014.

[271] Z. Z. Sun, C. A. Hayes, J. Shin, F. Caschera, R. M. Murray, and V. Noireaux, "Protocols for Implementing an Escherichia Coli Based TX-TL Cell-free Expression System for Synthetic Biology," *Journal of Visualized Experiments: JoVE*, no. 79, 2013.

[272] J. U. Bowie, S. Sherkhanov, T. P. Korman, M. A. Valliere, P. H. Opgenorth, and H. Liu, "Synthetic Biochemistry: the Bio-inspired Cell-free Approach to Commodity Chemical Production," *Trends in Biotechnology*, vol. 38, no. 7, pp. 766–778, 2020.

[273] N. Horvath, M. Vilkhovoy, J. A. Wayman, K. Calhoun, J. Swartz, and J. D. Varner, "Toward a Genome Scale Sequence Specific Dynamic Model of Cell-free Protein Synthesis in Escherichia Coli," *Metabolic Engineering Communications*, vol. 10, e00113, 2020.

[274] D. Garenne, S. Thompson, A. Brisson, A. Khakimzhan, and V. Noireaux, "The All-E. Coli TXTL Toolbox 3.0: New Capabilities of a Cell-Free Synthetic Biology Platform," *Synthetic Biology*, 2021.

[275] J. Garamella, R. Marshall, M. Rustad, and V. Noireaux, "The All E. coli TX-TL Toolbox 2.0: a Platform for Cell-free Synthetic Biology," *ACS Synthetic Biology*, vol. 5, no. 4, pp. 344–355, 2016.

[276] Y.-C. Kwon and M. C. Jewett, "High-throughput Preparation Methods of Crude Extract for Robust Cell-free Protein Synthesis," *Scientific Reports*, vol. 5, no. 1, pp. 1–8, 2015.

[277] J. D. Orth, I. Thiele, and B. Ø. Palsson, "What is Flux Balance Analysis?" *Nature Biotechnology*, vol. 28, no. 3, pp. 245–248, 2010.

[278] A. Khodayari, A. R. Zomorrodi, J. C. Liao, and C. D. Maranas, "A Kinetic Model of Escherichia Coli Core Metabolism Satisfying Multiple Sets of Mutant Flux Data," *Metabolic Engineering*, vol. 25, pp. 50–62, 2014.

[279] E. Karzbrun, J. Shin, R. H. Bar-Ziv, and V. Noireaux, "Coarse-grained Dynamics of Protein Synthesis in a Cell-free System," *Physical Review Letters*, vol. 106, no. 4, p. 048 104, 2011.

[280] R. Marshall and V. Noireaux, "Quantitative Modeling of Transcription and Translation of an All-E. coli Cell-free System," *Scientific Reports*, vol. 9, no. 1, pp. 1–12, 2019.

[281] D. Siegal-Gaskins, Z. A. Tuza, J. Kim, V. Noireaux, and R. M. Murray, "Gene Circuit Performance Characterization and Resource Usage in a Cell-free "Breadboard"," *ACS Synthetic Biology*, vol. 3, no. 6, pp. 416–425, 2014.

[282] W. Poole, D. L. Gibbs, I. Shmulevich, B. Bernard, and T. A. Knijnenburg, "Combining Dependent P-values with an Empirical Adaptation of Brown's Method," *Bioinformatics*, vol. 32, no. 17, pp. i430–i436, 2016.

[283] R. J. Ellis, "Macromolecular Crowding: Obvious but Underappreciated," *Trends in Biochemical Sciences*, vol. 26, no. 10, pp. 597–604, 2001.

[284] X. Ge, D. Luo, and J. Xu, "Cell-free Protein Expression Under Macromolecular Crowding Conditions," *PloS One*, vol. 6, no. 12, e28707, 2011.

[285] M. Basan, "Resource Allocation and Metabolism: the Search for Governing Principles," *Current Opinion in Microbiology*, vol. 45, pp. 77–83, 2018.

[286] D. Dutta and S. Saini, "Phenomenological Models as Effective Tools to Discover Cellular Design Principles," *Archives of Microbiology*, vol. 201, no. 3, pp. 283–293, 2019.

[287] B. O. Palsson and E. N. Lightfoot, "Mathematical Modelling of Dynamics and Control in Metabolic Networks. I. On Michaelis-Menten Kinetics," *Journal of Theoretical Biology*, vol. 111, no. 2, pp. 273–302, 1984.

[288] W. Halter, F. Allgower, R. M. Murray, and A. Gyorgy, "Optimal Experiment Design and Leveraging Competition for Shared Resources in Cell-free Extracts," in *2018 IEEE Conference on Decision and Control (CDC)*, IEEE, 2018, pp. 1872–1879.

[289] T. Matsuura, N. Tanimura, K. Hosoda, T. Yomo, and Y. Shimizu, "Reaction Dynamics Analysis of a Reconstituted Escherichia Coli Protein Translation System by Computational Modeling," *Proceedings of the National Academy of Sciences*, vol. 114, no. 8, E1336–E1344, 2017.

[290] T. Matsuura, K. Hosoda, and Y. Shimizu, "Robustness of a Reconstituted Escherichia Coli Protein Translation System Analyzed by Computational Modeling," *ACS Synthetic Biology*, vol. 7, no. 8, pp. 1964–1972, 2018.

[291] M. Weitz and F. C. Simmel, "Synthetic In Vitro Transcription Circuits," *Transcription*, vol. 3, no. 2, pp. 87–91, 2012.

[292] S. W. Schaffter and R. Schulman, "Building in Vitro Transcriptional Regulatory Networks by Successively Tntegrating Multiple Functional Circuit Modules," *Nature chemistry*, vol. 11, no. 9, pp. 829–838, 2019.

[293] A. Baccouche, K. Montagne, A. Padirac, T. Fujii, and Y. Rondelez, "Dynamic DNA-toolbox Reaction Circuits: A Walkthrough," *Methods*, vol. 67, no. 2, pp. 234–249, 2014.

[294] J. R. Karr, J. C. Sanghvi, D. N. Macklin, M. V. Gutschow, J. M. Jacobs, B. Bolival Jr, N. Assad-Garcia, J. I. Glass, and M. W. Covert, "A Whole-cell Computational Model Predicts Phenotype from Genotype," *Cell*, vol. 150, no. 2, pp. 389–401, 2012.

[295] D. N. Macklin, T. A. Ahn-Horst, H. Choi, N. A. Ruggero, J. Carrera, J. C. Mason, G. Sun, E. Agmon, M. M. DeFelice, I. Maayan, *et al.*, "Simultaneous Cross-evaluation of Heterogeneous E. Coli Datasets via Mechanistic Simulation," *Science*, vol. 369, no. 6502, 2020.

[296] J. Shin and V. Noireaux, "Efficient Cell-free Expression with the Endogenous E. Coli RNA Polymerase and Sigma Factor 70," *Journal of Biological Engineering*, vol. 4, no. 1, pp. 1–9, 2010.

[297] R. P. Maharjan and T. Ferenci, "Global Metabolite Analysis: the Influence of Extraction Methodology on Metabolome Profiles of Escherichia Coli," *Analytical Biochemistry*, vol. 313, no. 1, pp. 145–154, 2003.

[298] A. Vargha and H. D. Delaney, "The Kruskal-Wallis Test and Stochastic Homogeneity," *Journal of Educational and Behavioral Statistics*, vol. 23, no. 2, pp. 170–192, 1998.

[299] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman, "Emcee: the MCMC Hammer," *Publications of the Astronomical Society of the Pacific*, vol. 125, no. 925, p. 306, 2013.

[300] M. Kaiser, F. Jug, T. Julou, S. Deshpande, T. Pfohl, O. K. Silander, G. Myers, and E. Van Nimwegen, "Monitoring Single-cell Gene Regulation Under Dynamically Controllable Conditions with Integrated Microfluidics and Software," *Nature Communications*, vol. 9, no. 1, pp. 1–16, 2018.