UNIVERSITY OF CALIFORNIA SAN DIEGO

Modeling Reaction-Diffusion Systems with Dynamic Boltzmann Distributions

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Physics

by

Oliver K Ernst

Committee in charge:

Professor Terrence Sejnowski, Chair
Professor Henry Abarbanel, Co-Chair
Professor Michael Holst
Professor Eric Mjolsness
Professor Jeremie Palacci
Professor Gabriel Silva

2021

The dissertation of Oliver K Ernst is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

EPIGRAPH

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

First of all, I would like to thank my wife Sunny, without whom none of this would have been possible alone, and with whom I am discovering all the most worthwhile things in life.

My deepest thanks are to my advisors, Professors Terrence Sejnowski and Eric Mjolsness. Both provided me with seemingly endless support and guidance for this research, and even more than that, endless patience for countless meetings. I joined the lab almost six years ago because of the spark that Tom Bartol put in me when I first met him at tea and he showed me the astonishing models of synapses that he had been developing. It was a moment where I felt that something from a science fiction book was coming alive, not only as art but also as science. Tom later introduced me to Eric, who suggested to me machine learning methods which I felt brought even more science fiction to life, and which eventually grew into this thesis. I think all four of us felt that great new synapses could form from this small collaboration, and I am grateful that they lent me their time and energy, the results of which are documented here.

I also want to thank the CNL lab for helping to get me to do something other than science, at the very least for an hour every day at 3:30, and for the time spent listening to my often less than polished lab talks.

Finally I want to thank all the members on my committee for putting up with my also less than polished committee meeting presentations, which I hope have improved slightly over time through their guidance.

Chapter 4 is a reprint of material, with minor edits as it appears in: O.K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, "Learning dynamic Boltzmann distributions as reduced models of spatial chemical kinetics", *The Journal of Chemical Physics*, vol. 149, no. 3, pp. 034107, 2018.

Chapters 5 and 6 are a reprint of material, with minor edits as it appears in:

O.K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, "Learning moment closure in reaction-diffusion systems with spatial dynamic Boltzmann distributions", *Phys. Rev. E*, vol. 99, no. 6, pp. 063315, Jun. 2019.

Chapters 6 also contains a reprint of material, with minor edits as it appears in: O.K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, "Deep Learning Moment Closure Approximations using Dynamic Boltzmann Distributions", *arXiv*:1905.12122, 2019.

The author of the dissertation was the primary author of these papers.

VITA

2021	Ph. D. in Physics, University of California San Diego
2014	B. S. in Physics, Case Western Reserve University

PUBLICATIONS

O. K. Ernst, T. Bartol, T. Sejnowski and E. Mjolsness, "Deep learning moment closure approximations using dynamic Boltzmann distributions," *arXiv* 1905.12122, Sep. 2019.

O. K. Ernst, T. Bartol, T. Sejnowski and E. Mjolsness, "Learning moment closure in reaction-diffusion systems with spatial dynamic Boltzmann distributions," *Phys. Rev. E* **99**, 063315, June 2019.

O. K. Ernst, T. Bartol, T. Sejnowski and E. Mjolsness, "Learning Dynamic Boltzmann Distributions as Reduced Models of Spatial Chemical Kinetics," *J. Chem. Phys.* **149**, 034107, July 2018.

ABSTRACT OF THE DISSERTATION

Modeling Reaction-Diffusion Systems with Dynamic Boltzmann Distributions

by

Oliver K Ernst

Doctor of Philosophy in Physics

University of California San Diego, 2021

Professor Terrence Sejnowski, Chair
Professor Henry Abarbanel, Co-Chair

Computational models are an essential tool to understand biological systems. A common challenge in this field is to find reduced models that offer a simpler effective description of a system with increased computational efficiency. Recent revived interest in applications of machine learning has produced algorithms that are naturally suited for this task. This thesis introduces dynamic Boltzmann distributions (DBDs) for model reduction of chemical reaction networks. DBDs are an unsupervised learning method, framed in the language of probabilistic graphical models. This allows a close connection to be made between DBDs and the description of chemical reaction networks by master equations. In

this framework, this thesis shows how the physics of the system can be incorporated into otherwise application-agnostic machine learning algorithms. DBDs and their accompanying physics-informed machine learning algorithms provide a new path forward to apply reduced modeling methods to study reaction pathways at scale in synaptic neuroscience and other applications in biology.

Chapter 1

Introduction

Multiscale modeling in biology has become a forefront topic. In neuroscience, advances in imaging have created new datasets, from whole brain imaging data from light sheet microscopy down to reconstructions of synapses from electron microscopy (EM) data. These advances have been accompanied by a demand for capable multiscale modeling frameworks that can bridge the scales, connecting chemical signaling pathways at synapses up to spiking activity in whole neurons and networks.

This hierarchy is driven by biochemistry at the synapse occurring at microsecond and nanometer scales. EM has been applied to study synapses for more than half a century [1], and serial EM has been used to reveal the structure of dendritic spines and synapses for almost as long [2]. Serial EM datasets can be used to generate 3D reconstructions of neuropil, including axons, dendrites, astrocytes and mitochondria [3]. In addition to what the geometry can reveal about these systems [4], these reconstructions can be used to simulate the spatial chemical reaction networks inside synapses that underlie learning and memory.

The reaction-diffusion software MCell [5, 6] is specially designed for this task, treating each molecule in a grid-free random walk. Across a $6 \times 6 \times 5 \mu$m cube of recon-

structed rat hippocampal neuropil, this software was used to account for all major sinks & sources of signaling molecules and interactions with ion channels. This includes AMPA- and NMDA-type glutamate receptors, L- and R-type voltage-dependent Ca^{2+} channels, Na^{+}/Ca^{2+} exchangers, plasma membrane Ca^{2+} ATPases, smooth endoplasmic reticulum Ca^{2+} ATPases, immobile Ca^{2+} buffers, and calbindin [3]. Reconstitution experiments such as this are a leap forward in understanding synaptic neuroscience, as they integrate signaling pathways which were discovered independently of one another in experiments. No current imaging technology can simultaneously capture how this symphony of signaling molecules and pathways works together - for this problem, the simulation *is* the experiment.

These models of the biochemistry at synapses generate tantalizing questions. For example, the changing shape and size of synapses is thought to be the essential process of learning and memory in the brain [7, 4]. How do the signaling pathways at the μs scale interact with actin filament dynamics over minutes to regulate synaptic morphodynamics? Across spines on different dendrites, how do the biochemical pathways and spike trains work together to explain the observed [4] correlations in spine head size? These question requires multiscale models of synaptic biochemistry, but despite tools such as MCell, this remains a challenge.

Particle-based methods such as MCell are optimally suited to describe the low and spatially heterogeneous concentrations of molecules in dendritic spines over the timescales of μs to seconds. They allow predictions to be made based on changes to the reaction pathways involved, and can even reproduce the fluorescent dyes used in in vivo experiments. However, this approach is too computationally demanding over long timescales, such as those of morphological changes to the spine that occur during learning on the order of minutes to hours. Furthermore, it is also too demanding to be scaled in space beyond a handful of synapses, since each tracked particle requires ray tracing and the evaluation of a potentially large reaction network. Finally, while it is not the focus of this thesis,

incorporating physics other than the physics of reaction diffusion systems into multi-physics models is a further challenge. An example of a multi-physics problem is to model the electrodiffusion of ions in 3D in the electric field generated by an action potential, which is commonly modeled by a cable equation in one spatial dimension [8, 9].

Alternatively, differential equations for the observables can be derived from the chemical master equation (CME) describing the system. Ordinary differential equations typically arise when spatial organization is not considered important. Alternatively, partial differential equations require a carefully constructed mesh that is refined near ion channels and other confined environments, balanced with larger grid sizes in the dendritic shaft.

A further pervasive problem in the differential equation setting is moment closure: for most reaction networks, the differential equations for the moments derived from the CME do not close. Instead, differential equations for low order moments depend on higher order observables, leading to an infinite hierarchy of coupled differential equations [10]. To solve this system, the hierarchy must be closed at some level by approximating high order moments by lower order ones. However, the approximation becomes inaccurate as higher order terms become relevant to the dynamics. Can it be determined from stochastic simulations which higher order moments are relevant to the dynamics at any time, and to find the corresponding optimal moment closure approximation?

At the same time as the development of these realistic models of synaptic biochemistry, a revival has been underway in *machine learning* algorithms and applications. Since the field of machine learning is both broad and growing, in this thesis the term is specifically used to denote the subset of methods concerned with *statistical inference*: given a set of data, these methods seek to infer a useful effective probability distribution describing the data. Here, the meaning of “useful" is task dependent. For example, for an image, the goal may be that the probability distribution yields a latent representation that performs better in a classification task than the original image. While classification

tasks such as this commonly operate on human labeled datasets, statistical inference methods operate on *unlabeled* training data. Applying statistical inference methods to model reduction, the objective is to use examples of the true system (the fine scale) to find a coarse scale model which reduces computational cost. A closely related machine learning task is dimensionality reduction, popularized by principal component analysis (PCA) [11], variational autoencoders (VAEs) [12], and Boltzmann machines [13].

This thesis develops dynamic Boltzmann distributions (DBDs) and their learning algorithms as a new class of model reduction methods. Modeling with DBDs comprises two parts: (1) a graphical model defining a reduced model probability distribution, and (2) a differential equation model describing the time evolution of the interactions in the graphical model.

DBDs learn a parameterized differential equation system describing a probability distribution from data. For model reduction of reaction networks, this approach straddles the divide between stochastic simulations and solving the moment differential equation systems. The stochastic simulations, for example as performed using MCell for spatial systems, represent the fine scale ground truth, as they use the exact geometries, are grid free, naturally treat the stochasticity of individual ion channels, and do not require a closure approximation. From spatial stochastic simulations, DBDs allow a spatial reduced model (a PDE system) or a non-spatial reduced model (an ODE system) to be learned. From non-spatial stochastic simulations, for example using the Gillespie algorithm [14], a non-spatial reduced model can be learned. The resulting reduced model can be used for simulations over longer timescales. Alternatively, it can be used in multigrid methods such as V-cycle or W-cycle methods [15] that iteratively switch between stochastic simulations and differential equations by model reduction (learning) or model prolongation (sampling) as appropriate.

The latent variables in the reduced model probability distribution are trained to

represent the moments relevant to the describe dynamics. DBDs learn the distribution over latent variables from data, as opposed to other dimensionality reduction approaches such as VAEs that assume a fixed and known distribution. Throughout this thesis, the latent variables are also referred to as latent species or latent particles in the context of chemical reaction networks. The interpretation of the interaction of these latent species through an energy function is one of the key advantages of DBDs. Further, DBDs can be modified to describe well-mixed systems, systems on a spatial grid, and systems in continuous space (grid free).

The choice of the differential equation model is naturally of key importance and a particular focus of this work. Fundamentally, two options are possible: (1) a generic approach using basis functions, and (2) a parameterization based on the physics of the problem under consideration. Both options are explored in this thesis. The latter option is of particular interest for scientific applications, is it incorporates domain specific knowledge into machine learning. *Physics-informed machine learning* methods such as this can be more interpretable than otherwise "blackbox" machine learning methods. Further, the use of prior knowledge suggests that they can be lower dimensional and train on fewer examples than a generic approach. Most importantly, incorporating the physics of the problem improves the generalization of machine learning methods.

Chapter 2 of this thesis reviews the description of reaction networks by the chemical master equation (CME) in both well-mixed and spatial settings. Chapter 3 reviews relevant background on graphical models, statistical inference and Boltzmann machines [13]. Chapter 4 introduces dynamic Boltzmann distributions and explores systems which are exactly solvable in this framework, largely taken from Ref. [16] with minor edits. Chapter 5 introduces the differential equation constrained learning problem for DBDs, largely taken from Ref. [17] with minor edits. Some example problems are studied, in particular physics-based Gaussian graphical models. Chapter 6 introduces lattice models for reaction diffusion

systems, and studies DBDs for restricted Boltzmann machines, largely taken from Ref. [17] with minor edits. Further, the extension to deep dynamic Boltzmann distributions with many layers of latent species is developed, largely taken from Ref. [18] with minor edits. Chapter 7 returns to biology, applying DBDs to model calcium oscillations in non-excitable cells. These inositol trisphosphate (IP_3)-induced oscillations show unique characteristics captured only by stochastic models, and demonstrate the advantage of introducing domain-specific physics knowledge into machine learning problems.

Chapter 2

Master equations and moment closure

Reaction networks are described by a chemical master equation (CME). This includes spatial reaction networks, for which a powerful perturbative approach follows by applying the techniques from quantum field theory. In this chapter, the CME and its solution is reviewed. The challenges for solving the CME and the implications to modeling in biology are laid out, including the moment closure problem, and alternative stochastic simulation methods. Finally, the motivation for introducing machine learning approaches is made clear.

2.1 Chemical Master Equation

Consider a system described by $\boldsymbol{n} = \{n_{\mathcal{R}_1}, \ldots, n_{\mathcal{R}_M}\}$ particles of M different species $\mathcal{R}$, where each species count $n_{\mathcal{R}_i} = 0, 1, \ldots$ is non-negative. Let the probability of being in state $\boldsymbol{n}$ at time t be given by the distribution $p(\boldsymbol{n}, t)$. Additionally, introduce a set of reactions indexed by $r = 1, \ldots, R$. The probability density then evolves in time according

to the CME:

$$\frac{dp(\boldsymbol{n},t)}{dt} = \sum_{r=1}^{R} \sum_{\boldsymbol{n}'} \left[T_r(\boldsymbol{n}|\boldsymbol{n}')p(\boldsymbol{n}',t) - T_r(\boldsymbol{n}'|\boldsymbol{n})p(\boldsymbol{n},t) \right], \tag{2.1}$$

where $T_r(\boldsymbol{n}|\boldsymbol{n}')$ is the transition rate from state $\boldsymbol{n}'$ into state $\boldsymbol{n}$ under the reaction indexed by r. The sum over $\boldsymbol{n}'$ in this chapter is used to denote a sum over all possible counts for each species, keeping each count non-negative. The master equation has a number of interesting properties. It captures the idea of the Markov property for discrete variables, but in continuous time. It is linear in reactions, which will become important in the model reduction formalism introduced later.

Unfortunately, while it is easy to write down, there are few problems for which the CME can be solved analytically. Perturbative methods extend this set of problems, but most applications in biology are not practically accessible this way. Further, the time evolution of some quantity of interest $X(\boldsymbol{n})$ can be readily calculated by summing over states as:

$$\frac{d\langle X(\boldsymbol{n})\rangle(t)}{dt} = \sum_{\boldsymbol{n}} X(\boldsymbol{n}) \frac{dp(\boldsymbol{n},t)}{dt}. \tag{2.2}$$

However, unless the transitions involve only unimolecular reactions, this equation does not close, meaning that observables of higher order than $X(\boldsymbol{n})$ appear on the right hand side. This leads to an infinite hierarchy of moments that can only be solved approximately.

2.1.1 Guiding examples

Consider a simple birth-death process involving $M = 1$ species A:

$$\begin{aligned} X &\underset{k_b}{\rightharpoonup} A, \\ A &\underset{k_d}{\rightharpoonup} \varnothing \end{aligned} \quad . \tag{2.3}$$

Here, birth occurs from a reservoir X, which is assumed to be a fixed number of particles $\overline{n}_X$ that never decreases. Without considering the spatial distribution of particles, the state of the system is described by just the number of particles $\boldsymbol{n} = \{n_A\}$.

The right hand side of the CME (2.1) has contributions from each of the reactions. For example, the unimolecular death reaction contributes:

$$k_d\left((n_A+1)p(n_A+1,t) - n_A p(n_A,t)\right), \tag{2.4}$$

because any if any of the n_A+1 particles in the state $\{n_A+1\}$ decay, the resulting state will be $\{n_A\}$, giving the probability flow into this state. Similarly, the possible decay of any of the n_A particles gives the probability flow out of this state. The full CME for this reaction network is:

$$\begin{aligned}\frac{d}{dt}p(n_A,t) =& k_b\left(\overline{n}_X p(n_A-1,t) - \overline{n}_X p(n_A,t)\right) \\ &+ k_d\left((n_A+1)p(n_A+1,t) - n_A p(n_A,t)\right)\end{aligned}. \tag{2.5}$$

This is one of the few chemical master equations for which an analytic solution exists for certain initial distributions. One way to find such solutions is by using generating functions. In this one dimensional case, introduce a generating function of the form:

$$g(z,t) = \sum_{n_A=0}^{\infty} p(n_A,t) z^{n_A}. \tag{2.6}$$

This has the usual convenient property of a generating function that observables can be obtained by differentiating and setting $z=1$. Further, by reindexing:

$$\begin{aligned}\sum_{n_A=0}^{\infty} n_A p(n_A,t) z^{n_A} &= z\frac{\partial g(z,t)}{\partial z}, \\ \sum_{n_A=0}^{\infty} (n_A-1) p(n_A-1,t) z^{n_A} &= z^2\frac{\partial g(z,t)}{\partial z}.\end{aligned} \tag{2.7}$$

Using (2.5), the generating function therefore evolves according to:

$$\frac{\partial g(z,t)}{\partial t} = (z-1)\left(k_b \overline{n}_X g(z,t) - k_d \frac{\partial g(z,t)}{\partial z}\right). \tag{2.8}$$

This PDE can be solved using the method of characteristics. For the specific case where initially there are no particles $p(n_A, t=0) = \delta_{n_A,1}$, the solution is:

$$\begin{aligned} g(z,t) &= \sum_{n_A=0}^{\infty} e^{-\lambda(t)} \frac{\lambda(t)^{n_A}}{n_A!}, \\ \lambda(t) &= \frac{\overline{n}_X k_b}{k_d}(1 - e^{-k_d t}), \end{aligned} \tag{2.9}$$

from which the probability distribution is identified as Poisson:

$$p(n_A, t) = e^{-\lambda(t)} \frac{\lambda(t)^{n_A}}{n_A!}. \tag{2.10}$$

Next, consider the Lotka-Volterra system, described by the reactions:

$$\begin{aligned} H &\underset{k_d}{\rightharpoonup} \varnothing, \\ P &\underset{k_b}{\rightharpoonup} 2P, \\ H + P &\underset{k_e}{\rightharpoonup} 2H. \end{aligned} \tag{2.11}$$

This is a reaction network involving $M = 2$ species, with $\mathcal{R} = \{H, P\}$ denoting the hunter and prey, and system state $\boldsymbol{n} = \{n_H, n_P\}$. The CME is:

$$\begin{aligned} \frac{d}{dt}p(n_H, n_P, t) =& k_b \left((n_P - 1)p(n_H, n_P - 1, t) - n_P p(n_H, n_P, t)\right) \\ &+ k_d \left((n_H + 1)p(n_H + 1, n_P, t) - n_H p(n_H, n_P, t)\right) \\ &+ k_e \left((n_H + 1)(n_P + 1)p(n_H + 1, n_P + 1, t) - n_H n_P p(n_H, n_P, t)\right). \end{aligned} \tag{2.12}$$

Despite only containing a single bimolecular reaction more than the birth-death process, no analytic solution to this model exists. As an indicator of how generally challenging analytic solutions are to find, consider that the solution to the general reaction scheme $A + B \leftrightharpoons C$ was worked out only relatively recently in 2000 [19].

2.1.2 Perturbative approach

A very powerful approach to solve the CME was developed by Masao Doi [20, 21] (and independently by Luca Peliti [22]) by applying the techniques of quantum field theory. The Doi-Peliti formalism can also describe systems of particles diffusion in continuous 3D space. The application of these methods have been used in applications such as modeling actin filament growth [23], as well as to derive stochastic simulation algorithms from first principles [24].

The crucial concept is the introduction of a *probability state vector* that acts as a generating function. Let the state of the system consist of $|\boldsymbol{n}\rangle$ particles as previously, now using the common bra-ket notation, then the probability state vector is defined as:

$$|\Psi(t)\rangle = \sum_{\boldsymbol{n}} p(\boldsymbol{n}, t)\, |\boldsymbol{n}\rangle\,. \tag{2.13}$$

Further, associate with every state $|\boldsymbol{n}\rangle$ the $\boldsymbol{n}$-th state of the M dimensional harmonic oscillator. The raising operator $a^\dagger_{\mathcal{R}_i}$ creates a particle of species $\mathcal{R}_i$, and the lowering operator $a_{\mathcal{R}_i}$ annihilates it:

$$\begin{aligned} a^\dagger_{\mathcal{R}_i}\, |\boldsymbol{n}\rangle &= |\boldsymbol{n} + \boldsymbol{e}_i\rangle\,, \\ a_{\mathcal{R}_i}\, |\boldsymbol{n}\rangle &= n_{\mathcal{R}_i}\, |\boldsymbol{n} - \boldsymbol{e}_i\rangle\,, \end{aligned} \tag{2.14}$$

where $\boldsymbol{e}_i$ is the i-th unit vector of length M. The usual commutation relation $[a_{\mathcal{R}_i}, a^\dagger_{\mathcal{R}_j}] = \delta_{i,j}$ holds. Additionally, the special vacuum state $|\boldsymbol{0}\rangle$ obeys $a_{\mathcal{R}_i}\, |\boldsymbol{0}\rangle = \boldsymbol{0}$.

Similar to a generating function, the probability state vector $|\Psi(t)\rangle$ can be used to yield the moments of p. However, the normalization of this state requires a reference state $\langle\Phi|$ which is a special case of the Glauber state, specifically the eigenvector of $a^\dagger_{\mathcal{R}_i}$ with eigenvalue 1:

$$\langle\Phi| = \langle\mathbf{0}|\exp\left[\sum_{i=1}^{M} a_{\mathcal{R}_i}\right] = \sum_{\boldsymbol{n}} \langle\boldsymbol{n}|\,. \tag{2.15}$$

The reference state shows probability conservation $\langle\Phi|\Psi(t)\rangle = 1$. Further, any observable $\langle X\rangle(t)$ can now be obtained as:

$$\langle X\rangle(t) = \langle\Phi|\,X(\boldsymbol{a},\boldsymbol{a}^\dagger)\,|\Psi(t)\rangle\,, \tag{2.16}$$

where vector notation denotes $\boldsymbol{a} = \{a_{\mathcal{R}_i} \forall i = 1,\ldots,M\}$, and similarly for $\boldsymbol{a}^\dagger$.

The state vector satisfies a differential equation similar analogous to the generating function:

$$\frac{\partial}{\partial t}\,|\Psi(t)\rangle = \sum_{r=1}^{R} W_r(\boldsymbol{a},\boldsymbol{a}^\dagger)\,|\Psi(t)\rangle\,, \tag{2.17}$$

where W_r is the quantum Hamiltonian corresponding to the reaction indexed by r. For example, for the hunter-prey reaction (2.11), the operator is:

$$W_e = k_e\left(\left(a^\dagger_H\right)^2 - a^\dagger_H a^\dagger_P\right) a_H a_P. \tag{2.18}$$

Equation (2.17) has formal solution:

$$|\Psi(t)\rangle = \exp\left[\sum_{r=1}^{R} W_r\right]|\Psi(t=0)\rangle\,. \tag{2.19}$$

Still, this solution is only possible exactly for few reaction operators because the majority of reaction operators do not commute. For a general solution, the operator $W = \sum_{r=1}^{R} W_r$ is split as $W = W_0 + W_1$ into a diagonal part W_0 which is at most quadratic in the raising

and lowering operators, and a perturbative part W_1 which contains higher order terms. The time ordered product expansion gives the solution to (2.17) as:

$$\begin{aligned} |\Psi(t)\rangle &= e^{tW_0} T(t) \, |\Psi(t=0)\rangle \,, \\ T(t) &= 1 + \int_{t'=0}^{t} dt' W_T(t') + \int_{t'=0}^{t} dt' W_T(t') \int_{t''=0}^{t'} dt'' W_T(t'') + \ldots, \\ W_T(t) &= e^{-tW_0} W_1 e^{tW_0}. \end{aligned} \tag{2.20}$$

From this follows the standard procedure of expanding the exponentials, and combining terms that are of equal order in the number $a, a^\dagger$ (other terms do not contribute). Each such vacuum expectation value (VeV) is by Wick's theorem equivalent to a sum over all non-vanishing permutations of pairs of operators in normal ordering, where each term in the sum can be represented diagrammatically. Ultimately, since the typical goal is to calculate some observable of the distribution rather than the full distribution, then the quantity to be evaluated is the log of the partition function from which the observables are attainable. For this, only connected diagrams contribute, simplifying the number of diagrams that must be summed up to some order in the perturbation.

2.1.3 Generating functions and operators

A convenient mapping exists between the generating function notation introduced earlier and the operator algebra of the Doi-Peliti formalism that allows an easy to calculate the time evolution of the observables. By letting:

$$|\boldsymbol{n}\rangle \to \prod_{i=1}^{M} z_{\mathcal{R}_i}^{n_{\mathcal{R}_i}}, \tag{2.21}$$

the probability state vector becomes the moment generating function:

$$|\Psi(t)\rangle \to g(\boldsymbol{z},t) = \sum_{\boldsymbol{n}} p(\boldsymbol{n},t) \prod_{i=1}^{M} z_{\mathcal{R}_i}^{n_{\mathcal{R}_i}}, \tag{2.22}$$

where $\boldsymbol{z} = \{z_{\mathcal{R}_i} \forall i = 1, \ldots, M\}$. Further, the ladder operators become:

$$\begin{aligned} a_{\mathcal{R}_i}^{\dagger} &\to z_{\mathcal{R}_i}, \\ a_{\mathcal{R}_i} &\to \frac{\partial}{\partial z_{\mathcal{R}_i}}. \end{aligned} \tag{2.23}$$

An observable $\langle X \rangle(t)$ evolves according to:

$$\frac{d\langle X \rangle}{dt} = \langle \Phi | \, X(\boldsymbol{a}, \boldsymbol{a}^{\dagger}) W(\boldsymbol{a}, \boldsymbol{a}^{\dagger}) \, |\Psi(t)\rangle \to X_{\boldsymbol{z}} W_{\boldsymbol{z}} g(\boldsymbol{z},t) \Big|_{z=1}, \tag{2.24}$$

where $X_{\boldsymbol{z}}, W_{\boldsymbol{z}}$ are transformed by (2.23). For example, for the birth reaction in (2.11):

$$W(\boldsymbol{a}, \boldsymbol{a}^{\dagger}) = k_b (a_P^{\dagger} - 1) a_P^{\dagger} a_P \to W_{\boldsymbol{z}} = k_b (z_P - 1) z_P \frac{\partial}{\partial z_P} \tag{2.25}$$

and taking X as the operator for the number of particles:

$$X(\boldsymbol{a}, \boldsymbol{a}^{\dagger}) = a_p^{\dagger} a_P \to X_{\boldsymbol{z}} = z_P \frac{\partial}{\partial z_P} \tag{2.26}$$

gives the familiar result for exponential growth:

$$\frac{d\langle n_P \rangle}{dt} = k_b \langle n_P \rangle. \tag{2.27}$$

2.2 Moment closure

Consider again the predator-prey reaction:

$$H + P \underset{k_e}{\rightharpoonup} 2H, \tag{2.28}$$

for which the operator is:

$$W_e = k_e(a_H^\dagger - a_P^\dagger)a_H^\dagger a_H a_P. \tag{2.29}$$

The time evolution of the means can be easily calculated from (2.24) as:

$$\begin{aligned} \frac{d\langle n_P\rangle}{dt} &= -k_e\langle n_H n_P\rangle, \\ \frac{d\langle n_H\rangle}{dt} &= k_e\langle n_H n_P\rangle. \end{aligned} \tag{2.30}$$

This depends on the next highest order term $\langle n_H n_P\rangle$. Similarly, this observable evolves according to:

$$\frac{d\langle n_H n_P\rangle}{dt} = -k_e\langle n_H^2 n_P\rangle + k_e\langle n_H n_P^2\rangle - k_e\langle n_H n_P\rangle. \tag{2.31}$$

This continuous infinitely, where the time evolution of any observable depends on higher order terms appearing on the right hand side.

Equations (2.30,2.31) are an example of an infinite hierarchy of moment equations. In order to solve this hierarchy, an approximation must be introduced that closes this infinite hierarchy at some order. This *moment closure approximation* turns the system into a closed system of ordinary differential equations.

Several popular approaches exist for the moment closure approximation. The mean field approach $\langle n_H n_P\rangle \approx \langle n_H\rangle\langle n_P\rangle$ greatly simplifies the moment equations, but the approximation is generally inaccurate as it closes the hierarchy at a low level. One way to close the hierarchy at a higher level is the Gaussian moment closure approximation. For

an M dimensional multivariate Gaussian distribution defined by the mean vector $\boldsymbol{\mu}$ and covariance matrix Σ, all higher order observables can be expressed in terms of Σ and $\boldsymbol{\mu}$. For example:

$$\langle n_{\mathcal{R}_i} n_{\mathcal{R}_j} n_{\mathcal{R}_k} \rangle = \mu_i \mu_j \mu_k + \mu_i \Sigma_{j,k} + \mu_j \Sigma_{i,k} + \mu_k \Sigma_{i,j}, \tag{2.32}$$

for any species indexes i, j, k in $1, \ldots, M$. This approximation is more accurate than mean-field, since it closes the hierarchy at one order higher.

The Gaussian moment closure scheme is an example of a more general approach to moment closure based on maximizing entropy (MaxEnt). This has been introduced several times in the literature [25, 16], and is sometimes called zero-information moment closure. The information entropy at an instant in time for the random variables $\boldsymbol{n}$ is:

$$H(t) = -\sum_{\boldsymbol{n}} p(\boldsymbol{n}, t) \ln p(\boldsymbol{n}, t) \tag{2.33}$$

Define some set of observables at this point in time $\{\langle \mathcal{O}_i \rangle(t)\}$ for $i = 1, \ldots, N_{\mathcal{O}}$ that should be tracked explicitly, i.e. the entropy will be maximized under these constraints for the moments. Introducing a Lagrange multiplier ν_i for every constraint, then the unconstrained optimization problem is to maximize at a specific instant in time:

$$S(t) = H(t) - \sum_{i=1}^{N_{\mathcal{O}}} \nu_i(t) \left(\langle \mathcal{O}_i \rangle(t) - \sum_{\boldsymbol{n}} \mathcal{O}_i p(\boldsymbol{n}, t) \right). \tag{2.34}$$

The solution to this MaxEnt problem is:

$$\begin{aligned} \tilde{p}(\boldsymbol{n}, t) &= \frac{1}{Z(t)} \exp\left[-\sum_{i=1}^{N_{\mathcal{O}}} \nu_i(t) \mathcal{O}_i \right], \\ Z(t) &= \sum_{\boldsymbol{n}} \exp\left[-\sum_{i=1}^{N_{\mathcal{O}}} \nu_i(t) \mathcal{O}_i \right]. \end{aligned} \tag{2.35}$$

Here the observables can be obtained from the partition function in the usual way: $\langle \mathcal{O}_i \rangle_{\tilde{p}}(t) =$

$\partial \ln Z(t)/\partial \nu_i$. For the Gaussian moment closure example, the observables are:

$$\begin{aligned}
\langle \mathcal{O}_{1,\mathcal{R}_i} \rangle(t) &= \langle n_{\mathcal{R}_i} \rangle(t), \\
\langle \mathcal{O}_{2,\mathcal{R}_i,\mathcal{R}_j} \rangle(t) &= \begin{cases} \langle n_{\mathcal{R}_i} n_{\mathcal{R}_i} \rangle(t) & \text{if} \quad i \neq j, \\ \left\langle \binom{n_{\mathcal{R}_i}}{2} \right\rangle(t) & \text{if} \quad i = j, \end{cases}
\end{aligned} \tag{2.36}$$

The resulting MaxEnt distribution has the form:

$$\tilde{p}(\boldsymbol{n}, t) = \frac{1}{Z(t)} \exp\left[-\sum_{i=1}^{M} \nu_{1,i} n_{\mathcal{R}_i} - \sum_{i=1}^{M} \nu_{2,i,i} \binom{n_{\mathcal{R}_i}}{2} - \sum_{i=1}^{M} \sum_{j>i} \nu_{2,i,j} n_{\mathcal{R}_i} n_{\mathcal{R}_j} \right]. \tag{2.37}$$

where the Lagrange multipliers obey $\nu_{2,i,j} = \nu_{2,j,i}$.

Equation (2.37) is an uncommon way to write a Gaussian distribution. However, this notation reveals an important connection to graphical models, where the Lagrange multipliers are the interactions in the graph. Figure 2.1 shows an example of such a graphical model. Every random variable $n_{\mathcal{R}_i}$ denotes a node in the graph, with edges representing interactions. The pairwise interactions $\nu_{2,\mathcal{R}_i,\mathcal{R}_j}$ are interactions between two different nodes, while $\nu_{2,\mathcal{R}_i,\mathcal{R}_i}$ are self interactions.

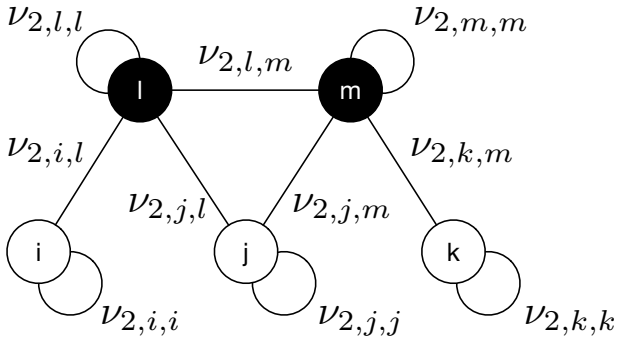

Figure 2.1: Example of a Gaussian graphical model (2.37). Open circles represent observed random variables $n_{\mathcal{R}_i}, n_{\mathcal{R}_j}, \ldots$, while filled circles denote latent random variables. Edges denote interactions. The introduction of latent variables will enable the moment closure approximation to be learned from simulations, as shown in later chapters.

Learning in graphical models is the problem of *statistical inference*, and at the heart of machine learning [26]. Given observables from experimental data or simulations of these reaction networks, the problem of determining the interactions in the graph is also sometimes called the *inverse Ising problem* in statistical physics.

Almost all machine learning models revolve around latent random variables - in this case, additional species that are not observed in the data, but introduced to explain the data. As shown in later chapters, the introduction of latent random variables allows the moment closure approximation to be learned from data.

The next section will show how stochastic simulations can be used to circumvent the moment closure problem. Averaging over stochastic simulations is a popular technique for modeling biological systems, but comes at the cost of significant computational overhead.

2.3 Stochastic simulations

As mentioned earlier, one of the most popular alternatives to solving the CME are stochastic simulations. A single trajectory of a stochastic simulation corresponds to a sample from the probability distribution function that is the solution of the master equation.

2.3.1 The Gillespie algorithm

The Gillespie algorithm [14] is one of the most popular numerical methods for simulation stochastic trajectories of a reaction network. The original formulation is for well-mixed system and does not treat a spatial distribution of particles, although it is also often applied to model systems in the reaction limited regime where diffusion is fast.

The fundamental premise of the Gillespie algorithm is that reactions occur at random, which is valid when the system is at thermal equilibrium. This leads to the notion

of a reaction probability density function:

$$q(\tau, r; t, \boldsymbol{n}) = a(r; t, \boldsymbol{n}) \exp\left[-\tau \sum_{r'=1}^{R} a(r'; t, \boldsymbol{n})\right]. \tag{2.38}$$

Given the current state of the system $\boldsymbol{n}$ at time t, then $q(\tau, r; t, \boldsymbol{n})d\tau$ is the probability that a reaction indexed by r will occur in infinitesimal the time interval $[t+\tau, t+\tau+d\tau]$. Here $\boldsymbol{n}$ is as before the number of particles of each species $\boldsymbol{n} = \{n_{\mathcal{R}_1}, \ldots, n_{\mathcal{R}_M}\}$.

The Gillespie algorithm follows from iteratively sampling from q. The time of the next reaction τ is exponentially distributed, and the reaction that occurs has probabilities given by $a(r; t, \boldsymbol{n}) / \sum_{r'=1}^{R} a(r'; t, \boldsymbol{n})$. Finally, to calculate the propensity of a reaction, consider a reaction indexed by r of the form:

$$\sum_{i=1}^{N^{(r)}} m_i^{(r)} \mathcal{R}_i \underset{\gamma^{(r)}}{\rightarrow} \ldots, \tag{2.39}$$

where $N^{(r)}$ is the number of reactants denoted by the set $\mathcal{R}$, $m_i^{(r)} \in \{1, 2, \ldots\}$ are the integer multiplicities of each reactant, and $\gamma^{(r)}$ is the stochastic reaction rate. The reaction propensity is given by:

$$a(r; t, \boldsymbol{n}) = \gamma^{(r)} \prod_{i=1}^{N^{(r)}} \left(n_{\mathcal{R}_i}\right)_{m_i^{(r)}}, \tag{2.40}$$

where $(x)_n$ denotes the falling factorial. Note that all possible combinations of particles gives a binomial coefficient, from which a factor $\prod_{i=1}^{N^{(r)}} m_i^{(r)}$ has been absorbed into the rate $\gamma^{(r)}$.

The Gillespie algorithm is a powerful and popular alternative to solving the master equation. Moments from the distribution are obtained by averaging over many stochastic simulations. A limitation of the algorithm and generally all such particle-based methods is that it scales poorly with the number of particles, as this causes the reaction propensities to increase, leading $\tau \to 0$ and an explosive number of reactions to be evaluated. Several

variants of the algorithm exist to address this, such as *tau-leaping* [27], which approximates the number of reactions that occur in a fixed time-step of size τ. The step size can be fixed or variable, with further algorithms guiding an efficient choice of τ [28]. Alternatively, *R-leaping* methods proceed in reaction space, sampling how a fixed total number of reactions are distributed across the different reaction channels at each step [29]. Finally, exact R-leaping (ER-leaping) is an accelerated method that works without approximations by sampling reactions from the same distribution as the original Gillespie algorithm [30].

2.3.2 Spatial stochastic simulations

Applying Gillespie methods to spatial systems is one of the most important modifications to make when modeling many real systems in biology. For example, in the dendritic spine head, the organization of ion channels in the membrane is highly precise. The influx of calcium through voltage gated calcium channels quickly leads to the activation of small conductance potassium channels (SK channels) because these channels are highly co-located. The original Gillespie algorithm for well-mixed systems is ill-suited to accurately model these spatial effects.

A simple modification of the Gillespie algorithm is a compartmental version, where space is divided into compartments according to some prescription. Inside each compartment, the Gillespie algorithm proceeds independently of the other compartments, modeling the compartment as well-mixed. Additionally, a hopping rate constant lets particles hop between neighboring compartments, approximating diffusion. Compartmental Gillespie approaches are simple to implement and highly suited for parallel implementations, but still are only coarse approximations to the true particle-particle interactions.

MCell [5, 6] is a highly successful open source modeling platform for spatial stochastic simulations. MCell tracks individual particles as they diffuse and interact, both on 2D surfaces and in 3D volumes, whose surfaces are defined by triangulated meshes. Importantly,

particles are not constrained to a grid or otherwise compartments, but are free to move to points in 2D or 3D with arbitrary precision. Ray tracing is used to accurately model diffusion in confined volumes, such as those of dendritic spines.

The main principles of MCell's core algorithm for spatial stochastic simulations are briefly reviewed next. The algorithm lets particles diffuse inside a volume in 3D or on a 2D surface. Each species has a unique diffusion constant, for which step lengths Δr are sampled from a probability distribution $p_{\text{step}}(\Delta r|\Delta t)$ for a set timestep Δt, along with a random direction in which to move. The event of particles encountering one another is called a collision, for which there exists a possibility for a reaction. Let the probability of a particular reaction indexed by r given that a collision occurred be $p_{\text{rxn}}(r|\text{coll}, \Delta t)$. These two probabilities are the main quantities of interest that will be derived from given reaction rates and diffusion constants.

Consider first just the diffusion process. From the master equation follows Fick's 2^{nd} law:

$$\frac{\partial C(\boldsymbol{x}, t)}{\partial t} = D\nabla^2 C(\boldsymbol{x}, t), \tag{2.41}$$

where $C(\boldsymbol{x}, t)$ is the concentration for a species at a point $\boldsymbol{x}$ at time t, and D the diffusion constant. If the concentration is assumed to be radially symmetric $C(\boldsymbol{x}, t) \approx C(r, t)$, then the solution for a point source of N particles is:

$$C(r, t) = \frac{N}{\lambda(t)^3 \pi^{3/2}} e^{-r^2/\lambda(t)^2}, \tag{2.42}$$

where $\lambda(t) = 2\sqrt{Dt}$. Therefore, the probability that a single particle is located in the spherical shell from radii $[\Delta r, \Delta r + dr]$ of infinitesimal volume $4\pi r^2 dr$ is:

$$p_{\text{step}}(\Delta r|\Delta t) = \frac{1}{\lambda(\Delta t)^3 \pi^{3/2}} e^{-(\Delta r)^2/\lambda(\Delta t)^2} \times 4\pi(\Delta r)^2 dr. \tag{2.43}$$

At every timestep, a displacement can be sampled from $p_{\text{step}}(\Delta r|\Delta t)$ for every particle, along with a random direction for the diffusion.

Next, the probability of a particular reaction per collision $p_{\text{rxn}}(r|\text{coll})$ will be derived. Consider a bimolecular reaction of the form:

$$A + B \underset{\gamma_r}{\rightarrow} C, \tag{2.44}$$

occurring when a particle of species A diffuses and encounters in space a particle of species B. Initially, let B be fixed at one point in space - later this restriction will be lifted. This general reaction is the most interesting to consider: unimolecular reactions occur independently of the spatial location of particles following the Gillespie algorithm, and higher order reactions are rare events that are often not modeled.

Assign to every particle an interaction radius r_{ixn}. From a single diffusion step of length Δr sampled from $p_{\text{step}}(\Delta r)$, the A particle traces out a 3D cylinder, with volume $V = \pi r_{\text{ixn}}^2 \Delta r$. The number of B particles in this cylinder is then $n_{B\in V} = c_A \times V \times [B]$, where c_A is Avogadro's constant. Integrating over all possible step lengths gives the total collision probability:

$$p_{\text{rxn}}(\text{coll}|\Delta t) = \int_{\Delta r=0}^{\infty} d\Delta r \, (4\pi(\Delta r)^2) \, p_{\text{step}}(\Delta r) \times n_{B\in V} = 2c_A\sqrt{\pi} r_{\text{int}}^2 [B] \times \lambda_A(\Delta t). \tag{2.45}$$

The total probability of the reaction r occuring then is:

$$p_{\text{rxn}}(r|\Delta t) = p(\text{coll}|\Delta t) \times p_{\text{rxn}}(r|\text{coll}, \Delta t) = 2c_A\sqrt{\pi} r_{\text{int}}^2 [B] \times p_{\text{rxn}}(r|\text{coll}) \times \lambda_A(\Delta t). \tag{2.46}$$

Mass action kinetics gives the binding rate for this reaction in a time interval Δt as:

$$p_{MA}(r|\Delta t) = k_r [B] \Delta t, \tag{2.47}$$

where k_r is the reaction rate (the relationship between k_r and γ_r is discussed is Section 2.3.3). In order for the algorithm to reproduce the true observables, $p_{\text{rxn}}(r|\Delta t)$ must equal $p_{MA}(r|\Delta t)$:

$$p_{\text{rxn}}(r|\text{coll}, \Delta t) = \frac{k_r}{2c_A\sqrt{\pi}r_{\text{int}}^2 v_A(\Delta t)}, \tag{2.48}$$

where $v_A(\Delta t) = \lambda(\Delta t)/\Delta t$ is the characteristic velocity of the particle over Δt. Finally, equation (2.48) can be easily generalized to the case where both particles are moving:

$$p_{\text{rxn}}(r|\text{coll}, \Delta t) = \frac{k_r}{2c_A\sqrt{\pi}r_{\text{int}}^2 \left(v_A(\Delta t) + v_B(\Delta t)\right)}. \tag{2.49}$$

Equations (2.43) and (2.49) together give the core of the particle based stochastic simulation algorithm MCell. Given the simulation step size Δt and diffusion constants for every species, equation (2.43) is used to diffuse particles in space. Ray tracing is used to detect collisions between particles as they sweep out some effective interaction volume. When a collision occurs, equation (2.48) is used to evaluate whether a reaction occurs with a given rate constant.

The choice for the interaction radius is a practical one. First note that by substituting (2.49) into (2.46), the probability of a reaction is seen to be independent of the interaction radius. Still, if the interaction radius is too small, too few collisions will occur. On paper this is rectified by the probability of a reaction per collision (2.49) surpassing unity, but since this is meaningless, the interaction radius must be chosen as larger. On the other hand, if it is very large, the particle collides with very many other molecules in a single timestep, but the probability of a reaction per collision tends to zero. Since this is numerically inefficient, the radius must be set smaller. In practice, a good choice for the radius is one for which all reactions have a probability of reaction per collision (2.49) as $\mathcal{O}(10^{-1})$, yet strictly less than unity.

This approach to spatial stochastic simulations has been validated [5, 6] by comparing

the simulated stochastic trajectories to the true solutions derived from the master equations. This includes well-mixed systems that are reaction-limited, i.e. where diffusion is fast, and spatial systems with simple boundary conditions that admit analytic solutions.

MCell has been used extensively in modeling synaptic neuroscience. Reconstitution experiments [3] using 3D reconstructions from serial EM microscopy study the many sources and sinks of signaling molecules such as calcium. These simulations integrate the many signaling pathways which were discovered independently of one another in experiments - for this problem, the simulations *are* the experiment.

2.3.3 Stochastic reaction constants

Up to this point, two reaction constants have appeared. One is the reaction rate, which is a coefficient in the master equation or in mass action kinetics. The other is a stochastic reaction constant, which appears in the formulation of a stochastic simulation algorithm such as the Gillespie algorithm. To clearly differentiate the two, let the constant appearing in the master equation be the *concentration-based* reaction rate, and the constant in the stochastic algorithm be the *molecular-based* reaction rate. What is the relationship between these two rates?

Consider again the notation of (2.39) for a reaction indexed by r of the form

$$\sum_{i=1}^{N^{(r)}} m_i^{(r)} \mathcal{R}_i \underset{\gamma^{(r)}}{\rightarrow} \ldots, \tag{2.50}$$

Here, $\gamma^{(r)}$ is the molecular-based reaction rate. Associated with this reaction is the stoichiometry vector $\boldsymbol{\nu}^{(r)}$ of length equal to the number species M, whose integer components $\nu_i^{(r)}$ describe the change in the number of particles of $\mathcal{R}_i$. The goal is to relate $\gamma^{(r)}$ to the

concentration-based reaction rate $k^{(r)}$ appearing in mass action kinetics:

$$\frac{d[\mathcal{R}_j]}{dt} = -\nu_j^{(r)} k^{(r)} \prod_{i=1}^{N^{(r)}} ([\mathcal{R}_i])^{m_i^{(r)}} + \ldots, \tag{2.51}$$

for $j \in \{1, 2, \ldots, N^{(r)}\}$, and where the $\ldots$ denote other possible reactions. Here, $k^{(r)}$ is the concentration-based reaction rate.

The propensity term for the reaction (2.50) in units of molecules per time is approximated at large particle numbers as:

$$a(r; t, \boldsymbol{n}) = \gamma^{(r)} \prod_{i=1}^{N^{(r)}} \left(n_{\mathcal{R}_i}\right)_{m_i^{(r)}} \approx \gamma^{(r)} \prod_{i=1}^{N^{(r)}} n_{\mathcal{R}_i}^{m_i^{(r)}}. \tag{2.52}$$

In the mass action equation (2.51), the reaction-based rate of change converted to units of molecules per time is (without the stoichiometry vector):

$$c_A V \times k^{(r)} \prod_{i=1}^{N^{(r)}} ([\mathcal{R}_i])^{m_i^{(r)}} = k^{(r)} \frac{\prod_{i=1}^{N^{(r)}} n_{\mathcal{R}_i}^{m_i^{(r)}}}{(c_A V)^{-1+\sum_{i=1}^{N^{(r)}} m_i^{(r)}}}, \tag{2.53}$$

where we have substituted the definition of the concentration $[\mathcal{R}_i] = n_{\mathcal{R}_i}/(c_A \times V)$ for $n_{\mathcal{R}_i}$ particles in volume V where c_A is Avogadros constant. Equating this with the approximation for the approximation for the propensity (2.52) gives the relation:

$$\gamma^{(r)} = k^{(r)} (c_A V)^{1-\sum_{i=1}^{N^{(r)}} m_i^{(r)}}. \tag{2.54}$$

Crucially, at low particle counts, the approximation for the propensity (2.52) is inaccurate, such that mass action kinetics and stochastic simulations do not agree.

2.4 Spatial formulations

So far, only well-mixed systems described by $\boldsymbol{n}$ particles of different species have been considered. The master equation and the Doi-Peliti formalism can similarly describe spatial systems. In this section, the notation will be introduced for two formulations that are used later in this thesis: (1) for discrete spatial systems where particles hop on a lattice, and (2) for continuous spatial systems where particles diffuse in 3D space.

2.4.1 Spatial system on a lattice

Consider a system where particles live on a lattice in the single-particle occupancy limit. The state of the system is that of an n-vector model from statistical physics, defined as follows. Assign a unique index i to each of the N sites in the lattice. Let the vector of possible species $\mathcal{R}$ be of size M in some arbitrary ordering, excluding $\varnothing$ to denote an empty site. Spins at site i are multinomial units, represented as a vector $\boldsymbol{s}_i$ of length M, where entries $s_{i,\alpha} \in \{0,1\}$ for $i = 1,\ldots,M$ denote the absence (0) or presence (1) of a particle of species $\mathcal{R}_\alpha$. The single-occupancy limit corresponds to the implicit constraint $c(i) \in \{0,1\}$ for all sites $i = 1,\ldots,N$ that the vectors are of unit length:

$$c(i) = \sum_{\alpha=1}^{M} s_{i,\alpha}. \tag{2.55}$$

The state of the system is given by the matrix $\boldsymbol{S}$ of size $N \times M$, where each row denotes a lattice site.

The dynamics of the system can be most easily described in the Doi-Peliti formalism. The form of the differential equation obeyed by the probability state vector (2.13) is unchanged:

$$\frac{d}{dt}\left|\Psi(t)\right\rangle = \sum_{r=1}^{R} W_r \left|\Psi(t)\right\rangle, \tag{2.56}$$

but the probability state vector itself now becomes:

$$|\Psi(t)\rangle = \sum_{\boldsymbol{S}} p(\boldsymbol{S},t)\,|\boldsymbol{S}\rangle\,, \tag{2.57}$$

where the sum $\sum_{\boldsymbol{S}}$ maintains the implicit constraint $c(i) \in \{0,1\}$ for $i = 1,\ldots,N$. The raising and lowering operators become:

$$\begin{aligned} a^{\dagger}_{i,\alpha}\,|\boldsymbol{S}\rangle &= (1-c(i))\,\Big|\boldsymbol{S}+I_{i,\alpha}\Big\rangle\,, \\ a_{i,\alpha}\,|\boldsymbol{S}\rangle &= s_{i,\alpha}\,\Big|\boldsymbol{S}-I_{i,\alpha}\Big\rangle\,, \end{aligned} \tag{2.58}$$

where I_{ij} is the $N \times M$ single-entry matrix with entries zero everywhere except at index (i,j) where it is one.

Diffusion occurs by particles hopping to neighboring lattice sites. Let the neighbors of lattice site i be the sites with indexes neigh(i), for $i = 1,\ldots,N$. Assuming all species hop with the same rate h, then diffusion operator is:

$$W_{\text{diff}} = h \sum_{\alpha=1}^{M} \sum_{i=1}^{N} \sum_{j \in \text{neigh}(i)} (a^{\dagger}_{j,\alpha} a_{i,\alpha} - a^{\dagger}_{i,\alpha} a_{i,\alpha}). \tag{2.59}$$

Unimolecular reactions can occur anywhere on the lattice, but bimolecular reactions can only occur between particles on neighboring lattice sites. For example, the operator for the predator-prey reaction $H + P \to 2H$ is

$$W_e = \kappa_e \sum_{\langle i,j \rangle} \Big((a^{\dagger}_{j,H} - a^{\dagger}_{j,P}) a^{\dagger}_{i,H} a_{j,P} a_{i,H} + (a^{\dagger}_{i,H} - a^{\dagger}_{i,P}) a^{\dagger}_{j,H} a_{i,P} a_{j,H} \Big)\,, \tag{2.60}$$

where $\sum_{\langle i,j \rangle}$ sums over unique neighbors sites $i,j = 1,\ldots,N$.

The equivalent generating function formalism now represents states and the proba-

bility state vector as:

$$
\begin{aligned}
|\boldsymbol{S}\rangle &\to \prod_{i=1}^{N}\prod_{\alpha=1}^{M} z_{i,\alpha}^{s_{i,\alpha}}, \\
|\Psi(t)\rangle &\to g(\boldsymbol{z},t) = \sum_{\boldsymbol{S}} p(\boldsymbol{S},t) \prod_{i=1}^{N}\prod_{\alpha=1}^{M} z_{i,\alpha}^{s_{i,\alpha}},
\end{aligned}
\tag{2.61}
$$

and the raising and lowering operators and counting operator $c(i)$ become:

$$
\begin{aligned}
a_{i,\alpha}^{\dagger} &\to (1-c(i))\, z_{i,\alpha}, \\
a_{i,\alpha} &\to \frac{\partial}{\partial z_{i,\alpha}}, \\
c(i) &\to \sum_{\alpha=1}^{M} z_{i,\alpha} \frac{\partial}{\partial z_{i,\alpha}}.
\end{aligned}
\tag{2.62}
$$

With these transformations, the time evolution of observables can be easily calculated as before analogously to equation (2.24).

2.4.2 Spatial systems in continuous space

Define the state of the system in continuous space as consisting of n particles, located at positions $\boldsymbol{x}$, of species $\boldsymbol{\alpha}$. Here $\boldsymbol{x}$ is an $n \times q$ matrix, where q is the dimension of the space (e.g. $q = 3$ for 3D), and $\boldsymbol{\alpha}$ is a vector of length n containing the species label $\alpha_i \in \mathcal{R}$ for $i = 1, \ldots, n$. Note also that the state is unchanged by the ordering of the particles, i.e. when both $\boldsymbol{\alpha}, \boldsymbol{x}$ are reordered.

The form of the differential equation obeyed by the probability state vector (2.13) is again unchanged:

$$
\frac{d}{dt}|\Psi(t)\rangle = \sum_{r=1}^{R} W_r |\Psi(t)\rangle, \tag{2.63}
$$

where the probability state vector (2.13) now has become:

$$|\Psi(t)\rangle = \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x}\, p(n, \boldsymbol{\alpha}, \boldsymbol{x}, t)\, |n, \boldsymbol{\alpha}, \boldsymbol{x}\rangle\,. \tag{2.64}$$

The sum over $\boldsymbol{\alpha}$ considers for every particle all possible species, while the integral over position considers for every particle all possible points in the volume, or otherwise over $(-\infty, \infty)$. The time evolution operators W_r are now expressed in terms of the lowering and raising operators that act as:

$$\begin{aligned} a^{\dagger}_{\beta}(y)\, |n, \boldsymbol{\alpha}, \boldsymbol{x}\rangle &= |n+1, (\boldsymbol{\alpha}, \boldsymbol{x}) \cup \{(\beta, y)\}\rangle\,, \\ a_{\beta}(y)\, |n, \boldsymbol{\alpha}, \boldsymbol{x}\rangle &= n_{\alpha}(x)\, |n-1, (\boldsymbol{\alpha}, \boldsymbol{x}) \backslash \{(\beta, y)\}\rangle\,, \end{aligned} \tag{2.65}$$

where the operation $(\boldsymbol{\alpha}, \boldsymbol{x}) \cup \{(\beta, y)\}$ denotes appending the new element β to the vector $\boldsymbol{\alpha}$ and similarly for y and $\boldsymbol{x}$, and the operation $(\boldsymbol{\alpha}, \boldsymbol{x}) \backslash \{(\beta, y)\}$ denotes removing the particle of species β at located at position y, and:

$$n_{\beta}(y) = \sum_{i=1}^{n} \delta_{\beta, \alpha_i} \delta(x_i - y) \tag{2.66}$$

counts the number of particles of species α at a point x in 3D space.

For example, the diffusion operator is:

$$W_{\text{diff}} = \sum_{\alpha \in \mathcal{R}} D_{\alpha} \int dx\, a^{\dagger}_{\alpha}(x) \nabla^2_x a_{\alpha}(x) \to \sum_{\alpha \in \mathcal{R}} D_{\alpha} \int dx\, z_{\alpha}(x) \nabla^2_x \frac{\delta}{\delta z_{\alpha}(x)}. \tag{2.67}$$

As an example, consider a system with only one species with diffusion constant D. Appendix A shows how the formalism can be used to derive the equation for the first order moments, namely Fick's second law of diffusion:

$$\frac{\partial \langle n(x) \rangle}{\partial t} = D \nabla^2_x \langle n(x) \rangle, \tag{2.68}$$

where x is a point in 3D space. This derivation is easiest by using the generating function formalism defined by the transformations:

$$\begin{aligned} |n, \boldsymbol{\alpha}, \boldsymbol{x}\rangle &\to \prod_{i=1}^{n} z_{\alpha_i}(x_i), \\ |\Psi(t)\rangle &\to g[\boldsymbol{z}](t) = \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x}\, p(n, \boldsymbol{\alpha}, \boldsymbol{x}, t) \prod_{i=1}^{n} z_{\alpha_i}(x_i). \end{aligned} \tag{2.69}$$

Here the generating function variables z from before have turned into functions of a spatial coordinate, and the moment generating function has turned into a functional. Similarly, the raising and lowering operators transform as:

$$\begin{aligned} a_{\alpha}^{\dagger}(x) &\to z_{\alpha}(x), \\ a_{\alpha}(x) &\to \frac{\delta}{\delta z_{\alpha}(x)}, \end{aligned} \tag{2.70}$$

where the lowering operator is now expressed of functional derivatives.

Chapter 3

Graphical models and statistical inference

Machine learning is a broad field, encompassing neural networks inspired by neuroscience, as well as abstract constructions such as reinforcement learning that considers agents acting in an environment. To make the discussion precise, in this thesis the word machine learning is used specifically to refer to the problem of *statistical inference*: given a set of data, the objective is to estimate a probability distribution that abstracts the data in a meaningful way. Here, the notion of meaningful is problem specific. It could have to do with explaining the data, for example finding the directions of maximum variance as in principal component analysis (PCA). Alternatively, it could have to do with obtaining a useful representation of the data, for example a representation of images that can be used to train a classifier as in deep Boltzmann machines (DBMs).

At the heart of these problems are graphical models. In the last chapter, the idea to introduce a graphical model for learning a moment closure approximation from data was introduced. In this chapter, the necessary concepts about graphical models are developed. The focus is on undirected graphical models, also known as Markov random fields (MRFs),

as they are relevant for the model reduction work to follow. The foundations of MRFs are discussed, as well as their connection to Ising models in statistical physics. Learning in MRFs is introduced, ending in the Boltzmann machine learning algorithm and its relevance to training neural networks.

3.1 From Markov Random Fields to Boltzmann machines

Undirected graphical models offer a way to represent probabilistic relationships between random variables represented by nodes. Every undirected graphical model carries with it two notions: one about conditional independence, and one about factorization. Two random variables X and Y are conditionally independent of a third variable C if every path between nodes X and Y must pass through C. Consider for example the graphical model of Figure 3.1 that describes four random variables n_A, n_B, n_C, n_D corresponding to the number of particles of four species A, B, C, D. We can identify the following conditional independence properties among the four random variables n_A, n_B, n_C, n_D:

$$\begin{aligned} &(n_A \perp n_D) \mid n_C, \\ &(n_B \perp n_D) \mid n_C. \end{aligned} \tag{3.1}$$

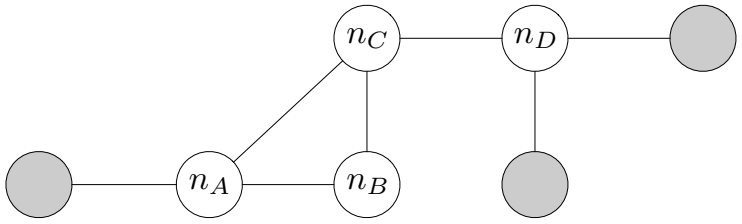

Figure 3.1: Example of a graphical model consisting of four random variables n_A, n_B, n_C, n_D and three other unspecified random variables (gray).

An undirected graphical model defines the concept of factorization through *cliques*.

A clique is any subset of nodes for which a link exists between all pairs of nodes in the subset, also known as a fully connected set of nodes. An example of a clique in Figure 3.1 is $\{n_A, n_B, n_C\}$. Adding the node n_D makes this not a clique, since no edge exists between nodes n_A and n_D. This makes the clique $\{n_A, n_B, n_C\}$ a *maximal clique*, which is a clique which would cease to be clique if any other node is added.

Maximal cliques are a key concept, as they allow the joint distribution $\tilde{p}(\boldsymbol{n})$ over random variables $\boldsymbol{n}$ defined by the graph to be written as a product of potential functions $\psi_C(\boldsymbol{n}_C)$ where the subscript refers to the C-th clique:

$$\begin{aligned} \tilde{p}(\boldsymbol{n}) &= \frac{1}{Z}\prod_C \psi_C(\boldsymbol{n}_C), \\ Z &= \sum_{\boldsymbol{n}}\prod_C \psi_C(\boldsymbol{n}_C). \end{aligned} \tag{3.2}$$

Note that $\psi_C(\boldsymbol{n}_C) \geq 0$ to ensure that the joint probability is strictly positive.

A famous result is the *Hammersley-Clifford* theorem, which states that the set of distributions defined by the conditional independence property is the same set of distributions defined by the factorization property. This lends a simple analogy: an undirected graphical model is a machine that takes as input all possible probability distributions, and outputs the family of distributions which obey factorization properties (3.2) or equivalently conditional independence properties such as (3.1).

The joint distribution (3.2) can be expressed in a more common form:

$$\tilde{p}(\boldsymbol{n}) = \frac{1}{Z}\exp\left[-\sum_C E(\boldsymbol{n}_C)\right], \tag{3.3}$$

where E is the energy function. Note that is always possible since the potential functions must be strictly positive. Every MRF therefore defines a family of *Boltzmann distributions*.

Several examples of Boltzmann distributions corresponding to graphical models

have already been encountered in Chapter 2, for example Gaussian graphical models:

$$\begin{aligned} E(\boldsymbol{n}) &= -\frac{1}{2}(\boldsymbol{n}-\boldsymbol{\mu})^{\intercal}\Sigma^{-1}(\boldsymbol{n}-\boldsymbol{\mu}) \\ &= -\frac{1}{2}\left(\boldsymbol{n}+B^{-1}\boldsymbol{\nu}_1-\frac{1}{2}B^{-1}\mathrm{diag}(B)\right)^{\intercal} B \left(\boldsymbol{n}+B^{-1}\boldsymbol{\nu}_1-\frac{1}{2}B^{-1}\mathrm{diag}(B)\right), \end{aligned} \tag{3.4}$$

where $\boldsymbol{\mu}, \Sigma$ are the mean and covariance matrix. The second line is an equivalent but less common way to write a Gaussian distribution that has already been encountered in Chapter 2, but makes the connection to graphical models precise. The precision matrix B has elements:

$$B = \begin{pmatrix} \nu_{2,\mathcal{R}_1,\mathcal{R}_1} & \nu_{2,\mathcal{R}_1,\mathcal{R}_2} & \cdots & \nu_{2,\mathcal{R}_1,\mathcal{R}_M} \\ \nu_{2,\mathcal{R}_1,\mathcal{R}_2} & \nu_{2,\mathcal{R}_2,\mathcal{R}_2} & \cdots & \nu_{2,\mathcal{R}_2,\mathcal{R}_M} \\ \cdots & & & \\ \nu_{2,\mathcal{R}_1,\mathcal{R}_M} & \nu_{2,\mathcal{R}_2,\mathcal{R}_M} & \cdots & \nu_{2,\mathcal{R}_M,\mathcal{R}_M} \end{pmatrix}, \tag{3.5}$$

where every entry $\nu_{2,\mathcal{R}_1,\mathcal{R}_2}$ is an interaction represented by a line in the graphical model, and $\boldsymbol{\nu}_1$ is the vector of bias terms for each variable. The structure of the graphical model therefore constrains the structure of the precision matrix, setting missing interactions to zero.

The two notations are related by:

$$\begin{aligned} B &= \Sigma^{-1}, \\ \boldsymbol{\nu}_1 &= -\Sigma^{-1}\boldsymbol{\mu} + \frac{1}{2}\mathrm{diag}(\Sigma^{-1}). \end{aligned} \tag{3.6}$$

A second popular example from statistical physics is the Ising model shown in Figure 3.2(a). Consider a 1D lattice of spins $\boldsymbol{s}$ for lattice sites $i = 1, \ldots, N$, where each $s_i \in \{-1, 1\}$. This may represent particles of a single species in the single occupancy limit,

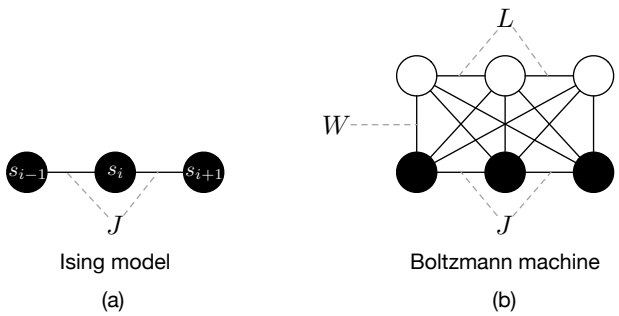

Figure 3.2: (a) 1D Ising model (biases not shown). (b) Boltzmann machine.

where s_i the represents the presence or absence of a particle. The energy function is:

$$E(\boldsymbol{s}) = -\sum_{i=1}^{N} a_i s_i - \sum_{i=1}^{N-1} J_{i,i+1} s_i s_{i+1}, \tag{3.7}$$

where a_i is a bias term (an external field), and $J_{i,i+1}$ are nearest neighbor interactions.

An important extension of the Ising model is to allow latent variables. This leads to a Boltzmann machine [13, 31], as shown in Figure 3.2(b). The energy function takes the form:

$$E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = -\boldsymbol{a}^\intercal \boldsymbol{v} - \boldsymbol{b}^\intercal \boldsymbol{h} - \frac{1}{2}\boldsymbol{v}^\intercal J \boldsymbol{v} - \frac{1}{2}\boldsymbol{h}^\intercal L \boldsymbol{h} - \boldsymbol{v}^\intercal W \boldsymbol{h}. \tag{3.8}$$

If the visible variables are of size N_v and the latent variables are of size N_h, then $\boldsymbol{a}$ and $\boldsymbol{b}$ are the bias vectors of size N_v and N_h. The matrix J is of size $N_v \times N_v$ and L is of size $N_h \times N_h$, both of which have diagonal terms set to zero such that there are no self-interactions. The visible-hidden interactions are given by the matrix W of size $N_v \times N_h$.

Boltzmann machines are an important model in machine learning. The discovery of how to efficiently train Boltzmann machines helped drive the recent revival in neural networks, as discussed later in Chapter 3.4.1

3.2 Boltzmann machine learning algorithm

At the heart of statistical inference is to determine the interactions in a graphical model from a given set of data. For example, let $\boldsymbol{n}$ represent the data. Let the unknown data distribution that $\boldsymbol{n}$ is obtained from be p, and the graphical model define the distribution $\tilde{p}$. For $\tilde{p}$ to approximate p, we seek to minimize the Kullback-Leibler (KL) divergence from the the model to the data distribution:

$$\mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = \sum_{\boldsymbol{n}} p(\boldsymbol{n}) \ln \frac{p(\boldsymbol{n})}{\tilde{p}(\boldsymbol{n};\boldsymbol{\theta})}. \tag{3.9}$$

In statistical physics, this problem is sometimes called the inverse Ising problem if all units are visible.

The KL divergence is not a metric because it is not symmetric and does not obey the triangle inequality. However, it has an important interpretation is terms of the likelihood of the data - from the property of the logarithm:

$$\mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = H(p) - \sum_{\boldsymbol{n}} p(\boldsymbol{n}) \ln \tilde{p}(\boldsymbol{n};\boldsymbol{\theta}), \tag{3.10}$$

where $H(p)$ is the entropy. If data $\boldsymbol{n}$ are sampled from p, then the second term is directly the negative log likelihood. Minimizing the KL divergence is therefore equivalent to maximizing the log likelihood.

The gradient of the KL divergence with respect to the parameters is:

$$\begin{aligned}\frac{\partial \mathcal{D}_{\mathcal{KL}}(p||\tilde{p})}{\partial \boldsymbol{\theta}} &= -\sum_{\boldsymbol{n}} \frac{p(\boldsymbol{n})}{\tilde{p}(\boldsymbol{n};\boldsymbol{\theta})} \frac{\partial \tilde{p}(\boldsymbol{n};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \\ &= -\sum_{\boldsymbol{n}} \frac{p(\boldsymbol{n})}{\tilde{p}(\boldsymbol{n};\boldsymbol{\theta})} \frac{1}{Z(\boldsymbol{\theta})} \frac{\partial \exp(-E(\boldsymbol{n};\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}} + \sum_{\boldsymbol{n}} \frac{p(\boldsymbol{n})}{Z(\boldsymbol{\theta})} \frac{\partial Z(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \\ &= \sum_{\boldsymbol{n}} p(\boldsymbol{n}) \frac{\partial E(\boldsymbol{n};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} - \sum_{\boldsymbol{n}} p(\boldsymbol{n}) \sum_{\boldsymbol{n}'} \tilde{p}(\boldsymbol{n}';\boldsymbol{\theta}) \frac{\partial E(\boldsymbol{n}';\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \\ &= \left\langle \frac{\partial E(\boldsymbol{n};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle_p - \left\langle \frac{\partial E(\boldsymbol{n};\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right\rangle_{\tilde{p}},\end{aligned} \tag{3.11}$$

where the final result follows because p is normalized, and $\langle X \rangle_p$ denotes an average taken over the distribution p, and similarly $\langle X \rangle_{\tilde{p}}$ over $\tilde{p}$.

Equation (3.11) is a popular result that forms the core of what is known as the *Boltzmann machine learning algorithm* (BMLA). The BMLA is a first order optimization method to minimize the KL divergence from $\tilde{p}$ to p. Note that $\mathcal{D}_{\mathcal{KL}}(p||\tilde{p})$ is convex in $\tilde{p}$ for fixed p. The average over the data distribution p is also known as the *wake phase moment*, and the average over the model distribution $\tilde{p}$ is also known as the *sleep phase moment*. In practice, these moments are typically approximated by Markov chain Monte Carlo (MCMC) sampling [13], with separate chains used for the two observables. This makes the BMLA a *stochastic* first order optimization method. Finally, while (3.11) considers discrete random variables $\boldsymbol{n}$, the BMLA similarly applies to continuous random variables, with sums replaced by integrals.

3.2.1 Connection to expectation maximization

The general way to learn the interaction parameters in a graph with latent variables is by expectation maximization (EM). Splitting the random variables into observed $\boldsymbol{n}_v$ and latent variables $\boldsymbol{h}$, it is based on the following decomposition, valid for any distribution

$q(\boldsymbol{n}_h)$:

$$\ln \tilde{p}(\boldsymbol{n}_v;\boldsymbol{\theta}) = -\mathcal{D}_{\mathcal{KL}}(q(\boldsymbol{n}_h)||\tilde{p}(\boldsymbol{n}_v,\boldsymbol{n}_h;\boldsymbol{\theta})) + \mathcal{D}_{\mathcal{KL}}(q(\boldsymbol{n}_h)||\tilde{p}(\boldsymbol{n}_h|\boldsymbol{n}_v;\boldsymbol{\theta})). \tag{3.12}$$

Since the KL divergence is strictly positive, then the first term $-\mathcal{D}_{\mathcal{KL}}(q(\boldsymbol{n}_h)||\tilde{p}(\boldsymbol{n}_v,\boldsymbol{n}_h))$ is a lower bound on the log likelihood. The EM algorithm [26] therefore proceeds in two steps:

- The expectation step, where $\boldsymbol{\theta}^{\text{old}}$ is fixed at the current values, and q is chosen to minimize the second term. Clearly, this occurs for $q(\boldsymbol{n}_h) = \tilde{p}(\boldsymbol{n}_h|\boldsymbol{n}_v;\boldsymbol{\theta}^{\text{old}})$, which makes the bound tight.

- The maximization step, where q is fixed and the first term is maximized in $\boldsymbol{\theta}$. Substituting the solution of the expectation step into the first term gives the update step:

  $$\boldsymbol{\theta} \leftarrow \max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta},\boldsymbol{\theta}^{\text{old}}), \tag{3.13}$$

  where

$$Q(\boldsymbol{\theta},\boldsymbol{\theta}^{\text{old}}) = -\mathcal{D}_{\mathcal{KL}}(q(\boldsymbol{n}_h)||\tilde{p}(\boldsymbol{n}_v,\boldsymbol{n}_h;\boldsymbol{\theta})) = \sum_{\boldsymbol{n}_h} \tilde{p}(\boldsymbol{n}_h|\boldsymbol{n}_v;\boldsymbol{\theta}^{\text{old}}) \ln \tilde{p}(\boldsymbol{n}_v,\boldsymbol{n}_h;\boldsymbol{\theta}) + \text{const.} \tag{3.14}$$

EM and the BMLA are closely connected and maximize the same objective [32]. Evaluating $\tilde{p}(\boldsymbol{n}_h|\boldsymbol{n}_v;\boldsymbol{\theta}^{\text{old}})$ in the expectation step is often intractable and done by sampling as in the BMLA. With this, the gradient of BMLA (3.11) is the same as the gradient of the objective function in the maximization step (3.14).

3.3 Contrastive divergence

Consider again the Boltzmann machine (3.8). Minimizing the KL-divergence using the BMLA leads to the gradients of the usual form (3.11) - a difference between expectations under the model and data distributions:

$$\begin{aligned}\frac{\partial E}{\partial W} &= \left\langle \boldsymbol{v}\boldsymbol{h}^{\mathsf{T}} \right\rangle_{\tilde{p}} - \left\langle \boldsymbol{v}\boldsymbol{h}^{\mathsf{T}} \right\rangle_{p}, \\ \frac{\partial E}{\partial L} &= \left\langle \boldsymbol{v}\boldsymbol{v}^{\mathsf{T}} \right\rangle_{\tilde{p}} - \left\langle \boldsymbol{v}\boldsymbol{v}^{\mathsf{T}} \right\rangle_{p}, \\ \frac{\partial E}{\partial J} &= \left\langle \boldsymbol{h}\boldsymbol{h}^{\mathsf{T}} \right\rangle_{\tilde{p}} - \left\langle \boldsymbol{h}\boldsymbol{h}^{\mathsf{T}} \right\rangle_{p},\end{aligned} \tag{3.15}$$

and similarly for the bias vectors $\boldsymbol{a}$ and $\boldsymbol{b}$. While the exact observables in this model are intractable, an efficient MCMC sampling method known as Gibbs sampling [26] can be used to approximate them from a small chosen number of chains. Gibbs sampling iteratively draws from the marginal distributions:

$$\begin{aligned}p(v_i = 1 | \boldsymbol{v}_{-i}, \boldsymbol{h}) &= \sigma_{\text{sigmoid}}\left(a_i + \sum_{k \neq i}^{N_v} J_{ik} v_k + \sum_{j=1}^{N_h} W_{ij} h_j \right), \\ p(h_j = 1 | \boldsymbol{v}, \boldsymbol{h}_{-j}) &= \sigma_{\text{sigmoid}}\left(b_j + \sum_{l \neq j}^{N_h} L_{jl} h_l + \sum_{i=1}^{N_v} W_{ij} v_i \right),\end{aligned} \tag{3.16}$$

where $\boldsymbol{v}_{-i}$ denotes $\boldsymbol{v}$ without the i-th element, and the sigmoid is:

$$\sigma_{\text{sigmoid}}(x) = \left(1 + e^{-x}\right)^{-1}. \tag{3.17}$$

Despite the fact that the chains can be run in parallel, letting them run to convergence is a major computational bottleneck. Instead, the idea to terminate the chains early was popularized by [33] in 2002. In the original algorithm, known as contrastive divergence (CD), only a single step of the Gibbs sampler for both chains is run at each optimization

step. CD therefore does not minimize the KL divergence, but rather minimizes [34]:

$$\mathcal{D}_{\mathcal{KL}}(p_0||\tilde{p}) - \mathcal{D}_{\mathcal{KL}}(p_1||\tilde{p}) \tag{3.18}$$

where p_0 is the data distribution and p_1 is the distribution after a single step of Gibbs sampling. CD greatly improves the efficiency of the BMLA. A second advantage of the CD is the stochasticity of the moments - stochastic gradient descent can improve optimization in high dimensional parameter spaces, because there is a chance that the optimizer can escape local minima through noisy gradients [35].

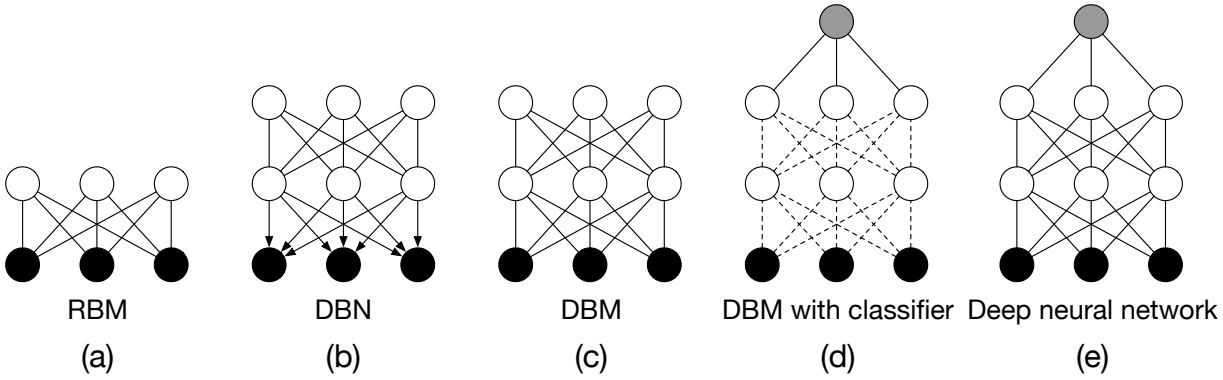

Figure 3.3: (a) Restricted Boltzmann machine (RBM) with visible units in black and latent units in white. (b) Deep belief network (DBN). (c) Deep Boltzmann machine (DBM). (d) DBM with classifier on top, indicated by a gray unit for binary classification. Other weights are commonly fixed (dashed lines) while training the classifier, or otherwise only allowed to adjust with a small learning rate. (e) A deep neural network for a binary classification task trained end-to-end.

To see CD in action, consider a popular form of a Boltzmann machine known as restricted Boltzmann machines (RBMs), in which the visible-visible and hidden-hidden interactions are discarded. The energy function becomes:

$$E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = -\boldsymbol{a}^\intercal \boldsymbol{v} - \boldsymbol{b}^\intercal \boldsymbol{h} - \boldsymbol{v}^\intercal \boldsymbol{W} \boldsymbol{h}. \tag{3.19}$$

For a given set of data samples $\{\boldsymbol{v}^{[0]}\}$, CD sampling proceeds by for every sample:

- Sampling the hidden units given the visible:

$$p(h_j^{[0]} = 1|\boldsymbol{v}^{[0]}) = \sigma_{\text{sigmoid}}\left(b_j + \sum_{i=1}^{N_v} W_{ij} v_i^{[0]}\right). \tag{3.20}$$

- The sample $(\boldsymbol{v}^{[0]}, \boldsymbol{h}^{[0]})$ is used as one of the samples to evaluate the moments under the data distribution.

- Sampling the visible units from the hidden, followed by (3.20) again:

$$\begin{aligned} p(v_i^{[1]} = 1|\boldsymbol{h}^{[0]}) &= \sigma_{\text{sigmoid}}\left(a_i + \sum_{j=1}^{N_h} W_{ij} h_j^{[0]}\right), \\ p(h_j^{[1]} = 1|\boldsymbol{v}^{[1]}) &= \sigma_{\text{sigmoid}}\left(b_j + \sum_{i=1}^{N_v} W_{ij} v_i^{[1]}\right). \end{aligned} \tag{3.21}$$

- The sample $(\boldsymbol{v}^{[1]}, \boldsymbol{h}^{[1]})$ is used as one of the samples to evaluate the moments under the model distribution.

Another important variant on CD is persistent CD (PCD) [36], in which the Gibbs chains for estimating the model and data distributions are separated, and the model distribution chain is not restarted every optimization step. Instead, the latent variables $\boldsymbol{h}^{[1]}$ of the last optimization step are used in the next optimization step as the initial $\boldsymbol{h}^{[0]}$ in the second step of CD sampling (3.21). The data chains are restarted at every optimization step as before and given by (3.20).

3.4 Deep belief networks and deep Boltzmann machines

RBMs have been used in applications from representing the quantum many body wavefunction [37] to modeling lattice proteins [38]. An obvious extension is to have multiple layers of hidden units to learn richer latent representations. A deep Boltzmann machine (DBM) [31] has energy function:

$$E(\{\boldsymbol{s}^{(l)}\}) = -\sum_{l=0}^{L-1}(\boldsymbol{a}^{(l)})^{\intercal}\boldsymbol{s}^{(l)} - \sum_{l=0}^{L-2}(\boldsymbol{s}^{(l)})^{\intercal}W^{(l,l+1)}\boldsymbol{s}^{(l+1)}, \tag{3.22}$$

where units $\boldsymbol{s}^{(l)}$ of size $N^{(l)}$ have been organized by layer $l = 0, \ldots, L-1$. Despite the direct generalization from RBMs, DBMs were initially non-trivial to train. When multiple layers are involved, the signals appearing in the activation function can fluctuate wildly because the layers are coupled to eachother. Small learning rates can be used to compensate the large fluctuations in the gradients, but this dramatically slows down learning. The tricks developed to train DBMs reveal some of the key ingredients that current neural networks use to train end-to-end on supervised tasks.

One early strategy to circumvent this is layer-by-layer pre-training of stacked RBMs [39], where a single RBM layer is trained at a time. After it is trained, its weights and biases are frozen to train the next RBM in the stack. The visible units for training this next RBM are obtained by a forward pass through the lower RBMs in the stack. The resulting model is known as a deep belief network (DBN). A similar pre-training strategy for DBMs was one of the first methods that allowed deep representations to be learned [31].

Despite this success, pre-training is labor intensive and requires careful tuning for the solution of several optimization problems in series. A more robust strategy is to use a *centering transformation* [40, 41]. A centered DBM with parameters $\tilde{\boldsymbol{a}}^{(l)}, \tilde{W}^{(l,l+1)}$ has

energy function:

$$E(\{\boldsymbol{s}^{(l)}\}) = -\sum_{l=0}^{L-1}(\tilde{\boldsymbol{a}}^{(l)})^{\intercal}\left(\boldsymbol{s}^{(l)}-\boldsymbol{\mu}^{(l)}\right) - \sum_{l=0}^{L-2}\left(\boldsymbol{s}^{(l)}-\boldsymbol{\mu}^{(l)}\right)^{\intercal}\tilde{W}^{(l,l+1)}\left(\boldsymbol{s}^{(l+1)}-\boldsymbol{\mu}^{(l+1)}\right), \quad (3.23)$$

where $\boldsymbol{\mu}^{(l)}$ are the centers in layer l. The centers are additional parameters that are chosen (rather than learned) as the mean unit activation in each layer. Crucially, every regular DBM can be transformed to a centered DBM by transforming parameters as:

$$\begin{aligned}\tilde{W}^{(l,l+1)} &= W^{(l,l+1)},\\ \tilde{\boldsymbol{a}}^{(l)} &= \boldsymbol{a}^{(l)} + (W^{(l-1,l)})^{\intercal}\boldsymbol{\mu}^{(l-1)} + W^{(l,l+1)}\boldsymbol{\mu}^{(l+1)}.\end{aligned} \quad (3.24)$$

This can be used to derive the *centered gradient* [41]: After sampling the moments of a regular DBM, transform to a centered DBM, calculate the gradient with respect to the centered parameters, and transform back to obtain the gradient for the regular DBM parameters. The result is

$$\begin{aligned}\frac{\partial E}{\partial W^{(l,l+1)}} &= \left\langle(\boldsymbol{s}^{(l)}-\boldsymbol{\mu}^{(l)})(\boldsymbol{s}^{(l+1)}-\boldsymbol{\mu}^{(l+1)})^{\intercal}\right\rangle_p - \left\langle(\boldsymbol{s}^{(l)}-\boldsymbol{\mu}^{(l)})(\boldsymbol{s}^{(l+1)}-\boldsymbol{\mu}^{(l+1)})^{\intercal}\right\rangle_{\tilde{p}},\\ \frac{\partial E}{\partial a_\alpha^{(l)}} &= \left\langle \boldsymbol{s}^{(l)}\right\rangle_p - \left\langle \boldsymbol{s}^{(l)}\right\rangle_{\tilde{p}} - \left(\frac{\partial E}{\partial W^{(l-1,l)}}\right)^{\intercal}\boldsymbol{\mu}^{(l-1)} - \frac{\partial E}{\partial W^{(l,l+1)}}\boldsymbol{\mu}^{(l+1)}.\end{aligned} \quad (3.25)$$

To reduce noise, the centers are updated as the average unit's state with an exponential sliding average with sliding parameter $r \in [0,1]$:

$$\boldsymbol{\mu}^{(l)} \leftarrow (1-r)\times\boldsymbol{\mu}^{(l)} + r\times\left\langle \boldsymbol{s}^{(l)}\right\rangle_{\tilde{p},\text{batch}}, \quad (3.26)$$

where the average is over the batch of training data used in a single optimization step.

The centering transformations enables training DBMs without pre-training. The availability of multiple layers of hidden units allows more useful representations to be learned than in RBMs.

3.4.1 To supervised learning

Many common applications of machine learning are to supervised tasks, i.e. tasks where data are labeled, and the goal is to predict a target output for a given input. The Boltzmann machine learning algorithm applies to unsupervised problems, i.e. where data are unlabeled, and the goal is to learn a more useful representation of the data. These representations directly enable supervised applications, most commonly by adding a classifier that takes as inputs the latent variables in the final layer. Figure 3.3 shows an illustration of a DBM where a binary classifier is trained from the final hidden layer. Commonly, the weights and biases in the DBM are held fixed while the classifier is trained.

Figure 3.3 is an important concept, because it shows that every supervised task can be viewed as containing an unsupervised task of learning useful representations. Separately training DBMs and then a classifier was one of the first methods to effectively train deep neural networks. Training a deep neural network “end-to-end" in a supervised fashion suffers from the vanishing gradient problem. This arises from applying backpropagation to calculate the gradients: small gradients in one layer will lead to increasingly small gradients in subsequent layers because they are multiplicative. However, since the development of DBMs, several advancements have allowed the training of networks end-to-end, without explicitly separating out an unsupervised task:

- Replacing the sigmoid activation function (3.17) with rectified linear units (ReLUs), given by:

$$\sigma_{\text{ReLU}}(x) = \begin{cases} 1 & \text{if } x \geq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.27}$$

  Note that ReLUs can be represented in Boltzmann machines, e.g. in RBMs, by an

energy function of the form [42]:

$$E(\boldsymbol{v},\boldsymbol{h}) = -\boldsymbol{v}^\intercal W \boldsymbol{h} - \boldsymbol{a}^\intercal \boldsymbol{v} + \sum_{j=1}^{M} \mathcal{U}(h_j),$$
$$\mathcal{U}(h) = \begin{cases} \frac{h^2}{2} + h\theta & \text{for } h \geq 0, \\ \infty & \text{for } h < 0. \end{cases} \tag{3.28}$$

- The use of dropout, where during training, units are set to zero after passing the activation function with some probability α. Common choices are $\alpha \sim 0.5$ for most layers, but smaller rates $\alpha \sim 0.1$ in the input layer to prevent discarding signals too early. After training the network, the dropout layers are discarded, such that the evaluation of the network on a given input is deterministic. However, during training, dropout reduces the average magnitudes of signals passing between layers by a factor α. To account for this, after training, weights are rescaled by the factor α corresponding to each layer.

  Dropout greatly improves the generalization of neural networks to samples not seen during training. It is commonly combined with regularization of the weights.

- Regularization of weights, commonly by introducing an L_1 norm in the objective function, or otherwise by clipping weights beyond a cutoff magnitude.

- Batch normalization [43], which applies a similar transformation as centering, where the input to every activation function is first standardized as:

$$x \to x_0 = \frac{x - \mu}{\sqrt{\sigma^2}}, \tag{3.29}$$

and then additionally, an affine transformation is performed:

$$\hat{x} = \gamma x_0 + \beta, \tag{3.30}$$

and the resulting $\hat{x}$ passed to the activation function as before. Note that here all parameters are scalars, i.e. the input to every neuron is standardized and transformed independently. The mean μ and variance σ^2 are estimated over batches of data, while the shift and scale parameters β, γ are learned as part of backpropagation.

Batch normalization directly standardizes the magnitudes of signals passed between layers and prevents gradients from vanishing. Note that batch normalization and dropout are not used together. During training, the approximate magnitude of the signal seen in a layer is reduced by the dropout factor α. This corrupts the magnitudes of the mean and variance estimated in batch normalization.

3.5 Machine learning for model reduction

Previous work has shown the applicability of machine learning to model reduction. In particular, the "Graph Constrained Correlation Dynamics" (GCCD) framework [10] uses a Markov Random Field (MRF) of plausible state variables and interactions as input, incorporating human expertise into the model reduction process. The probability distribution associated with this MRF is written in a form that separates the time evolution $\mu(t)$ from the graph structure $V_\alpha(s)$:

$$\tilde{p}(s,t;\{\mu\}) = \frac{1}{\mathcal{Z}(\mu(t))} \exp\left[-\sum_\alpha \mu_\alpha(t) V_\alpha(s)\right]. \tag{3.31}$$

where s are the random variables, μ are time-dependent interactions, and $V_\alpha(s)$ are clique potentials. At each point in time, the interactions $\mu(t)$ are learned separately from the

data using the Boltzmann machine learning algorithm (BMLA) [13] introduced earlier. To link snapshots in time, a differential equation model is introduced for the interactions:

$$\frac{d\mu_\alpha(t)}{dt} = f_\alpha(\mu(t)) = \sum_A \theta_{\alpha,A} f_A(\mu(t)), \tag{3.32}$$

which is linear in basis functions f_A with parameter matrix θ to be learned. The basis functions are elementary functions such exponentials and polynomials in μ, which are identified to be relevant to the problem from two and four-state binding models. Later work on model identification [44] used a similar linear model, but without the connection to a reduced model probability distribution (3.31).

After learning parameters μ from data separately using BMLA at each timepoint $t = 0, 1, \ldots, T$ and calculating their derivatives in time, GCCD minimizes the L_2 loss:

$$S = \sum_\alpha \sum_{t=0}^{T} \left| \sum_A \theta_{\alpha,A} f_A(\mu(t)) - \left(\frac{d\mu_\alpha(t)}{dt} \right)_{\text{BMLA}} \right|^2 + \lambda \sum_\alpha |\theta_{\alpha,A}| \tag{3.33}$$

for the parameter matrix θ and where λ is a sparsity regularizer.

The choice of the MRF for (3.31) is guided by human knowledge. For example, to model the activation of Ca2+/calmodulin-dependent protein kinase II (CaMKII) by calcium, the graphical model reflected the known structure of CaMKII, resulting in a large degree of model reduction [10]. The ability to introduce prior knowledge into the problem to guide the choice of the MRF is a key advantage of the method.

The two subproblems in GCCD are solved *separately*:

- The state estimation problem to determine the interactions in the graph (3.31) at an instant in time.
- The time evolution problem to determine the optimal coefficients in the differential equation model (3.32).

Solving these problems independently is a limitation for the representations that can be learned, as these problems are actually linked. The idea to formulate a single differential equation constrained optimization problem is starting point for the introduction of dynamic Boltzmann distributions in the next chapter. Furthermore, it is often the case that the dynamics studied are not well-described by linear combinations of available basis functions [44]. One solution is to extend the set of basis functions, but the growing success of machine learning suggests that a more promising approach is to learn non-linear combinations of basis functions through a neural network. This idea will be explored in later chapters of this thesis.

Chapter 4

Dynamic Boltzmann distributions

The previous chapter introduced the Boltzmann machine learning algorithm for statistical inference in graphs. In this chapter, dynamic Boltzmann distributions will be introduced as reduced models for stochastic chemical reaction networks. The advantage of this approach is the strong parallels between the fine scale model of chemical kinetics described in Chapter 1 and the reduced model defined by a graphical model described in Chapter 2. Therefore, the physics of the system can be incorporated into the problem as shown for several example systems. This chapter is taken with minor edits from [16].

4.1 Spatial dynamic Boltzmann distributions

Consider a system of n particles. Each particle carries with it a species label α_i, such that the vector $\boldsymbol{\alpha}$ of length n describes the species labels of all particles. Further, let the particles be distributed in 3D space, such that the position of all particles is described by the matrix $\boldsymbol{x}$ of size $n \times d$ where $d = 3$ is the dimension of the space. The fine-scale state of the system at time t is then described by $|n, \boldsymbol{\alpha}, \boldsymbol{x}, t\rangle$, where we have used the common bra/ket notation. The time evolution of this state can be expressed in the Doi-Peliti formalism as shown in Chapter 2.

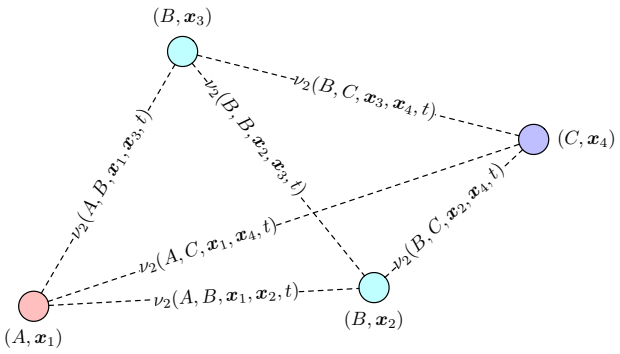

Figure 4.1: Example of time-dependent pairwise interactions ν_2 between four particles of species (A, B, B, C) at positions $(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \boldsymbol{x}_4)$.

In the spirit of a Markov random field (MRF), construct states in a coarse-scale model:

$$\left|\{\nu_k\}_{k=1}^K, t\right\rangle = \frac{1}{\mathcal{Z}\left[\{\nu_k\}_{k=1}^K\right]} \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x} \, \exp\left[-\sum_{k=1}^{K} \sum_{\langle i \rangle_k^n} \nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)\right] |n, \boldsymbol{\alpha}, \boldsymbol{x}, t\rangle, \quad (4.1)$$

where $\langle i \rangle_k^n = \{i_1 < i_2 < \cdots < i_k : i \in [1, n]\}$ denotes ordered subsets of k indexes each in $\{1, 2, \ldots, n\}$, and $\nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)$ are *k-particle interaction functions* up to a cutoff order K. We note that $\{\ldots\}_{k=1}^K$ is used to denote an index-ordered set in this context. In other words, equation (4.1) considers all self-interactions ν_1 for every particle, all pairwise interactions ν_2 between all possible pairs of particles, etc., continuing up to a cutoff interaction order K. Figure 4.1 shows an example of such pairwise interactions. This expansion of n-body interactions is a specific case of more general dimension-wise decompositions, such as analysis of least variance (ANOVA) [45].

The probability of being in a state $|n, \boldsymbol{\alpha}, \boldsymbol{x}, t\rangle$ is given by a dynamic and instantaneous Boltzmann distribution:

$$\tilde{p}(n, \boldsymbol{\alpha}, \boldsymbol{x}, t) = \left\langle n, \boldsymbol{\alpha}, \boldsymbol{x}, t \middle| \{\nu_k\}_{k=1}^K, t \right\rangle = \frac{\exp\left[-\sum_{k=1}^K \sum_{\langle i \rangle_k^n} \nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)\right]}{\mathcal{Z}\left[\{\nu_k\}_{k=1}^K\right]}. \quad (4.2)$$

The probability distribution $p(n, \boldsymbol{\alpha}, \boldsymbol{x}, t)$ over the fine scale states $|n, \boldsymbol{\alpha}, \boldsymbol{x}, t\rangle$ evolves ac-

cording to the CME. To describe the time evolution of the reduced model, introduce a differential equation system for the interaction functions $\{\nu_k\}_{k=1}^K$:

$$\frac{d}{dt}\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t) = \mathcal{F}_k\left[\{\nu(\boldsymbol{\alpha}, \boldsymbol{x}, t)\}\right], \tag{4.3}$$

where each $\mathcal{F}_k$ is a functional. The arguments to this functional are:

$$\{\nu(\boldsymbol{\alpha}, \boldsymbol{x}, t)\} = \left\{\nu_{k'}(\boldsymbol{\alpha}_{\langle j\rangle_{k'}^n}, \boldsymbol{x}_{\langle j\rangle_{k'}^n}, t)\ \forall\ \langle j\rangle_{k'}^n : 1 \leq k' \leq K\right\}, \tag{4.4}$$

which denotes all possible ν functions evaluated at the given arguments. The right hand side $\mathcal{F}_k$ may be a global functional, in the sense that the arguments $\{\nu(\boldsymbol{\alpha}, \boldsymbol{x}, t)\}$ are not restricted to the arguments appearing on the left hand side of (4.3).

4.1.1 What makes the reduced model a good choice

In addition to the connection to MRFs, we note several advantages of the form of this reduced model (4.2,4.3):

1. Since the states $\left|\{\nu_k\}_{k=1}^K, t\right\rangle$ define a grand canonical ensemble (GCE), (4.2) exactly describes equilibrium systems, and is expected to reasonably approximate systems approaching equilibrium.

2. If the interactions between two groups of particles are independent, their joint probability distribution equals the product of their probabilities, and their interaction functions ν_k in (4.2) sum. The Boltzmann distribution thus preserves the locality of interactions.

3. A further important result pertains to linearity, stated in the following proposition.

Proposition 1. *Given a reaction network and a fixed collection of K interaction functions*

$\{\nu_k\}_{k=1}^K$, *the linearity of the CME in reaction operators extends to the functionals* $\mathcal{F}_k = \sum_{r=1}^R \mathcal{F}_k^{(r,closed)}$ *calculated under the zero-information moment closure approximation.*

Proof. The dynamic Boltzmann distribution $\tilde{p}(n, \boldsymbol{\alpha}, \boldsymbol{x}, t)$ is a maximum entropy (MaxEnt) distribution, where each interaction function $\nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)$ controls a corresponding moment $\mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)\}$, given by:

$$\mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) = \sum_{n'=0}^{\infty} \sum_{\boldsymbol{\alpha}'} \int d\boldsymbol{x}'\, p(n', \boldsymbol{\alpha}', \boldsymbol{x}', t) \sum_{\langle j \rangle_k^{n'}} \delta(\boldsymbol{x}_{\langle i \rangle_k^n} - \boldsymbol{x}'_{\langle j \rangle_k^{n'}}) \delta(\boldsymbol{\alpha}_{\langle i \rangle_k^n} - \boldsymbol{\alpha}'_{\langle j \rangle_k^{n'}}). \quad (4.5)$$

Here, $\delta(\boldsymbol{x})$ denotes a multi-dimensional Dirac delta function. Note that there are $L = \sum_{k=1}^K \binom{n}{k}$ interaction terms and equally many moments they control. Switching to vector notation, let $\boldsymbol{\nu}$ of length L denote the interaction functions, and $\boldsymbol{\mu}$ the corresponding moments.

Relating the interaction functions to the moments constitutes an inverse Ising problem. Let the solution to this problem be

$$\nu_l = \phi_l(\boldsymbol{\mu}) \quad (4.6)$$

for some functions ϕ_l for $l = 1, \ldots, L$. This solution depends only on the interaction functions, and not on the reaction operators appearing in the CME. For a single reaction process, let the differential equations for the moments be $\dot{\boldsymbol{\mu}}^{(r)}$, resulting from $\dot{p}^{(r)} = W^{(r)} p^{(r)}$, where $\dot{x}$ denotes a time derivative. Taking the time derivatives of both sides of (4.6) gives:

$$\dot{\nu}_l^{(r)} = \sum_{l'=1}^{L} \frac{\partial \phi_l(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}_{l'}} \dot{\boldsymbol{\mu}}_{l'}^{(r)} \quad (4.7)$$

The differential equations for the moments $\dot{\boldsymbol{\mu}}_{l'}^{(r)}$ typically do not close; under the zero-information closure approximation, we replace p with $\tilde{p}$, closing these equations under the

assumed form of $\tilde{p}$. Let the closed form be $\dot{\boldsymbol{\mu}}_{l'}^{(r,\text{closed})}$. For the full network of reactions then:

$$\dot{\nu}_l = \sum_{l'=1}^{L} \frac{\partial \phi_l(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}_{l'}} \dot{\boldsymbol{\mu}}_{l'} = \sum_r \sum_{l'=1}^{L} \frac{\partial \phi_l(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}_{l'}} \dot{\boldsymbol{\mu}}_{l'}^{(r,\text{closed})} = \sum_r \dot{\nu}_l^{(r,\text{closed})} \tag{4.8}$$

gives the desired linearity property by the definitions $\mathcal{F}_l = \dot{\nu}_l$ and $\mathcal{F}_l^{(r,\text{closed})} = \dot{\nu}_l^{(r,\text{closed})}$.

□

Due to Proposition 1, the functions $F^{(r,\text{closed})}$ can be referred to as *basis functionals*. The utility of this property will be explored further in a machine learning context.

4.2 The two problems for learning dynamic Boltzmann distributions

The learning problem for dynamic Boltzmann distributions consists of two parts:

1. The *state estimation problem* to determine the interaction functions in the energy function (4.2) at an instant in time.

2. The *reduced model estimation problem* to determine the right hand sides of the differential equations (4.3).

In practice, the form of the differential must be specified in terms of ordinary parameters in order to solve an optimization problem. Before we do this, we will write down a general solution to this problem, and then consider how it can help to guide the choice for the differential equations.

While the two problems can be solved separately as in GCCD, the two depend on each-other. Define the action as the KL-divergence between the true and reduced models

(extending Ref. [10]):

$$\begin{aligned} S &= \int_0^\infty dt\, \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}), \\ \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) &= \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x}\, p(n,\boldsymbol{\alpha},\boldsymbol{x},t) \ln \frac{p(n,\boldsymbol{\alpha},\boldsymbol{x},t)}{\tilde{p}(n,\boldsymbol{\alpha},\boldsymbol{x},t)}. \end{aligned} \tag{4.9}$$

Next, we introduce notation to define a higher-order variational problem. Since the interaction functions are defined by specifying the set of functionals $\mathcal{F}_k[\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\}]$ for all $k = 1,\ldots,K$, we use the notation $\nu_k[\![\{\mathcal{F}\}]\!]$ to denote that ν_k is a higher-order generalization of a functional. The action is a functional of the set of all interaction functions, which we denote by $S[\{\nu[\![\{\mathcal{F}\}]\!]\}]$, where $\{x\} = \{x_k\}_{k=1}^K$.

The higher-order variational problem for the basis functionals is given by the chain rule:

$$\frac{\hat{\delta} S[\{\nu[\![\{\mathcal{F}\}]\!]\}]}{\hat{\delta}\mathcal{F}_k\left[\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\}\right]} = \sum_{k'=1}^{K} \sum_{\boldsymbol{\alpha}'} \int d\boldsymbol{x}' \int dt'\, \frac{\delta S[\{\nu\}]}{\delta \nu_{k'}(\boldsymbol{\alpha}',\boldsymbol{x}',t')} \frac{\hat{\delta}\nu_{k'}(\boldsymbol{\alpha}',\boldsymbol{x}',t')}{\hat{\delta}\mathcal{F}_k\left[\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\}\right]} = 0, \tag{4.10}$$

where we use the notation $\hat{\delta}$ to denote that this is not an ordinary variational problem, in the sense that a variation with respect to a functional is implied. Equation (4.10) should therefore be regarded as a purely notation solution, generalizing the well-known chain rule for functionals where a variational derivative is taken of a functional of a functional: $\frac{\delta F[G[\phi]]}{\delta\phi(y)} = \int dx\, \frac{\delta F[G]}{\delta G(x)} \frac{\delta G[\phi]}{\delta\phi(y)}$. The first term is a variational derivative analogous to that appearing in the derivation of the BM learning algorithm [13], giving:

$$\begin{aligned} &\frac{\hat{\delta} S[\{\nu[\![\{\mathcal{F}\}]\!]\}]}{\hat{\delta}\mathcal{F}_k\left[\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\}\right]} \\ &= \sum_{k'=1}^{K} \sum_{\boldsymbol{\alpha}'} \int d\boldsymbol{x}' \int_0^\infty dt' \, \left(\mu_{k'}(\boldsymbol{\alpha}',\boldsymbol{x}',t') - \tilde{\mu}_{k'}(\boldsymbol{\alpha}',\boldsymbol{x}',t')\right) \frac{\hat{\delta}\nu_{k'}(\boldsymbol{\alpha}',\boldsymbol{x}',t')}{\hat{\delta}\mathcal{F}_k\left[\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\}\right]} \\ &= 0, \end{aligned} \tag{4.11}$$

where the moments μ are defined in (4.5), with $\tilde{\mu}$ having p replaced by $\tilde{p}$. Next, we consider well-mixed systems where the de-escalation from functionals to ordinary functions F_k makes this problem (4.11) well-defined. In Section 4.3.3, we parameterize the functional form of $\mathcal{F}$ to consider spatially distributed systems.

4.3 Analytic MaxEnt solutions

4.3.1 Well-mixed systems in one species

In the case of well-mixed systems in one species, the state of the system is entirely characterized by the number of individuals $|n,t\rangle$. Dropping the species and position labels in the dynamic Boltzmann distribution gives:

$$\tilde{p}(n,t) = \frac{1}{\mathcal{Z}\left(\{\nu\}\right)} \exp\left[-\sum_{k=1}^{K} \binom{n}{k} \nu_k(t)\right], \tag{4.12}$$

where we use the notation $\{\nu\} = \{\nu_{k'}\}_{k'=1}^{K}$. The time evolution is now described by basis functions forming the autonomous differential equation system:

$$\begin{aligned} \frac{d}{dt}\nu_k(t) &= F_k\left(\{\nu\}\right), \\ \text{with I.C.:} \quad \nu_k(t=0) &= \eta_k, \end{aligned} \tag{4.13}$$

where F_k are now functions rather than functionals $\mathcal{F}_k$. The variational problem (4.11) for the basis functions becomes:

$$\frac{\delta S[\{\nu[\{F\}]\}]}{\delta F_k(\{\nu\})} = \sum_{k'=1}^{K} \int_0^\infty dt' \left(\left\langle \binom{n}{k'} \right\rangle_{p(t')} - \left\langle \binom{n}{k'} \right\rangle_{\tilde{p}(t')} \right) \frac{\delta \nu_{k'}(t')}{\delta F_k(\{\nu\})} = 0, \tag{4.14}$$

where $\langle X \rangle_p (t') = \sum_{n=0}^{\infty} X p(n,t')$ and similarly for $\tilde{p}$.

The variational term on the RHS of (4.14) may be eliminated in adjoint method, as

discussed in Chapter 5. In this chapter, the direct sensitivity approach is used to evaluate the desired gradients. The equation for the sensitivities can determined by a number of methods, including by an ODE formulation derived in Appendix B.1.1, a PDE formulation derived from applying the chain rule at the initial condition, and using a Lie series approach (Appendix B.3.2). The first of these and arguably the most practical is:

$$\begin{aligned} \frac{d}{dt'}\left(\frac{\delta\nu_{k'}(t')}{\delta F_k(\{\nu\})}\right) =& \sum_{l=1}^{K} \frac{\partial F_{k'}(\{\nu(t')\})}{\partial \nu_l(t')} \frac{\delta\nu_l(t')}{\delta F_k(\{\nu\})} + \delta_{k',k}\delta(\{\nu\} - \{\nu(t')\}), \\ \text{with I.C.:} \quad \frac{\delta\nu_{k'}(t'=0)}{\delta F_k(\{\nu\})} =& 0. \end{aligned} \tag{4.15}$$

An algorithmic solution to (4.14) is therefore possible in the form of a PDE-constrained optimization problem: Solve (4.14,4.15) subject to the PDE-constraint (4.13). An example algorithm using simple gradient descent is given by Algorithm 1.

We note the implicit connection between this approach and using Boltzmann machines, such as in GCCD, by the algorithm's objective function. Here, the whole trajectory of moments from stochastic simulations is used to directly estimate time evolution operators, rather than estimating the interaction parameters at each time step. We make this connection explicit in Algorithm 2 in Section 4.4.3 below.

Further improvements to Algorithm 1 are possible, such as to replace ordinary gradient descent by an accelerated version, e.g. Nesterov accelerated gradient descent [46]. Furthermore, the wealth of methods available to solve PDE-constrained optimization problems, e.g. adjoint methods [47], offer rich possibilities for further development.

Example: Mean of the Galton-Watson Branching Process

As a simple illustrative example, consider a reduced model that captures the time-evolving mean of the Galton-Watson branching process, consisting of the birth process $A \to A + A$ with rate k_b and death $A \to \varnothing$ with rate k_d.

Algorithm 1 Gradient Descent for Learning Basis Functions Governing Well-Mixed Dynamics

Initialize
  Grid of values $\{\nu\}$ to solve over.
  $F_k(\{\nu\})$ for $k = 1, \ldots, K$.
  Max. integration time T.
  A formula for the learning rate λ.
while not converged **do**
  Initialize $\Delta F_k(\{\nu\}) = 0$ for all $k, \{\nu\}$.
  Generate a sample of random initial conditions $\{\eta\}$.
  for $\eta_i \in \{\eta\}$ **do**
    ▷ *Generate trajectory in reduced space* $\{\nu\}$*:*
    Solve the PDE constraint (4.13) with IC η_i for $0 \leq t \leq T$.
    Solve (4.15) for variational term $\delta\nu_{k'}(t)/\delta F_k(\{\nu\})$.
    ▷ *Sampling step:*
    Evaluate moments $\left\langle \binom{n}{k'} \right\rangle_{\tilde{p}(t')}$ of the Boltzmann distribution by sampling or analytically.
    Evaluate true moments $\left\langle \binom{n}{k'} \right\rangle_{p(t')}$ by stochastic simulation or analytic solution.
    ▷ *Evaluate the objective function:*
    Update $\Delta F_k(\{\nu\})$ as the cumulative moving average of (4.14) over initial conditions.
  ▷ *Update to decrease objective function:*
  Update $F_k(\{\nu\})$ to decrease the objective function: $F_k(\{\nu\}) \to F_k(\{\nu\}) - \lambda \Delta F_k(\{\nu\})$.

In this case, there are only self-interactions ($K = 1$) described by $\nu(t)$ with basis function $F(\nu(t))$. The dynamic Boltzmann distribution is:

$$\tilde{p}(n,t) = \frac{1}{\tilde{\mathcal{Z}}} \exp[-n\nu(t)]. \tag{4.16}$$

Using the fact that

$$\langle n \rangle_{\tilde{p}} = \frac{1}{e^{\nu} - 1} \tag{4.17}$$

and from the CME

$$\frac{d\langle n\rangle_p}{dt} = (k_b - k_d)\langle n\rangle_p \tag{4.18}$$

gives the analytic solution for the basis functions

$$F(\nu) = (k_b - k_d)(e^{-\nu} - 1). \tag{4.19}$$

This solution is reproduced using Algorithm 1, as shown in Figure 4.2 for $k_d = 3k_b/2$. Here, the solution is constructed on a grid of $\nu \in [0^+, 3.0]$ with spacing $\Delta\nu = 0.1$, with maximum integration time $T = 1$ (arbitrary units). The learning rate is decreased exponentially over iterations to improve convergence. The convergence of the algorithm is shown in Figure 4.3.

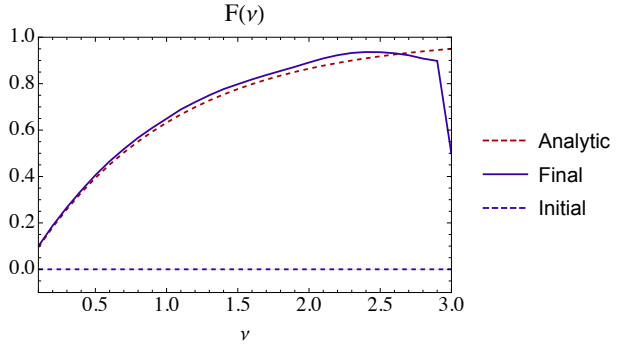

Figure 4.2: Learned basis function for the simple annihilation process $A \to \varnothing$ after 40 iterations, from a uniform initial condition.

Example: Two Basis Functions Controlling Mean and Variance

Consider again the process of the previous section, but with $K = 2$ basis functions $\nu_1(t)$ and $\nu_2(t)$ controlling the mean and variance in the number of particles. The dynamic Boltzmann distribution is:

$$\tilde{p}(n,t) = \frac{1}{\mathcal{Z}} \exp\left[-n\nu_1(t) - \binom{n}{2}\nu_2(t)\right]. \tag{4.20}$$

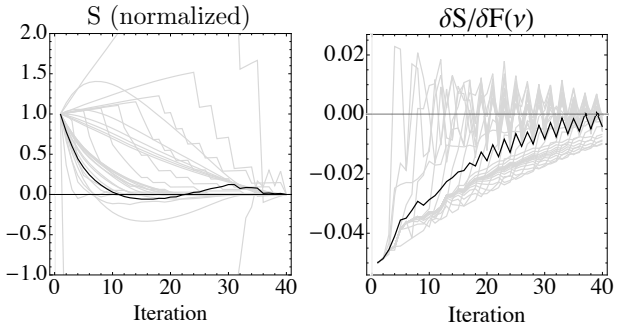

Figure 4.3: *Left:* The convergence of the action S as it is minimized over iterations following Algorithm 1. Trajectories (grey) for individual η, normalized to start at one and end at zero, and their mean (black). *Right:* The minimization of the variation in the action $\delta S/\delta F(\nu)$, as a function of the position to vary ν. Trajectories (grey) at each ν, and their mean (black).

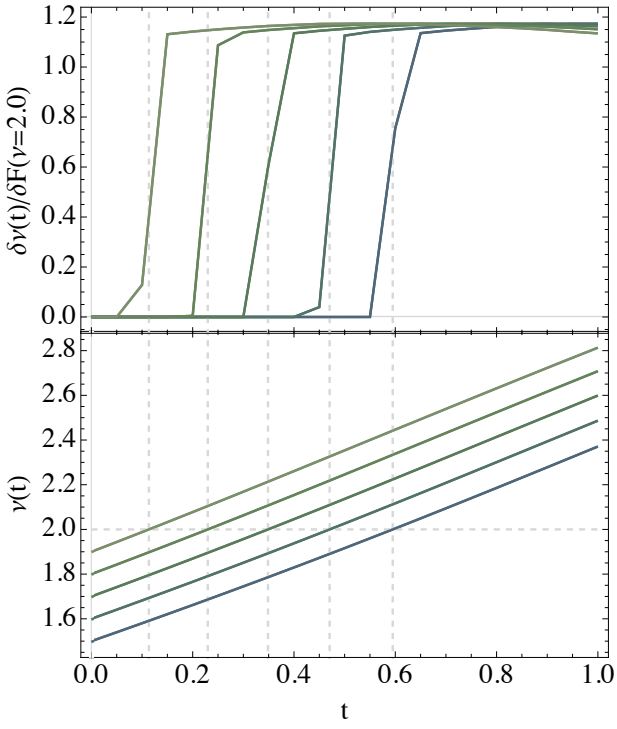

Figure 4.4: *Top:* The variational term $\delta\nu(t)/\delta F(\nu = 2.0)$ for several initial conditions $\eta = 1.5, \ldots, 1.9$ as a function of time, obtained by solving (4.15) numerically. *Bottom:* The solution trajectories $\nu(t)$ starting from these η. Only when the solution trajectory is close to $\nu(t) = 2.0$ and thereafter does varying the basis function F at this point have a non-zero effect.

This may be interpreted as a Gaußian distribution in the number of particles, provided we treat n as continuous and extend its range to $\pm\infty$, or consider systems with means far from $n = 0$. In this case, the mean μ and variance σ^2 can be related to the interaction functions as $\mu = 1/2 - \nu_1/\nu_2$ and $\sigma^2 = 1/\nu_2$. The differential equations derived from the CME for the moments of this system are:

$$\begin{aligned} \frac{d\mu}{dt} &= (k_b - k_d)\mu, \\ \frac{d\sigma^2}{dt} &= 2(k_b - k_d)\sigma^2 + (k_b + k_d)\mu, \end{aligned} \tag{4.21}$$

which can be converted to analytic solutions for the basis functions:

$$\begin{aligned} F_1(\nu_1, \nu_2) =& \nu_1 \left(k_d - k_b + (k_b + k_d)\nu_1\right) - \frac{\nu_2}{2}\left(k_b - k_d + (k_b + k_d)\nu_1\right), \\ F_2(\nu_1, \nu_2) =& -\frac{\nu_2}{2}\Big(k_d\left(-4 - 2\nu_1 + \nu_2\right) + k_b\left(4 - 2\nu_1 + \nu_2\right)\Big). \end{aligned} \tag{4.22}$$

These are shown in Figure 4.5.

Figure 4.6 shows the variational terms $\delta\nu_1(t)/\delta F_1(\nu_1, \nu_2)$ and $\delta\nu_2(t)/\delta F_1(\nu_1, \nu_2)$, resulting from Algorithm 1 and determined by (4.15). Interestingly, the self-varying term $\delta\nu_1(t)/\delta F_1(\nu_1, \nu_2)$ more closely resembles the multivariate delta-function appearing in (4.15), while the cross term $\delta\nu_2(t)/\delta F_1(\nu_1, \nu_2)$ shows a greater temporal memory of the solution trajectory.

4.3.2 Solvable systems

We next consider special cases where analytic solutions for the basis functionals are possible, to motivate a parameterization leading to a solvable version of the variational problem (4.11).

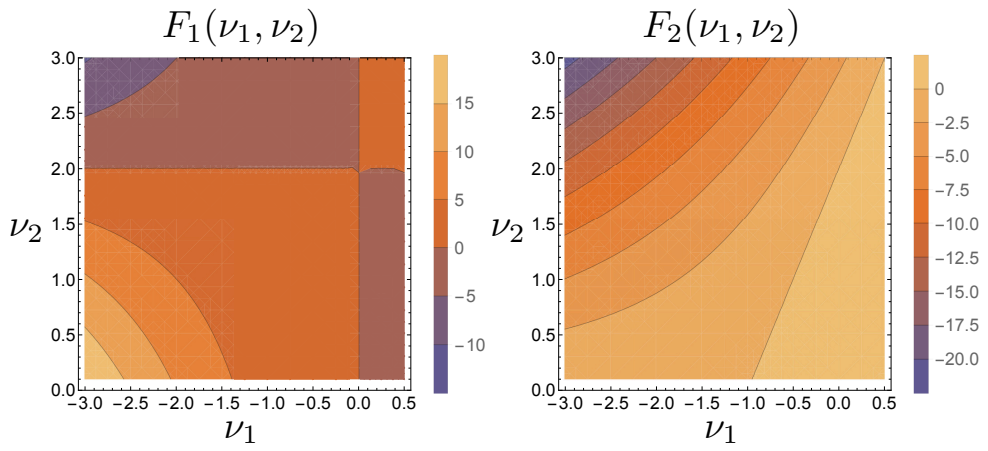

Figure 4.5: The true basis functions (4.22) for the Galton-Watson system. *Left:* $F_1(\nu_1,\nu_2)$. *Right:* $F_2(\nu_1,\nu_2)$. The reaction rates used are $k_d = 3k_b/2 = 1.5$.

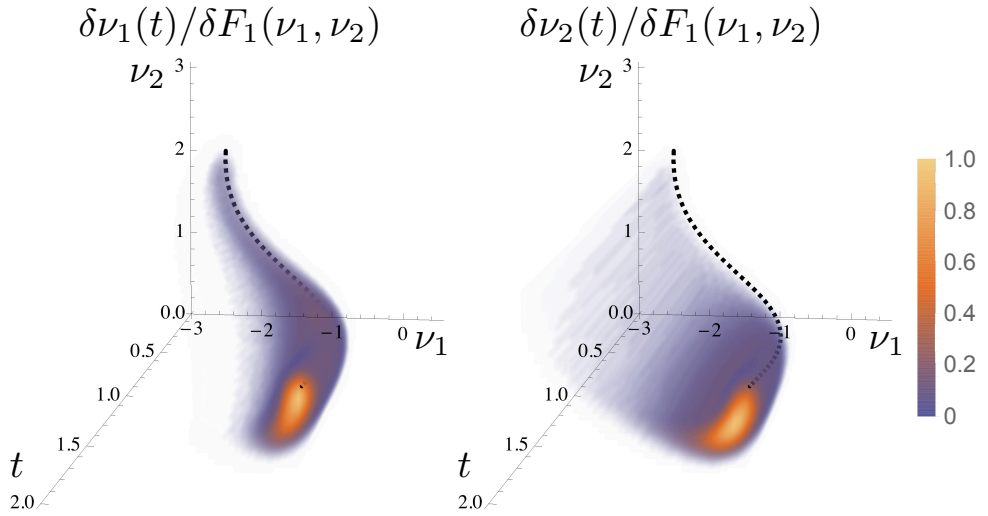

Figure 4.6: Two of the four variational terms for the Galton-Watson system using two basis functions. *Left:* $\delta\nu_1(t)/\delta F_1(\nu_1,\nu_2)$. *Right:* $\delta\nu_2(t)/\delta F_1(\nu_1,\nu_2)$. The black dashed line shows the solution trajectory. The initial conditions are $(\eta_1,\eta_2) = (-2.5, 2.0)$, with reaction rates $k_d = 3k_b/2 = 1.5$. The effect of the mixing term $\delta\nu_2(t)/\delta F_1(\nu_1,\nu_2)$ is lower in magnitude but persists longer over time. Note that the absolute magnitude of the spread is related to the approximation chosen for the delta function in (4.15), a normalized multivariate Gaußian with variance of 0.1 in both directions ν_1,ν_2.

Gaußian Distributions

The well-mixed case (4.12) is the MaxEnt distribution consistent with $\left\langle \binom{n}{k} \right\rangle_p$ for $k = 1,\ldots,K$. If $K = 2$, then (4.12) may be interpreted as a Gaußian distribution in

continuous n, as discussed in the previous section. Generalizing these results, the basis functions are generally given by:

$$\begin{aligned} F_1(\nu_1,\nu_2) &= -\nu_2\frac{d\mu}{dt} - \nu_1\nu_2\frac{d\sigma^2}{dt}, \\ F_2(\nu_1,\nu_2) &= -\nu_2^2\frac{d\sigma^2}{dt}, \end{aligned} \tag{4.23}$$

where $d\mu/dt$, $d\sigma^2/dt$ are evaluated from the CME and expressed in terms of ν_1, ν_2. Here, a moment closure approximation must be applied if the reactions are greater than unimolecular in number of reagents. For example, the higher order moments appearing in the CME may be approximated by those of the reduced model $\tilde{p}$ and expressed in terms of lower order μ, σ^2 following the well known property of Gaußian distributions. This closure technique is described further in Section 4.4.1.

Diffusion from Point Source

In the spatial case, consider a diffusion process of a fixed number of particles n with diffusion constant D spreading out from a point source at x_0. The analytic solution to the CME is:

$$p(\boldsymbol{x},t) = (4\pi Dt)^{-n/2}\exp\left[-\sum_{i=1}^{n}\frac{(x_i-x_0)^2}{4Dt}\right], \tag{4.24}$$

reflecting that only self interactions ($K = 1$) are necessary to describe the process. The reduced model (4.2) becomes:

$$\tilde{p}(\boldsymbol{x},t) = \frac{1}{\mathcal{Z}}\exp\left[-\sum_{i=1}^{n}\overline{\nu}(x_i,t)\right]. \tag{4.25}$$

It is straightforward to verify that $p(\boldsymbol{x},t) = \tilde{p}(\boldsymbol{x},t)$ if

$$\overline{\nu}(y,t) = \ln\left(1+\frac{1}{n}\right) + \frac{1}{2}\ln(4\pi Dt) + \frac{(y-x_0)^2}{4Dt}. \tag{4.26}$$

Consequentially, from $\partial_t \overline{\nu}(y,t)$, the basis functional is:

$$F[\overline{\nu}(y,t)] = D\partial_y^2 \overline{\nu}(y,t) - D\left(\partial_y \overline{\nu}(y,t)\right)^2. \tag{4.27}$$

Unimolecular Reaction-Diffusion

For reaction networks that involve only diffusion and unimolecular reactions, two key properties hold for the CME solution:

1. Separable spatial and particle number distributions $p(n,\boldsymbol{x}) = p(n)p(\boldsymbol{x})$ where each distribution is normalized $\sum_n p(n) = 1$ and $\int d\boldsymbol{x}\, p(\boldsymbol{x}) = 1$.

2. Independence of spatial distribution $p(\boldsymbol{x}) = p(x_1)p(x_2)\dots p(x_n)$ where each $\int dx\, p(x) = 1$ is normalized. This assumes that initial $p(x_i)$ are independent - otherwise, a fixed mixture of independent components must be considered.

Analogous to the purely diffusive process above, this allows analytic solutions to the inverse Ising problem by imposing these conditions upon the dynamic Boltzmann distribution $\tilde{p}$. Here, we exploit the fact that multiplication of Boltzmann distributions results in addition of the energy functions.

Introduce a single interaction function $\overline{\nu}(x,t)$ to capture the diffusion process and the usual $\nu_1(t),\dots,\nu_K(t)$ to describe the reactions (for brevity, omit further time arguments in this section). Furthermore, impose the normalization $\int dx \exp[-\overline{\nu}(x)] = 1$. The dynamic Boltzmann distribution becomes:

$$\begin{aligned}
\tilde{p}(n,\boldsymbol{x}) &= \tilde{p}(n)\tilde{p}(\boldsymbol{x}) = \tilde{p}(n)\tilde{p}(x_1)\tilde{p}(x_2)\dots\tilde{p}(x_n), \\
\tilde{p}(n) &= \frac{1}{\mathcal{Z}} \exp\left[-\sum_{k=1}^{K} \binom{n}{k} \nu_k\right], \\
\tilde{p}(x) &= \exp[-\overline{\nu}(x)],
\end{aligned} \tag{4.28}$$

where the partition function is

$$\mathcal{Z} = \sum_n \int d\boldsymbol{x}\, \tilde{p}(n, \boldsymbol{x}) = \sum_n \tilde{p}(n). \tag{4.29}$$

The distribution $\tilde{p}(n, \boldsymbol{x})$ is the MaxEnt distribution consistent with the moments $\langle \binom{n}{k} \rangle$ for all $k = 1, \ldots, K$, as well as the spatial moment:

$$\left\langle \sum_{i=1}^{n} \delta(y - x_i) \right\rangle = \sum_n \int d\boldsymbol{x} \sum_{i=1}^{n} \delta(y - x_i) \tilde{p}(n, \boldsymbol{x}) = \exp[-\overline{\nu}(y)] \langle n \rangle, \tag{4.30}$$

such that the solution to the inverse Ising problem is:

$$\overline{\nu}(y) = \ln \left(\frac{\langle n \rangle}{\langle \sum_{i=1}^{n} \delta(y - x_i) \rangle} \right). \tag{4.31}$$

The solution for the inverse Ising problem for $\langle \binom{n}{k} \rangle$ is independent of this spatial moment, and analytically possible e.g. for $K = 1$ or 2, as demonstrated in Sections 4.3.1 and 4.3.2 above.

Taking the time derivatives of these solutions $\dot{\overline{\nu}}$ and $\dot{\nu_k}$ and using the CME to derive differential equations for the moments gives the basis functionals. For unimolecular reactions, the diffusion process does not affect the reactions, such that the functional controlling $\overline{\nu}$ is always that of diffusion (4.27). For example, for a branching random walk consisting of diffusion from a point source and the Galton-Watson process with $K = 2$, the basis are the functional (4.27) and the functions (4.22).

4.3.3 Parameterizations for Spatially Heterogeneous Systems

For reaction-diffusion systems that involve reactions greater than unimolecular in number of reagents, it generally becomes difficult to analytically solve the inverse Ising problem and consequentially identify basis functionals. However, an algorithmic solution

remains possible, where we guess a local parameterization of the functional (4.3) based on the analytic solutions presented above.

Let $\boldsymbol{\beta}, \boldsymbol{y}$ be of length k, and use the notation

$$\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\} = \left\{ \nu_{k'}(\boldsymbol{\beta}_{\langle i \rangle_{k'}^{k}}, \boldsymbol{y}_{\langle i \rangle_{k'}^{k}}, t) \; \forall \; \langle i \rangle_{k'}^{k} : 1 \leq k' \leq k \right\}. \tag{4.32}$$

Then choose the *spatially local* parameterization of $\mathcal{F}_k$ in (4.3):

$$\begin{aligned} \frac{d}{dt}\nu_k(\boldsymbol{\beta}, \boldsymbol{y}, t) = & F_k^{(0)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\}) + \sum_{\lambda=1}^{k} \Bigg(F_k^{(1,\lambda)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\}) \sum_{\langle i \rangle_{\lambda}^{k}} \sum_{m=1}^{\lambda} \left(\partial_m \nu_\lambda(\boldsymbol{\beta}_{\langle i \rangle_{\lambda}^{k}}, \boldsymbol{y}_{\langle i \rangle_{\lambda}^{k}}, t) \right)^2 \\ & + F_k^{(2,\lambda)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\}) \sum_{\langle i \rangle_{\lambda}^{k}} \sum_{m=1}^{\lambda} \partial_m^2 \nu_\lambda(\boldsymbol{\beta}_{\langle i \rangle_{\lambda}^{k}}, \boldsymbol{y}_{\langle i \rangle_{\lambda}^{k}}, t) \Bigg), \end{aligned} \tag{4.33}$$

with initial condition $\nu_k(\boldsymbol{\beta}, \boldsymbol{y}, t = 0) = \eta_k(\boldsymbol{\beta}, \boldsymbol{y})$. Here, ∂_m denotes the derivative with respect to the m-th component of $\boldsymbol{y}_{\langle i \rangle_{\lambda}^{k}}$, and $F_k^{(\gamma)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\})$ for $(\gamma) = (0), (1, \lambda), (2, \lambda)$ are local functions, i.e. functions of the arguments on the left hand side of (4.33).

The variational problem (4.11) now becomes

$$\frac{\delta S[\{\nu[\{F\}]\}]}{\delta F_k^{(\gamma)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y})\})} = \sum_{k'=1}^{K} \sum_{\boldsymbol{\beta}'} \int d\boldsymbol{y}' \int dt' \left(\mu_{k'}(\boldsymbol{\beta}', \boldsymbol{y}', t') - \tilde{\mu}_{k'}(\boldsymbol{\beta}', \boldsymbol{y}', t') \right) \frac{\delta \nu_{k'}(\boldsymbol{\beta}', \boldsymbol{y}', t')}{\delta F_k^{(\gamma)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y})\})} = 0 \tag{4.34}$$

for $(\gamma) = (0), (1, \lambda), (2, \lambda)$, where $\boldsymbol{\beta}', \boldsymbol{y}'$ are of length k'.

Analogously to the well-mixed case, it is possible to derive a PDE system governing the variational term $\delta \nu_{k'}(\boldsymbol{\beta}', \boldsymbol{y}', t') / \delta F_k^{(\gamma)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y})\})$. In Appendix B.1.2, an illustrative example is derived for a diffusion process.

Equations (4.33,4.34) together form a PDE-constrained optimization problem, which may be solved analogously to Algorithm 1, with additional spatial axes.

Example: Branching Random Walk

Consider a branching random walk consisting of the Galton-Watson process and diffusion from a point source with rate D in one spatial dimension and one species. From the true solutions for the basis functionals (4.27,4.22), use one spatial interaction function $\overline{\nu}(y,t)$ and two purely temporal $\nu_1(t), \nu_2(t)$, and further restrict the parameterization (4.33) of the basis functionals to be

$$\begin{aligned} \frac{d\overline{\nu}(y,t)}{dt} =& \overline{F}[\overline{\nu}(y,t)] = \overline{F}^{(1)}(\overline{\nu}(y,t)) \left(\partial_y \overline{\nu}(y,t)\right)^2 + \overline{F}^{(2)}(\overline{\nu}(y,t)) \partial_y^2 \overline{\nu}(y,t), \\ \frac{d\nu_k(t)}{dt} =& F_k[\nu_1(t), \nu_2(t)] = F_k^{(0)}(\nu_1(t), \nu_2(t)) \end{aligned} \tag{4.35}$$

for $k = 1, 2$. The variational problem is

$$\frac{\delta S}{\delta \overline{F}^{(\gamma)}(\overline{\nu})} = \int dy' \int dt' \, \left(\mu_1(y',t') - \tilde{\mu}_1(y',t')\right) \frac{\delta \overline{\nu}(y',t')}{\delta \overline{F}^{(\gamma)}(\overline{\nu})} = 0, \tag{4.36}$$

$$\frac{\delta S}{\delta F_k^{(0)}(\nu_1,\nu_2)} = \sum_{k'=1}^{2} \int dt' \, \left(\left\langle \binom{n}{k'} \right\rangle_{p(t')} - \left\langle \binom{n}{k'} \right\rangle_{\tilde{p}(t')} \right) \frac{\delta \nu_{k'}(t')}{\delta F_k^{(0)}(\nu_1,\nu_2)} = 0 \tag{4.37}$$

for $\gamma = 1, 2$.

Differential equations governing the variational terms $\delta\overline{\nu}(y',t')/\delta\overline{F}^{(\gamma)}(\overline{\nu})$ for $\gamma = 1,2$ are derived in Appendix B.1.2, given by (B.15). Differential equations governing $\delta\nu_{k'}(t')/\delta F_k^{(0)}(\nu_1,\nu_2)$ are given by (4.15).

The optimization problem (4.36,4.37) subject to the PDE-constraints (4.35) may be solved algorithmically using Algorithm 1 in each $F_k, \overline{F}^{(1)}, \overline{F}^{(2)}$, analogously to the well-mixed case. We note that the true solutions are given by (4.27,4.22), in particular: $\overline{F}^{(1)} = D$ and $\overline{F}^{(2)} = -D$.

Figure 4.7 plots the spatial variational terms resulting from the true basis functionals. Here, the reaction rates used are as before $k_d = 3k_b/2 = 1.5$, with a diffusion constant of $D = 1$. Contrary to the well-mixed case, these terms do not resemble step functions, but

rather exhibit some extended temporal dynamics.

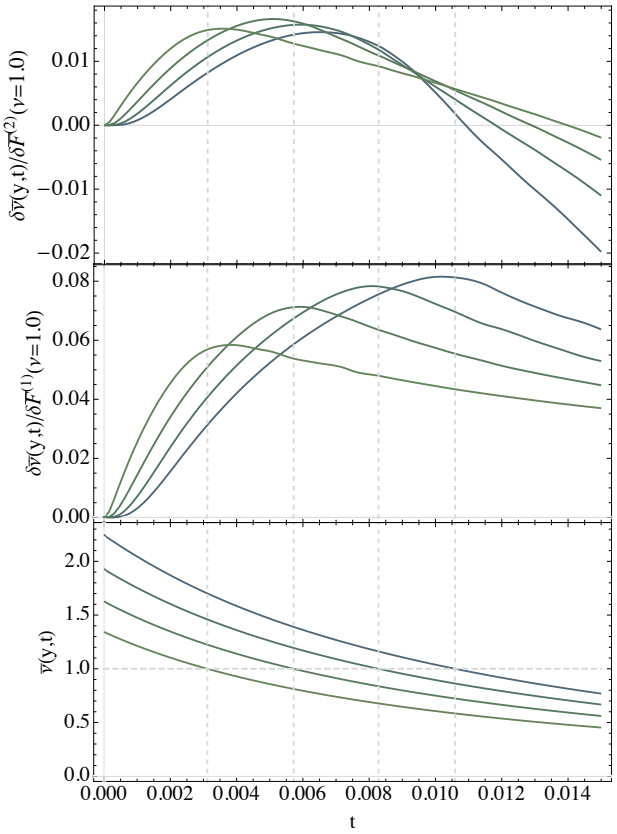

Figure 4.7: Branching random walk with diffusion in 1D estimated by Algorithm 1. *Top:* The variational term $\delta\overline{\nu}(y,t)/\delta\overline{F}^{(2)}(\nu=1.0)$ as a function of time at several spatial locations $y=0.25,0.5,0.75,1$. Here, $\overline{F}^{(1)}=D,\overline{F}^{(2)}=-D$ are the true solutions. *Middle:* $\delta\overline{\nu}(y,t)/\delta\overline{F}^{(1)}(\nu=1.0)$. *Bottom:* The solution trajectories $\overline{\nu}(y,t)$, starting from a point source. Contrary to the well-mixed case, the variational terms do not resemble step functions at $\nu=1.0$, but rather exhibit some extended temporal dynamics.

4.4 Lattice systems in 1D

The PDE-constrained optimization problems above are the general solution for finding the basis functionals that govern the time evolution of the reduced MaxEnt model. Here, we present a more efficient machine learning approach for learning the basis functions from the solutions of simple, analytically solvable models. In Section 4.4.1, we present a

method for finding such analytic solutions in the discrete lattice limit, and present examples for a variety of simple processes in Section 4.4.2. In Section 4.4.3, we demonstrate the utility of using such analytic solutions in a Boltzmann machine-like learning algorithm, and further in Section 4.4.4 to learn non-linear combinations of solutions using artificial neural networks (ANNs).

4.4.1 Mapping to Spin Glass Systems in 1D

At low particle densities, a feasible model of a reaction-diffusion system in one spatial dimension and one species is that of a 1D lattice in the single occupancy limit. Let the spin values occupying each lattice site be $s_i \in \{0,1\}$, for all $i = 1,\ldots,N$, denoting the absence or presence of a particle.

The reduced model (4.2) now becomes the discrete analogue. We note that this model is consistent with the continuous version in some parameter regime where the separation between molecules is large compared to the interaction radius. By including only self-interactions described by an interaction function $h(t)$, and two particle nearest-neighbor interactions $J(t)$, we obtain the well known Ising model, with partition function:

$$\mathcal{Z} = \sum_{\{s\}} \exp\left[h(t) \sum_{i=1}^{N} s_i + J(t) \sum_{i=1}^{N-1} s_i s_{i+1} \right]. \tag{4.38}$$

This may be evaluated explicitly using the standard transfer matrix method. In the thermodynamic limit, $\ln \mathcal{Z} \approx \lambda_+^N$ is analytically accessible, where λ_+ is the largest eigenvalue of the transfer matrix.

The inverse Ising problem has the solution:

$$\begin{pmatrix} \left\langle \sum_{i=1}^{N} s_i \right\rangle (t) \\ \left\langle \sum_{i=1}^{N-1} s_i s_{i+1} \right\rangle (t) \end{pmatrix} = \begin{pmatrix} \partial_h \ln \mathcal{Z} \\ \partial_J \ln \mathcal{Z} \end{pmatrix}. \tag{4.39}$$

Taking the derivatives of both sides of (4.39)

$$\begin{pmatrix} \frac{d}{dt}\left\langle \sum_{i=1}^{N} s_i \right\rangle \\ \frac{d}{dt}\left\langle \sum_{i=1}^{N-1} s_i s_{i+1} \right\rangle \end{pmatrix} = \begin{pmatrix} \partial_h^2 \ln \mathcal{Z} & \partial_h \partial_J \ln \mathcal{Z} \\ \partial_h \partial_J \ln \mathcal{Z} & \partial_J^2 \ln \mathcal{Z} \end{pmatrix} \begin{pmatrix} \frac{dh}{dt} \\ \frac{dJ}{dt} \end{pmatrix}. \tag{4.40}$$

The time derivatives of the moments on the left may be obtained directly from the CME $\dot{p} = Wp$ using the Doi-Peliti formalism described in Chapter 2. If the system is *linear*, these may be expressed further in terms of h, J using (4.39), and the basis functions are given directly by inverting (4.40). If the system is *non-linear*, the presence of a moment hierarchy requires an approximation in the form of a moment closure technique. Here, we choose to express the higher order moments that appear through the CME in terms of h, J, which is possible for any higher order moment since the partition function (4.38) is analytically accessible. As a result of inverting (4.40):

$$\begin{pmatrix} \tilde{F}_h(h,J) \\ \tilde{F}_J(h,J) \end{pmatrix} = \begin{pmatrix} \partial_h^2 \ln \mathcal{Z} & \partial_h \partial_J \ln \mathcal{Z} \\ \partial_h \partial_J \ln \mathcal{Z} & \partial_J^2 \ln \mathcal{Z} \end{pmatrix}^{-1} \times \begin{pmatrix} \frac{d}{dt}\left\langle \sum_{i=1}^{N} s_i \right\rangle \\ \frac{d}{dt}\left\langle \sum_{i=1}^{N-1} s_i s_{i+1} \right\rangle \end{pmatrix}, \tag{4.41}$$

where the RHS has been expressed in terms of h, J as described above, and we use the notation $\tilde{F}_h, \tilde{F}_J$ to indicate that these are generally only approximations to the true basis functions F_h, F_J, and only exact for systems with closed moments. Effectively, we have replaced the probability distribution p in the CME $\dot{p} = Wp$ by the dynamic Boltzmann distribution $\tilde{p}$, and evaluated the effect of the operator on the RHS on this new distribution. The analytic solution to the 1D inverse Ising problem therefore provides an elegant approach to moment closure (see Ref. [10],[25] for related MaxEnt approaches to moment closure). Similar extensions to 2D Ising models [48] are likewise possible, and possibly to 3D as well [49].

Furthermore, we note that analogous to the continuous case proven in Proposition 1,

the linearity of reaction operators in the CME extends to the basis function approximations $\tilde{F}$ (regardless of whether $\mathcal{Z}$ is analytically accessible as in the 1D case). This requires that the inverse Ising problem has not changed, as discussed further in Section 4.4.3.

4.4.2 Analytic Approximations to Basis Functions of Simple Reaction Motifs

Figure 4.8 shows the basis function approximations calculated using the 1D Ising model (4.41) for several simple unimolecular reaction processes. Note that the reaction rates/diffusion constant provide an overall multiplicative factor to each process. Computer algebra systems can be used to determine these analytic forms, which contain sums on the order of ten to a hundred terms in length, depending on the operator.

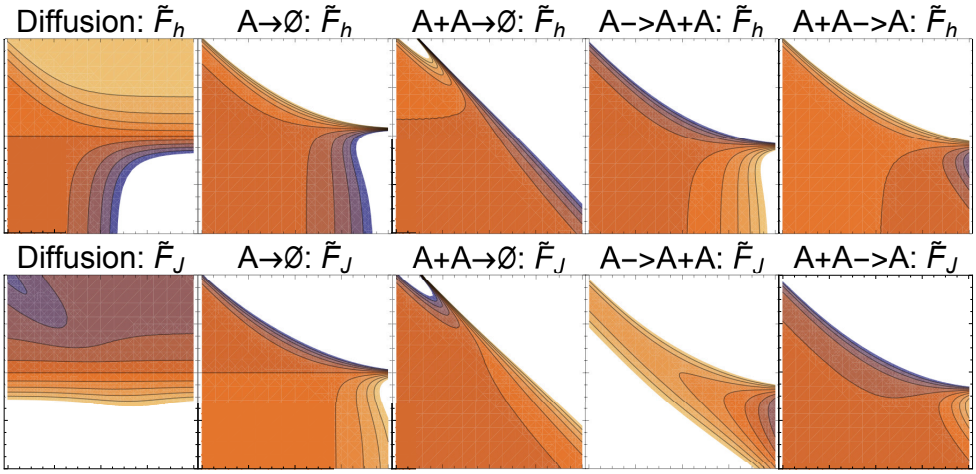

Figure 4.8: Basis functions (4.41) for several simple reaction schemes in one species. Horizontal, vertical axis: $h, J \in [-4, 4]$. The magnitudes have been scaled to $[-1, 1]$, since the reaction rate/diffusion constant provides an arbitrary scaling factor.

Generalizing these simple systems, we solve for the basis function approximations of the trivalent reaction $A + B \to C$ with its reverse process $C \to A + B$. This process is fundamentally important as a generalization of many simple biochemical processes, and has been studied extensively [50, 51]. For example, it is the building block of the broadly

applicable substrate-enzyme-product (SEP) motif $S+E \rightleftharpoons C \to P+E$, where S,E,P denote the substrate, enzyme, and product (see Section 4.4.4 below).

In the Ising model formalism, the description of this process involves 9 time dependent interaction functions $h_A, h_B, h_C, J_{AA}, J_{AB}, J_{AC}, J_{BB}, J_{BC}, J_{CC}$, forming the reduced model:

$$\mathcal{Z} = \sum_{\{s\}} \sum_{\{\alpha\}} \exp\left[\sum_{i=1}^{N} h_{\alpha_i}(t) s_i + \sum_{i=1}^{N-1} J_{\alpha_i,\alpha_{i+1}}(t) s_i s_{i+1} \right] \tag{4.42}$$

where the species label $\alpha_i \in \{A,B,C\}$, and we implicitly note that the sum $\sum_{\{\alpha\}}$ runs only over occupied sites $s_i = 1$. Figure 4.9 shows several 2D slices of three of the nine basis function approximations for the forward process $A+B \to C$.

By including species labels, (4.41) leads to analytic expressions containing on the order of hundreds of terms. Here, we used a numerical strategy as described in Appendix B.2 for evaluating the basis functions over the chosen domain. While a computer algebra system may be employed as before, this strategy is computationally faster.

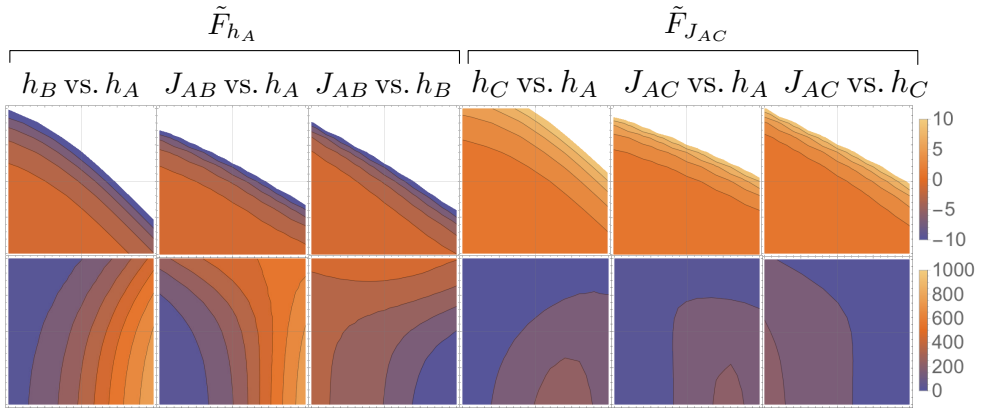

Figure 4.9: Basis function approximations $\tilde{F}_{h_A}, \tilde{F}_{J_{AC}}$ corresponding to the forward trivalent reaction $A+B \to C$ with rate $k=1$. Each is a 9 dimensional function, of which 2D slices are shown, holding all other parameters at zero. The top row shows the basis function, while the bottom row shows the corresponding moments controlled by these parameters h_A, J_{AC}. The chain length used is $N = 1000$. The ranges for all horizontal, vertical axes are $[-2,2]$.

4.4.3 Boltzmann Machine-Style Learning Algorithm for Dynamics

The basis function approximations derived above constitute a space of possible reduced dynamics. Here, we consider using these analytic insights to describe large spatially distributed reaction networks in 1D. This approach faces two key problems:

1. For non-linear systems, $\tilde{p}$ obeying (4.41) will over time diverge from the MaxEnt distribution consistent with the CME moments due the moment closure approximation made. As a fundamental consequence of this moment hierarchy, it is not possible to find exact basis functions over the *entire* interaction parameter space (e.g. h, J). Another way to see this is that trajectories of the CME system will intersect in h, J space.

   However, we postulate that it may be possible to learn approximately well the basis functions for a *single trajectory* (from a single initial condition) which does not self-intersect over some domain. This model may be used for extrapolation with reasonable accuracy close to the stochastic trajectory.

2. For large reaction networks, the basis functions are generally not linear in the basis functions of individual processes because the collection of interaction functions is not fixed, violating the assumption in Proposition 1. For example, consider the process $A \to B \to C$. Here, nine basis functions are required to capture all means and nearest neighbor correlations, such that (4.41) is nine dimensional. Denote these by $\boldsymbol{\beta} = A^{-1}\boldsymbol{m}$ where $\boldsymbol{\beta}$ denotes the basis functions, $\boldsymbol{m}$ the time evolving moments, and A the matrix of partition function derivatives.

   Next, consider the separate processes $A \to B$ and $B \to C$, described by five basis functions each. Let these be denoted by $\boldsymbol{\beta}^{(r)} = (A^{(r)})^{-1}\boldsymbol{m}^{(r)}$ for each of the two reactions r. Clearly, not all nine basis functions in $\boldsymbol{\beta}$ are present in each $\boldsymbol{\beta}^{(r)}$.

Furthermore, for those that are present in both, it is not necessarily true that the i-th basis function is expressible as $\boldsymbol{\beta}_i \neq \boldsymbol{\beta}_j^{(1)} + \boldsymbol{\beta}_k^{(2)}$ for appropriate j, k.

Generally, a reaction network involves more interaction parameters than each of the individual processes, such that Proposition 1 does not apply. It is only for a subset of networks, such as reaction networks in one species, where the linearity in the CME extends *exactly* to the basis functions. Regardless, we postulate that many networks may be described *approximately* well by linear combinations of basis functions corresponding to individual processes.

In light of these postulates, we return to the variational problem (4.14) and its PDE-constraint. In the discrete lattice case considered in Section 4.4.1, it becomes for each $\gamma = h, J$:

$$\int_0^\infty dt' \, \left(\tilde{\mu}(t') - \mu(t')\right) \frac{\delta h(t')}{\delta F_\gamma(h, J)} + \int_0^\infty dt' \, \left(\tilde{\Delta}(t') - \Delta(t')\right) \frac{\delta J(t')}{\delta F_\gamma(h, J)} = 0, \tag{4.43}$$

where we have used the notation μ, Δ to denote the average number of particles, nearest neighbors (NN) over p, and similarly $\tilde{\mu}, \tilde{\Delta}$ to denote averages over $\tilde{p}$.

Here, we exploit the analytic results derived above to simplify this problem and derive an efficient Boltzmann-machine type learning algorithm for the dynamics. In particular, we assume that the true basis functions are linear combinations of the approximations derived in Section 4.4.2 above, given by:

$$\begin{aligned} \frac{dh}{dt} &= F_h(h, J) = \sum_r \theta^{(r)} \tilde{F}_h^{(r)}, \\ \frac{dJ}{dt} &= F_J(h, J) = \sum_r \theta^{(r)} \tilde{F}_J^{(r)}. \end{aligned} \tag{4.44}$$

Here, the reaction rates and diffusion constant are all set to unity, such that the coefficients θ indicate the rates. The variational problem now turns into a regular optimization problem

for the coefficients θ that will yield at all times the MaxEnt distribution consistent with the CME moments. The optimization problem becomes: Subject to the PDE constraint (4.44), solve:

$$\int_0^\infty dt' \left(\tilde{\mu}(t') - \mu(t')\right) \frac{\partial h(t')}{\partial \theta^{(s)}} + \int_0^\infty dt' \left(\tilde{\Delta}(t') - \Delta(t')\right) \frac{\partial J(t')}{\partial \theta^{(s)}} = 0, \tag{4.45}$$

where the derivative terms are given by the solution to the ordinary differential equation system

$$\begin{aligned} \frac{\partial}{\partial t'}\left(\frac{\partial h(t')}{\partial \theta^{(s)}}\right) =& \tilde{F}_h^{(s)} + \frac{\partial h(t')}{\partial \theta^{(s)}} \sum_r \theta^{(r)} \frac{\partial \tilde{F}_h^{(r)}}{\partial h} + \frac{\partial J(t')}{\partial \theta^{(s)}} \sum_r \theta^{(r)} \frac{\partial \tilde{F}_h^{(r)}}{\partial J}, \\ \frac{\partial}{\partial t'}\left(\frac{\partial J(t')}{\partial \theta^{(s)}}\right) =& \tilde{F}_J^{(s)} + \frac{\partial h(t')}{\partial \theta^{(s)}} \sum_r \theta^{(r)} \frac{\partial \tilde{F}_J^{(r)}}{\partial h} + \frac{\partial J(t')}{\partial \theta^{(s)}} \sum_r \theta^{(r)} \frac{\partial \tilde{F}_J^{(r)}}{\partial J}, \end{aligned} \tag{4.46}$$

with initial condition $\partial h(0)/\partial \theta^{(s)} = \partial J(0)/\partial \theta^{(s)} = 0$.

Parameter estimation is greatly simpler to solve than the function estimation (4.43). Furthermore, the variational problem (4.46) is significantly simplified, since $\tilde{F}^{(r)}$ and consequentially its derivatives are analytically accessible. We capitalize upon these practical qualities in Algorithm 2, which solves this problem in a Boltzmann-machine learning style approach.

As an illustrative example, we apply Algorithm 2 to a branching and annihilating random walk (BARW) on a 1D lattice, described by the three processes: $A \to A + A$ with rate $k_b = 10$, $A + A \to 0$ with rate $k_a = 10$, and diffusion with constant $D = 10$. Extensive theoretical work has been dedicated to studying BARWs in the context of universality classes, in particular the directed percolation universality class [52, 53].

Stochastic simulations are used to generate training data for this system on a chain of length $N = 100$ for maximum time of $T = 1$ with timestep $dt = 0.01$. Here, we follow the numerical procedure described in Ref. [52]. The basis functions used in (4.44) are those of

Algorithm 2 Boltzmann Machine-Style Learning of Dynamics

Initialize
  Initial $\theta^{(r)}$ for all r.
  Max. integration time T.
  A formula for the learning rate λ.
  Time-series of lattice spins $\{s\}(t)$ from stochastic simulations from some known IC h_0, J_0.
  Fully visible MRF with NN connections and as many units as lattice sites N.
while not converged **do**
  ▷ *Generate trajectory in reduced space:*
  Solve the PDE constraint (4.44) with IC h_0, J_0 for $0 \leq t \leq T$.
  ▷ *Awake phase:*
  Evaluate true moments $\mu(t), \Delta(t)$ from the stochastic simulation data $\{s\}(t)$.
  ▷ *Asleep phase:*
  Evaluate moments $\tilde{\mu}(t), \tilde{\Delta}(t)$ of the Boltzmann distribution by Gibbs sampling.
  ▷ *Update to decrease objective function:*
  Solve (4.46) for derivative terms.
  Update $\theta^{(s)}$ to decrease the objective function for all s by taking: $\theta^{(s)} \to \theta^{(s)} - \lambda \times (4.45)$.

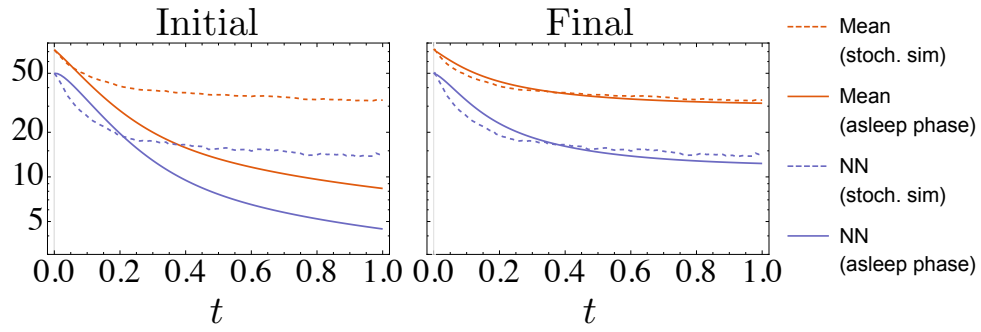

Figure 4.10: The 1st and 2nd (mean and NN) moments of the BARW system obtained from stochastic simulation (dashed) and by integrating the PDE constraint (4.44) and using Gibbs sampling in the asleep phase of Algorithm 2 (solid). *Left:* using initial $\theta_0^{(s)}$ reveals the limitations of moment closure approximation. *Right:* after 400 iterations, the coefficients have adjusted to more accurately capture the true CME dynamics.

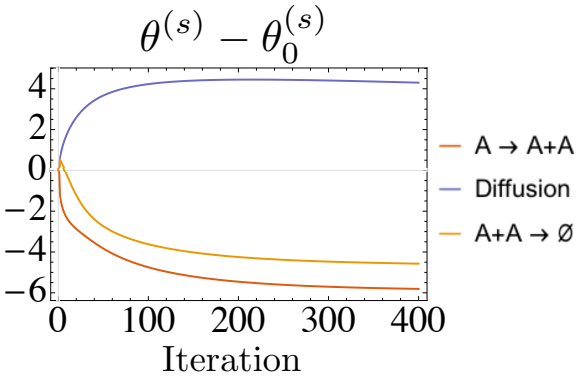

Figure 4.11: The coefficients in the PDE constraint converging over 400 iterations of Algorithm 2 applied to the BARW system, starting from the parameters used in the stochastic simulations.

the three processes present, as shown in Figure 4.8. The initial coefficients $\theta_0^{(s)}$ used are the known reaction rates.

Figure 4.10 shows the moments of the BARW system. Due to the moment closure problem, the system predicted by solving the constraint equations diverges from the true, even though the true reaction rates are used as coefficients $\theta_0^{(s)}$. After running 400 iterations of Algorithm 2, the new coefficients lead to much closer agreement to the true system.

Figure 4.11 shows the coefficients converge over the iterations. In particular, the effective rates for bimolecular annihilation and branching have decreased, while the effective diffusion constant has increased. Since the final values are sensitive to the initial $\theta_0^{(s)}$ chosen, an L_2 regularization term is included in the action. A further constraint in Algorithm 2 to keep $\theta^{(s)}$ positive enforces the connection to effective reaction rates.

4.4.4 Learning Non-linear Combinations of Basis Functions

As a more general approach than linear combinations, we use ANNs (artificial neural networks) to describe non-linear combinations of basis functions. Consider the SEP system

diffusing on a 1D lattice, described by the reactions:

$$S + E \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \xrightarrow{k_2} P + E. \tag{4.47}$$

The full Ising model for this system consists of four self interactions and 10 NN coupling parameters.

Figure 4.12 shows several moments of this system evolving in time from stochastic simulations. Here, the parameters used are: $k_1 = 10, k_{-1} = 0.1, k_2 = 0.5$, max. time $T = 1$ with timestep 0.01, and lattice length $N = 100$. The system evolves from an initial lattice generated by Gibbs sampling with parameters $h_S = 0.5, h_E = 1, h_C = -1, h_P = -1$, and all NN terms set to zero.

The input to the ANN are the basis functions for the three separate processes, each of which belongs to the trivalent reaction motif of Figure 4.9 thereby contributing 9 basis functions. Additionally, the two basis functions for the diffusion of each of the four species is included from Figure 4.8, for a total of 35 inputs. The other layers in the ANN are two layers of 40 units, and an output layer of 14 units, with tanh activation functions between each layer. Two thirds of the total length T of the timeseries are used for training. These are converted to trajectories in interaction parameter space using Boltzmann machine learning, and smoothed using a low-pass filter before being used to evaluate the 35 input basis functions. The corresponding outputs to be learned are the time derivatives of these 14 parameters, also smoothed by a low-pass filter.

The network learns the dynamics of these parameters to high precision. We infer from the fast training times that the usage of these analytic solutions as input greatly reduces the difficulty of training the network from the interaction parameters directly.

Figure 4.12 shows the extrapolated parameters and corresponding moments, compared to the remaining third of the simulation time. These extrapolations are generally

linear in interaction space, and may diverge quickly, such as for h_S. However, the moments show considerable robustness to these variations, suggesting that using ANNs for extrapolation is possible. This has promising implications for further development in multiscale simulation algorithms.

A further feature learned by the ANN is a moment closure approximation for the dynamics of J_{SP}, and the corresponding NN moment it controls. This parameter is not included in any of the basis functions or inputs to the ANN. The basis function learned, shown in Figure 4.12, therefore expresses the dynamics of this moment in terms of the interactions made available as input to the network. Similar extensions to higher order moments are likewise possible.

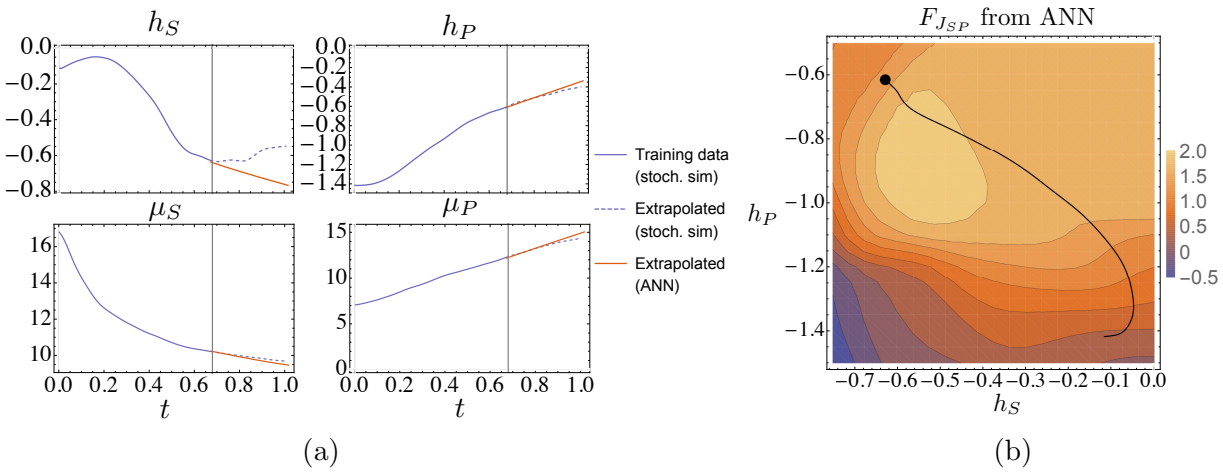

Figure 4.12: (a) Trajectories of the SEP system from stochastic simulation, and extrapolated values from the trained ANN. The divergence of the predicted and true values in moment space is smaller than in interaction parameter space, suggesting a stability in the observable quantities of the model to small errors. (b) The derivative learned by the ANN for the moment. A two dimensional slice is shown through this 14 dimensional function. The black line shows the trajectory of the training data, while the dot indicates the evaluation point for this slice, chosen at the end of the training data (gray vertical line in (a)). All other parameters than h_S, h_P are held fixed at this point.

4.5 Discussion and Conclusions

This chapter presented a new approach to model reduction of spatial chemical systems. Slowly time-evolving MaxEnt models are employed to capture the key correlations in the system. This approach is particularly useful for multiscale problems, where different spatial and temporal correlations become more or less relevant over time to accurately describe the system. For example, in synaptic level neuroscience, the stochastic influx of signaling molecules in the post-synaptic spine produces complex spatial correlations between ion channels and downstream targets, but these are less relevant during quiescent periods. We anticipate that such problems stand to benefit greatly from modeling approaches that are able to adjust which correlations are included to optimize simulation efficiency and accuracy.

A general model that is functional in nature is introduced to describe dynamic Boltzmann distributions. This extends and formalizes ideas originally developed in GCCD in Ref. [10] - in particular:

1. A general variational problem has been formulated to determine the functions in the dynamical system controlling the interaction parameters. This takes the form of a PDE-constrained optimization problem.

2. The reduced model has been extended to capture spatial correlations, with particular relevance to Biological applications. By motivating parameterizations of the functionals from analytically solvable cases, practical optimization algorithms for learning the dynamics of spatial systems are made possible.

3. ANNs have been employed to learn *non-linear* combinations of basis functions, derived for individual reaction processes using the aid of computer algebra systems.

Mapping the chemical system onto a spin lattice allows a direct connection to the more traditionally formulation of a Boltzmann machine. Here, the connection to the new

learning algorithm is evident in (4.43), and we anticipate this will suggest numerous further applications to diverse areas of machine learning where estimating the dynamics of a time series is required. Including arbitrary spatial correlations beyond NN in the lattice model may be of further interest in pursuit of 3D simulations.

Numerous strategies are possible for improving the efficiency of the PDE-constrained optimization problem formulated here, such as adjoint methods [47]. In this work, we have shown that the complexity of this problem can be greatly reduced by instead learning linear and non-linear combinations of analytically accessible approximations. Deconstructing the problem in this way can offer physical insight into a complex reaction system, such as in Section 4.4.3 where effective reaction rates are learned. Future work in this direction may further explore these principled methods for integrating human intuition with machine inference in the model reduction process.

4.6 Acknowledgements

Chapter 4 is a reprint of material, with minor edits as it appears in: O.K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, "Learning dynamic Boltzmann distributions as reduced models of spatial chemical kinetics", *The Journal of Chemical Physics*, vol. 149, no. 3, pp. 034107, 2018.

The author of the dissertation was the primary author of this paper.

Chapter 5

Learning problem for spatial dynamic Boltzmann distributions

In this chapter, we introduce a general learning problem for spatial dynamic Boltzmann distributions. In the previous chapter, we have discussed the state estimation subproblem to determine the parameters in the energy function. At an instant in time, this can be done using the Boltzmann machine learning algorithm, or using expectation maximization. A second subproblem is to determine the parameters in the differential equations that link snapshots in time. A single PDE-constrained optimization problem is formulated in this chapter to address both subproblems. This chapter is taken with edits from [17]. Further work shows examples for physics-based Gaussian graphical models.

5.1 Spatial dynamic Boltzmann distributions

In this section, we introduce the reduced model for a spatiotemporal distribution and its dynamics in continuous space from [16], and formulate the learning problem using adjoint methods. We consider the specific application of a reaction-diffusion system, but

note that the methods are also applicable to other spatiotemporal systems.

The state of a reaction-diffusion system at some time t is described by n particles of species labels $\boldsymbol{\alpha}$ located at positions $\boldsymbol{x}$ in generally continuous 3D space (each x_i for $i = 1, \ldots, n$ is a coordinate in 3D space). Let the true distribution over system states be denoted by $p(n, \boldsymbol{\alpha}, \boldsymbol{x}, t)$, which evolves in time according to the chemical master equation (CME).

To define the reduced model, introduce k-particle interaction functions

$$\nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t), \tag{5.1}$$

where $\langle i \rangle_k^n$ denotes any ordered subset of k indexes with each index in $\{1, \ldots, n\}$. Given a set of such interaction functions $\{\nu\}_{k=1}^K$ up to cutoff order K, define a *spatial dynamic Boltzmann distribution* as one of the form:

$$\tilde{p}(n, \boldsymbol{\alpha}, \boldsymbol{x}, t; \{\nu\}) = \frac{1}{Z[\{\nu\}]} \exp\left[-\sum_{k=1}^{K} \sum_{\langle i \rangle_k^n} \nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) \right], \tag{5.2}$$

where the sum over $\langle i \rangle_k^n$ iterates over unique k-th order interactions between n particles, and the partition function is

$$Z[\{\nu\}] = \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x} \, \exp\left[-\sum_{k=1}^{K} \sum_{\langle i \rangle_k^n} \nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) \right]. \tag{5.3}$$

Boltzmann distributions are maximum entropy (MaxEnt) distributions, where each interaction function $\nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)$ controls a corresponding moment $\mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)$, given by:

$$\mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) = \sum_{n'=0}^{\infty} \sum_{\boldsymbol{\alpha}'} \int d\boldsymbol{x}' \, p(n', \boldsymbol{\alpha}', \boldsymbol{x}', t) \sum_{\langle j \rangle_k^{n'}} \delta(\boldsymbol{x}_{\langle i \rangle_k^n} - \boldsymbol{x}'_{\langle j \rangle_k^{n'}}) \delta(\boldsymbol{\alpha}_{\langle i \rangle_k^n} - \boldsymbol{\alpha}'_{\langle j \rangle_k^{n'}}), \tag{5.4}$$

that is, the average number of k-sized tuplets of particles of species $\boldsymbol{\alpha}_{\langle i \rangle_k^n}$ at locations $\boldsymbol{x}_{\langle i \rangle_k^n}$. Note that $\boldsymbol{\alpha}'$ and $\boldsymbol{x}'$ are of size n'. Section 5.1.3 shows an algorithm for sampling these spatial moments.

5.1.1 Moment matching

Given a set of training data drawn from $p(n, \boldsymbol{\alpha}, \boldsymbol{x}, t)$ at some instant in time, the BM learning algorithm determines parameters in the energy function such that the instantaneous distribution (5.2) is the MaxEnt dist. consistent with the moments in the dataset. To learn a reduced model of a system that evolves in both time and space continuously, we seek the distribution that is *at all times* the MaxEnt solution. Define as the action the KL-divergence between the true and reduced models, p and $\tilde{p}$, over all times:

$$S = \int_{t_0}^{t_f} dt\, \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}), \tag{5.5}$$

where the Lagrangian is $\mathcal{L}(t; \{\nu\}) = \mathcal{D}_{\mathcal{KL}}(p||\tilde{p})$ for

$$\mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) = \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x}\; p(n, \boldsymbol{\alpha}, \boldsymbol{x}, t) \ln \frac{p(n, \boldsymbol{\alpha}, \boldsymbol{x}, t)}{\tilde{p}(n, \boldsymbol{\alpha}, \boldsymbol{x}, t; \{\nu\})}. \tag{5.6}$$

Minimizing S is thus equivalent to maximizing the log-likelihood of the observed data given the interaction functions, i.e. $L(\{\nu\}; \boldsymbol{\alpha}, \boldsymbol{x}, t) = \log \tilde{p}(\boldsymbol{\alpha}, \boldsymbol{x}, t; \{\nu\})$. Other approaches for modeling time series are discussed in Section 6.1.1.

The condition for extremizing the action follows from the chain rule as

$$\delta S = \int_{t_0}^{t_f} dt \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x} \sum_{k=1}^{K} \sum_{\langle i \rangle_k^n} \Delta\mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) \delta\nu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) = 0, \tag{5.7}$$

where

$$\Delta\mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) = \tilde{\mu}_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) - \mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t), \tag{5.8}$$

where μ and $\tilde{\mu}$ are averages taken over p and $\tilde{p}$. This appearance of a difference of moments is the common result from using the KL-divergence in the objective functional.

5.1.2 An adjoint method learning problem for spatial dynamic Boltzmann distributions

Introduce for each interaction function $\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t)$ a *functional* model:

$$\frac{d}{dt}\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t) = \mathcal{F}_k[\{\nu\}](\boldsymbol{\alpha}, \boldsymbol{x}, t), \tag{5.9}$$

over some domain $\boldsymbol{x} \in \Omega$ with boundary Γ, with initial condition $\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t_0) = \eta_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n})$, and where the notation $\{\nu\}$ denotes $\{\nu\} = \{\nu_k\}_{k=1}^K$. We use $\mathcal{F}$ to denote a functional, allowing for example a PDE model to be introduced. Note that the arguments to the left hand side may also appear on the right, for example through a spatial derivative term $\nabla\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t)$.

In practice, boundary conditions on Γ for (5.9) must also be introduced. As these are problem specific, they are intentionally left out of the derivations in this section. In Section 5.3, an example of the learning problem is presented where boundary conditions are included.

Introduce vector notation[1] $\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t)$ and $\boldsymbol{\mathcal{F}}[\{\nu\}](\boldsymbol{\alpha}, \boldsymbol{x}, t)$ for the left and right hand sides of (5.9), which contain $N = \sum_{k=1}^K \binom{n}{k}$ entries, one for every possible $(k, \langle i\rangle_k^n)$ in some order $i = 1, \ldots, N$. To enforce the constraint (5.9), define the Lagrangian as the functional:

$$\begin{aligned}\mathcal{L}[\{\nu\}, \{\xi\}](t) =& \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) \\ &+ \sum_{n=0}^{\infty}\sum_{\boldsymbol{\alpha}}\int d\boldsymbol{x}\, \boldsymbol{\zeta}^\intercal(\boldsymbol{\alpha}, \boldsymbol{x}, t)\left(\frac{d\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}{dt} - \boldsymbol{\mathcal{F}}[\{\nu\}](\boldsymbol{\alpha}, \boldsymbol{x}, t)\right),\end{aligned} \tag{5.10}$$

[1] In this notation, the dot product is: $\boldsymbol{a}^\intercal(\boldsymbol{\alpha}, \boldsymbol{x})\boldsymbol{b}(\boldsymbol{\alpha}, \boldsymbol{x}) = \sum_{k=1}^K \sum_{\langle i\rangle_k^n} a(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}) b(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n})$.

where we have introduced Lagrange multiplier functions $\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t)$ corresponding to $\boldsymbol{\nu}$. Since the constraint is satisfied, then the action is as before $S = \int_{t_0}^{t_f} dt\, \mathcal{L}[\{\nu\}, \{\xi\}](t)$ where $\{\xi\} = \{\xi_k\}_{k=1}^{K}$.

Introducing perturbations $\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t)$ to the interaction functions gives as condition for extremizing the action:

$$\delta S = \int_{t_0}^{t_f} dt \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x}\, \delta\boldsymbol{\nu}^{\intercal}(\boldsymbol{\alpha}, \boldsymbol{x}, t) \left\{ \Delta\boldsymbol{\mu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) - \frac{d\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}{dt} - \frac{\delta\mathcal{J}[\{\nu\}, \{\zeta\}](t)}{\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t)} \right\} = 0, \tag{5.11}$$

where the boundary terms from the integration by parts in the second term have vanished due to the boundary condition for the adjoint variables $\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t_f) = 0$, and we have defined:

$$\mathcal{J}[\{\nu\}, \{\zeta\}](t) = \sum_{n'=0}^{\infty} \sum_{\boldsymbol{\alpha}'} \int d\boldsymbol{x}'\, \boldsymbol{\zeta}^{\intercal}(\boldsymbol{\alpha}', \boldsymbol{x}', t) \boldsymbol{\mathcal{F}}[\{\nu\}](\boldsymbol{\alpha}', \boldsymbol{x}', t). \tag{5.12}$$

We therefore obtain the adjoint system

$$\frac{d\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}{dt} = \Delta\boldsymbol{\mu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) - \frac{\delta\mathcal{J}[\{\nu\}, \{\zeta\}](t)}{\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}. \tag{5.13}$$

Depending on the form of the functional, additional boundary conditions may be enforced to evaluate the term on the right. Equations (5.9,5.13) can be equivalently expressed by the Hamiltonian system

$$\begin{aligned} \frac{d\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}{dt} &= \frac{\delta H[\{\nu\}, \{\zeta\}](t)}{\delta\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}, \\ \frac{d\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}{dt} &= -\frac{\delta H[\{\nu\}, \{\zeta\}](t)}{\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}, \end{aligned} \tag{5.14}$$

where

$$H[\{\nu\}, \{\zeta\}](t) = -\mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) + \mathcal{J}[\{\nu\}, \{\zeta\}](t). \tag{5.15}$$

Given a reduced model for the dynamics (5.9), equation (5.11) gives the necessary

condition for extremizing the action. In a typical model reduction setting, however, the reduced model is not know beforehand. What should the form of the model (5.9) be extremize the action? Consider the case where the functional is specified in terms of some ordinary functions. We next set up a variational problem for these functions appearing on the right hand side of the differential equation. Variational problems of this form have been studied previously: first in the context of optimal control theory [54, 55], and later didactically in [56].

Let the functional be of the form:

$$\frac{d}{dt}\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t) = \mathcal{F}_k[\{\nu\},\{F_k\}](\boldsymbol{\alpha}, \boldsymbol{x}, t), \tag{5.16}$$

where the M_k ordinary functions appearing on the right hand side are $F_k^{(s)}(\{\nu(\boldsymbol{\alpha}, \boldsymbol{x}, t)\})$ for $s = 1, \ldots, M_k$, denoted by $\{F_k\} = \{F_k^{(s)}\}_{s=1}^{M_k}$. For arbitrary perturbations $\delta F_k^{(s)}$, extremizing the action gives

$$\delta S = -\int_{t_0}^{t_f} dt \sum_{n=0}^{\infty}\sum_{\boldsymbol{\alpha}}\int d\boldsymbol{x} \sum_{k=1}^{K}\sum_{\langle i\rangle_k^n}\sum_{s=1}^{M_k} \frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta F_k^{(s)}(\{\nu(\boldsymbol{\alpha}, \boldsymbol{x}, t)\})}\delta F_k^{(s)}(\{\nu(\boldsymbol{\alpha}, \boldsymbol{x}, t)\}) = 0. \tag{5.17}$$

Equation (5.17) is the variational calculus form of the sensitivity equation obtained by the adjoint method when the functional model is specified in terms of some parameter vector [57]. This is particularly clear if we consider the specific form of (5.16) as the autonomous ordinary differential equation (ODE) system:

$$\frac{d}{dt}\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t) = F_k(\{\nu(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t)\}), \tag{5.18}$$

where $\{\nu(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t)\}$ denotes all ν of all possible arguments appearing on the left hand

side. In this case, (5.17) becomes

$$\delta S = -\int_{t_0}^{t_f} dt \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x}\, \boldsymbol{\zeta}^{\mathsf{T}}(\boldsymbol{\alpha}, \boldsymbol{x}, t) \delta \boldsymbol{F}(\{\nu(\boldsymbol{\alpha}, \boldsymbol{x}, t)\}) = 0, \tag{5.19}$$

where as before we have used vectors of length N to denote possible $(k, \langle i \rangle_k^n)$ as before. This resembles the adjoint method sensitivity equation, where variational terms δF_k and δS replace ordinary derivatives with respect to parameters. This will be pursued further in Section 6.1.1. The result of (5.19) is that extremizing the action requires that the adjoint variables vanish everywhere $\zeta_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) = 0$. One case when this is satisfied is if the adjoint system is source free $\Delta\mu_k(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t) = 0$, i.e. the moment matching condition is enforced.

From the Euler-Lagrange equations (5.13), the adjoint variables obey:

$$\frac{d\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t)}{dt} = \Delta\boldsymbol{\mu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) - G^{\mathsf{T}}(\boldsymbol{\alpha}, \boldsymbol{x}, t)\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t), \tag{5.20}$$

where the elements of the $N \times N$ matrix G are

$$G_{i,i'}(\boldsymbol{\alpha}, \boldsymbol{x}, t) = \frac{\partial F_k(\{\nu(\boldsymbol{\alpha}_{\langle i \rangle_k^n}, \boldsymbol{x}_{\langle i \rangle_k^n}, t)\})}{\partial \nu_{k'}(\boldsymbol{\alpha}_{\langle i \rangle_{k'}^n}, \boldsymbol{x}_{\langle i \rangle_{k'}^n}, t)}, \tag{5.21}$$

where $(k, \langle i \rangle_k^n)$ corresponds to index i and $(k', \langle i \rangle_{k'}^n)$ corresponds to index i'. Appendix C.1 gives the formal solution to (5.20) and makes explicit the connection between the conditions for extrema (5.19) and (5.7).

5.1.3 Sampling spatial Boltzmann distributions

It remains to discuss how to sample moments from a spatial Boltzmann distribution like $\tilde{p}(n, \boldsymbol{x})$ (without loss of generality, we drop the species label in this discussion). This type of distribution is challenging to sample - not only are there n random variables $\boldsymbol{x}$, but

the number of unknowns n is also unknown. Because of this, popular sampling techniques in ML such as Gibbs sampling cannot be used. Instead, a special Markov chain Monte Carlo (MCMC) method called reversible jump MCMC (RJMCMC) [58] can be employed.

Metropolis Hastings algorithm

To start, we review the Metropolis Hastings algorithm. This does not allow for jumps in the dimension n, but sets up the RJMCMC method. For a fixed n, the algorithm proceeds as follows:

1. Starting at a given state $(n, \boldsymbol{x}^{(t)})$ at iteration t, draw a new proposal state $n, \boldsymbol{x}^*$ from the proposal (jumping) distribution $J_t(n, \boldsymbol{x}^* | n, \boldsymbol{x}^{(t)})$.

2. Compute acceptance ratio:

$$\begin{aligned} r &= \frac{\tilde{p}(n, \boldsymbol{x}^*)}{\tilde{p}(n, \boldsymbol{x}^{(t)})} \times \frac{J_t(n, \boldsymbol{x}^{(t)} | n, \boldsymbol{x}^*)}{J_t(n, \boldsymbol{x}^* | n, \boldsymbol{x}^{(t)})} \\ &= \exp\Big[- \Big(E(n, \boldsymbol{x}^*, t) - E(n, \boldsymbol{x}^{(t)}, t) \Big) \Big] \times \frac{J_t(n, \boldsymbol{x}^{(t)} | n, \boldsymbol{x}^*)}{J_t(n, \boldsymbol{x}^* | n, \boldsymbol{x}^{(t)})}. \end{aligned} \tag{5.22}$$

3. Accept the new state $(n, \boldsymbol{x}^{(t+1)}) = (n, \boldsymbol{x}^*)$ with probability $\min(r, 1)$.

Reversible jump MCMC

Sampling distributions where "we don't know the number of things we don't know" [58] is possible using reversible jump MCMC (RJMCMC). Here, the number of things we don't know is $n^{(t)}$, the number of particles, i.e. the number of random variables, which we now allow to vary. In this case, RJMCMC generalizes the Metropolis-Hastings algorithm as follows:

1. Start at a given state $(n^{(t)}, \boldsymbol{x}^{(t)})$. Let the probability for moving to a new space of dimension n^* be $q(n^* | n^{(t)})$.

2. Generate a random vector $\boldsymbol{u}$ of length n_u according to: $q_{n_u}(\boldsymbol{u})$. Note: no restriction is made on the length of $\boldsymbol{u}$.

3. Construct the new proposal state $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ from $(\boldsymbol{x}^{(t)}, \boldsymbol{u})$ using the *one-to-one* function $(\boldsymbol{x}^*, \boldsymbol{u}^*) = g_{n^*|n^{(t)}}(\boldsymbol{x}^{(t)}, \boldsymbol{u})$. Here $\boldsymbol{x}^*$ is of length n^*, and $\boldsymbol{u}^*$ is of length n_u^*, which satisfy:

$$n^{(t)} + n_u = n^* + n_u^*. \tag{5.23}$$

   Here, $\boldsymbol{u}^*$ are the random variable indexes and displacements needed to go backward according to: $(\boldsymbol{x}^{(t)}, \boldsymbol{u}) = g^*_{n^{(t)}|n^*}(\boldsymbol{x}^*, \boldsymbol{u}^*)$ (also *one-to-one*).

   - $g_{n^*|n^{(t)}}$ *maps* $\mathcal{R}^{n^{(t)}} \times \mathcal{R}^{n_u} \to \mathcal{R}^{n^*} \times \mathcal{R}^{n_u^*}$.
   - $g^*_{n^{(t)}|n^*}$ *maps* $\mathcal{R}^{n^*} \times \mathcal{R}^{n_u^*} \to \mathcal{R}^{n^{(t)}} \times \mathcal{R}^{n_u}$.

4. Form the Jacobian matrix $\boldsymbol{J}$ with components:

$$\boldsymbol{J}_{ij} = \frac{\partial(\boldsymbol{x}^*, \boldsymbol{u}^*)_i}{\partial(\boldsymbol{x}^{(t)}, \boldsymbol{u})_j}. \tag{5.24}$$

5. Calculate the acceptance probability

$$r = \frac{\tilde{p}(n^*, \boldsymbol{x}^*)}{\tilde{p}(n^{(t)}, \boldsymbol{x}^{(t)})} \times \frac{q(n^{(t)}|n^*) q^*_{n_u^*}(\boldsymbol{u}^*)}{q(n^*|n^{(t)}) q_{n_u}(\boldsymbol{u})} \times |\det(\boldsymbol{J})|. \tag{5.25}$$

6. Accept the new state $(n^{(t+1)}, \boldsymbol{x}^{(t+1)}) = (n^*, \boldsymbol{x}^*)$ with probability $\min(r, 1)$.

The missing piece in RJMCMC is the choice for the proposal distribution q to jump between spaces of different dimensions. This is best chosen specific to the system considered. Since this work studies reaction-diffusion systems, a natural choice is to use a reaction system to generate proposals.

Proposal distribution through annihilation and synthesis reactions

One proposal distribution for the RJMCMC sampler are annihilation and synthesis reactions.

1. Starting from $(n^{(t)}, \boldsymbol{x}^{(t)})$, the annihilation reaction moves to a new space of dimension $n^* = n^{(t)} - 1$, while the synthesis reaction moves to $n^* = n^{(t)} + 1$. Let synthesis be chosen with rate α and annihilation with rate $1 - \alpha$, then choose which reaction occurs from the candidate distribution:

$$\begin{aligned} q(n^{(t)} - 1 | n^{(t)}) &= \begin{cases} 1 - \alpha & \text{if} \quad n^{(t)} \neq 0, \\ 0 & \text{otherwise}, \end{cases} \\ q(n^{(t)} + 1 | n^{(t)}) &= \begin{cases} \alpha & \text{if} \quad n^{(t)} \neq 0, \\ 1 & \text{otherwise}, \end{cases} \end{aligned} \tag{5.26}$$

2. For the annihilation reaction, generate a discrete random variable u according to:

$$q_{\text{ann}}(u = i) = \frac{1}{n^{(t)}} \quad \text{where} \quad i = 0, 1, n^{(t)} - 1. \tag{5.27}$$

   For the synthesis reaction, generate two discrete random variables (u_1, u_2) where u_1 is discrete in $i = 0, 1, \ldots, n^{(t)} + 1$ denoting which index the new particle will be inserted at, and u_2 is the point in continuous space at which the particle will be located:

$$q_{\text{syn}}(u_1 = i, u_2) = \frac{1}{n^{(t)} + 1}. \tag{5.28}$$

3. For the annihilation reaction, construct the new proposal state $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ of length

$n^{(t)}+1$ from $(\boldsymbol{x}^{(t)}, \boldsymbol{u})$ of length $n^{(t)}+1$ using:

$$\begin{aligned} \boldsymbol{x}^* &= \boldsymbol{x}^{(t)} \backslash \{x_u^{(t)}\} \quad \text{of length} \quad n^{(t)}-1, \\ \boldsymbol{u}^* &= (u, x_u^{(t)}) \quad \text{of length} \quad 2. \end{aligned} \tag{5.29}$$

For the synthesis reaction, construct the new proposal state $(\boldsymbol{x}^*, u^*)$ of length $n^{(t)}+2$ from $(\boldsymbol{x}^{(t)}, u_1, u_2)$ of length $n^{(t)}+2$ using:

$$\begin{aligned} \boldsymbol{x}^* &= \{x_0^{(t)}, \ldots, x_{u_1-1}^{(t)}, u_2, x_{u_1+1}^{(t)}, x_{n^{(t)}-1}^{(t)}\} \quad \text{of length} \quad n^{(t)}+1, \\ u^* &= (u_1) \quad \text{of length} \quad 1. \end{aligned} \tag{5.30}$$

4. Form the Jacobian, which works out to $|\det(J)| = 1$ for both annihilation and synthesis reactions.

5. The acceptance probability is then: for the annihilation reaction:

$$r = \frac{\tilde{p}(n^*, \boldsymbol{x}^*)}{\tilde{p}(n^{(t)}, \boldsymbol{x}^{(t)})} \times c, \tag{5.31}$$

where for annihilation:

$$c = \frac{q(n^{(t)}|n^{(t)}-1) q^*_{\text{syn}}(u_1, u_2)}{q(n^{(t)}-1|n^{(t)}) q_{\text{ann}}(u)} = \frac{\alpha(1/n^{(t)})}{(1-\alpha)(1/n^{(t)})} = \frac{\alpha}{1-\alpha}, \tag{5.32}$$

and for the synthesis reaction for $n^{(t)} \neq 0$:

$$c = \frac{q(n^{(t)}|n^{(t)}+1) q^*_{\text{ann}}(u)}{q(n^{(t)}+1|n^{(t)}) q_{\text{syn}}(u_1, u_2)} = \frac{(1-\alpha)(1/(n^{(t)}+1))}{\alpha(1/(n^{(t)}+1))} = \frac{(1-\alpha)}{\alpha}, \tag{5.33}$$

and for the synthesis reaction for $n^{(t)} = 0$:

$$c = \frac{(1-\alpha)(1/(0+1))}{(1)(1/(0+1))} = \frac{1-\alpha}{2}. \tag{5.34}$$

Since the ratio of Boltzmann distributions results in a difference of energies, then the RJMCMC sampler is very efficient because only terms in the energy function involving the particle being annihilated or synthesized contribute.

Sampling particle positions through diffusion

We can change the positions of the particles one at a time, keeping n fixed, using a Gibbs sampler. The univariate conditional distributions are:

$$p(x_j^{(t)}|n^{(t)}, \boldsymbol{x}^{(-j)(t)}, t) = \frac{\exp\left[-\sum_{k=1}^{K}\sum_{\langle i\rangle_{k-1}^{n}} \nu_k\left(x_j^{(t)}, x_{\langle i\rangle_{k-1}^{n}}^{(-j)(t)}, t\right)\right]}{\int dy\, \exp\left[-\sum_{k=1}^{K}\sum_{\langle i\rangle_{k-1}^{n}} \nu_k\left(y, x_{\langle i\rangle_{k-1}^{n}}^{(-j)(t)}, t\right)\right]}, \tag{5.35}$$

where $\boldsymbol{x}^{(-j)}$ denotes the vector $\boldsymbol{x}$ without the j-th element. However, this requires calculating the normalization factor in the denominator, which due to the integration can be difficult/imprecise. A more robust strategy therefore is a similar Metropolis-Hastings sampler.

To move a single particle at a time with a diffusion-like proposal state, construct a new state as $\boldsymbol{x}_i^* = \boldsymbol{x}^{(t)} \backslash \{x_i^{(t)}\} \cup \{x^*\}$ for index $i = 0, \ldots, n^{(t)} - 1$ and new position x^*. The proposal distribution is Gaussian:

$$J_t(n^{(t)}, \boldsymbol{x}_i^* | n^{(t)}, \boldsymbol{x}^{(t)}) = G(x_i^{(t)}, x^*, \xi), \tag{5.36}$$

and the ratio in the Metropolis-Hastings algorithm (5.22) is unity.

Example

Consider a Gaussian in n particles with mean n_0 distributed in space as a Gaussian about the point x_0 with variance σ^2 in each direction:

$$p(n, \boldsymbol{x}) \propto \exp\left[-\frac{(n-n_0)^2}{2\zeta^2}\right] \exp\left[-\sum_{i=1}^{n} \frac{|x_i - x_0|^2}{2\sigma^2}\right] \propto \exp\left[-\sum_{i=1}^{n} \nu_1(x_i) - \binom{n}{2}\nu_2\right], \quad (5.37)$$

where

$$\begin{aligned} \nu_1(x_i) &= \frac{1/2 - n_0}{\zeta^2} + \frac{|x_i - x_0|^2}{2\sigma^2}, \\ \nu_2 &= \frac{1}{\zeta^2}. \end{aligned} \quad (5.38)$$

Figure 5.1 shows the sampled number of particles from the RJMCMC sampler. The parameters were $n_0 = 100, \zeta^2 = 2, x_0 = \mathbf{0}, \sigma^2 = 1$, and the initial distribution contained 200 particles spread uniformly in a box of side length 10. The method converges rapidly over 500 sampling steps.

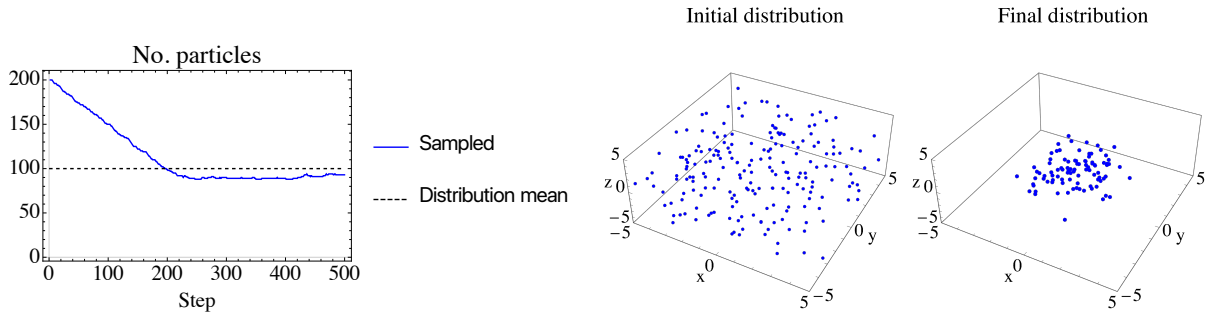

Figure 5.1: *Left:* No particles sampled from the distribution (5.37) using the RJMCMC sampler over 500 steps. At each step, annihilation and synthesis are chosen with equal probability $\alpha = 0.5$, and 10 steps of diffusion are run. *Middle, right:* The initial uniform distribution of 200 particles and final Gaussian distribution of 100 particles.

5.2 PDE-constrained optimization in machine learning

In the previous section, an adjoint method learning problem was introduced for spatial dynamic Boltzmann distributions. First order gradient methods are popular for solving optimization problems in machine learning, but there are many more methods which are popular in PDE-constrained optimization. Here, a short review and brief discussion about solving PDE-constrained optimization problems in machine learning is presented.

Consider again the problem to extremize S given by (5.17) subject to the ODE constraint (5.16). Ultimately, to solve such a problem on a computer, the ODE constraint must be parameterized in terms of *design* or *control variables* $\boldsymbol{r}$. The problem can be written as:

$$\begin{aligned} \min_{\boldsymbol{\nu},\boldsymbol{r}} \quad & S(\boldsymbol{\nu},\boldsymbol{r}), \\ \text{s.t.} \quad & \boldsymbol{c}(\boldsymbol{r},\boldsymbol{\nu}) = \frac{d\boldsymbol{\nu}}{dt} - \boldsymbol{F}(\boldsymbol{\nu},\boldsymbol{r}) = 0, \\ & \boldsymbol{\nu}(t=0) = \boldsymbol{\eta}, \end{aligned} \tag{5.39}$$

where we use vector notation and drop arguments for convenience. The vector $\boldsymbol{\nu}$ is referred to as the *state variables*, and S the *objective function*. Note that the objective function considered in this thesis has no explicit dependence on the controls $S(\boldsymbol{\nu},\boldsymbol{r}) = S(\boldsymbol{\nu})$, but we keep the notation more general in this section.

Two approaches exist for solving (5.39):

1. The solution to the differential equation constraint can be used to define an implicit function from control variables to state variables $\boldsymbol{\nu}(\boldsymbol{r})$, which can be used to eliminate the state constraints. This is often called *black-box optimization* (in the context of PDE-constrained problems), or *nested analysis and design* (NAND) [59, 60].

The *reduced formulation* of the optimization problem becomes:

$$\min_{\boldsymbol{r}} \quad S^*(\boldsymbol{r}), \tag{5.40}$$

where the reduced objective function is:

$$S^*(\boldsymbol{r}) = S(\boldsymbol{r}, \boldsymbol{\nu}(\boldsymbol{r})). \tag{5.41}$$

The mapping $\boldsymbol{\nu}(\boldsymbol{r})$ exists for all constraints that will be considered in this thesis. More precisely, the implicit function theorem guarantees that the mapping $\boldsymbol{\nu}(\boldsymbol{r})$ exists if the matrix $\partial \boldsymbol{c}/\partial \boldsymbol{\nu}$ is invertible (i.e. exists and is nonsingular) [59].

2. The constraint can be kept explicitly in the optimization problem, such that *both* $\boldsymbol{\nu}$ and $\boldsymbol{r}$ are optimized. The PDE-constraint is therefore generally not satisfied during training; only at the final optimization step do we require (if the problem has been solved) that the constraint is satisfied. This is often called *all-at-once optimization*, or *simultaneous analysis and design* (SAND) [59, 60].

The second method is increasingly popular for PDE-constrained optimization problems because it does not necessarily require solving the PDE constraint at each optimization step, which can be a computational bottleneck. However, the second method relies on using powerful optimization methods to perform well, particularly second order methods [60]. However, in machine learning applications, there are other factors that limit the choice of optimization method:

- Methods that require evaluating the objective function are typically avoided in machine learning. A cornerstone idea is to estimate gradients from small batches, leading to the popular *stochastic gradient descent*. Computing the objective function over the entire dataset during training is computationally expensive, but using a

small batch size is likely to be noisy. Further, computing the objective function for the dynamic Boltzmann distribution requires the calculation of the normalization constant. Methods such as annealed importance sampling (AIS) or Chib-inspired estimators [61] can be used to estimate of the partition function, but these require a large number of sampling steps, making them impractical to use during training.

- Line search methods are not common because they require evaluating the objective function. This will have important implications for the treatment of state inequality constraints as discussed later.

- Second order gradient methods require estimating higher order moments. For example, in deep Boltzmann machines, the Hessian involves estimating three and four-spin moments. Higher order moments require larger sample sizes to estimate reliably at significant computational expense.

- Stochastic quasi-Newton methods, i.e. methods that combine stochastic optimization with quasi-Newton methods that estimate the Hessian, are a forefront research topic [62, 63] and discussed further in Chapter 5.4.1. Quasi-Newton methods usually involve curvature pairs, i.e. differences in the gradient signal obtained over successive optimization steps. In machine learning, even the true first order gradient is not available - only a noisy approximation is possible over a small batch size. Since differentiation amplifies noise, the approximation of the Hessian by small batch sizes is highly inaccurate. One way to mitigate this is by using large batch sizes, but this is difficult to balance with computation time [64]. Instead, methods like stochastic Broyden–Fletcher–Goldfarb–Shanno (BFGS) update curvature pairs without changing the batch across neighboring optimization steps to mitigate the noise. Despite this, limitations remain as they require line searches for accurate estimates of the Hessian (c.f. Wolfe conditions).

Both first order methods and stochastic quasi-Newton methods are used in the next sections.

5.3 Learning diffusion constant

To model a simple diffusion process in 3D from a point source of particles, let the spatial dynamic Boltzmann distribution be of the form:

$$\tilde{p}(\boldsymbol{x},t) = \exp\left[-\sum_{i=1}^{n} \nu_1(x_i,t)\right]. \tag{5.42}$$

Note that to take the limit from a generally variable number of particles n to a fixed number of particles, a thin Gaussian in n can be introduced in the interactions (Appendix C.3.1).

The diffusion operator acting on $\tilde{p}$ for an initial Gaussian distribution at $t = t_0$ leads to the following differential equation for the interactions (derived in Appendix C.3.2), where x is a point inside Ω with boundary Γ:

$$\begin{aligned} \partial_t \nu_1(x,t) &= D\nabla^2 \nu_1(x,t) - D\left(\nabla \nu_1(x,t)\right)^2, \\ \nu_1(x,t=t_0) &= \frac{(x-\mu)^2}{4Dt_0} + \frac{3}{2}\ln(4\pi D t_0). \end{aligned} \tag{5.43}$$

Additionally, the boundary conditions must be specified. Let these be Neumann conditions for x on the boundary $x \in \Gamma$ at all times $t \in [t_0, t_f]$:

$$\nabla_i \nu_1(x,t) \cdot \hat{n} = g(x,t), \tag{5.44}$$

for some specified $g(x,t)$.

To derive the adjoint equation, introduce a Lagrange multiplier $\xi(x,t)$ to enforce the differential equation constraint for $\nu_1(x,t)$, and similarly a Lagrange multiplier func-

tion $\eta(x,t)$ to enforce the boundary condition. The adjoint equations are derived in Appendix C.3.4, giving for points x in the interior:

$$\begin{aligned}\partial_t \xi(x,t) &= \Delta\mu(x,t) - D\nabla^2\xi(x,t) - 2D\xi(x,t)\nabla^2\nu_1(x,t) - 2D\nabla\xi(x,t)\cdot\nabla\nu_1(x,t),\\ \xi(x,t_f) &= 0,\end{aligned} \tag{5.45}$$

and for points on the boundary $x \in \Gamma$:

$$\begin{aligned}-D\xi(x,t) + \eta(x,t) &= 0,\\ \nabla\xi(x,t)\cdot\hat{n} + 2\xi(x,t)g(x,t) &= 0.\end{aligned} \tag{5.46}$$

Note that since the Lagrange multiplier η only appears in the third equation, it can be ignored in practice unless we are interested in determining its value.

To solve the diffusion equation numerically, for example using FEniCS [65], the weak formulation of the forward and backward problems are required. Approximating time derivatives using a backward Euler scheme:

$$\frac{\nu_1^{(n+1)}(x) - \nu_1^{(n)}(x)}{\Delta t} = D\nabla^2\nu_1^{(n+1)}(x) - D\left(\nabla\nu_1^{(n+1)}(x)\right)^2, \tag{5.47}$$

gives the weak formulations for the forward problem as derived in Appendix C.3.3:

$$\begin{aligned}&\int_\Omega dx\, \nu_1^{(n+1)}(x)v(x) + \Delta t D\int_\Omega dx\, \nabla\nu_1^{(n+1)}(x)\cdot\nabla v(x)\\ &+ \Delta t D\int_\Omega dx\, \left(\nabla\nu_1^{(n+1)}(x)\right)^2 v(x) - \int_\Omega dx\, \nu_1^{(n)}(x)v(x) = 0,\end{aligned} \tag{5.48}$$

and for the backward problem as derived in Appendix C.3.5:

$$\begin{aligned}&\int dy\, \frac{\xi^{(n)}(y)}{\Delta t}v(y) + D\int dy\, \nabla\xi^{(n)}(y)\cdot\nabla v(y) + 2D\int dy\, \xi^{(n)}(y)\nabla\nu_1^{(n)}(y)\cdot\nabla v(y)\\ &- \int dy\, \frac{\xi^{(n+1)}(y)}{\Delta t}v(y) + \int dy\, \Delta\mu^{(n)}(y)v(y) = 0.\end{aligned} \tag{5.49}$$

Finally, the optimality condition is derived in Appendix C.3.6 and given by:

$$\begin{aligned} 0 = \frac{dS}{dD} = &- \int_{t_0}^{t_f} dt \int_{\Gamma} dx\, g(x,t)\xi(x,t) \\ &+ \int_{t_0}^{t_f} dt \int dx \left(\nabla \nu_1(x,t) \cdot \nabla \xi(x,t) + (\nabla \nu_1(x,t))^2 \xi(x,t) \right). \end{aligned} \tag{5.50}$$

Figure 5.2 gives an example where the diffusion constant of a point source of particles is learned from stochastic simulations in MCell [5]. The weak forms of the equations are solved using FEniCS [65] on the mesh show in panel (b), where the boundary condition is taken as the zero flux condition $g(x,t) = 0$ everywhere on the boundary at all times. The learned diffusion constant matches the true diffusion constant of the stochastic simulations. In general, this approach can be used to learn an effective diffusion constant that may arise from particles diffusion in a confined environment, such as for ions in the post-synaptic spine head.

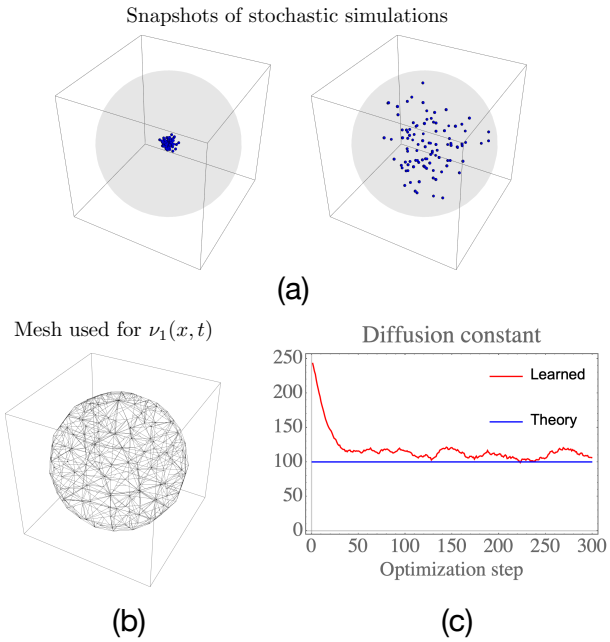

Figure 5.2: (a) Snapshots of stochastic simulations of a point source of particles diffusion in a 3D sphere. Simulations are done using MCell [5] in continuous space. *Left:* initial distribution. *Right:* final timepoint. (b) Mesh used for learning $\nu_1(x,t)$. (c) Learned diffusion constant converging to the true constant from stochastic simulations, starting from an initial guess $D = 250$ as a function of the optimization step.

5.4 Physics-based Gaussian graphical models

An important class of dynamic Boltzmann distribution models are Gaussian distributions for well mixed systems, described by the number of particles $\boldsymbol{n} = \{n_A, n_B, \dots\}$ of each species $\mathcal{R} = \{A, B, \dots\}$ of length M. The reduced model distribution is:

$$\tilde{p}(\boldsymbol{n}; \boldsymbol{\nu}(t)) \propto \exp\left[-\frac{1}{2}\left(\boldsymbol{n} + B^{-1}\boldsymbol{\nu}_1 - \frac{1}{2}B^{-1}\mathrm{diag}(B)\right)^{\intercal} B \left(\boldsymbol{n} + B^{-1}\boldsymbol{\nu}_1 - \frac{1}{2}B^{-1}\mathrm{diag}(B)\right)\right], \tag{5.51}$$

where $\boldsymbol{\nu}_1$ are bias terms for each species, and the precision matrix B has elements:

$$B = \begin{pmatrix} \nu_{2,\mathcal{R}_1,\mathcal{R}_1} & \nu_{2,\mathcal{R}_1,\mathcal{R}_2} & \cdots & \nu_{2,\mathcal{R}_1,\mathcal{R}_M} \\ \nu_{2,\mathcal{R}_1,\mathcal{R}_2} & \nu_{2,\mathcal{R}_2,\mathcal{R}_2} & \cdots & \nu_{2,\mathcal{R}_2,\mathcal{R}_M} \\ \cdots & & & \\ \nu_{2,\mathcal{R}_1,\mathcal{R}_M} & \nu_{2,\mathcal{R}_2,\mathcal{R}_M} & \cdots & \nu_{2,\mathcal{R}_M,\mathcal{R}_M} \end{pmatrix}. \tag{5.52}$$

Here, the time dependence of parameters ν, B has been omitted. This uncommon form of a Gaussian distribution makes clear the connection to graphical models - ν_1 are single species interaction terms, while ν_2 are edges in graph, including self interactions. See also Chapter 3.1. Missing edges correspond to zeros in the precision matrix.

To describe the time evolution of the dynamic Gaussian distribution, differential equations are introduced for $\boldsymbol{\nu}$. The physics of reaction networks can be introduced into the learning problem through the linearity proposition of the previous chapter. For a chosen set of reactions, approximations for the time evolution of $\boldsymbol{\nu}$ are derived under the MaxEnt closure approximation. For example, for the reaction $A + B \to C$:

1. Derive from the CME the differential equations for the moments controlled by the

interactions $\boldsymbol{\nu}$ in the MaxEnt sense, i.e.

$$\begin{aligned} \frac{d}{dt}\langle n_{\mathcal{R}_i}\rangle_p \quad & \text{for } i=1,\dots,M, \\ \frac{d}{dt}\left\langle \binom{n_{\mathcal{R}_i}}{2} \right\rangle_p \quad & \text{for } i=1,\dots,M, \\ \frac{d}{dt}\langle n_{\mathcal{R}_i} n_{\mathcal{R}_j}\rangle_p \quad & \text{for } i \neq j. \end{aligned} \tag{5.53}$$

Here, missing terms in the precision matrix are not included, and p denotes that these are the fine scale differential equations derived from the CME. The scaling factor that appears in these equations from a reaction rate is omitted.

2. Close the system of differential equations by using MaxEnt closure (aka "zero information" closure), i.e. by expressing all moments through the Boltzmann distribution (5.51):

$$\langle \dots \rangle_p \to \langle \dots \rangle_{\tilde{p}}. \tag{5.54}$$

3. Convert back to the parameter frame using the relations:

$$\begin{aligned} B &= \Sigma^{-1}, \\ \boldsymbol{\nu}_1 &= -B\boldsymbol{\mu} + \frac{1}{2}\mathrm{diag}(B). \end{aligned} \tag{5.55}$$

If there are missing edges in the graphical model, i.e. if the precision matrix has a certain structure with some zero elements, this conversion is more involved. Appendix C.4.1 derives the necessary equations for this conversion.

The result is a closed differential equation system for the parameters:

$$\tilde{\boldsymbol{F}}^{(A+B\to C)}(\boldsymbol{\nu}(t)) = \left(\frac{d\boldsymbol{\nu}}{dt}\right)^{(A+B\to C)}, \tag{5.56}$$

where we use $\tilde{\boldsymbol{F}}$ to denote that this is an approximation under the MaxEnt closure

approximation for this reaction. This approximation is a function of all generally parameters $\boldsymbol{\nu}$ in the model.

Since the linearity of the CME in reactions extends to this form of the approximations (Chapter 4), then a linear model is a physically relevant choice for combining $\tilde{\boldsymbol{F}}$ under different reactions. Let the differential equations thus be of the form:

$$\frac{d\boldsymbol{\nu}}{dt} = \boldsymbol{F}(\boldsymbol{\nu}(t); \boldsymbol{u}) = \sum_{r=1}^{R} u_r \, \tilde{\boldsymbol{F}}^{(r)}(\boldsymbol{\nu}(t)), \tag{5.57}$$

for reactions indexed by $r = 1, \ldots, R$. The coefficients $\boldsymbol{u}$ learned are directly the reaction rates corresponding to different reactions. This type of model can be used for both system identification, where the space of reactions introduced includes those of the stochastic simulations, or for model reduction, where the reaction approximations chosen are a simpler reaction network than that of the simulations.

In addition to the differential equation constraints, two additional inequality constraints are required. First, since the distribution describes particle counts, the mean of the distribution $\boldsymbol{\mu}$ should be non-negative. Second, while the initial distribution is assumed to have a valid precision matrix B that is positive semidefinite, the learned differential equations must keep the precision matrix positive semidefinite at all timepoints.

The full constrained optimization is:

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & S = \int_0^T dt \, \mathcal{D}_{\mathrm{KL}}(p||\tilde{p}), \\ \text{subject to} \quad & \dot{\boldsymbol{\nu}}(t) = \boldsymbol{F}(\boldsymbol{\nu}(t); \boldsymbol{u}) = \sum_{r=1}^{R} u_r \, \tilde{\boldsymbol{F}}^{(r)}(\boldsymbol{\nu}(t)), \\ & \boldsymbol{\nu}(t=0) = \boldsymbol{\nu}_0, \\ & \boldsymbol{\mu}(t) \geq 0, \\ & B(t) \succ 0, \end{aligned} \tag{5.58}$$

where $\boldsymbol{u}$ are the parameters to be learned. The awake and asleep phase moments for the Gaussian distribution that arise in the gradients in S are derived in Appendix C.4.2.

The optimization problem (5.58) now contains not only differential equation constraints, but also inequality constraints on the state variables (rather than directly on the controls $\boldsymbol{u}$). A key problem is that the inequality constraints must be satisfied during training, not just at the final iterate. Otherwise, the calculation of the reaction approximations is numerically unstable because the distribution (5.51) is ill-formed. The optimization method must therefore be a *interior point method*.

To deal with the inequality constraints, introduce the log barrier function:

$$\Phi(x) = -\ln(x). \tag{5.59}$$

To deal with the positive semidefinite constraint, use the eigenvalue decomposition $\Sigma = Q\Lambda Q^\intercal$ where $\boldsymbol{\lambda} = \mathrm{diag}(\Lambda)$ are the eigenvalues. The eigenvalues of the precision matrix are simply their inverse $\boldsymbol{\lambda}^{-1}$, which are constrained to be positive at all times along with the mean number of particles:

$$S \to \int_0^T dt\, \mathcal{D}_{\mathrm{KL}}(p||\tilde{p}) + \zeta_\mu \int_0^T dt\, \Phi(\boldsymbol{\mu}) + \zeta_\lambda \int_0^T dt\, \Phi(\boldsymbol{\lambda}^{-1}), \tag{5.60}$$

where ζ are hyperparameters to be tuned, and vector notation denotes $\Phi(\boldsymbol{\mu}) = \sum_{i=1}^M \Phi(\mu_i)$.

Equation (5.60) requires evaluating the gradients of the log barrier with respect to the state variables. For the eigenvalues in particular, this is given by:

$$\frac{\partial \Phi(\lambda_i^{-1})}{\partial \nu^*} = -\lambda_i^{-1}\frac{\partial \lambda_i}{\partial \nu^*} = \begin{cases} 0 & \text{if } \nu^* = \nu_{1\alpha} \\ (Q_{:,i}\cdot \boldsymbol{e}_\alpha)(Q_{:,i}\cdot \boldsymbol{e}_\beta) = Q_{\alpha,i}Q_{\beta,i} & \text{if } \nu^* = \nu_{2\alpha\beta} \end{cases} \tag{5.61}$$

where $Q_{:,i}$ is the eigenvector given by the i-th column of Q. In practice, placing a barrier

on $\log_{10}(\boldsymbol{\lambda}^{-1})$ rather than $\boldsymbol{\lambda}^{-1}$ was found to be numerically more stable.

Since an interior point method is required, an addition Lagrange multiplier $\boldsymbol{\xi}$ is introduced as the dual variable for each inequality constraint:

$$\begin{aligned} \mu_i \xi_{\mu,i} &= \zeta_\mu, \\ \lambda_i \xi_{\lambda,i} &= \zeta_\lambda, \end{aligned} \tag{5.62}$$

for $i = 1, \ldots, M$. The dual variables are updated in parallel to the control parameters in the optimization problem. Further, the step sizes must be limited, i.e. if the update step is of the form:

$$\boldsymbol{u} \to \boldsymbol{u} + \alpha \times \Delta \boldsymbol{u}, \tag{5.63}$$

where $\Delta \boldsymbol{u}$ is the update resulting from some optimization method, then $\alpha \in [0, 1]$ must chosen such that $\xi \geq 0$ to ensure that the new point is interior.

5.4.1 Stochastic L-BFGS

Some of the most powerful methods for solving differential equation constrained optimization problems such as (5.58) are quasi-Newton methods. Newton's method is prohibitively expensive, as the second gradient in the KL-divergence involves high order moments that require many samples to estimate. Instead, quasi-Newton methods such as the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm or it's limited memory variant L-BFGS construct an approximation to the inverse Hessian. Let the term *curvature pairs* at the n-th optimization step denote the pair:

$$\begin{aligned} \boldsymbol{s}_{n-1} &= \boldsymbol{u}_n - \boldsymbol{u}_{n-1}, \\ \boldsymbol{y}_{n-1} &= \nabla S(\boldsymbol{u}_n) - \nabla S(\boldsymbol{u}_{n-1}). \end{aligned} \tag{5.64}$$

Algorithm 3 Standard L-BFGS

Initialize
    Given: batch size η, no. curvature pairs m to store
    Initialize: $\boldsymbol{u}_0$, $n = 0$
while not converged **do**
    Choose batch of data R_n of size η.
    Calculate gradients $\nabla S(\boldsymbol{u}_n)$ as in Algorithm 1 using R_n.
    Update curvature pairs, retaining the last m pairs:
        $\boldsymbol{s}_{n-1} = \boldsymbol{u}_n - \boldsymbol{u}_{n-1}$
        $\boldsymbol{y}_{n-1} = \nabla S(\boldsymbol{u}_n) - \nabla S(\boldsymbol{u}_{n-1})$
    Calculate search direction $\boldsymbol{p}_n$ from $\{\boldsymbol{s}\}, \{\boldsymbol{y}\}$ using standard L-BFGS two-loop recursion [66].
    Line search and update: $\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \alpha \boldsymbol{p}_n$.
    $n \to n+1$.

In L-BFGS, the search direction is calculated [66] from the m most recent curvature pairs, where m is usually chosen to be small $m \sim 5$. Algorithm 3 reviews the L-BFGS algorithm. An important aspect is a line search that ensures that the Wolfe conditions are met - this ensures that estimate of the inverse Hessian is positive definite.

The KL-divergence objective function has an unfortunate challenge for the L-BFGS algorithm. One reason is that the gradients are noisy because they are estimated from averaging over a small batch size. This noise is amplified by the curvature pairs (5.64) because they essentially compute the derivative of the noisy gradients. Furthermore, not only are the gradients noisy, but even the objective function cannot be easily estimated, since it requires evaluating a partition function. This makes the line search in the L-BFGS algorithm infeasible, but this is needed to ensure a positive definite inverse Hessian.

The solution to these problems is a stochastic L-BFGS algorithm [67], also sometimes called an online L-BFGS algorithm. Let the batch used for the gradients at step $n-1$ be R_{n-1}, and the gradients at n be estimated from the batch R_n. The main idea is to estimate the curvature pairs (5.64) from the *overlap* $O_n = R_n \cap R_{n-1}$ to reduce the noise in the estimates.

Algorithm 4 Stochastic (online) L-BFGS

Initialize
  Given: batch size η, no. curvature pairs m to store
  Initialize: $\boldsymbol{u}_0$, $n = 0$
while not converged **do**
  Batch R_n of size η that overlaps with R_{n-1}: $O_n = R_n \cap R_{n-1}$.
  Calculate gradients $\nabla S(\boldsymbol{u}_n)$ as in Algorithm 1 using R_n.
  Update curvature pairs, retaining the last m pairs:
    $\boldsymbol{s}_{n-1} = \boldsymbol{u}_n - \boldsymbol{u}_{n-1}$
    $\boldsymbol{y}_{n-1} = \nabla S(\boldsymbol{u}_n) - \nabla S(\boldsymbol{u}_{n-1})$ from overlap O_n
  Ensure that the new curvature pair satisfies $\boldsymbol{s}_{n-1}^\mathsf{T}\boldsymbol{y}_{n-1} > 0$, else drop this curvature pair.
  Calculate search direction $\boldsymbol{p}_n$ from $\{\boldsymbol{s}\}, \{\boldsymbol{y}\}$ using standard L-BFGS two-loop recursion [66].
  No line search: $\boldsymbol{u}_{n+1} = \boldsymbol{u}_n + \boldsymbol{p}_n$.
  $n \to n+1$.

A further regularization strategy is needed to ensure the estimate of the inverse Hessian remains positive definite for the new curvature pairs. A simple strategy is to just drop curvature pairs that do not satisfy $\boldsymbol{s}_{n-1}^\mathsf{T}\boldsymbol{y}_{n-1} > 0$. However, if too many curvature pairs are dropped, the estimate of the inverse Hessian can lag behind and give wrong curvature information. A better strategy is to introduce a regularization parameter λ and replace the curvature estimate (5.64) by $\boldsymbol{y}_{n-1} = \nabla S(\boldsymbol{u}_n) - \nabla S(\boldsymbol{u}_{n-1}) + \lambda \boldsymbol{s}_{n-1}$, where λ is chosen at each optimization step to satisfy $\boldsymbol{s}_{n-1}^\mathsf{T}\boldsymbol{y}_{n-1} > 0$. A similar strategy using Powell damping is also possible [68].

Algorithm 4 summarizes the complete stochastic L-BFGS method.

5.4.2 Lotka-Volterra

As an example problem, consider again the Lotka-Volterra system:

$$\begin{aligned} H &\underset{k_d}{\rightarrow} \varnothing, \\ H + P &\underset{k_e}{\rightarrow} 2H, \\ P &\underset{k_b}{\rightarrow} 2P. \end{aligned} \tag{5.65}$$

Stochastic simulations were run for this system using the Gillespie algorithm to generate training data. The reaction rates in the simulations were $k_d = 0.2, k_e = 0.00166$, and $k_b = 1.0$ (arbitrary units). The simulation time was 30 (arbitrary time units), with the population of hunter and prey stored at intervals of 0.25 (arbitrary time units) resulting in 120 timepoints. The initial distribution of particles was Gaussian with mean for hunter and prey $\{600, 250\}$ and covariance matrix $\text{diag}(\{25, 25\})$, and simulations were run for 300 random seeds.

Figure 5.3 shows the learned coefficients in the differential equations (5.57) where the reaction approximations used are those of the stochastic simulations (5.65). The coefficients reproduce the true reaction rates as expected after ~ 50 optimization steps, with initial coefficients set to zero. Figure 5.4 shows the awake and asleep phase moments learned, showing that the reduced model differential equations (5.57) under the closure approximation accurately reproduce the observed moments.

5.4.3 Recovering unobserved species

In the previous section, all species $\mathcal{R} = \{H, P\}$ were observed from the stochastic simulations. Instead, latent species may also be introduced. For example, consider again the Lotka-Volterra system (5.65), but only observe the prey P, and introduce a latent species X in the Gaussian distribution. The reaction approximations in the reduced model

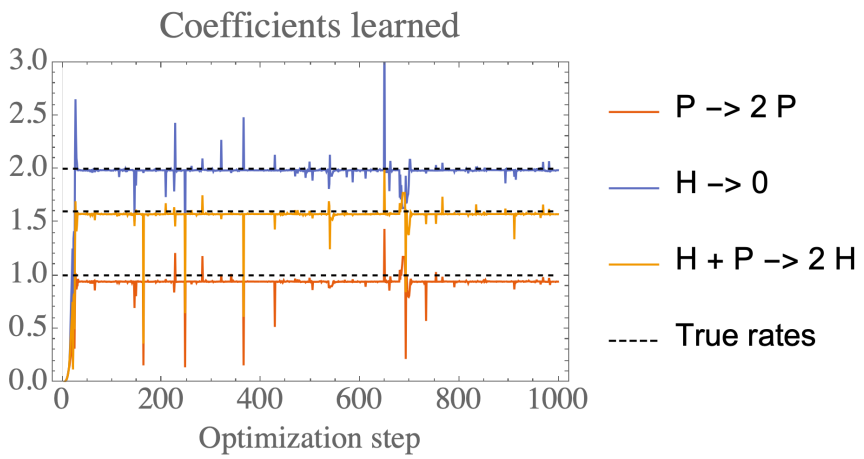

Figure 5.3: Learned coefficients in the differential equation model (5.57) for the Lotka-Volterra system (5.65). The reaction system used in the reduced model is that of the stochastic simulations; the learned coefficients converge to the values from the simulations. The rate for the hunter death reaction is scaled by a factor 10; the rate for the predator-prey reaction is scaled by a factor 1000. Noise is due to the stochastic L-BFGS algorithm used.

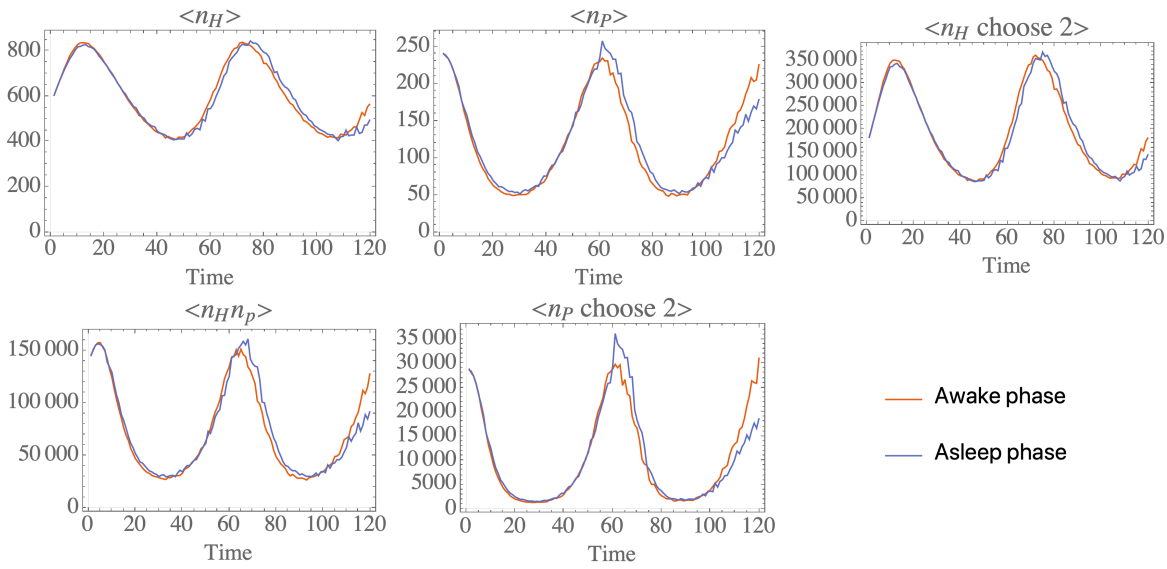

Figure 5.4: Moments learned after 1000 optimization steps for the Lotka-Volterra system (5.65). The asleep phase moments reproduce the awake phase moments.

are now:

$$\begin{aligned} X &\to \varnothing, \\ X + P &\to 2X, \\ P &\to 2P. \end{aligned} \tag{5.66}$$

Figure 5.5 shows the learned coefficients for this model, showing that the model can rediscover X as a hunter species, with the correct rates from the stochastic simulations.

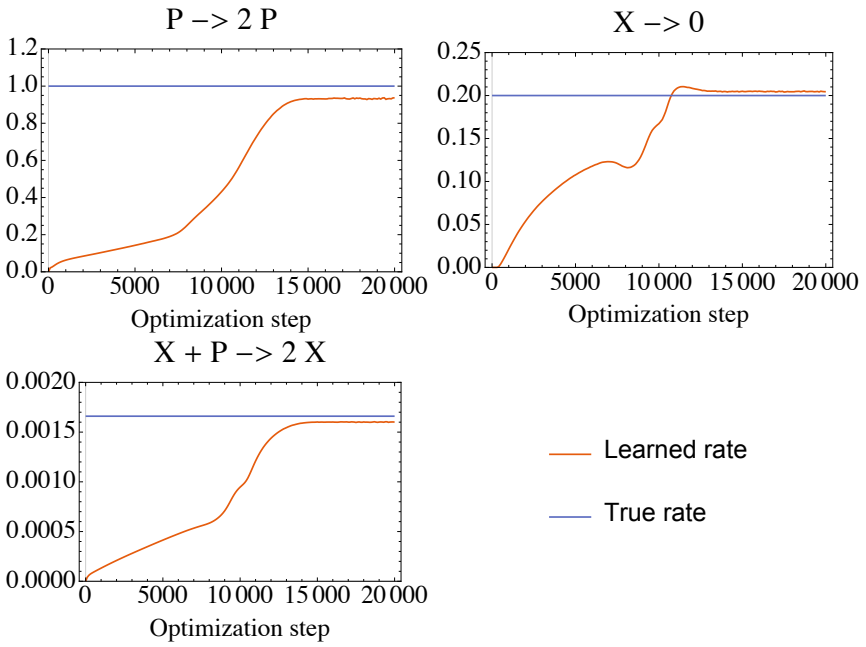

Figure 5.5: Learned coefficients in the differential equation model (5.57) for the Lotka-Volterra system (5.65). The reaction system used in the reduced model is (5.66), where the hunter population is not observed, but instead represented by a latent species X. The model learns that X should play the role of a hunter species, and rediscovers the rate constants from the stochastic simulations. The Adam optimizer [69] was used for 2×10^4 optimization steps, with learning rates 10^{-3} for the birth reaction coefficient, 10^{-4} for the death reaction, and 10^{-6} for the predator-prey reaction.

5.4.4 Competitive Lotka-Volterra

As a second example, the Lotka-Volterra system is learned again, but this time using a competitive Lotka-Volterra system for the reaction approximations of the reduced

model (5.57):

$$\begin{aligned} H &\to \varnothing, \\ P &\to \varnothing, \\ H + P &\to 2H, \\ P + H &\to 2P, \\ H &\to 2H, \\ P &\to 2P. \end{aligned} \tag{5.67}$$

Each species can therefore be the hunter or prey. Figure 5.6 shows the learned coefficients in this case. While the true reaction rates are not obtained, an equivalent reaction network is learned for the system. An extra Lasso regularization term has been shown to help identify the true system [44].

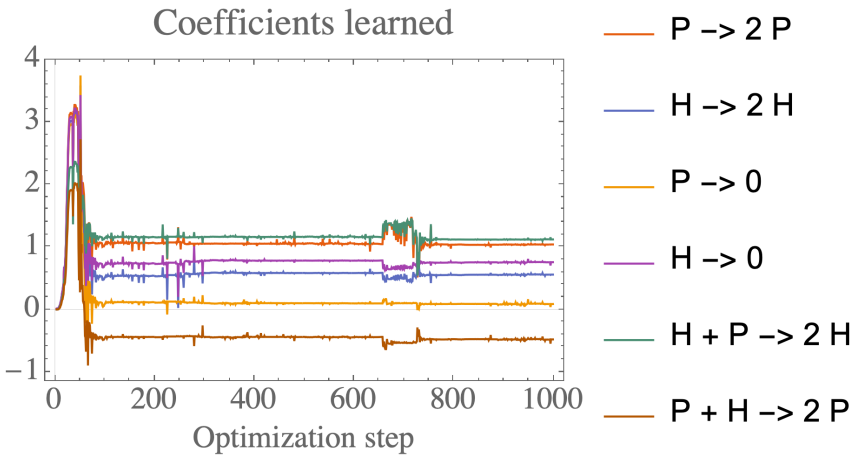

Figure 5.6: Learned coefficients in the differential equation model (5.57) for the Lotka-Volterra system (5.65). The reaction system used in the reduced model is a competitive Lotka-Volterra system (5.67) where each species can be hunter or prey. The rates for the death reactions are scaled by a factor 10; the rates for the predator-prey reactions are scaled by a factor 1000. Noise is due to the stochastic L-BFGS algorithm used.

5.5 Discussion

We have presented a learning problem for spatiotemporal distributions that estimates differential equation systems controlling a time-varying Boltzmann distribution. The ability to enforce a reduced physical model to be estimated makes the method interesting for many modeling applications, including chemical kinetics as presented here. Mapping to a differential equation model can be likewise be useful for engineering applications, allowing constraints to be efficiently introduced into BM learning as discussed in Section 6.1.1.

The moment closure approximation presented in Section 5.1 is broadly applicable due to the use latent variables that can be trained to capture relevant higher order correlations, rather than deciding a priori what correlations to include as in typical closure schemes. Minimizing the KL divergence between the reduced and true models at all times is closely related to entropic matching, but differs by the introduction of a differential equation system. We also make the connection to spatially continuous reaction systems explicit.

A popular alternative class of generative models to RBMs are variational autoencoders (VAEs). An adaptation of the proposed method may be possible for these models - however, the main advantages of the current RBM framework is that the form of the energy function can be used interpret the reduced model [16], and that the distribution over the latent variables is not chosen as in VAEs (typically a standard normal distribution), but rather learned from data.

A closely related problem to model reduction is the problem of data assimilation, where noisy measurements and an incomplete model for the dynamics are combined to estimate the true state of the system and unknown parameters in the model [70]. Model reduction methods complement the data assimilation problem by replacing the physical model with a reduced one which can increase the efficiency of data assimilation methods.

We view the present work as progress toward linking models across scales in biology [71]. Reaction-diffusion systems illustrate many of the common problems in this field.

While much machinery (CME or field-theoretic methods) exists to formulate problems for observables, their solution is non-trivial in most applications. Even without analytic challenges such as moment closure, the numerical solution of PDE systems is difficult for systems with high spatial organization, or where interactions with other scales (e.g. molecular dynamics) or physics (e.g. electrodiffusion) become relevant. Learning reduced models in the form of spatial dynamic Boltzmann distributions may abstract much of these non-trivial interactions.

5.6 Acknowledgements

Sections 5.1 and 5.5 is a reprint of material, with minor edits as it appears in: O.K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, “Learning moment closure in reaction-diffusion systems with spatial dynamic Boltzmann distributions", *Phys. Rev. E*, vol. 99, no. 6, pp. 063315, Jun. 2019.

The author of the dissertation was the primary author of this paper.

Chapter 6

Deep learning moment closure approximations

In this chapter, applications to reaction-diffusion systems on a lattice are explored. These lattice models of reaction-diffusion systems are close to Boltzmann machines in their formalism, and many interesting problems arise from the connection. Building to deep dynamic Boltzmann distributions in particular is a challenge that is solved in this chapter using the centering transformation [40, 41], a trick introduced in Chapter 3 to limit the magnitude of signals passed between layers. The multiple layers learn the long range spatial correlations which are relevant to the moment closure problem. It is further shown how the differential equations can be parameterized using basis functions from finite elements. This chapter is taken with modifications from Ref. [18].

6.1 Restricted Boltzmann machines

We next consider a specific case of the formalism of Section 5.1 where the system is described by discrete random variables. A Boltzmann distribution on a state $\boldsymbol{v} =$

$\{v_1, \ldots, v_N\}$ of N discrete random variables is of the form:

$$\tilde{p}(\boldsymbol{v}) = \frac{1}{Z} \exp[-E(\boldsymbol{v})], \tag{6.1}$$

where Z is the partition function, and the energy function $E(\boldsymbol{v})$ is typically defined by a chosen Markov random field (MRF). For example, a Boltzmann machine (BM) [13] is a binary MRF, where binary units update their state based on a bias and pairwise connections to other units. A MRF where all variables $\boldsymbol{v}$ are driven by data is fully visible; otherwise units $\boldsymbol{h}$ which are not driven by data are denoted as hidden.

A restricted Boltzmann machine (RBM) [72] is a BM in which hidden and visible units are organized into layers, where a layer is defined by the property that there are no interactions among units in the same layer. For example, a typical energy function for an RBM is of the form:

$$E(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{\theta}) = -\sum_i b_i v_i - \sum_j b'_j h_j - \sum_{\{i,j\}} W_{i,j} v_i h_j, \tag{6.2}$$

where the summation $\{i, j\}$ is determined by the graph edges, and $\boldsymbol{\theta}$ is the vector of length K of all interaction parameters in the graph. This defines a joint distribution over $\boldsymbol{v}$ and $\boldsymbol{h}$:

$$\tilde{p}(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp[-E(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{\theta})]. \tag{6.3}$$

Each parameter θ_k in this MaxEnt distribution controls a corresponding moment $\tilde{\mu}_k$, given by $\tilde{\mu}_k = \partial \ln Z(\boldsymbol{\theta}) / \partial \theta_k$.

Define a *dynamic Boltzmann distribution* as one with time-dependent interaction parameters:

$$\tilde{p}(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\theta}(t)) = \frac{1}{Z(\boldsymbol{\theta}(t))} \exp[-E(\boldsymbol{v}, \boldsymbol{h}, \boldsymbol{\theta}(t))]. \tag{6.4}$$

For example, the energy function of the RBM becomes:

$$E(\boldsymbol{v},\boldsymbol{h},\boldsymbol{\theta}(t)) = -\sum_i b_i(t)v_i - \sum_j b'_j(t)h_j - \sum_{\{i,j\}} W_{i,j}(t)v_i h_j. \tag{6.5}$$

This is a specific case of a spatial dynamic Boltzmann distribution (5.2) in the discrete lattice limit. To see this, assign to every visible unit v_i a spatial location x_i. By taking self interaction functions $\nu_1(x,t) = \sum_i b_i(t)\delta_{x,x_i}$ in (5.2), we recover the first term in (6.5) with $v_i \in \{0,1\}$, where δ_{x,x_i} is unity if the coordinates are coincident and zero otherwise.

Similarly, hidden units can also be represented in continuous space. Let the species labels α_v denote visible units and β_h denote hidden units, and assign to every hidden unit h_j a spatial location y_j. The weights between layers are then obtained by taking pairwise interactions $\nu_2(x,y,\alpha,\beta,t) = \sum_{i,j} W_{i,j}(t)\delta_{x,x_i}\delta_{y,y_j}\delta_{\alpha,\alpha_v}\delta_{\beta,\beta_h}$ in (5.2).

6.1.1 An adjoint method learning problem for restricted Boltzmann machines

Introduce for each interaction parameter θ_k, $k = 1,\ldots,K$, in the interaction graph a *time-evolution function* F_k forming an autonomous ODE system (analogous to (5.18)):

$$\frac{d}{dt}\theta_k(t) = F_k(\boldsymbol{\theta}(t)), \tag{6.6}$$

with initial condition $\theta_k(t_0) = \theta_{k,0}$. To obtain from the variational problem derived in Section 5.1.2 an ordinary optimization problem for parameters, further consider the parameterization by the vectors $\boldsymbol{u}_k$ of size M_k, generally unique for every k:

$$\frac{d}{dt}\theta_k(t) = F_k(\boldsymbol{\theta}(t);\boldsymbol{u}_k). \tag{6.7}$$

Algorithm 5 Stochastic Gradient Descent for Learning Restricted Boltzmann Machine Dynamics

Initialize
  Parameters $\boldsymbol{u}_k$ controlling the functions $F_k(\boldsymbol{\theta};\boldsymbol{u}_k)$ for all $k=1,\dots,K$.
  Time interval $[t_0,t_f]$, a formula for the learning rate λ.
while not converged **do**
  Initialize $\Delta F_{k,i}=0$ for all $k=1,\dots,K$ and parameters $i=1,\dots,M_k$.
  for sample in batch **do**
    ▷ *Generate trajectory in reduced space* $\boldsymbol{\theta}$*:*
    Solve the PDE constraint (6.7) for $\theta_k(t)$ with a given IC $\theta_{k,0}$ over $t_0\le t\le t_f$, for all k.
    ▷ *Awake phase:*
    Evaluate moments $\mu_k(t)$ of the data for all k,t.
    ▷ *Asleep phase:*
    Evaluate moments $\tilde{\mu}_k(t)$ of the Boltzmann distribution.
    ▷ *Solve the adjoint system:*
    Solve the adjoint system (6.11) for $\phi_k(t)$ for all k,t.
    ▷ *Evaluate the objective function:*
    Update $\Delta F_{k,i}$ as the cumulative moving average of the sensitivity equation (6.10) over the batch.
  ▷ *Update to decrease objective function:*
  $u_{k,i}\to u_{k,i}-\lambda\Delta F_{k,i}$ for all k,i.

Analogously to the continuous case, define as the objective function the KL-divergence between the true and reduced models, p and $\tilde{p}$, over all times (analogous to (5.5)):

$$\begin{aligned} S &= \int_{t_0}^{t_f} dt\, \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}), \\ \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) &= \sum_{\boldsymbol{z}} p(\boldsymbol{z}) \ln \frac{p(\boldsymbol{z})}{\tilde{p}(\boldsymbol{z};\{\boldsymbol{u}\})}. \end{aligned} \tag{6.8}$$

where $\{\boldsymbol{u}\}=\{\boldsymbol{u}_k\}_{k=1}^K$. Minimizing S is thus equivalent to maximizing the log-likelihood of the observed data given the parameters, i.e. $L(\{\boldsymbol{u}\};\boldsymbol{z})=\log\tilde{p}(\boldsymbol{z};\{\boldsymbol{u}\})$. A more common approach is to instead maximize the conditional likelihood of observations conditioned on the first observation: $L(\{\boldsymbol{u}\};z_2,z_3,\dots|z_1)=\log\tilde{p}(z_2,z_3,\dots|z_1;\{\boldsymbol{u}\})$, or similar causal relations.

For Markov chains, this approach is highly successful (leading to e.g. Kalman filters; see [73] for an introduction). If a prior is available, Bayesian methods that compute the posterior $\tilde{p}(\{\boldsymbol{u}\};\boldsymbol{z}) \propto \tilde{p}(\boldsymbol{z};\{\boldsymbol{u}\}) \times \tilde{p}(\{\boldsymbol{u}\})$ can provide further improvements. The advantage of the current approach is that a reduced physical model can be enforced through the parameterization (6.7). This model can be based on prior information, such as reaction networks with known solutions [16]. A second advantage is that the generalization to spatially continuous systems follows naturally using PDEs as in (5.9).

The time integral in S can be lead to undesired extrema, for example for periodic systems where the objective function may not minimize the KL-divergence pointwise. One algorithmic strategy for eliminating these in practice is to shift the limits of integration during the optimization, as further explored in Section 6.2.1.

Minimizing the objective function defines a PDE-constrained optimization problem: minimize (6.8) subject to the PDE-constraint (6.7). Define the Lagrangian function (analogous to (5.10)):

$$\mathcal{L}(t;\{\boldsymbol{u}\}) = \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) + \sum_{k=1}^{K} \phi_k(t) \left(\frac{d}{dt}\theta_k(t) - F_k(\boldsymbol{\theta}(t);\boldsymbol{u}_k) \right), \tag{6.9}$$

where we have introduced the adjoint variables ϕ_k associated with each θ_k. Taking the derivative of the objective function $S = \int_{t_0}^{t_f} dt\, \mathcal{L}(t;\{\boldsymbol{u}\})$ with respect to a parameter gives the sensitivity equation (analogous to (5.19)):

$$\frac{dS}{du_{k,i}} = -\int_{t_0}^{t_f} dt\, \frac{\partial F_k(\boldsymbol{\theta}(t);\boldsymbol{u}_k)}{\partial u_{k,i}} \phi_k(t), \tag{6.10}$$

and taking the derivative with respect to θ gives the ODE system obeyed by the adjoint

variables (analogous to (5.20)):

$$\frac{d}{dt}\phi_k(t) = \tilde{\mu}_k(t) - \mu_k(t) - \sum_{l=1}^{K} \frac{\partial F_l(\boldsymbol{\theta}(t); \boldsymbol{u}_l)}{\partial \theta_k(t)} \phi_l(t), \tag{6.11}$$

where $\mu_k(t')$ and $\tilde{\mu}_k(t')$ are averages taken over to p and $\tilde{p}$ at time t', and the boundary condition is $\phi_k(t_f) = 0$.

Algorithm 5 outlines how this optimization problem can be solved in practice. The inner loop of an "awake" and "asleep" phase of sampling are identical to that of BM learning. Standard algorithmic improvements are possible, such as the use of accelerated gradient descent methods such as Adam [74], and using persistent contrastive divergence (PCD) method [36] to estimate the moments of the reduced model $\tilde{\mu}_k(t')$.

Adjoint methods such as these for solving PDE-constrained optimization problems are also called "black-box" methods [75, 76], since the PDE constraint (6.7) is eliminated in the derivation of the sensitivity equation (6.10). A competing class of methods (sometimes referred to as "all-at-once" methods) treat the constraint explicitly in the optimization, and may offer a computational advantage over this approach. These include sequential quadratic programming (SQP) and augmented Lagrangian methods.

Additional constraints or regularization terms can be included in the optimization, such as conserved quantities identified from the left null space of the net stoichiometry matrix. For example, L_2 regularization can be incorporated into the objective function:

$$S = \int_{t_0}^{t_f} dt\, \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) + \lambda_r \int_{t_0}^{t_f} dt\, \sum_{k=1}^{K} \Big(\theta_k(t) - \overline{\theta}_k(t)\Big)^2, \tag{6.12}$$

where $\overline{\theta}_k(t)$ are some specified functions or otherwise constant, and λ_r is a regularization

parameter. In this case, the adjoint variables are given by:

$$\frac{d}{dt}\phi_k(t) = \tilde{\mu}_k(t) - \mu_k(t) + 2\lambda_r\left(\theta_k(t) - \overline{\theta}_k(t)\right) - \sum_{l=1}^{K}\frac{\partial F_l(\boldsymbol{\theta}(t);\boldsymbol{u}_l)}{\partial\theta_k(t)}\phi_l(t). \tag{6.13}$$

6.1.2 Finite element parameterization

What choice should be made for the parameterization (6.7) of the right hand sides of the differential equations? In [16], we considered simple reaction-diffusion systems from which general forms of approximate models could be inferred that maintain physical interpretations. A second approach also explored in [16] is to use a separate moment closure approximation to derive analytic solutions for simple reaction systems on 1D lattices, where the inverse Ising problem is analytically solvable. The form of (6.7) can then be taken as either linear or non-linear combinations of known solutions.

Here, we take a *finite element method* (FEM) [77] approach to the parameterization that is more aligned with the unsupervised learning problem in a Boltzmann machine. The space of solutions to the general variational problem (5.17), which is some Banach space, is therefore restricted to the space of finite element method solutions.

An important restriction is that the learning rule (6.10) requires C^1 finite elements. One choice for such elements is the Q_3 family of finite elements [78], which in dimensions higher than one are easily constructed as tensor products of 1D cubic polynomials[1]. For C^1 elements that control the value of the function and its derivative at the endpoints, these polynomials are just the Hermite polynomials, shown in Figure 6.1(d).

We therefore introduce for each time-evolution function in (6.7) a domain of hypercubic cells, with 4^d degrees of freedom, where d are the number of arguments to F_k. In practice, we found it is rarely necessary to have more than $d = 3$ arguments (see Section 4.5). For $d = 3$, each cube has 64 degrees of freedom (8 degrees of freedom at each vertex, specifying

[1] An alternative choice for tetrahedral meshes is the P_3 family of finite elements.

the function value and derivatives). For a cubic lattice of $V = L_1 \times L_2 \times L_3$ cells, there are $8V$ degrees of freedom total, with the parameterization taking the usual form in terms of the basis functions f_l associated with each degree of freedom:

$$F_k(\theta_1, \theta_2, \theta_3; \boldsymbol{u}_k) = \sum_{l=1}^{8V} u_l f_l(\theta_1, \theta_2, \theta_3). \tag{6.14}$$

Note that here, the right hand side of the differential equation is parameterized (as opposed to the solution of the differential equation), since the objective of the learning algorithm is to determine a suitable differential equation model.

6.2 Reaction-diffusion systems on lattices

Recall that the state of a reaction-diffusion system at some time is described by n particles of species $\boldsymbol{\alpha}$ located at positions $\boldsymbol{x}$ in generally continuous 3D space. To make an explicit connection to binary random variables, we consider a simpler model of particles hopping on a discrete lattice in the single-occupancy limit. To generate stochastic simulations of such a system, we adapt the method of Ref. [52] for a lattice-based variant of the popular Gillespie stochastic simulation algorithm (SSA) [14] as follows: at each timestep:

1. Perform unimolecular reactions following the standard Gillespie SSA.
2. Iterate over all particles in random order; for each:
   (a) Hop to a neighboring site, chosen at random with equal probability.
   (b) If the site is unoccupied, the move is accepted. If the site is occupied, a bimolecular reaction occurs with some probability; else, the move is rejected and the particle is returned to the original site.

The lattice on which particles hop is designated as the visible part of the MRF. Assign a unique index i to each of the N sites in the lattice, and let the vector of possible

species be $\boldsymbol{s}$ of size M in some arbitrary ordering (excluding $\varnothing$ to denote an empty site). Spins at a site i are now multinomial units, represented as a vector $\boldsymbol{v}_i$ of length M where entries $v_{i,\alpha} \in \{0,1\}$ for $\alpha = 1,\ldots,M$ denote the absence or presence of a particle of species s_α (an n-vector model in statistical mechanics). The single-occupancy limit corresponds to the implicit constraint that the vectors are of unit length, i.e. $\sum_{\alpha=0}^{M} v_{i,\alpha} = 1$, where $\alpha = 0$ denotes an empty site. The matrix $\boldsymbol{V}$ of size $N \times M$ describes the state of the visible part of the MRF, where each row denotes a lattice site.

Likewise introduce hidden layer species $\boldsymbol{s}'$ of size M', which may be different from $\boldsymbol{s}$. Indexing all hidden sites as $j = 1,\ldots,N'$, hidden unit vectors are $\boldsymbol{h}_j$ of length M'. The state of the hidden units is $\boldsymbol{H}$ of size $N' \times M'$, with the single occupancy constraint as before.

The dynamic Boltzmann distribution becomes:

$$\tilde{p}(\boldsymbol{V},\boldsymbol{H}|\boldsymbol{\theta}(t)) = \exp[-E(\boldsymbol{V},\boldsymbol{H},\boldsymbol{\theta}(t))]/Z(\boldsymbol{\theta}(t)), \tag{6.15}$$

where interaction parameters $\boldsymbol{\theta}(t)$ may also be species-dependent (excluding $\varnothing$). For example, the energy function for the RBM becomes:

$$E(\boldsymbol{V},\boldsymbol{H},\boldsymbol{\theta}(t)) = -\sum_{i=1}^{N}\sum_{\alpha=1}^{M} b_{i,\alpha}(t)v_{i,\alpha} - \sum_{j=1}^{N'}\sum_{\beta=1}^{M'} b'_{j,\beta}(t)h_{j,\beta} - \sum_{\{i,j\}}\sum_{\alpha,\beta} W_{i,j,\alpha,\beta}(t)v_{i,\alpha}h_{j,\beta}. \tag{6.16}$$

6.2.1 Learning hidden layers for moment closure

A typical problem in many-body systems is the appearance of a hierarchy of moments, where the time-evolution of a given moment depends on higher order moments. Moment closure approximations terminate this infinite hierarchy at some finite order. In this section, we develop the perspective of the learning problem (6.10) as a closure approximation using

a simple pedagogical example. We note some similarity to previously proposed closure schemes [25, 10], as well as to entropic matching [79], although the current approach differs in the objective function (6.8) and the formulation for spatially continuous systems in Section 5.1.

Consider a bimolecular-annihilation process on a 1D lattice of length N, where particles of a single species A hop and react according to $A + A \to \varnothing$. The time-evolution of the first two moments can be derived from the CME (see example in Appendix C.2) as:

$$\begin{aligned} \frac{d}{dt}\left\langle \sum_i v_i \right\rangle =& -2k_r \left\langle \sum_i v_i v_{i+1} \right\rangle, \\ \frac{d}{dt}\left\langle \sum_i v_i v_{i+1} \right\rangle =& 2D \left\langle \sum_i v_i v_{i+2} \right\rangle - 2k_r \left\langle \sum_i v_i v_{i+1} v_{i+2} \right\rangle + (k_r - 2D) \left\langle \sum_i v_i v_{i+1} \right\rangle, \end{aligned} \tag{6.17}$$

where k_r is the reaction rate and D the diffusion rate. The simplest graph to capture such observables is a fully visible Markov random field, i.e. a 1D Ising model including interactions up to some order. For example, including third order interactions, let:

$$E(\boldsymbol{v}, b(t), J(t), K(t)) = -b(t) \sum_{i=1}^{N} v_i - J(t) \sum_{i=1}^{N-1} v_i v_{i+1} - K(t) \sum_{i=1}^{N-2} v_i v_{i+1} v_{i+2}, \tag{6.18}$$

and let:

$$\begin{aligned} \dot{b} =& F_b(b, J, K; \boldsymbol{u}_b), \\ \dot{J} =& F_J(b, J, K; \boldsymbol{u}_J), \\ \dot{K} =& F_K(b, J, K; \boldsymbol{u}_K), \end{aligned} \tag{6.19}$$

for some parameter vectors $\boldsymbol{u}$ to be learned, where time derivatives are denoted as $\dot{x} = d/dt$. The corresponding graphical model is illustrated in Figure 6.1(b). The choice of the energy function in (6.18) defines which moments are explicitly captured by the reduced model. The additional choice of the form of the differential equations F_γ defines the moment closure

approximation made.

We next show through computational experiments that the introduction of hidden layers can improve upon a fully visible closure model:

1. In any closure scheme, moments beyond a certain order are not captured explicitly by the model, so that their approximation may be poor. The representation power of hidden layers [80] can be used to incorporate information about which higher order moments are relevant to the dataset.
2. Two distinct states having the same lower order moments are indistinguishable in the reduced model (the model is not sufficiently high dimensional). Hidden layers may be able to separate such states if their connectivity is suitably chosen to represent relevant higher order correlations, even if the model remains low order.
3. The number of higher-order terms appearing on the right of (6.17) grows with the order on the left. This problem is compounded if species labels are included. Hidden layers and a restriction on the number of species M' allowed to occupy hidden units may be used to approximate such higher order interactions with fewer parameters.

It is generally difficult to choose the optimal close approximation, i.e. to know which moments are relevant to the time-evolution of a given dataset. A key advantage of the present approach is that the connectivity of the hidden layers may be chosen based on the differential equations derived from the chemical master equation. For example, consider to the bimolecular annihilation system (6.17): if the goal is to accurately model the mean number of particles, then the right hand side of (6.17) shows that the nearest-neighbor moment is relevant to the time evolution. The graphical model of the reduced system could therefore introduce a hidden unit for every pair of neighboring lattice sites, with corresponding energy function:

$$E(\boldsymbol{v},\boldsymbol{h},b(t),W(t),b'(t)) = -b(t)\sum_{i=1}^{N} v_i - b'(t)\sum_{j=1}^{N-1} h_j - W(t)\sum_{i=1}^{N}\sum_{j=i-1,i} v_i h_j, \tag{6.20}$$

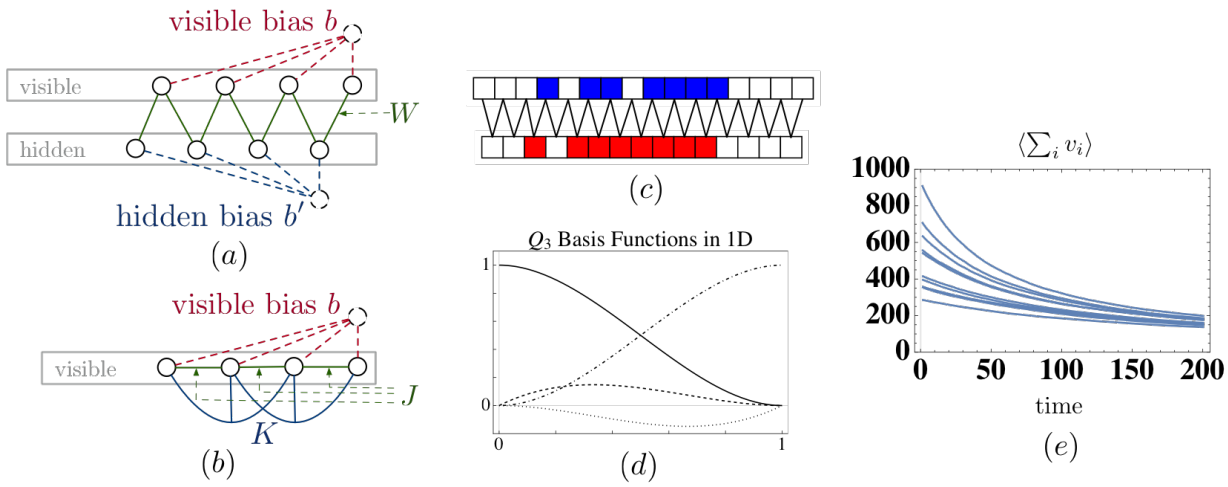

Figure 6.1: Comparison of a fully visible and a latent variable model for capturing local correlations in a 1D lattice. (a) 1D lattice with one hidden layer (similar to an RBM). Note that in this simplified example, W is a single translation invariant parameter rather than a matrix as common in RBMs. (b) Fully visible model for a 1D lattice including nearest neighbor (NN) interactions J and next-nearest neighbors (NNN) K. (c) An example state of the hidden layer model, where blue color indicates the presence of a particle in the visible layer, and likewise red for the hidden layer. By learning the parameters, the hidden layer can be tuned to capture the presence of NNs. (d) The basis functions of the Q_3 family of C^1 finite elements in 1D (Hermite polynomials), used to parameterize the right-hand sides of (6.19,6.21). Basis functions in higher dimensions are constructed as tensor products of the 1D polynomials. (e) Moments of stochastic simulations for 10 of the 50 initial conditions used for training (each trajectory obtained from averaging over 50 lattices simulated from the same initial condition).

and differential equation model:

$$\begin{aligned} \dot{b} &= F_b(b, b', W; \boldsymbol{u}_b), \\ \dot{b}' &= F_{b'}(b, b', W; \boldsymbol{u}_{b'}), \\ \dot{W} &= F_W(b, b', W; \boldsymbol{u}_W). \end{aligned} \tag{6.21}$$

The corresponding graphical model is shown in Figure 6.1(a,c).

The time-evolution functions for (6.19) and (6.21) are learned using Algorithm 1 and compared in Figure 6.2. For the visible model, cells of size $0.5 \times 0.5 \times 0.5$ in (b, J, K) are used, and for the hidden layer model cells of size $0.5 \times 0.5 \times 0.05$ in (b, W, b'), as shown in Figure 6.2.

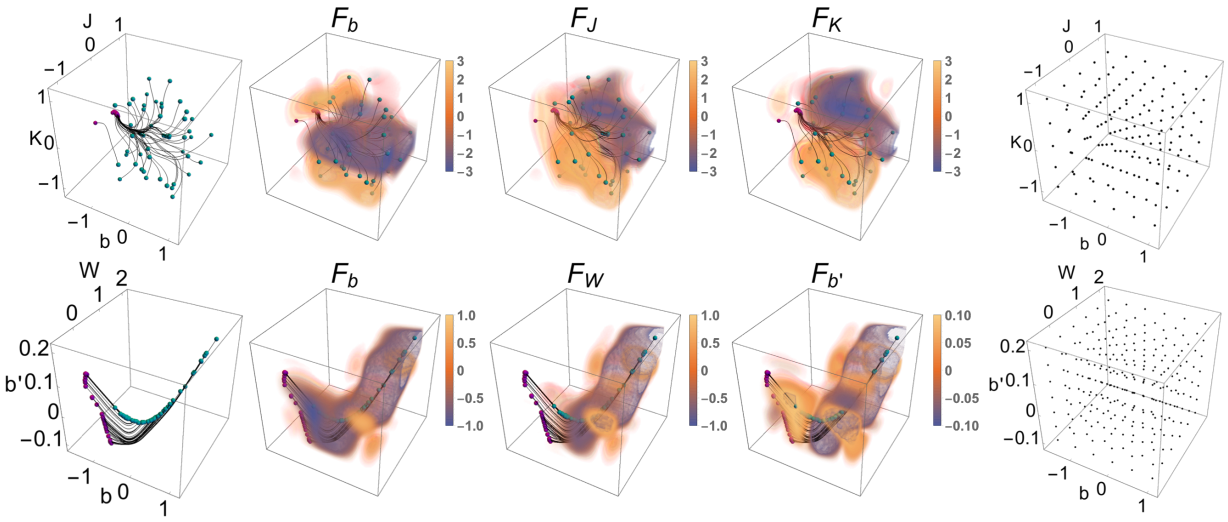

Figure 6.2: *Top row:* Learned time-evolution functions for the fully visible model (6.19), using the Q_3, C^1 finite element parameterization (6.14) with cells of size $0.5 \times 0.5 \times 0.5$ in (b, J, K). Left panel: Training set of initial points (b, J, K) (cyan) sampled evenly in $[-1, 1]$. Stochastic simulations for each initial point are used as training data (learned trajectories shown in black, endpoints in magenta). Middle three panels: the time evolution functions learned, where the heat map indicates the value of F_γ in (6.19). Right panel: vertices of the finite element cells used. *Bottom row:* Hidden layer model (6.21) and parameterization (6.14) with cells of size $0.5 \times 0.5 \times 0.05$ in (b, W, b'). Initial points are generated by BM learning applied to the points of the visible model. Note that the coefficients corresponding to the other seven degrees of freedom at each vertex are also learned (not shown), i.e. the first derivatives in each parameter.

As training data, 50 points (b, J, K) are sampled evenly over $(b, J, K) \in [-1, 1]^3$. Each point corresponds to an initial distribution (6.18), from each of which 50 lattices of length $N = 1000$ are sampled (top left panel of Fig 6.2). The corresponding initial conditions in (b, W, b') space are learned separately using the BM learning algorithm (bottom left panel of Fig 6.2). Each lattice is simulated for 200 timesteps of size $\Delta t = 0.01$ with $p_r = 0.01$, as shown in Figure 6.1(e). These trajectories are pooled for Algorithm 1. Note that a single set of parameter vectors $\{\boldsymbol{u}\}$ in (6.19,6.21) is learned, i.e. the parameter vectors are shared among trajectories from all initial conditions.

For the fully visible model, asleep phase moments are estimated by running a Gibbs sampler for a single step. Similarly, for the hidden model, awake and asleep phase moments

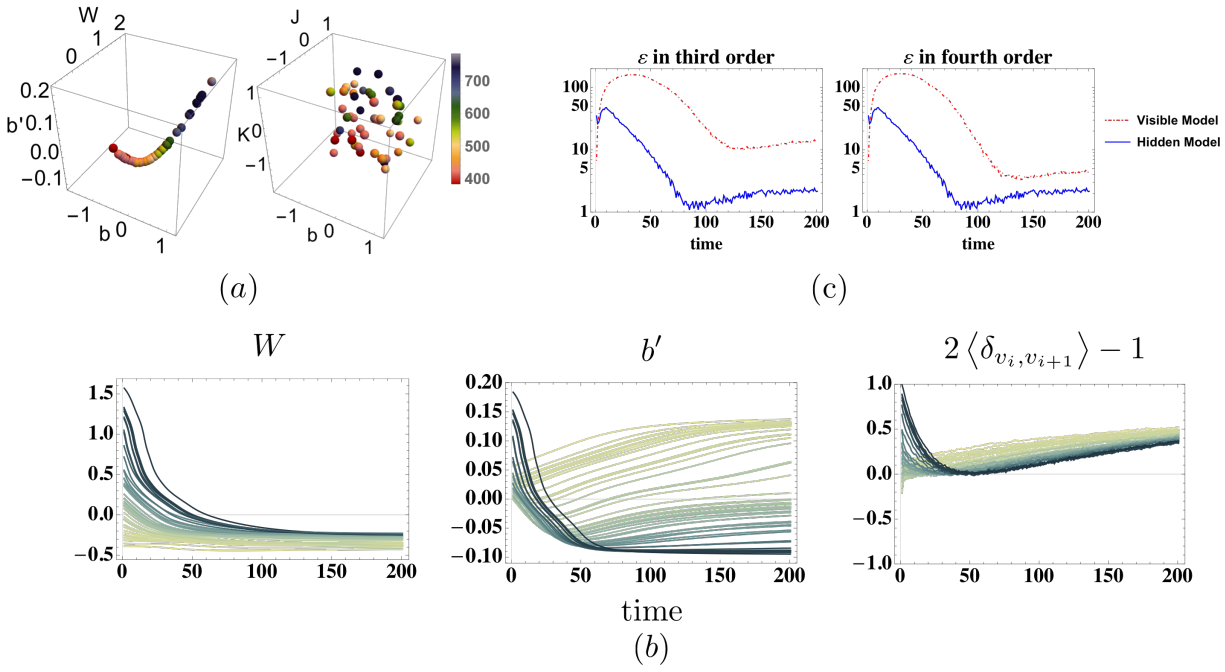

Figure 6.3: (a) Nearest neighbor moment $\langle \sum_i v_i v_{i+1} \rangle$ of the two models. The more compact representation learned by the hidden layer model (left) captures low range spatial correlations, while the fully visible model (right) shows no apparent organization. (b) The parameters W and b' for the hidden layer model for the 50 initial conditions (b is monotonically decreasing for all trajectories). The learned parameters encode the spatial correlation $2\langle v_i v_{i+1} \rangle$ shown on the right. This shows the moment closure approximation learned by the reduced model (see text). (c) RMSE in the third order moment $\langle \sum_i v_i v_{i+1} v_{i+2} \rangle$ and fourth order moment $\langle \sum_i v_i v_{i+1} v_{i+2} v_{i+3} \rangle$, calculated from a set of test trajectories (not shown). Both models reproduce the observables with reasonable accuracy, however, the error in the hidden layer model is lower due to the more compact representation learned.

are estimated by a single step of contrastive divergence, i.e. CD-1. The parameters to Algorithm 1 are learning rate $\lambda = 1$ for 200 optimization steps for both models.

The time integral in the action (6.8) can lead to undesired extrema, e.g. for periodic trajectories. We use an on-line algorithmic solution is to shift the limits of integration in (6.10) as new data is available:

$$\frac{dS}{du_{k,i}} = \int_{\tau}^{\tau+\Delta\tau} dt \, \frac{\partial F_k(\boldsymbol{\theta}(t); \boldsymbol{u}_k)}{\partial u_{k,i}} \phi_k(t), \tag{6.22}$$

where $\Delta\tau$ is fixed, and τ is gradually incremented $t_0 \leq \tau \leq t_f - \Delta\tau$. In this case, the PDE constraint (6.7) is solved from t_0 to τ, decreasing the size of the trajectories early in the training. Further, the adjoint system (6.11) only has to be solved backwards from $\phi(\tau + \Delta\tau) = 0$ to $\phi(\tau)$, which also controls the magnitude of the update steps as the length of the trajectory grows, allowing a constant learning rate to be used. For the annihilation system, we found that fixing $\Delta\tau = 5$ timesteps and shifting $\tau \to \tau + 1$ every 2 optimization steps gave fast convergence.

Figure 6.2 shows the learned time-evolution functions and trajectories of the training data. For the visible model, these show an expected symmetric structure. As particles diffuse and nearest-neighbor (NN) and next-nearest-neighbor (NNN) moments decay, F_J and F_K force $J, K \to 0$ everywhere, while the bias term tends to negative infinity. The representation learned by the hidden layer model is more compact. Figure 6.3(a) shows the nearest neighbor moment $\langle \sum_i v_i v_{i+1} \rangle$ overlaid onto the initial conditions, showing an almost monotonic organization from low to high values by which the model can distinguish these states (no organization is apparent in the visible model). Figure 6.3(b) shows the learned parameter trajectories: b monotonically decreases (not shown), W asymptotically approaches a negative value, and b' either increases monotonically or initially decreases before increasing again. This division corresponds to the decay of spatial correlations $2\langle v_i v_{i+1} \rangle - 1$ (such that 1 corresponds to a fully correlated lattice, and -1 to a fully anti-correlated lattice), also shown in Figure 6.3(b). The two types of trajectories of b' have a clear correspondence to two types of trajectories in the correlation function, and the separation is visible in $F_{b'}$ in the negative and positive regimes. We conclude that the moment closure approximation learned by the model therefore captures relevant low range spatial correlations to approximate the right hand sides of the moment equations (6.17) identified from the CME.

To assess the accuracy of the reduced models, we generate a test set of points

(b, J, K) and the learn the corresponding points (b, W, b') as before. These are evolved in time using the learned DE systems (6.19,6.21). Define $\varepsilon(t) = \sqrt{\langle(\mu(t) - \tilde{\mu}(t))^2\rangle}$ as the root mean square error (RMSE) between some moments of the reduced model $\tilde{\mu}$ and the stochastic simulations μ, where the moments are approximated by averaging over 50 samples. Figure 6.3(c) shows the RMSE for the third order moment $\langle\sum_i v_i v_{i+1} v_{i+2}\rangle$ and fourth order moment $\langle\sum_i v_i v_{i+1} v_{i+2} v_{i+3}\rangle$. Both models have relatively low error in reproducing the observables, however, the error in the hidden layer model is lower than in the visible model. This is because the representation learned by the hidden layer model is more compact, in that states initially distributed uniformly in (b, J, K) space are mapped to an approximately 1D curve in (b, W, b') space. Yet higher accuracies may be possible by further tailoring that parameterizations of the differential equations from the cubic finite elements used here.

6.2.2 Learning the Rössler oscillator

The Williamowski-Rössler oscillator system [81] is a chemical version of a spiral oscillator in three species. The original formulation requires additional species that are fixed at constant concentration. We follow recent work [82] on a volume-excluding version where these constraints are incorporated into pseudo-first order reaction rates. The oscillator for the species A, B, C is dictated by the reaction system:

$$\begin{gathered} A \underset{p_1}{\overset{k_1}{\rightleftharpoons}} 2A \qquad A + B \xrightarrow{p_2} 2B \qquad A + C \xrightarrow{p_3} \varnothing \\ B \xrightarrow{k_2} \varnothing \qquad C \underset{p_4}{\overset{k_3}{\rightleftharpoons}} 2C \end{gathered} \tag{6.23}$$

where the unimolecular reaction rates used are $k_1 = 30, k_2 = 10, k_3 = 16.5$ (arbitrary units), and the probabilities for bimolecular reactions are $p_1 = 0.1, p_2 = 0.4, p_3 = 0.24, p_4 = 0.36$. We simulate this system on a 3D lattice of size $10 \times 10 \times 10$ sites in the single occupancy

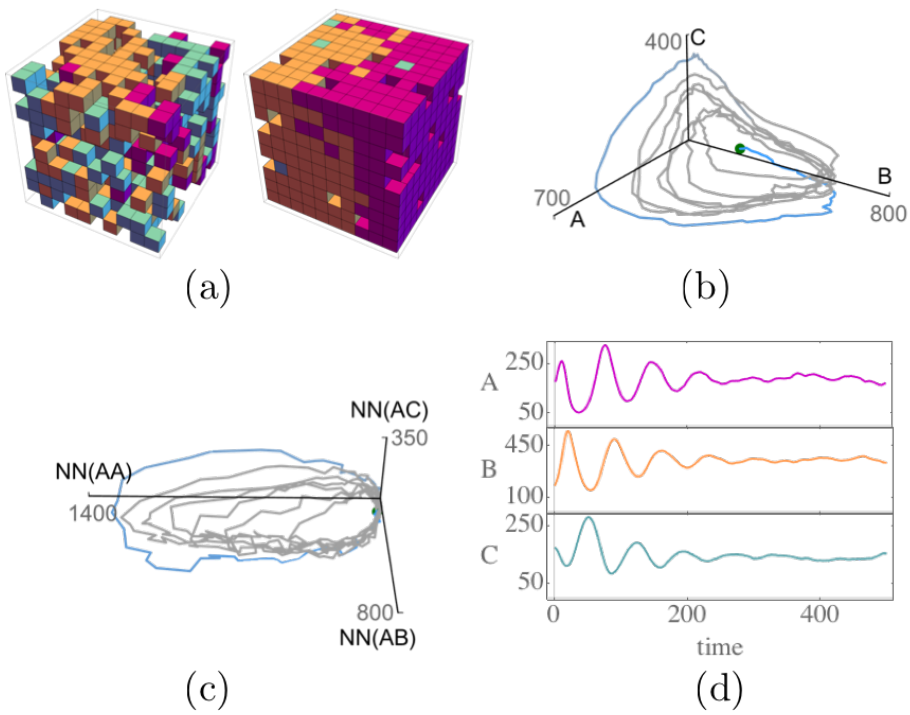

Figure 6.4: Rössler oscillator on a 3D lattice. (a) Snapshots of a stochastic simulation on a $10 \times 10 \times 10$ lattice (A, B, C in pink, orange, cyan). (b) Moments from a single simulation over 500 timesteps, producing a stochastic version of the characteristic attractor of the well-known deterministic model. (c) Nearest neighbor moments in the simulation of (b) show similar structure. (d) Relaxation to a stationary distribution, indicated by the convergence of the means from averaging over 300 stochastic simulations.

limit as before. Figure 6.4 shows snapshots of such a stochastic simulation. Panel (b) in particular shows the characteristic shape of the Rössler oscillator, with further structures evident in higher order moments shown in (c). A snapshot of the spatial waves that occur during transitions between A, B and C-dominated regimes is shown in panel (a).

The time evolution of the mean number of particles in A, B, C, denoted by μ_α, is related to the number of nearest neighbors, denoted by $\Delta_{\alpha\beta}$, as follows:

$$\begin{aligned}
\frac{d}{dt}\mu_A &= k_1\mu_A - \kappa_1\Delta_{AA} - \kappa_2\Delta_{AB} - \kappa_3\Delta_{AC}, \\
\frac{d}{dt}\mu_B &= \kappa_2\Delta_{AB} - k_2\mu_B, \\
\frac{d}{dt}\mu_C &= -\kappa_3\Delta_{AC} + k_3\mu_C - \kappa_4\Delta_{CC},
\end{aligned} \tag{6.24}$$

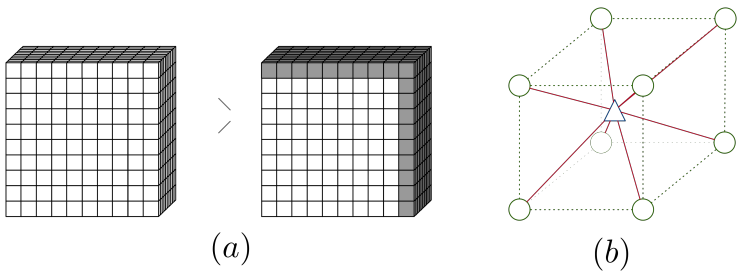

Figure 6.5: (a) Graph to learn for the Rössler oscillator. The lattice on the left corresponds to the visible layer, equivalent to the $10 \times 10 \times 10$ cube in Figure 6.4; the right corresponds to the hidden layer. Gray units in the hidden layer denote those units which implement periodic boundary conditions to the visible layer. (b) Connectivity of hidden layer. Each cube of 8 neighboring units in the visible layer (green circles) is connected to a single unit (blue triangle) in the hidden layer (connections shown in red), resembling a body-centred cubic structure. Biases for the units are not shown.

where $\kappa_1, \kappa_2, \kappa_3, \kappa_4$ are the reaction rates for the bimolecular reactions specified by probabilities p_1, p_2, p_3, p_4 above. As previously, this system is not closed, such that two close initial states in Figure 6.4(b) will diverge over their long term time-evolution. The challenge for the latent variables in the reduced differential equation model is to incorporate relevant higher order correlations to separate states which are close in their lower order moments.

As in Section 6.2.1, let the visible part of the graph be the lattice of Figure 6.4(a). For the hidden layer, we choose a connectivity that coarse-grains the visible lattice by one unit in each spatial dimension as shown in Figure 6.5. Note that the hidden layer is also of size $10 \times 10 \times 10$ units that implement periodic boundary conditions. The visible layer of the graph is multinomial in one of $\{A, B, C, \varnothing\}$, and similarly the hidden layer in $\{X, Y, Z, \varnothing\}$. The corresponding energy model is:

$$\begin{aligned} E(\boldsymbol{V}, \boldsymbol{H}, \boldsymbol{\theta}(t)) = &- \sum_i \sum_{\alpha \in \{A,B,C\}} b_\alpha v_{i,\alpha} - \sum_j \sum_{\alpha \in \{X,Y,Z\}} b_\alpha h_{j,\alpha} \\ &- \sum_{\{i,j\}} \Big(W_{AX} v_{i,A} h_{j,X} + W_{BY} v_{i,B} h_{j,Y} + W_{CZ} v_{i,C} h_{j,Z} \Big), \end{aligned} \tag{6.25}$$

where $\boldsymbol{H}$ refers to the hidden layer, and the sum over $\{i, j\}$ implements the connectivity

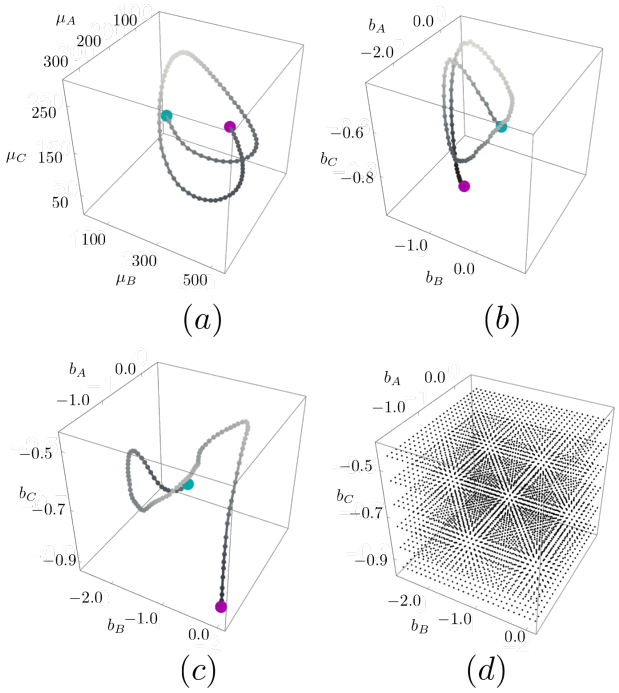

Figure 6.6: (a) The first 100 timesteps of the mean number of A, B, C in the Rössler oscillator system. (b) Interaction parameters for a MaxEnt model constrained on the moments in (a) given by equation (6.27). (c) The learned trajectory of (6.25) in (b_A, b_B, b_C)-space, with initial condition $(-\ln(2), -\ln(2), -\ln(2))$. The bias parameters have been tuned to control both the means and spatial correlations, together with the weights (not shown). Gray scale value indicates b_C component for clarity, scaled from dark ($\min(b_C)$) to light ($\max(b_C)$). Initial point is shown in cyan, and endpoint in magenta. (d) Vertices of the finite element cells of side length 0.1 used to parameterize the differential equations (6.26).

shown in Figure 6.5, and

$$\dot{\gamma} = F_\gamma(b_A, b_B, b_C; \boldsymbol{u}_\gamma), \tag{6.26}$$

for $\gamma \in \{b_A, b_B, b_C, W_{AX}, W_{BY}, W_{CZ}, b_X, b_Y, b_Z\}$. The right hand side of the differential equation is parameterized (6.14) by cubic C^1 finite elements as before. To reduce the complexity of the model, we have purposefully omitted interactions $W_{AY}, W_{AZ}, W_{BX}, W_{BZ}, W_{CX}, W_{CY}$. With this choice, the latent species X coarse grains the visible species A, and similarly for Y, B and C, Z. Note that all differential equation models share the same domain in (b_A, b_B, b_C) space. Note that while the biases h_A, h_B, h_C are the Lagrange multipliers corre-

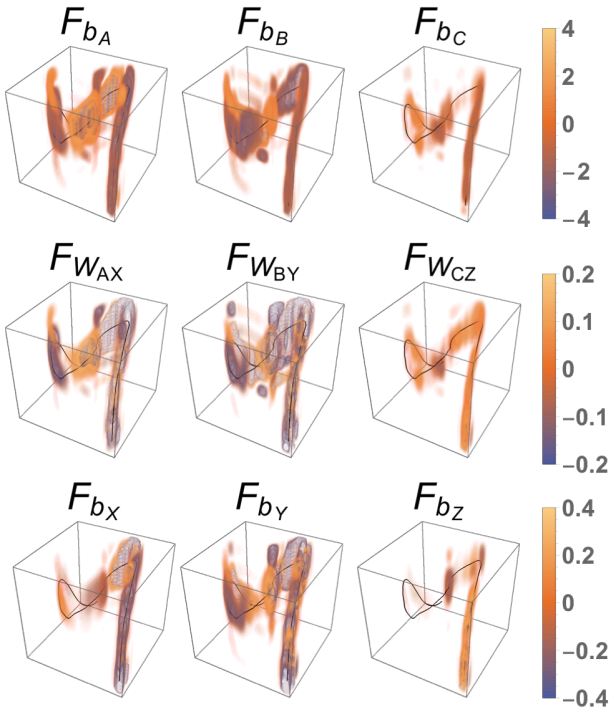

Figure 6.7: Learned time-evolution functions (6.26) in (b_A, b_B, b_C)-space (see Figure 6.6(d) for the vertices used), and the resulting trajectory in black (see Figure 6.6(c)).

sponding to the constraints for the number of particles of each species, through the energy function (6.25) both the biases and weights together also control all spatial correlations of the model.

Stochastic simulations are generated from an initial state with $b_A = b_B = b_C = -\ln(2)$, $W_{AX} = W_{BY} = W_{CZ} = W_{XY} = W_{YZ} = 0$, and $b_X = b_Y = b_Z = -\ln(1/7)$. By setting the initial weights to zero, this is the MaxEnt state given that the number of particles is $\mu_A = \mu_B = \mu_C = 200$, since with zero weight:

$$\mu_\alpha = 1000 \times \frac{e^{b_\alpha}}{1 + \sum_{\beta=A,B,C} e^{b_\beta}} \quad \text{for } \alpha \in \{A, B, C\}, \tag{6.27}$$

where the factor 1000 results from summing over all visible sites. With zero weight, the choice for the initial hidden layer bias is free - by choosing to set it to $-\ln(1/7)$, we are setting the target sparsity to approximately half of that of the visible layer with

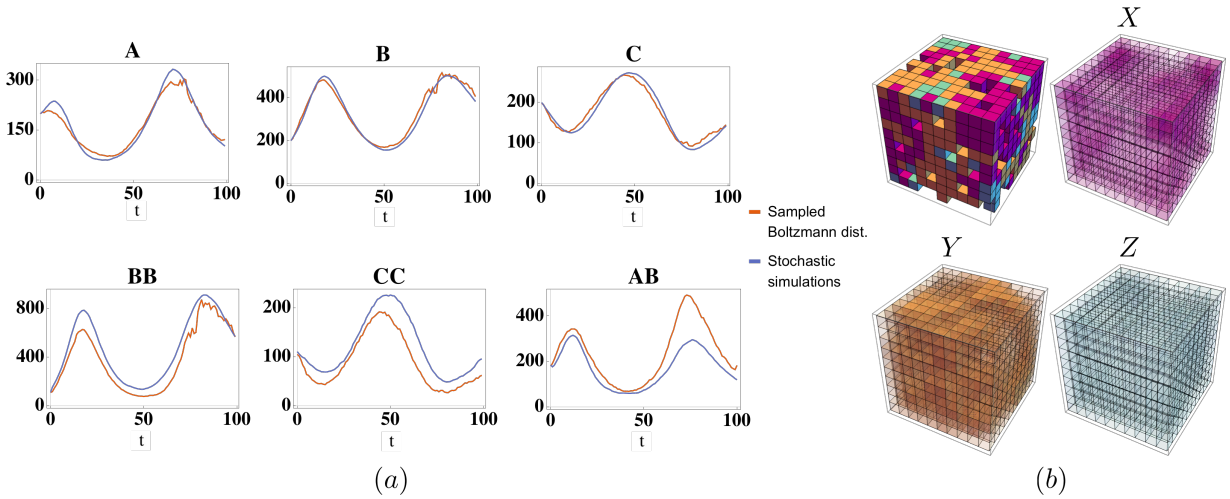

Figure 6.8: (a) Example of correlations learned by the reduced model compared to stochastic simulations, obtained by sampling over 100 samples. Top row: mean number of A, B, C particles. Bottom: neighboring pairs of $(B,B),(C,C)$, and (A,B). Short range spatial correlations relevant to the moment equations (6.24) are reasonably approximated due to the chosen connectivity. (a) Sampled state $\boldsymbol{V}$ from the learned model (top left), and the activated hidden layer probabilities $\tilde{p}(\boldsymbol{H}|\boldsymbol{V})$ at timepoint 20. After training, the hidden layers coarse grain nearest neighbors in the visible layer.

approximately 100 particles of each species as given by (6.27). Simulations are run for 500 timesteps of size $\Delta t = 0.01$. Figure 6.4(d) shows the relaxation of the distribution to equilibrium [83].

For training, we use Algorithm 1 with learning rate $\lambda = 0.05$ for the weights and $\lambda = 0.8$ for the biases for 10000 optimization steps. To estimate the awake phase moments, we sample $\tilde{p}(\boldsymbol{H} = 1|\boldsymbol{V})$ for each sample in a batch size of $\eta = 5$, where $\boldsymbol{V}$ is a data vector. To estimate the asleep phase moments, we alternate between sampling $\tilde{p}(\boldsymbol{H}^{(r)} = 1|\boldsymbol{V}^{(r)})$ and $\tilde{p}(\boldsymbol{V}^{(r)} = 1|\boldsymbol{H}^{(r-1)})$ for $r = 1, \ldots, 10$ steps, starting from a random configuration $\boldsymbol{V}^{(0)}$. Alternatively, we also found fast convergence using $k = 10$ steps of contrastive divergence (CD), as well as using persistent CD. To reduce the noise in the estimates, we use as is common raw probabilities instead of multinomial states for the hidden units when estimating both the awake and asleep phase moments.

As before, we use the online variant (6.22) of Algorithm 1 where the limits of

integration are shifted during training, with window size $\Delta\tau = 10$, and τ is gradually incremented $\tau \to \tau + 1$ every 100 optimization steps. To learn smooth trajectories and avoid jumps in the learned differential equation model, each timestep is divided into 10 substeps when solving the differential equations (6.25,6.26).

Figure 6.7 shows the learned time evolution functions for the Rössler oscillator over the first 100 timesteps. The side length of the cubic finite elements used was 0.1 on all sides, centered at the initial condition, as shown in Figure 6.6(d). We compare the learned trajectories to a simplified MaxEnt model in Figure 6.6(a)-(c). Panel (a) shows the mean number of particles over the first 100 timesteps, as in Figure 6.4(d). Panel (b) transforms these points to the parameters (b_A, b_B, b_C) of a simple MaxEnt model constrained on these lowest order moments as given by (6.27). Panel (c) shows the learned model (6.26), where the biases now control both the means and spatial correlations together with the weights. The trajectory no longer resembles a periodic trajectory, having learned to separate close states in panel (b).

The agreement between the stochastic simulations and reconstructed observables is shown in Figure 6.8(a). At each timepoint, 100 samples are drawn from the reduced model by running 25 steps of CD sampling, starting from a random configuration. Nearest neighbors, which determine the time evolution of the means in (6.24) are reasonably approximated, primarily due to the connectivity chosen in Figure 6.5.

Figure 6.8(b) shows a sampled state $\boldsymbol{V}$ from the learned model, and the activated hidden layer probabilities $\tilde{p}(\boldsymbol{H}|\boldsymbol{V})$ at timepoint 20. With the learned parameters, the hidden units coarse grain nearest neighbors in the lattice, as needed to approximate the right hand side of (6.24). A deeper network such as a deep Boltzmann machine (DBM) may approximate yet higher spatial correlations, and can therefore be used to close differential equation systems depending on higher order moments.

6.3 Notation for multiple hidden layers

In this section, the notation for multiple hidden layers is introduced. Let the lattice on which particles diffuse be the designated as the visible layer of the DBM. Let there be a total of L layers, where $l=0$ denotes the visible layer and $l=1,\ldots,L-1$ the hidden layers, each with $N^{(l)}$ units. Let the units in the l-th layer be one of $M^{(l)}$ species $\mathcal{R}^{(l)}=\{A,B,C,\ldots\}$. The state of each layer is described by $N^{(l)}\times M^{(l)}$ a matrix, where entries $s_{i,\alpha}\in\{0,1\}$ denote the absence or presence of species $\alpha\in\mathcal{R}^{(l)}$ at site $i=1,\ldots,N^{(l)}$. We consider lattices in the single-occupancy limit, corresponding to the implicit constraint $\sum_{\alpha\in\mathcal{R}^{(l)}} s_{i,\alpha}^{(l)}\in\{0,1\}$.

A general energy model for fully connected layers [17] has biases $a_{i,\alpha}^{(l)}$ for unit i in layer l occupied by species α, and weights $W_{i,\alpha,j,\beta}^{(l,l+1)}$ connecting unit i of species α in layer l with unit j of species β in layer $l+1$. In this chapter, we focus on learning hierarchical statistics with a smaller parameter space by making the following simplifications:

1. We consider locally connected layers instead of fully connected, where each unit in layer l is connected to $q^{(l,l+1)}$ units in layer $l+1$.
2. Biases and weights are shared across units in a layer: $a_{i,\alpha}^{(l)}\to a_{\alpha}^{(l)}$ and $W_{i,\alpha,j,\beta}^{(l,l+1)}\to W_{\alpha\beta}^{(l,l+1)}$.

Note that biases and weights remain species dependent. The energy function is:

$$E(\{\boldsymbol{S}^{(l)}\})=-\sum_{l=0}^{L-1}\sum_{\alpha\in\mathcal{R}^{(l)}}a_{\alpha}^{(l)}\sum_{i=1}^{N^{(l)}}s_{\alpha,i}^{(l)}-\sum_{l=0}^{L-2}\sum_{\alpha\in\mathcal{R}^{(l)}}\sum_{\beta\in\mathcal{R}^{(l+1)}}W_{\alpha\beta}^{(l,l+1)}\sum_{\langle ij\rangle}s_{\alpha,i}^{(l)}s_{\beta,j}^{(l+1)},\qquad(6.28)$$

where $\langle ij\rangle$ sums over the local connectivity between two layers. Since each site i in layer l is connected to $q^{(l,l+1)}$ sites in layer $l+1$, then this sum comprises $N^{(l)}\times q^{(l,l+1)}$ terms in total.

6.3.1 Learning rule for DBMs

Maximizing the log likelihood of observed data gives the well known learning rule for DBMs [31]:

$$\Delta W_{\alpha\beta}^{(l,l+1)} = \sum_{\langle ij \rangle} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)} s_{j,\beta}^{(l+1)}\right] \qquad \& \qquad \Delta a_{\alpha}^{(l)} = \sum_{i=1}^{N^{(l)}} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)}\right], \tag{6.29}$$

where $\Delta\mathbb{E}[X] = \langle X \rangle^{(m)} - \langle X \rangle^{(d)}$, and $\langle X \rangle^{(m)}$ denotes an average taken over the model distribution, and $\langle X \rangle^{(d)}$ denotes an average taken over the data distribution, and the sign convention in the update steps is: $a_{\alpha}^{(l)} \to a_{\alpha}^{(l)} - \lambda \Delta a_{\alpha}^{(l)}$ and similarly for $W_{\alpha\beta}^{(l,l+1)}$, with learning rate λ.

Estimating the moments can be done using the well-known wake-sleep algorithm [13]. The moments under the model distribution (sleep phase) are given by Gibbs sampling:

$$\begin{aligned} \tilde{p}(s_{i,\alpha}^{(l)} = 1 | \{\boldsymbol{S}^{(m \neq l)}\}) &= \exp\left[\phi_{i,\alpha}^{(l)}\right] \Big/ \left(1 + \sum_{\zeta \in \mathcal{R}^{(l)}} \exp\left[\phi_{i,\zeta}^{(l)}\right]\right), \\ \phi_{i,\alpha}^{(l)} &= a_{\alpha}^{(l)} + \sum_{\Delta l = \pm 1} \sum_{\beta \in \mathcal{R}^{(l+\Delta l)}} W_{\alpha\beta}^{(l+\Delta l)} \sum_{j \sim i} s_{\beta,j}^{(l+\Delta l)}, \end{aligned} \tag{6.30}$$

where $\sum_{j \sim i}$ sums over units j connected to unit i. Each step of sampling is performed in two phases: one pass for layers with even indexes, and one pass for layers with odd indexes. Estimating the moments under the data distribution (keeping the visible layer clamped at a data vector, i.e. the wake phase) can been done for DBMs by mean field methods [31], or else by Gibbs sampling [40].

6.3.2 Centering transformation and the centered gradient

A centered DBM [41, 40] with parameters $\tilde{a}_\alpha^{(l)}, W_{\alpha\beta}^{(l,l+1)}$ has the energy function:

$$\begin{aligned} E(\{\boldsymbol{S}^{(l)}\}) = &- \sum_{l=0}^{L-1} \sum_{\alpha\in\mathcal{R}^{(l)}} \tilde{a}_\alpha^{(l)} \sum_{i=1}^{N^{(l)}} \left(s_{\alpha,i}^{(l)} - \mu_\alpha^{(l)} \right) \\ &- \sum_{l=0}^{L-2} \sum_{\alpha\in\mathcal{R}^{(l)}} \sum_{\beta\in\mathcal{R}^{(l+1)}} \tilde{W}_{\alpha\beta}^{(l,l+1)} \sum_{\langle ij\rangle} \left(s_{\alpha,i}^{(l)} - \mu_\alpha^{(l)} \right) \left(s_{\beta,j}^{(l+1)} - \mu_\beta^{(l+1)} \right), \end{aligned} \tag{6.31}$$

where $\mu_\alpha^{(l)}$ are the species-dependent centers in layer l. Every regular DBM can be transformed to a centered DBM by transforming parameters as:

$$\tilde{W}_{\alpha\beta}^{(l,l+1)} = W_{\alpha\beta}^{(l,l+1)} \qquad \& \qquad \tilde{a}_\alpha^{(l)} = a_\alpha^{(l)} + \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} W_{\alpha\beta}^{(l,l+\Delta l)} \mu_\beta^{(l+\Delta l)}. \tag{6.32}$$

This can be used to derive the *centered gradient* [41]: After sampling the moments of a regular DBM (6.30), transform to a centered DBM, calculate the gradient with respect to the centered parameters, and transform back to obtain the gradient for the regular DBM parameters. The result is

$$\begin{aligned} \Delta W_{\alpha\beta}^{(l,l+1)} &= \sum_{\langle ij\rangle} \Delta\mathbb{E}\left[(s_{i,\alpha}^{(l)} - \mu_\alpha^{(l)})(s_{j,\beta}^{(l+1)} - \mu_\beta^{(l+1)}) \right], \\ \Delta a_\alpha^{(l)} &= \sum_{i=1}^{N^{(l)}} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)} \right] - \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} \Delta W_{\alpha\beta}^{(l,l+\Delta l)} \mu_\beta^{(l+\Delta l)}, \end{aligned} \tag{6.33}$$

as derived in Appendix D.2. To reduce noise, the centers are updated as the average unit's state with an exponential sliding average with sliding parameter $r \in [0,1]$:

$$\mu_\alpha^{(l)} \leftarrow (1-r)\mu_\alpha^{(l)} + r \times \frac{1}{N^{(l)}} \sum_{i=1}^{N^{(l)}} \left\langle s_{i,\alpha}^{(l)} \right\rangle^{(d)}. \tag{6.34}$$

6.4 Dynamic centered DBMs

Following the DBD framework, escalate to time-varying interactions $a_\alpha^{(l)}(t)$ and $W_{\alpha\beta}^{(l,l+1)}(t)$ in the energy function. To describe the time-evolving interactions, introduce the autonomous differential equation system:

$$\begin{aligned} \frac{d}{dt}a_\alpha^{(l)}(t) &= F_{a_\alpha}^{(l)}(\boldsymbol{\theta}(t);\boldsymbol{u}_{a_\alpha}^{(l)}), \\ \frac{d}{dt}W_{\alpha\beta}^{(l,l+1)}(t) &= F_{W_{\alpha\beta}}^{(l,l+1)}(\boldsymbol{\theta}(t);\boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}), \end{aligned} \tag{6.35}$$

for given initial conditions $a_\alpha^{(l)}(t=0)$, $W_{\alpha\beta}^{(l,l+1)}(t=0)$. Here $\boldsymbol{\theta}(t)$ is a chosen domain of interaction parameters (weights and biases), and F are functions with free parameter vectors $\boldsymbol{u}$ to be learned.

The functions F on the right side can be chosen based on the physics of the system under consideration to learn a reduced physical model. For reaction-diffusion systems, the forms of F can be derived from the CME when using a fully visible Boltzmann distributions [16]. A more blackbox aligned approach is to introduce basis functions $f_m(\boldsymbol{\theta}(t))$ to parameterize (6.35). Following [17] we use the $\mathbb{Q}_3$ family of finite elements [78], which has the advantage that in dimensions higher than one, basis functions are simply tensor products of 1D cubic polynomials $\mathbb{P}_3 \otimes \mathbb{P}_3 \otimes \ldots$. The learning algorithm in Section 6.4.2 requires C_1 functions - these polynomials are therefore the Hermite polynomials that in 1D control four degrees of freedom: the value and the first derivative at each endpoint. For $\boldsymbol{\theta}$ of length d, this gives 4^d degrees of freedom in total and corresponding coefficients $\boldsymbol{u}$ to be learned:

$$F_{a_\alpha}^{(l)}(\boldsymbol{\theta}(t);\boldsymbol{u}_{a_\alpha}^{(l)}) = \sum_{m=1}^{4^d} u_{a_\alpha,m}^{(l)} \times f_m(\boldsymbol{\theta}(t)) \tag{6.36}$$

and similarly for $W_{\alpha\beta}^{(l,l+1)}$, where f_m is the appropriate basis function.

6.4.1 The moment closure approximation for dynamic centered DBMs

The key advantage of the dynamic Boltzmann distribution setting is the moment closure approximation that can be learned from data. The time evolution of how any given moment $\langle X \rangle^{(m)}$ evolves is derived in Appendix D.3. The result is:

$$\begin{aligned}\frac{d\langle X \rangle^{(m)}}{dt} = &\sum_{l=0}^{L-1} \sum_{\alpha \in \mathcal{R}^{(l)}} F_{a_\alpha}^{(l)}(\boldsymbol{\theta}(t); \boldsymbol{u}_{a_\alpha}^{(l)}) \times \sum_{i=1}^{N^{(l)}} \mathrm{Cov}\left(X, s_{i,\alpha}^{(l)}\right) \\ &+ \sum_{l=0}^{L-2} \sum_{\alpha \in \mathcal{R}^{(l)}} \sum_{\beta \in \mathcal{R}^{(l+1)}} F_{W_{\alpha\beta}}^{(l,l+1)}(\boldsymbol{\theta}(t); \boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}) \times \sum_{\langle ij \rangle} \mathrm{Cov}\left(X, s_{i,\alpha}^{(l)} s_{j,\beta}^{(l+1)}\right),\end{aligned} \tag{6.37}$$

where $\mathrm{Cov}(X,Y) = \langle XY \rangle^{(m)} - \langle X \rangle^{(m)} \langle Y \rangle^{(m)}$. The learned differential equation F_ζ of every interaction ζ (weights and biases) contributes to the closure model, weighted by a covariance term between X and the observable for which ζ is the Lagrange multiplier. Equation (6.37) should be directly compared against (6.43). Higher order terms of visible units appearing in (6.43) are exchanged for correlations with latent random variables, whose activations are *learned*.

6.4.2 Adjoint method learning problem with centering transformation

We next formulate a learning problem for the parameters $\boldsymbol{u}$ defining the dynamical system (6.35). This is a specific case of a variational problem for the functions appearing on the right hand side of a differential equation [17, 56], as shown in Appendix D.1. To enforce the constraints (6.35), introduce adjoint variables $\phi_\alpha^{(l)}, \Lambda_{\alpha\beta}^{(l,l+1)}$ to the *centered* parameters $\tilde{a}_\alpha^{(l)}, \tilde{W}_{\alpha\beta}^{(l,l+1)}$. The adjoint equations is derived from the Hamiltonian in Appendix D.4. The

result is:

$$\begin{aligned}
\frac{d}{dt}\phi_\alpha^{(l)} =& \sum_{i=1}^{N^{(l)}} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)}\right] - \psi_{a_\alpha^{(l)}}, \\
\frac{d}{dt}\Lambda_{\alpha\beta}^{(l,l+1)} =& \sum_{\langle ij\rangle} \Delta\mathbb{E}\left[(s_{i,\alpha}^{(l)} - \mu_\alpha^{(l)})(s_{j,\beta}^{(l+1)} - \mu_\beta^{(l+1)})\right] - \psi_{W_{\alpha\beta}^{(l,l+1)}} \\
&+ q^{(l,l+1)}\left(\mu_\alpha^{(l)}\psi_{a_\beta^{(l+1)}} + \mu_\beta^{(l+1)}\psi_{a_\alpha^{(l)}} - \phi_\beta^{(l+1)}\frac{d\mu_\alpha^{(l)}}{dt} - \phi_\alpha^{(l)}\frac{d\mu_\beta^{(l+1)}}{dt}\right),
\end{aligned} \tag{6.38}$$

with boundary conditions $\phi_\alpha^{(l)}(t=\tau+\Delta\tau) = \Lambda_{\alpha\beta}^{(l,l+1)}(t=\tau+\Delta\tau) = 0$, and where

$$\begin{aligned}
\psi_\theta =& \sum_{m=0}^{L-2}\sum_{\zeta\in\mathcal{R}^{(m)}}\sum_{\eta\in\mathcal{R}^{(m+1)}} \Lambda_{\zeta\eta}^{(m,m+1)}\frac{\partial F_{W_{\zeta\eta}}^{(m,m+1)}}{\partial\theta} + \sum_{m=0}^{L-1}\sum_{\zeta\in\mathcal{R}^{(m)}} \phi_\zeta^{(m)}\times \\
&\left(\frac{\partial F_{a_\zeta}^{(m)}}{\partial\theta} + \sum_{\Delta m=\pm 1} q^{(m,m+\Delta m)}\sum_{\eta\in\mathcal{R}^{(m+\Delta m)}}\frac{\partial F_{W_{\zeta\eta}}^{(m,m+\Delta m)}}{\partial\theta}\mu_\eta^{(m+\Delta m)}\right).
\end{aligned} \tag{6.39}$$

While analytic expression for $d\mu_\alpha^{(l)}/dt$ are not available, in practice they can be easily estimated as: $d\mu_\alpha^{(l)}(t)/dt \approx (\mu_\alpha^{(l)}(t+\Delta t) - \mu_\alpha^{(l)}(t))/\Delta t$.

The sensitivity (update) equations for the parameters $\boldsymbol{u}$ to be learned are derived in Appendix D.4. The result is:

$$\begin{aligned}
\frac{dS}{d\boldsymbol{u}_{a_\alpha}^{(l)}} =& -\int_\tau^{\tau+\Delta\tau} dt\ \phi_\alpha^{(l)}\frac{\partial F_{a_\alpha}^{(l)}}{\partial \boldsymbol{u}_{a_\alpha}^{(l)}}, \\
\frac{dS}{d\boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}} =& -\int_\tau^{\tau+\Delta\tau} dt\ \left(\Lambda_{\alpha\beta}^{(l,l+1)} + q^{(l,l+1)}\phi_\alpha^{(l)}\mu_\beta^{(l+1)} + q^{(l,l+1)}\phi_\beta^{(l+1)}\mu_\alpha^{(l)}\right)\frac{\partial F_{W_{\alpha\beta}}^{(l,l+1)}}{\partial \boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}},
\end{aligned} \tag{6.40}$$

where the update step is: $\boldsymbol{u} \to \boldsymbol{u} - \lambda \times (dS/d\boldsymbol{u})$ with learning rate λ.

Algorithm 6 is an example of how the learning problem can be solved in practice. Alternative approaches for solving PDE-constrained optimization problems can be applied, such as sequential quadratic programming (SQP). A benefit of the current algorithm is its

simplicity - the inner loop at each timestep is equivalent to the wake/sleep phases of the Boltzmann machine learning algorithm [13]. A lower bound on the log-probability of test data can therefore be obtained using established methods such as Annealed Importance Sampling (AIS) [31]. It is naturally possible to use accelerated gradient descent methods such as Adam [69].

Algorithm 6 Learning algorithm for dynamic centered DBMs

Input: Initial conditions for all interactions parameters, time interval $[0,T]$, a formula for the learning rate λ, sliding factor r, batch size η, time window size $\Delta\tau$, a formula for sliding τ.
Initialize: $\tau = 0$, $\mu_\alpha^{(0)}$ to the data means, otherwise $\mu_\alpha^{(l)} = 1/(M^{(l)}+1)$.
while not converged **do**
    for timepoint t in $[\tau, \tau+\Delta\tau]$ with timestep Δt **do**
        ▷ Solve the constraints (6.35) for the current timepoint t from the previous timepoint $t-\Delta t$.
        ▷ Estimate wake & sleep phase moments (6.38) over the batch (e.g. Gibbs sampling).
        ▷ Update the centers $\mu_\alpha^{(l)}(t)$ according to (6.34).
        ▷ Calculate & store derivative terms $\partial F/\partial\theta$ in (6.38) and $\partial F/\partial \boldsymbol{u}$ in (6.40).
    ▷ Solve the adjoint system (6.38) backwards in time from $t=\tau+\Delta\tau$ to $t=\tau$.
    ▷ Update the parameters $\boldsymbol{u}_{a_\alpha}^{(l)}$ and $\boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}$ according to (6.40) with learning rate λ.
    ▷ Slide the time window at τ forward if necessary to eventually cover $[0,T]$.

6.5 Numerical example for the Lotka-Volterra system

We demonstrate Algorithm 6 for learning a moment closure approximation for a Lotka-Volterra system using stochastic simulations as training data. Note that we only

consider a single initial condition in this work - however, it is possible to learn a larger parameter space from stochastic simulations with varying initial conditions [17].

For completeness, we briefly review the appearance of a hierarchy of moments in a spatial probabilistic system. Consider the Lotka-Volterra predator-prey system, described by the reactions:

$$\varnothing \xrightarrow{k_1} P, \qquad H \xrightarrow{k_2} \varnothing, \qquad H + P \xrightarrow{k_3} H + H, \tag{6.41}$$

where H and P denote the predator (hunter) and prey populations and k_1, k_2, k_3 are reaction rates.

In a well mixed system where spatial effects are not included, the mean number of H and P, denoted by μ_H and μ_P, obey the following ordinary differential equation (ODE) system:

$$\frac{d\mu_P(t)}{dt} = k_1\mu_P(t) - k_3\mu_H(t)\mu_P(t) \quad \& \quad \frac{d\mu_H(t)}{dt} = k_3\mu_H(t)\mu_P(t) - k_2\mu_H(t). \tag{6.42}$$

These differential equations are solvable for given initial conditions $\mu_P(t=0)$ and $\mu_H(t=0)$.

In a spatial setting, consider a lattice in the single occupancy limit, where $s_{i,\alpha} \in \{0,1\}$ describes the i-th lattice site occupied by species $\alpha \in \{H, P\}$. The mean number of H and

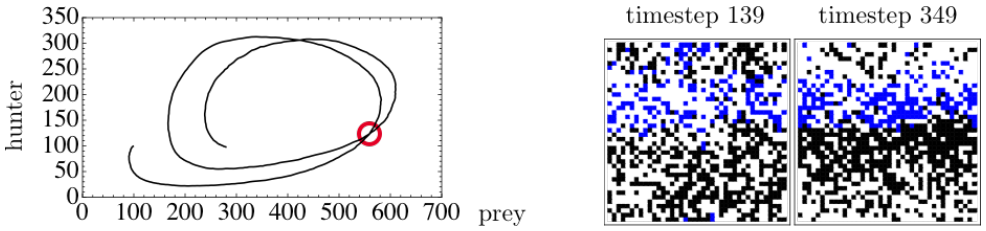

Figure 6.9: *Left:* Mean number of hunter and prey in the Lotka-Volterra system starting from 100 particles of each population (see text). *Right:* Two lattices at an intersection point in the moment space of the left panel (red circle). Each have close to the average number of particles (550 prey in black and 140 hunters in blue), but different spatial correlations lead to differing time derivatives (6.43).

P obey:

$$\begin{aligned}\frac{d}{dt}\left\langle \sum_i s_{i,P} \right\rangle &= k_1 \left\langle \sum_i s_{i,P} \right\rangle - k_3 \left\langle \sum_{\langle ij \rangle} s_{i,H} s_{j,P} \right\rangle, \\ \frac{d}{dt}\left\langle \sum_i s_{i,H} \right\rangle &= k_3 \left\langle \sum_{\langle ij \rangle} s_{i,H} s_{j,P} \right\rangle - k_2 \left\langle \sum_i s_{i,H} \right\rangle,\end{aligned} \tag{6.43}$$

where $\sum_i$ sums over lattice sites, and $\sum_{\langle ij \rangle}$ over neighboring sites. These equations do not close - a nearest neighbor moment appears on the right hand side. Further, the differential equation describing this term depends on yet higher order ones, e.g. next-nearest neighbors, or three particle correlations (if particles are allowed to diffuse, the diffusion constant would appear here). Moment closure methods seek to obtain tractable approximations to this infinite hierarchy of differential equations.

Alternative to the deterministic approach of solving differential equations such as (6.42), stochastic approaches can be used. For example, the Gillespie stochastic simulation algorithm (SSA) can be derived from the CME [14, 24]. Spatial adaptations of the Gillespie SSA can similarly be used to generate stochastic simulations of (6.43), including in continuous space [6]. Here we adopt a simple lattice based algorithm [17] in which particles hop on a grid, undergo unimolecular reactions following the Gillespie SSA, and bimolecular reactions with some chosen probability upon encounters.

To simulate the Lotka-Volterra system, we use a 2D lattice of 40×40 sites with von Neumann connectivity and periodic boundary conditions. For reaction rates, we use $k_1 = 0.025, k_3 = 0.06$, and bimolecular reaction probability 0.4 corresponding to k_2. The initial state has 100 particles of hunter and prey randomly distributed, and simulations are run for 500 timesteps with $\Delta t = 1$. Figure 6.9 shows snapshots of these simulations, which feature spatial patterns including waves.

We generate 100 simulations as training data used in the rest of this chapter. The mean number of hunter and prey is shown in Figure 6.9. Self intersections in this low order

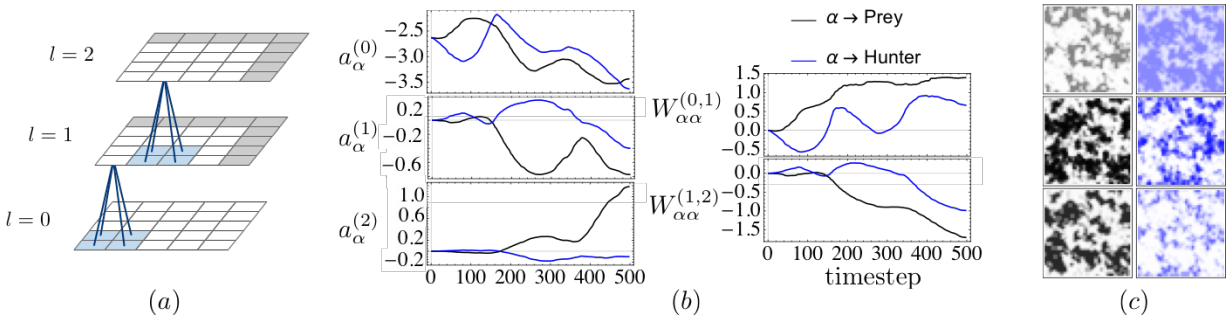

Figure 6.10: (a) Locally connected DBM in 3 layers with 40×40 units each (5×5 illustrated). Every 2×2 patch of units in layer l (blue) is connected to a single unit in layer $l+1$ with two species-dependent weights $W_{HH}^{(l,l+1)}$ and $W_{PP}^{(l,l+1)}$. Gray units implement periodic boundary conditions to the layer below. (b) Interactions learned for the Lotka-Volterra system using Algorithm 6 (see text). (c) Spatial patterns in the layers of (a) at timepoint 370 after training, obtained by 100 steps of Gibbs sampling from a random configuration (raw probabilities shown for prey in black, hunter in blue).

moment space reflect the dependence on spatial correlations which differ over time. The challenge for a deep learning problem is to identify relevant higher order correlations to separate states identical in low order moments, and to learn a closure model for expressing these correlations in terms of a finite number of parameters.

For the architecture of the DBM, we use a locally connected DBM with one visible and two hidden layers as shown in Figure 6.10(a). Each 2×2 patch in layer l is connected to a single unit in layer $l+1$. The number of units in each layer is held constant at 40×40, with boundary units implementing periodic boundary conditions to the layer below, reflecting those used in the stochastic simulations.

Many systems feature a large number of species, e.g. states of ion channels as different sub-units are activated. Having $M^{(l)}$ species in layer l and $M^{(l+1)}$ species in layer $l+1$ leads to $M^{(l)} \times M^{(l+1)}$ species dependent weights. To limit this parameter inflation, we consider two species H, P in each layer, and weights $W_{HH}^{(0,1)}, W_{PP}^{(0,1)}$ and $W_{HH}^{(1,2)}, W_{PP}^{(1,2)}$. We found the omitted cross-species weights to be less important, as a similar effect is produced by crowding in the hidden layers during sampling.

As initial condition, we use the maximum entropy distribution consistent with the 100

initial hunter and prey particles. This corresponds to initial parameters $a_H^{(0)} = a_P^{(0)} = -2.63$, and all other weights and biases set to zero. The algorithm therefore must learn to activate latent variables to control spatial correlations. Starting from zero weights and biases allows the hidden layers to enforce either a strengthening or suppressing (positive or negative weights) relationship with neighboring layers. We restrict the domain of the differential equations (6.35) to be $\boldsymbol{\theta} = (a_H^{(0)}, a_P^{(0)}, W_{HH}^{(0,1)}, W_{PP}^{(0,1)}, W_{HH}^{(1,2)}, W_{PP}^{(1,2)})$. Since the distribution starts with only the visible biases as non-zero, it is important to include these in the domain. When training on stochastic simulations from many initial conditions, using fewer parameters in the domain improves generalization [17]. The side length of the cubic cell (6.36) used in all dimensions was 0.1.

Algorithm 6 is implemented as a C++ library freely available online [84]. We use simple batch gradient descent with batch size $\eta = 5$, learning rate $\lambda = 2.5 \times 10^{-6}$ for all parameters for 1000 optimization steps, and sliding factor $r = 0.5$ (the center is averaged over all units allowing a larger factor than in non-dynamic centered DBMs). We use 10 steps of Gibbs sampling for estimating both the wake and sleep phase moments, with persistent chains in the sleep phase [36]. The differential equations (6.35,6.38) are solved using Euler's method with the step size from the stochastic simulations. The time window in (6.40) is of size $\Delta\tau = 10$ timesteps, and we slide $\tau \to \tau + 1$ every two optimization steps. Figure 6.10(b) shows the learned parameter trajectories. An example of the hidden layer states is shown in Figure 6.10(c), showing the learned hierarchical representation of spatial patterns in the hidden layers.

The moment closure approximation (6.37) is of particular interest. Here it can be analyzed which weighted covariance terms the system has learned that contribute to the time evolution of an observable. Consider for example the mean number of prey $\langle X \rangle^{(m)} = \sum_{i=1}^{N^{(0)}} \left\langle s_{i,\alpha}^{(0)} \right\rangle^{(m)}$. Figure 6.11, top-left, shows the time evolution of this observable under the trained model obtained by averaging over 100 sampled lattices (10 steps of Gibbs

sampling from a random configuration) using the learned parameters at each timestep. For testing, we compare the trajectory against stochastic simulations, which agree well (for training on varying initial conditions, a test data set of stochastic simulations may be used [17]). Next, we calculate each term in (6.37), shown in rows two and three (to reduce noise, we first smooth the interactions with a low-pass filter with cutoff 0.1 before sampling and calculating these terms). The sign of weight and bias terms is generally opposite - this reflects the selective activation of units based on the 2×2 patches in neighboring layers, rather than broad, spatially uncorrelated activations. To validate the accuracy of these terms, we also plot their sum in the bottom row, which is nearly identical to the true time derivative $d\langle X \rangle^{(d)}/dt$ calculated from the stochastic simulations.

The two oscillation cycles in Figure 6.9 evolve differently due to the dependence on higher order moments. How is this difference reflected in the learned moment closure approximation? In the first oscillation in Figure 6.11, the time evolution is primarily driven by the weight term $W_{PP}^{(0,1)}$ from the first hidden layer, indicating that mainly nearest neighbor structure relevant. In the second oscillation, the contribution from terms in the second hidden layer has grown significantly, indicating longer range correlations are relevant. Indeed, visually inspecting samples of the stochastic simulations (see Figure 6.9) shows that larger domains of hunter and prey are formed in the second oscillation.

Examining different quantities $\langle X \rangle^{(m)}$ in this fashion gives insight into the learned moment closure approximation. In the center column of Figure 6.11, we examine the mean number of nearest neighbors of prey. The terms in (6.37) are approximately scaled version of the terms for the mean number of prey. This is expected, since the reaction system does not explicitly give a source or sink for nearest neighbor correlations between prey. Not all terms are accurately captured. For example, the right column shows the mean number of next-nearest neighbors (Manhattan distance two) of hunter and prey, which are overestimated in the learned model. This is explained by weight terms from hunter

and prey both contributing positively to the time evolution, rather than competitively as previously.

6.6 Discussion

This chapter introduced a method for learning moment closure approximations from data using multiple hidden layers. The finite element parameterization is similar to the unsupervised learning setting of RBMs in the sense that it is independent of the system under consideration. For deeper architectures such as DBMs as discussed in Section 6.2.2, recycling the same time-evolution functions across multiple layers may be effective, similar to convolution layers in convolutional neural networks. Factoring weights has also been used effectively in deep learning [85], and may similarly reduce the computational burden here.

A key result is the closure equation (6.37), which replaces long range spatial correlations in the visible layer with correlations with latent variables, whose activation is learned. The learning problem is that for a dynamic Boltzmann distribution, combined with the architecture of a DBM. The centering transformation from centered DBMs is extended to the adjoint system for the dynamic case, such that pre-training is unnecessary. The hierarchical architecture in Figure 6.10(a) is tailored to reflect the moment equations derived CME (6.43), naturally capturing correlations relevant to the moment closure problem. A further important result is the use of multinomial variables in the hidden layers, which allows interpretable learned representations as in Figure 6.10(c).

Further avenues for improvement exist. It may be possible to adapt the "serendipitous" family of $\mathbb{Q}_3$ finite elements [78] to reduce the number of basis functions and therefore computational cost [86], although it is not C_1, requiring modifications to the learning problem. For modeling applications, physically informed parameterizations are particu-

larly interesting, e.g. for reaction-diffusion systems [17], and generally in mathematical modeling [71].

6.7 Acknowledgements

Sections 6.1 and 4.5 are a reprint of material, with minor edits as it appears in: O.K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, “Learning moment closure in reaction-diffusion systems with spatial dynamic Boltzmann distributions", *Phys. Rev. E*, vol. 99, no. 6, pp. 063315, Jun. 2019.

Sections 6.3,6.4,6.5, and 6.6 are a reprint of material, with minor edits as it appears in: O.K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, “Deep Learning Moment Closure Approximations using Dynamic Boltzmann Distributions", *arXiv*:1905.12122, 2019.

The author of the dissertation was the primary author of these papers.

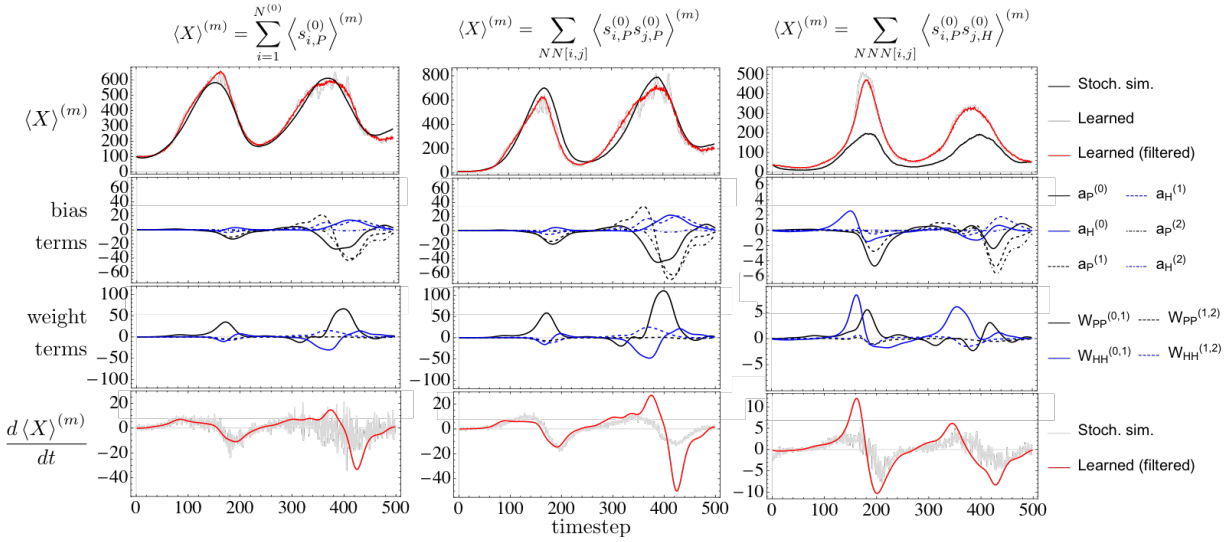

Figure 6.11: *Columns:* Quantities of interest $\langle X\rangle^{(m)}$. *Top row:* Comparison with stochastic simulations. *Second, third:* Terms in the moment closure approximation (6.37). *Bottom:* Derivative from summing the second & third rows as in (6.37) compared to the derivative from stochastic simulations.

Chapter 7

Physics-based machine learning for modeling stochastic IP3-dependent calcium dynamics

In the previous chapters, the DBD method for modeling reaction-diffusion systems has been introduced and its learning problem studied for different applications. In this chapter, we return to the main motivation of the method for physics-based machine learning. It is shown how domain-specific knowledge can be introduced into the framework through candidate functions. The ideas are demonstrated to model inositol 1,4,5-trisphosphate (IP_3) induced calcium oscillations that occur in non-excitable cells. The physics-based ML method is shown to outperform an equivalent domain-agnostic method in generalization performance, model size and interpretability of the learned representations.

7.1 Introduction

Modeling physical systems with machine learning is a growing research topic. Machine learning offers inference methods that can be computationally more efficient than first principles approaches, and that can generalize well from high dimensional datasets. Their successes in science span protein structure prediction [87] to solutions to the quantum many-body problem [37].

A key challenge is how to incorporate prior knowledge into the learning problem [88, 89, 90]. This includes physical laws, symmetries and conservation laws. For example, kernel methods have been improved by encoding symmetries [91], and convolutional neural networks (CNNs) have benefited from pose estimation [92]. However, it remains difficult to introduce domain knowledge such as physical laws into machine learning. Often, methods are used in a domain agnostic way [93, 94, 95, 96], in that physical processes are not introduced explicitly, but rather only implicitly present in the training data. For some applications this is an advantage [97, 98], but for scientific modeling it has at least three deficits. First, models can be challenging to train, having to internally rediscover already known function forms from large amounts of training data. Second, models can be difficult to interpret, requiring a large number of parameters to explain behavior that from first principles may be low dimensional. Third, the trained models may generalize poorly compared to approaches incorporating physical principles.

This chapter introduces a method for modeling stochastic reaction networks that incorporates knowledge from the chemical master equation (CME) into the inference problem. This is made possible by representing the right hand side of a differential equation by a neural network [10, 16, 17, 99, 100, 101]. By using analytically derived approximations as inputs, the network is shown to improve generalization for a classic model of IP_3 dependent calcium oscillations [102]. Additionally, reaction network conservation laws are incorporated into the framework. From a subset of stochastic simulations, the trained model

completes the full range of oscillations, and outperforms an equivalent domain-agnostic model. The proposed approach is one avenue to improve machine learning for scientific modelling with domain-specific knowledge.

7.1.1 Chemical kinetics at the fine scale

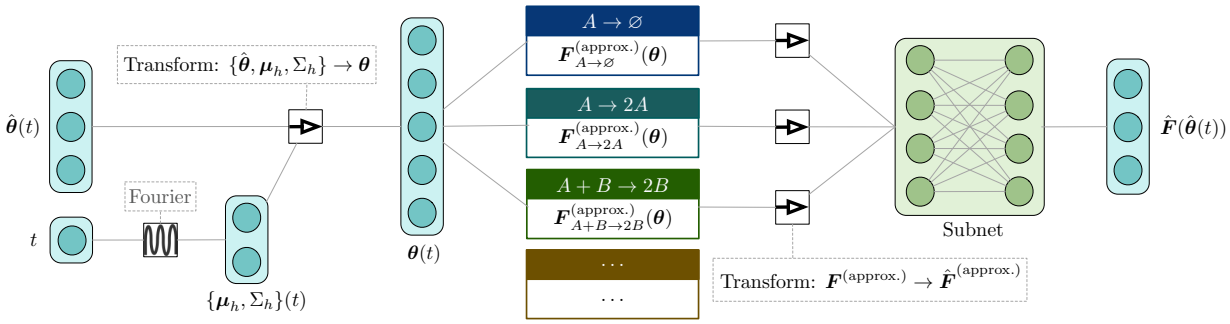

Figure 7.1: Architecture for the differential equation model (7.11), where the right hand side is parameterized by a neural network. Inputs and outputs of the subnet are also standardized.

Consider a system described by the number of particles $\boldsymbol{n} = \{n_A, n_B, \dots\}$ of species $\{A, B, \dots\}$. The time evolution of the probability distribution over states $p(\boldsymbol{n}, t)$ is described by the CME:

$$\frac{dp(\boldsymbol{n},t)}{dt} = \sum_{r=1}^{R} \sum_{\boldsymbol{n}'} \Big[T_r(\boldsymbol{n}|\boldsymbol{n}')p(\boldsymbol{n}',t) - T_r(\boldsymbol{n}'|\boldsymbol{n})p(\boldsymbol{n},t) \Big], \tag{7.1}$$

where $T_r(\boldsymbol{n}|\boldsymbol{n}')$ is the propensity for the transition $\boldsymbol{n}'$ to $\boldsymbol{n}$ under a reaction indexed by r.

Only the simplest reaction networks are solvable exactly or perturbatively in the Doi-Peliti operator formalism [103]. Further, the differential equations for moments derived from (7.1) generally do not close - equations for lower order moments depend on higher orders. For example, for $A + B \to 2B$:

$$\begin{aligned} \frac{d\langle n_A \rangle}{dt} &\propto -\langle n_A n_B \rangle, \\ \frac{d\langle n_A n_B \rangle}{dt} &\propto -\langle n_A n_B^2 \rangle + \langle n_A^2 n_B \rangle - \langle n_A n_B \rangle. \end{aligned} \tag{7.2}$$

This infinite hierarchy requires a moment closure approximation, such as the Gaussian closure approximation:

$$p(\boldsymbol{n}, t) \sim \mathcal{N}(\boldsymbol{n}|\boldsymbol{\mu}_p(t), \Sigma_p(t)), \tag{7.3}$$

where $\boldsymbol{\mu}_p, \Sigma_p$ are the mean and covariance under p at an instant in time. In practice, it is challenging to choose the optimal closure approximation, since it is not clear which higher order moments will become relevant over long times.

Alternatively, the Gillespie algorithm [14] can be used to simulate stochastic trajectories of reaction networks. This is popular in biology [6, 3], at the cost of computation time for collecting sufficient statistics. Motivated by data-driven methods, we next propose a framework to learn closure approximations from stochastic simulations.

7.2 Physics-based machine learning

7.2.1 Reduced model

We seek a reduced model that can be trained on stochastic simulations, but also incorporates physical knowledge to improve generalization. This connection can be made by a dynamic Boltzmann distribution (DBD) [16, 17, 18], consisting of an effective probability distribution with time-dependent interactions $\boldsymbol{\theta}(t)$ in the energy function:

$$\tilde{p}(\boldsymbol{n}; \boldsymbol{\theta}(t)) = \frac{1}{Z(t)} \exp[-E(\boldsymbol{n}; \boldsymbol{\theta}(t))], \tag{7.4}$$

and a differential equation system for the parameters:

$$\frac{d\boldsymbol{\theta}(t)}{dt} = \boldsymbol{F}(\boldsymbol{\theta}(t); \boldsymbol{u}), \tag{7.5}$$

for some functions $\boldsymbol{F}$ with parameters $\boldsymbol{u}$, with a given initial condition $\boldsymbol{\theta}(t=0)=\boldsymbol{\theta}_0$. The Boltzmann distribution *ansatz* is motivated by the connection to graphical models [10]. In this work, the reduced model (7.4) considered is that of probabilistic principal component analysis (PCA), a popular choice for dimensionality reduction [26]. The parameters in the energy function are:

$$\boldsymbol{\theta}(t)=\{\boldsymbol{b},W,\sigma^2,\boldsymbol{\mu}_h,\Sigma_h\}(t), \tag{7.6}$$

and the distribution is Gaussian:

$$\begin{aligned}\tilde{p}(\boldsymbol{n};\boldsymbol{\theta}(t))&=\mathcal{N}(\boldsymbol{n}|\boldsymbol{\mu}(t),C(t)),\\ \boldsymbol{\mu}(t)&=\begin{pmatrix}\boldsymbol{b}+W\boldsymbol{\mu}_h\\ \boldsymbol{\mu}_h\end{pmatrix}(t),\\ C(t)&=\begin{pmatrix}WW^\intercal+\sigma^2 I & W\Sigma_h\\ \Sigma_h W^\intercal & \Sigma_h\end{pmatrix}(t).\end{aligned} \tag{7.7}$$

Splitting the species into visible $\boldsymbol{n}_v$ of size N_v and hidden $\boldsymbol{n}_h$ of size N_h gives the more familiar form:

$$\begin{aligned}\tilde{p}(\boldsymbol{n}_h;\boldsymbol{\theta}(t))&=\mathcal{N}(\boldsymbol{n}_h|\boldsymbol{\mu}_h,\Sigma_h),\\ \tilde{p}(\boldsymbol{n}_v|\boldsymbol{n}_h;\boldsymbol{\theta}(t))&=\mathcal{N}(\boldsymbol{n}_v|\boldsymbol{b}+W(\boldsymbol{\mu}_h+\boldsymbol{n}_h),\sigma^2 I).\end{aligned} \tag{7.8}$$

7.2.2 Maximum likelihood at an instant in time

At an instant in time, $\boldsymbol{\mu}_h$ and Σ_h are arbitrary; across time, the differential equation (7.5) depends on these variables. For $\boldsymbol{\mu}_h=\boldsymbol{0}$ and $\Sigma_h=I$, the maximum likelihood

(ML) solution is:

$$\begin{aligned}
\hat{W}_{\mathrm{ML}}(t) &= U_q(t)(L_q(t) - \sigma^2_{\mathrm{ML}}(t)I)^{1/2}R, \\
\sigma^2_{\mathrm{ML}}(t) &= \frac{1}{N_v - q}\sum_{i=q+1}^{N_v}\lambda_i(t), \\
\hat{\boldsymbol{b}}_{\mathrm{ML}}(t) &= \frac{1}{M}\sum_{i=1}^{M}X_i(t),
\end{aligned} \tag{7.9}$$

where M is the number of samples, $X(t)$ is the data matrix of size $M \times N_v$, and $U_q(t)$ and $L_q(t)$ are the normalized eigenvectors and eigenvalues of the data covariance matrix for the $1 \leq q \leq N_v$ largest eigenvalues. R is a rotation matrix that can be taken as $R = I$. The transformation to arbitrary $\boldsymbol{\mu}_h, \Sigma_h$ is:

$$\begin{aligned}
\boldsymbol{b}_{\mathrm{ML}}(t) &= \hat{\boldsymbol{b}}_{\mathrm{ML}}(t) - \hat{W}_{\mathrm{ML}}(t)\Sigma_h^{-1/2}(t)\boldsymbol{\mu}_h(t), \\
W_{\mathrm{ML}}(t) &= \hat{W}_{\mathrm{ML}}(t)\Sigma_h^{-1/2}(t).
\end{aligned} \tag{7.10}$$

with matrix square root as $(A^{1/2})^\intercal A^{1/2} = A$. For convenience, let $\hat{\boldsymbol{\theta}}(t) = \{\hat{\boldsymbol{b}}, \hat{W}, \sigma^2\}(t)$ denote the *standard parameters*.

7.2.3 Linking snapshots in time

Given a set of training data, the ML parameters $\boldsymbol{\theta}_{\mathrm{ML}}(t)$ can be obtained at each timepoint. To link snapshots in time, the form of the differential equations (7.5) must be chosen. The known CME physics is used to guide this choice by deriving an approximation $\boldsymbol{F}^{(\mathrm{approx.})}$ to the true time evolution $\boldsymbol{F}$ as follows.

At any point in time, the distribution defined by $\boldsymbol{\theta}(t)$ has observables $\boldsymbol{\phi}(t) = \{\boldsymbol{\mu}, C\}(t)$. For a single reaction like $A + B \to 2B$, these evolve as $d\boldsymbol{\phi}_{A+B\to 2B}/dt$ according to a hierarchy of moments like (7.2), derived from the CME. Under the Gaussian closure approximation (7.3), the equations for the moments are closed $d\boldsymbol{\phi}_{A+B\to 2B}/dt \sim d\boldsymbol{\phi}^{(\mathrm{closed})}_{A+B\to 2B}/dt$.

To convert back to the parameter frame, only some observables are tracked exactly. While arbitrary, the natural choice is $d\{\boldsymbol{\mu}_v, C_{vh}, \mathrm{Tr}(C_v), \boldsymbol{\mu}_h, \Sigma_h\}/dt$ which match the dimensions of $\boldsymbol{\theta}$. The equations corresponding to this conversion $d\boldsymbol{\phi}^{(\mathrm{closed})}_{A+B\to 2B}/dt \to d\boldsymbol{\theta}^{(\mathrm{closed})}_{A+B\to 2B}/dt$ are obtained by differentiating (7.7). The result $\boldsymbol{F}^{(\mathrm{approx.})}_{A+B\to 2B} \equiv d\boldsymbol{\theta}^{(\mathrm{closed})}_{A+B\to 2B}/dt$ is an approximation to the time evolution of $\boldsymbol{\theta}(t)$ under this reaction (Appendix E.2.3).

By considering a variety of reaction processes in this manner, a set of candidates was generated and used to parameterize the differential equations (7.5). It has been shown that the linearity of the CME in reactions extends to this form of the reduced model [16]. However, a linear model for (7.5) generalizes poorly when the data is not well-represented by a sparse set of available candidates [44].

Instead, let the right hand side of the differential equations (7.5) be given by a neural network with a special architecture shown in Figure 7.1. At each point in time with standard parameters $\hat{\boldsymbol{\theta}}(t)$, the inputs are the different reaction approximations. The outputs are the derivatives:

$$\frac{d\hat{\boldsymbol{\theta}}(t)}{dt} = \hat{\boldsymbol{F}}(\hat{\boldsymbol{\theta}}(t); \boldsymbol{u}), \tag{7.11}$$

where the parameters $\boldsymbol{u}$ are those of the neural network. The model is trained to optimize the L_2 loss:

$$S = \sum_{t=1}^{T} \left(\frac{d\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t)}{dt} - \hat{\boldsymbol{F}}(\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t); \boldsymbol{u}) \right)^2. \tag{7.12}$$

The training data is obtained from the ML parameters $\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t)$ for $t = 1, \ldots, T$ by first computing $d\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t)/dt$ using total variation regularization [104] to differentiate the noisy signals. After training, the integration of (7.11) is stable if the Jacobian of the candidates $\partial \hat{\boldsymbol{F}}^{(\mathrm{approx.})}_{A+B\to 2B}/\partial \hat{\boldsymbol{\theta}}$ is small. To reduce the Jacobian, the data matrices $X(t)$ are transformed using a standardizing transformation (Appendix E.2.1).

In principle, the standard parameters $\hat{\boldsymbol{\theta}}$ where $\boldsymbol{\mu}_h = \mathbf{0}, \Sigma_h = I$ can be used to calculate the reaction approximations. Instead, to improve generalization, the latent parameters

$\boldsymbol{\mu}_h, \Sigma_h$ are learned as a Fourier series. For a fixed set of L frequencies $\boldsymbol{f}$, let Σ_h be diagonal and let:

$$\begin{aligned} \mu_{h,i}(t) &= s(\boldsymbol{a}^{(\mu,i)}, \boldsymbol{b}^{(\mu,i)}), \\ \Sigma_{h,i,i}(t) &= 1 + \epsilon + s(\boldsymbol{a}^{(\Sigma,i,i)}, \boldsymbol{b}^{(\Sigma,i,i)}), \\ s(\boldsymbol{a}, \boldsymbol{b}) &= \frac{\sum_{l=1}^{L} \left(a_l \cos(f_l t) + b_l \sin(f_l t)\right)}{\max\left(\sum_{l=1}^{L} (|a_l| + |b_l|), 1\right) + \epsilon}, \end{aligned} \tag{7.13}$$

where ϵ is small and coefficients $\boldsymbol{a}, \boldsymbol{b}$ are learned. This lets $\boldsymbol{\mu}_h$ oscillate in $[\boldsymbol{-1}, \boldsymbol{1}]$ and Σ_h around the identity. Finally, since $\boldsymbol{\mu}_h, \Sigma_h$ are unknown from the data, the approximations $\boldsymbol{F}_{\text{reaction}}^{(\text{approx.})}$ are converted back to the standard space $\hat{\boldsymbol{F}}_{\text{reaction}}^{(\text{approx.})}$ using (7.10).

7.3 IP3 dependent calcium oscillations

The proposed physics-based ML method is demonstrated for calcium oscillations in non-excitable cells [105]. These occur due to calcium influx into the cytoplasm from stores in the endoplasmic reticulum (ER) through IP_3 receptors (IP_3Rs) in the membrane. A classic model by Ref. [102] uses ordinary differential equations and treats the channel at equilibrium, as shown in Figures 7.2.

A key result is a bifurcation diagram for calcium oscillations, shown in Figure 7.2(c). A Hopf bifurcation occurs at $[IP_3] \sim 0.4\mu$M beyond which oscillations arise. Beyond $[IP_3] \sim 0.6\mu$M, a stable elevated level of calcium is observed.

Figure 7.2(c) compares the bifurcation diagram with the range of oscillations observed in a stochastic version of the Ref. [102] model. The receptor channel states and transport through the channel are simulated using the Gillespie method, with identical parameters to those in [102]. For the stochastic model, two cytoplasm volumes are considered: 10^{-12} L and 10^{-14} L, and the number of IP_3R is varied. The range of oscillations show the

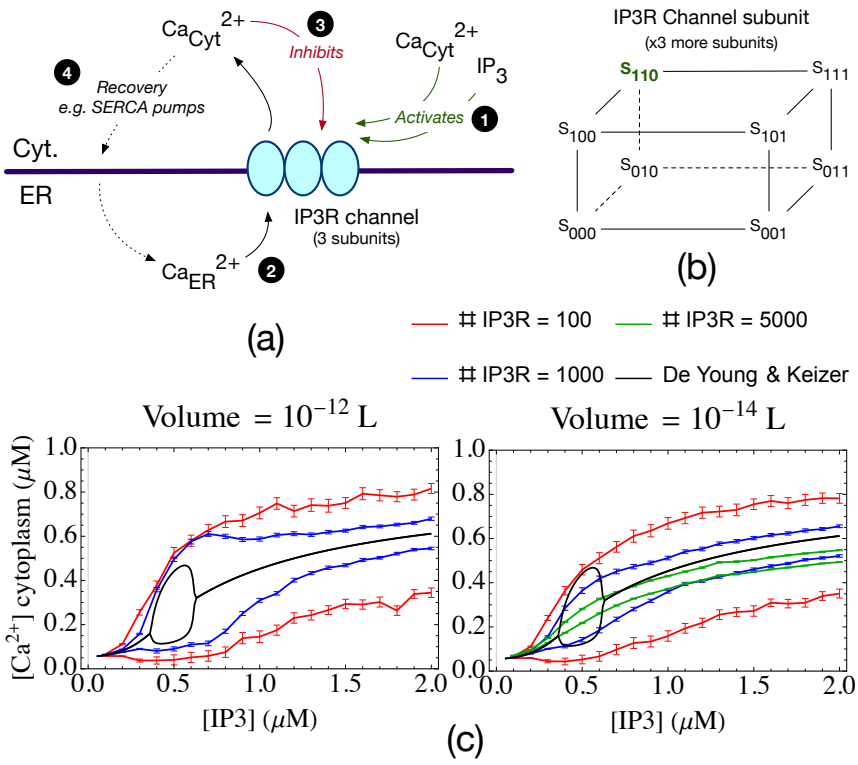

Figure 7.2: (a) Schematic of IP_3 dependent calcium oscillations. Clusters of IP_3Rs in the ER membrane are activated by cytosolic calcium and IP_3 (1), allowing calcium transport into the cytoplasm (2). Further binding inhibits the channel (3), and eventual recovery recycles the calcium store (4). (b) Channel states of one of three subunits (Appendix E.1.1). (c) Range of oscillations in the stochastic and deterministic [102] models for different volumes and numbers of IP_3Rs. Error bars indicate 95% confidence levels.

maximum/minimum over 40 s of the mean calcium concentration plus/minus a standard deviation. Spontaneous calcium spikes continue to arise in the stochastic model even at high IP_3 concentrations.

7.3.1 Learning calcium oscillations

The DBD architecture is applied to learn calcium oscillations over a subset of IP_3 concentrations. Figure 7.3 shows the range of oscillations learned for $V = 10^{-14}$L and 100 IP_3R receptors. The training data consists of simulations at IP_3 concentrations over $[0.4, 1]\mu$M in intervals of 0.1μM. Two subnet models are explored: a deep & wide subnet

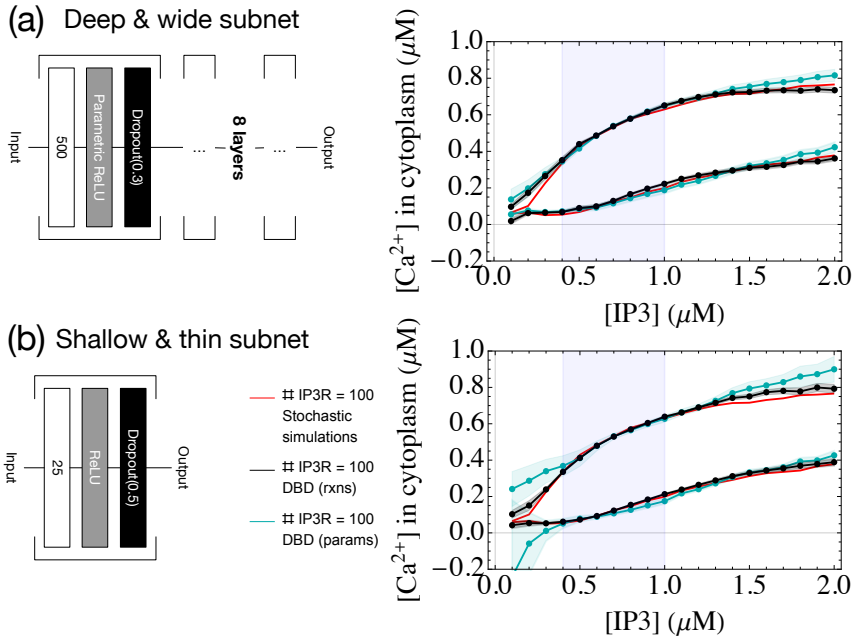

Figure 7.3: Incorporating reaction approximations improves generalization (black). Two subnets are compared: (a) a deep & wide subnet, and (b) a shallow & thin subnet. The IP_3 concentrations used as training data are highlighted (blue). A comparison architecture using the same subnet but missing reaction approximations is also shown (cyan). Shading shows 95% confidence intervals from 10 optimization trials.

consisting of 8 layers of width 500 units, and a shallow & thin subnet consisting of a single layer of 25 units, both using ReLU activation functions and dropout. Three species are used in the effective probability distribution (7.4): Ca^{2+},IP_3 and a latent species X. The reaction approximations used are those from enumerating the Lotka-Volterra system (Appendix E.3): $P \to 2P$, $H \to \varnothing$, and $H + P \to 2H$, allowing each combination of $\{H, P\}$ from $\{Ca^{2+}, IP_3, X\}$.

To demonstrate how domain-specific knowledge improves generalization, a comparison *parameter-only model* is shown, equivalent to Figure 7.1 but missing the reaction approximations (Appendix E.3.4). Both models are trained using the Adam optimizer [69] with batch size 64 and learning rate 10^{-3}. The deep subnet is trained for 25 rounds with weight clipping beyond a cutoff magnitude of 5; the shallow subnet for 200 rounds and weight cutoff 1. Between the parameter-only and the *reaction model*, the latter generalizes

better to IP_3 concentrations not observed during training. Further, the reaction model outperforms the comparison at keeping concentrations non-negative over the domain explored, although this is not explicitly enforced.

The generalization of the parameter-only model is better for the deep subnet than for the shallow subnet, partly because multiple layers of dropout improve generalization. However, the reaction model generalizes well even for the very low parameter shallow subnet. Figure 7.4(a) shows the integrated parameters at two slices of IP_3 using the Euler method. The reaction models learn the curves more exactly on both training and validation sets. This is quantified by a lower mean-squared error (MSE) shown in Figure 7.4(b).

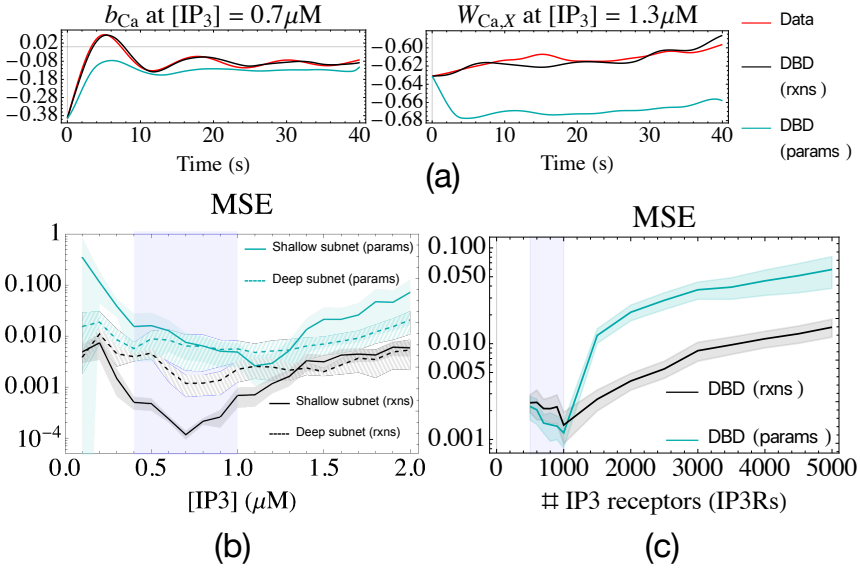

Figure 7.4: (a) Parameters from the shallow subnet model from slices of Figure 7.3 at $[IP_3] = 0.7\mu M$ and $[IP_3] = 1.3\mu M$. The reaction model learns a more detailed model. (b) MSE for the learned parameters $\hat{\boldsymbol{\theta}}$ in the models of Figure 7.3, with training data in blue. Models with reaction approximations decrease MSE by up to an order of magnitude. Shading shows 95% confidence intervals from 10 optimization trials. (c) MSE of a second model generalizing in IP_3 receptor number, with 95% confidence intervals from 40 trials, and training data in blue.

7.3.2 Encoding conservation

A second axis to generalize in is the number of IP_3Rs. The PCA model is now formulated for four species $\{Ca^{2+}, IP_3, IP_3R, X\}$ (the variance of IP_3R is set to a small constant 10^{-7} in the ML step). Since the receptor number is conserved in the simulations, the reaction approximations are extended with three of the form $A + IP_3R \to IP_3R$, where A is one of $\{Ca^{2+}, IP_3, X\}$. This explicitly conserves IP_3R in the input approximations.

Figure 7.4(c) shows the MSE over parameters for this model, trained on simulations at IP_3Rs over $[500, 1000]$ in intervals of 100. The reaction model outperforms the parameter-only model over the validation set covering 1000 to 5000 receptors. Training used the Adam optimizer for 25 rounds, with learning rate 10^{-3} and batch size 64. The subnet has 5 hidden layers of 150 units, ReLU activations, dropout rate 0.1, and weight cutoff 0.5.

7.4 Discussion

The power of DBDs is that knowledge of the domain can be explicitly built into the learning problem. This is possible due to the tight connection between reduced and fine scale models. Both are Markovian, depending only on the current point in parameter space. Moreover, because the reduced model is formulated by differential equations (7.5), reaction network physics could be built in through candidate functions derived from the master equation. Additionally, conservation laws for IP_3R were included in the network inputs. These connections to the underlying physics differentiate DBDs from how neural networks are commonly used for time series regression, and other methods including hidden Markov models (HMMs) and recurrent neural networks (RNNs). A further desired property is that the learned covariance matrix $C(t)$ is positive semidefinite at all times. This is the case for the PCA model (7.7) due to the transformation (7.10), but not for a generic Gaussian distribution. Additionally, the means $\boldsymbol{\mu}(t)$ should be non-negative to represent particle

counts. This is not enforced explicitly, but is observed for the reaction models in Figure 7.3.

Other methods have been proposed that learn a neural network representing a differential equation directly from parameters without explicitly incorporating domain-specific physics [99, 100, 101]. A related method that uses candidates is SINDy [44], but its differential equations are linear and struggle with model reduction, where candidates do not include the true dynamics. Further, its candidates are arbitrary polynomial forms, and not necessarily connected to underlying physics. For graphical models, graph-constrained correlation dynamics (GCCD) [10] has used polynomial and exponential candidates non-linearly with neural networks. Parameterizations using basis functions from finite elements [17, 18] have also been used. In these cases, for graphical models other than PCA (7.7), the ML parameters can be estimated by the Boltzmann machine learning algorithm [13] or by expectation maximization [26].

One avenue for improvement is to include approximations for small networks rather than just individual reactions. DBDs may also be extendable to delay differential equations to improve regression performance. Alternatively, this may be implemented using tailored input reaction motifs. Further closure approximations beyond the Gaussian closure can also be included as candidates.

While the models considered have no spatial dependence, the approach is equally valid for spatial systems [16, 17]. A spatial model of IP_3-dependent calcium oscillations may include plasma membrane pumps and feedback on IP_3 production. One application of DBDs is to synaptic neuroscience, where simulations of signaling pathways [3] could be used to build models that are computationally efficient and generalize well to new stimulation patterns. Beyond reaction-diffusion systems, applications to other domains such as neural populations [106] may be possible.

Chapter 8

Conclusion

This thesis has introduced a machine learning approach to model reduction for stochastic reaction networks. It is based on the Boltzmann machine learning algorithm [13] to simultaneously estimate a Boltzmann distribution with time dependent interactions, and a reduced differential equation model. The key strength of this DBD formalism is that prior knowledge can be incorporated into the machine learning framework. This is particularly promising for modeling applications in science, where a wealth of prior knowledge is typically available for the system under consideration. For example, for reaction-diffusion systems, the chemical master equation (CME) has been extensively studied, from perturbation theory to moment closure approximations. DBDs are one of a small but growing hybrid class of methods that incorporate both first principles physics and statistical inference methods.

The dynamic Boltzmann distribution is motivated by its connection to Markov random fields (MRFs). Prior knowledge can be encoded in the graphical model, such as the known structure of complexes like calcium/calmodulin-dependent protein kinase II (CaMKII) [10]. Another example is for lattice based systems, where the connectivity of the graph encodes the range of correlations that are relevant to the problem [17], with multiple

hidden layers capturing longer range spatial correlations as they appear in the differential equations for moments [18].

The differential equation models are motivated by considering analytic solutions to simple reaction motifs [16]. These may be exact solutions for simple systems, or approximate solutions, for example under a moment closure approximation. Linear combinations of these basis have a close connection to the true master equation physics - the coefficients are directly the reaction rates in the reduced model reaction network, as explored in Chapter 5. Non-linear combinations can also be learned, for example through neural networks in Chapter 7. Incorporating the candidate functions enables shallow networks with fewer parameters to be used, and the learned latent parameters are observed to converge more consistently, indicating that the network is more effective at discovering emergent order. Finally, effective partial differential equations may be learned on a given mesh directly from data as shown for lattice systems in Chapter 6.

The ability to introduce domain-specific knowledge sets DBDs apart from other methods for time series regression, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), autoregressive models and hidden Markov models (HMMs). Mapping to a differential equation model can likewise be useful for engineering applications, allowing constraints to be efficiently introduced into BM learning. A further advantage of this strategy is that it offers a natural description of systems where neither time nor space are discretized, i.e. the system is described by random variables representing space continuously and varying continuously in time. Spatially continuous descriptions are beneficial when confined geometries would introduce error into lattice-based methods, e.g. when modeling reaction-diffusion systems at synapses [3].

Chapter 7 in particular highlights an important point for scientific modeling. In the debate between whether models should be based on first principles physics or whether they should be data driven and discover emergent phenomena from examples, it is clearly

demonstrated that the best performance is achieved by hybrid models that can do both. Similar results have emerged in other areas of machine learning. Kernel methods such as support vector machines (SVMs) saw substantial performance boosts by encoding symmetries into kernels [91]. Image detection using convolutional neural networks (CNNs) has been improved by incorporating pose estimation, introducing $3D$ information about the image contents [92]. Automatic speech recognition models, too, have come full circle. The state of the art in this field used be systems which combined language models, pronunciation tables and acoustic models, rather than neural networks. The rise of end-to-end training methods, however, produced approaches such as deep learning of acoustic models in an end-to-end fashion [107] which allow the best of both worlds: encoding known structure and allowing blackbox inference.

Multiple avenues for improvement exist. For the learning problem of Chapter 5, parameterizations of the partial differential equations may be a promising way for modeling large simulations of synaptic biochemistry. Several candidates exist beyond those discussed. While interactions of arbitrary order can be included in the energy function, intuition suggests that models including only bias terms and pairwise interactions between two particles may be effective. Instead of considering arbitrary pairwise interactions, they may be parameterized, for example by the spherical harmonics such as those that arise in the wavefunctions in Hydrogen atoms. Alternatively, in density functional theory (DFT), the radially symmetry basis functions of Behler & Parrinello [108] have been highly successful.

For the physics-based machine learning method of Chapter 7, one avenue for improvement is to include approximations for small networks rather than just individual reactions. This may include tailored reaction motifs such as substrate-enzyme-product networks that encode longer time behavior. Further closure approximations beyond the Gaussian closure can also be included as candidates.

While the models considered in Chapter 7 have no spatial dependence, the approach

is equally valid for spatial systems. A spatial model of IP_3-dependent calcium oscillations may include plasma membrane pumps that can act as a clamp on the calcium concentration in the cytoplasm. Further, the feedback on IP_3 production modules calcium spiking over long timescales. Spatial models of the oscillations in MCell may be of broad interest to the elucidate how whole cell calcium spiking arises from smaller calcium puff events surrounding the receptors. The diffusion of calcium between channel clusters is speculated to synchronize these stochastic puffs, leading to more deterministic oscillations at the whole cell level [109].

More broadly, dynamic Boltzmann distributions are a promising avenue for models of synaptic biochemistry that bridge scales. Algebraic multigrid methods may have a key role in this, where for example W-cycle may iteratively switch between the fine scale stochastic simulation models and coarser scale dynamic Boltzmann distributions. Here, model reduction is done by solving the learning problem; prolongation is done by sampling the Boltzmann distribution.

A more direct method to link the reaction networks of ions with the assembly of larger complexes could also be possible. Consider for example synaptic morphodynamics, the changing size and shape of synapses that is thought to be the fundamental process of learning and memory in the brain. The synapse grows and shrinks on the timescale of minutes due to the reorganization of the actin cytoskeleton. These processes which include bundling of filamentous actin, branch fusion and nucleation are well described by a dynamical graph grammar [110]. The processes are regulated on a much shorter timescale over μs to s due to the influx of calcium into the post-synaptic spin head and its downstream signaling pathways including CaMKII-β. Constructing reduced models through dynamic Boltzmann distributions of each actin reorganization described by a dynamical graph grammar and calcium signaling described by a reaction network may offer a multiscale modeling approach to describe the morphological changes at the synapse that

result from different stimulation patterns.

The DBD approach developed in this thesis are part of this bigger goal: to develop multiscale modeling methods for synaptic neuroscience. Once established, these data-driven methods will be widely applicable in cellular neuroscience, and may aid in the development of pharmaceutical targets for learning deficits associated with aging and neurological disorders such as Alzheimers.

Appendix A

Derivation of Fick's second law from the master equation

Recall the generating function setting of Chapter 2.4.2. To ease notation, we consider only one species with diffusion constant D, and therefore drop any species labels. We derive the differential equations for the lowest order observables $\partial\langle n(y)\rangle/\partial t$, where $n(y)$ counts the number of particles at a point y in 3D space. In the generating function formalism:

$$\begin{aligned} g[z](t) &= \sum_{n=0}^{\infty} \int d\boldsymbol{x}\, p(n,\boldsymbol{x},t) \prod_{i=1}^{n} z(x_i), \\ n(y) &= z(y)\frac{\delta}{\delta z(y)}, \\ W_{\text{diff}} &= D\int dy\, z(y)\nabla_y^2 \frac{\delta}{\delta z(y)}. \end{aligned} \tag{A.1}$$

then for the diffusion operator:

$$\begin{aligned} W_{\text{diff}}\, g &= D\int dy\, z(y)\nabla_y^2 \sum_n \int d\boldsymbol{x}\, p(n,\boldsymbol{x}) \left(\sum_{i=1}^{n} \delta(x_i - y) \prod_{j\neq i} z(x_j) \right) \\ &= D\sum_n \int d\boldsymbol{x}\, \left(\sum_{i=1}^{n} \nabla_{x_i}^2 p(n,\boldsymbol{x}) \right) \left(\prod_{i=1}^{n} z(x_i) \right), \end{aligned} \tag{A.2}$$

where we have used the property of the variational derivative:

$$\int dx\, f(x)\delta(x-y) = f(y). \tag{A.3}$$

Applying the variational derivative gives:

$$\begin{aligned}
&z(y)\frac{\delta}{\delta z(y)}(W_{\text{diff}}\, g)\Big|_{z=1} \\
=&z(y)D\sum_n \int d\boldsymbol{x}\, \left(\sum_{i=1}^{n}\nabla_{x_i}^2 p(n,\boldsymbol{x})\right)\left(\sum_{i_1=1}^{n}\delta(x_{i_1}-y)\prod_{i_2\neq i_1} z(x_{i_2})\right)\Bigg|_{z=1} \\
=&D\sum_n \int d\boldsymbol{x}\, \left(\sum_{i=1}^{n}\nabla_{x_i}^2 p(n,\boldsymbol{x})\right)\left(\sum_{i_1=1}^{n}\delta(x_{i_1}-y)\right) \\
=&D\nabla_y^2\sum_n \int d\boldsymbol{x}\, p(n,\boldsymbol{x})\left(\sum_{i_1=1}^{n}\delta(x_{i_1}-y)\right) \\
=&D\nabla_y^2\langle n(y)\rangle\,,
\end{aligned} \tag{A.4}$$

gives the desired result:

$$\frac{\partial\langle n(y)\rangle}{\partial t} = D\nabla_y^2\langle n(y)\rangle. \tag{A.5}$$

Appendix B

Dynamic Boltzmann distributions

B.1 Derivation of Differential Equation System for Variational Term

B.1.1 Well-Mixed Case

Consider the differential equation system (4.13). Represent the solution as a functional of the basis functions F using the notation

$$\nu_{k'}(t') = J_{k'}[\{F\}], \tag{B.1}$$

where $\{F\} = \{F_l \mid l = 1, \ldots, K\}$, and J results from solving (4.13). Further, let $\{J[\{F\}]\} = \{J_l[\{F\}] \mid l = 1, \ldots, K\}$, then (4.13) is:

$$\frac{d}{dt'} J_{k'}[\{F\}] = F_{k'}(\{J[\{F\}]\}). \tag{B.2}$$

To find the variational term $\delta\nu_{k'}(t')/\delta F_k(\{\nu\})$, let $F_k \to F_k + \epsilon\eta$ using the notation

$$\{F'\} = \{F_l + \delta_{l,k}\epsilon\eta | l = 1, \ldots, K\}, \tag{B.3}$$

then:

$$\frac{d}{dt'}J_{k'}[\{F'\}] = F_{k'}(\{J[\{F'\}]\}) + \delta_{k',k}\epsilon\eta(\{J[\{F'\}]\}). \tag{B.4}$$

Differentiating with respect to ϵ at $\epsilon = 0$ gives:

$$\frac{d}{dt'}\left(\left.\frac{dJ_{k'}[\{F'\}]}{d\epsilon}\right|_{\epsilon=0}\right) = \sum_{l=1}^{K}\frac{\partial F_{k'}(\{\nu(t')\})}{\partial\nu_l(t')}\left(\left.\frac{dJ_l[\{F'\}]}{d\epsilon}\right|_{\epsilon=0}\right) + \delta_{k',k}\eta(\{\nu(t')\}). \tag{B.5}$$

Substitute the definition of the functional derivative

$$\left.\frac{dJ_{k'}[\{F'\}]}{d\epsilon}\right|_{\epsilon=0} = \int d\nu_1 \cdots \int d\nu_K \, \frac{\delta\nu_{k'}(t')}{\delta F_k(\{\nu\})}\eta(\{\nu\}) \tag{B.6}$$

to obtain (4.15):

$$\frac{d}{dt'}\left(\frac{\delta\nu_{k'}(t')}{\delta F_k(\{\nu\})}\right) = \sum_{l=1}^{K}\frac{\partial F_{k'}(\{\nu(t')\})}{\partial\nu_l(t')}\frac{\delta\nu_l(t')}{\delta F_k(\{\nu\})} + \delta_{k',k}\delta(\{\nu\} - \{\nu(t')\}). \tag{B.7}$$

B.1.2 Spatially Heterogeneous Example: Diffusion in 1D

Consider a diffusion process in 1D, described by single basis functional parameterized according to:

$$\begin{aligned}\frac{d}{dt'}\nu(y',t') =& F[\nu(y',t')] \\ =& F^{(1)}(\nu(y',t'))\left(\partial_{y'}\nu(y',t')\right)^2 + F^{(2)}(\nu(y',t'))\left(\partial_{y'}^2\nu(y',t')\right).\end{aligned} \tag{B.8}$$

Use the functional notation:

$$\nu(y',t') = J[\{F\}], \tag{B.9}$$

where $\{F\} = \{F^{(1)}, F^{(2)}\}$ and J results from solving (4.33), then:

$$\frac{d}{dt}J[\{F\}] = F^{(1)}(J[\{F\}])\left(\partial_{y'}J[\{F\}]\right)^2 + F^{(2)}(J[\{F\}])\partial_{y'}^2 J[\{F\}]. \tag{B.10}$$

To find the variational term $\delta\nu(y',t')/\delta F^{(\gamma)}(\omega)$ for $\gamma = 1,2$, let $F^{(\gamma)} \to F^{(\gamma)} + \epsilon\eta$. Use the notation:

$$\{F'\} = \{F^{(1)} + \delta_{\gamma,1}\epsilon\eta, F^{(2)} + \delta_{\gamma,2}\epsilon\eta\}, \tag{B.11}$$

then

$$\begin{aligned}\frac{d}{dt}J[\{F'\}] =& F^{(1)}(J[\{F'\}])\left(\partial_{y'}J[\{F'\}]\right)^2 + F^{(2)}(J[\{F'\}])\partial_{y'}^2 J[\{F'\}] \\ &+ \delta_{\gamma,1}\epsilon\eta(J[\{F'\}])\left(\partial_{y'}J[\{F'\}]\right)^2 + \delta_{\gamma,2}\epsilon\eta(J[\{F'\}])\partial_{y'}^2 J[\{F'\}].\end{aligned} \tag{B.12}$$

Take the derivative with respect to ϵ at $\epsilon = 0$:

$$\begin{aligned}\frac{d}{dt}\left(\left.\frac{dJ[\{F'\}]}{d\epsilon}\right|_{\epsilon=0}\right) =& \left(\frac{\partial F^{(1)}(\nu)}{\partial\nu}\left(\partial_{y'}\nu\right)^2 + \frac{\partial F^{(2)}(\nu)}{\partial\nu}\partial_{y'}^2\nu\right)\left(\left.\frac{dJ[\{F'\}]}{d\epsilon}\right|_{\epsilon=0}\right) \\ &+ \left(\delta_{\gamma,1}\left(\partial_{y'}\nu\right)^2 + \delta_{\gamma,2}\partial_{y'}^2\nu\right)\eta(\nu) \\ &+ 2F^{(1)}(\nu)\partial_{y'}\nu\frac{\partial}{\partial y'}\left(\left.\frac{dJ[\{F'\}]}{d\epsilon}\right|_{\epsilon=0}\right) \\ &+ F^{(2)}(\nu)\frac{\partial^2}{\partial y'^2}\left(\left.\frac{dJ[\{F'\}]}{d\epsilon}\right|_{\epsilon=0}\right),\end{aligned} \tag{B.13}$$

where $\nu = \nu(y',t')$ everywhere. Substituting the definition of the functional derivative

$$\left.\frac{dJ[\{F'\}]}{d\epsilon}\right|_{\epsilon=0} = \int d\omega\, \frac{\delta\nu(y',t')}{\delta F^{(\gamma)}(\omega)}\eta(\omega) \tag{B.14}$$

gives

$$\begin{aligned}\frac{d}{dt}\left(\frac{\delta\nu}{\delta F^{(\gamma)}(\omega)}\right) = &\left(\frac{\partial F^{(1)}(\nu)}{\partial\nu}\left(\partial_{y'}\nu\right)^2 + \frac{\partial F^{(2)}(\nu)}{\partial\nu}\partial_{y'}^2\nu\right)\left(\frac{\delta\nu}{\delta F^{(\gamma)}(\omega)}\right)\\ &+\left(\delta_{\gamma,1}\left(\partial_{y'}\nu\right)^2 + \delta_{\gamma,2}\partial_{y'}^2\nu\right)\delta(\nu-\omega)\\ &+2F^{(1)}(\nu)\partial_{y'}\nu\frac{\partial}{\partial y'}\left(\frac{\delta\nu}{\delta F^{(\gamma)}(\omega)}\right) + F^{(2)}(\nu)\frac{\partial^2}{\partial y'^2}\left(\frac{\delta\nu}{\delta F^{(\gamma)}(\omega)}\right).\end{aligned} \tag{B.15}$$

B.2 Evaluating Basis Functions Numerically

To compute the basis functions numerically using (4.41), an efficient method is possible if the eigenvalues of the transfer matrix M are singular. Let the eigenvalues be λ_i with corresponding eigenvectors $\boldsymbol{u}_i$. Define:

$$p_{ij}(\alpha) = \boldsymbol{u}_i^\intercal(\partial_\alpha M)\boldsymbol{u}_j^\intercal \tag{B.16}$$

for $\alpha = h, J$, where $\partial_\alpha M$ denotes component-wise differentiation of M. Also note that $p_{ij}(\alpha) = p_{ji}(\alpha)$ is symmetric. Then the derivatives of the eigenvalues are given by: [111]

$$\begin{aligned}\partial_\alpha\lambda_i &= p_{ii}(\alpha),\\ \partial_\alpha\partial_\beta\lambda_i &= \boldsymbol{u}_i^\intercal(\partial_\alpha\partial_\beta M)\boldsymbol{u}_i + 2\sum_{j\neq i}\frac{p_{ij}(\alpha)p_{ij}(\beta)}{\lambda_i-\lambda_j},\end{aligned} \tag{B.17}$$

for $\beta = h, J$. The principle advantage of this approach lies in the fact that the analytic expressions for $\partial_\alpha M$ and $\partial_\alpha\partial_\beta M$ are simpler to derive than differentiating the analytic expressions for the eigenvalues λ.

It is now straightforward to numerically evaluate the components $\partial_\alpha\partial_\beta \ln\mathcal{Z}$ of (4.41) in the thermodynamic limit $\ln\mathcal{Z} \approx N\ln\lambda_+$, where λ_+ is the largest eigenvalue of the transfer matrix and N the length of the chain.

B.3 Alternative Solutions for the Variational Terms

B.3.1 Well-Mixed Case: Alternate PDE Solution

Consider the differential equation system

$$\begin{aligned} \frac{d}{dt}\nu_k(t) &= F_k\left(\{\nu\}\right), \\ \text{with I.C.:} \quad \nu_k(t=0) &= \eta_k. \end{aligned} \tag{B.18}$$

Use the chain rule:

$$\frac{d}{dt}\nu_{k'}(t') = \sum_{l=1}^{K} \frac{\partial \nu_{k'}(t')}{\partial \eta_l}\frac{d\eta_l}{dt'} = \sum_{l=1}^{K} \frac{\partial \nu_{k'}(t')}{\partial \eta_l} F_l(\{\eta\}). \tag{B.19}$$

Now take the variational derivative $\delta/\delta F_k(\{\nu\})$ and use the product rule:

$$\begin{aligned} \frac{d}{dt}\left(\frac{\delta\nu_{k'}(t')}{\delta F_k(\{\nu\})}\right) =& \sum_{l=1}^{K}\left(\frac{\delta}{\delta F_k(\{\nu\})}\frac{\partial \nu_{k'}(t')}{\partial \eta_l}\right) F_l(\{\eta\}) + \sum_{l=1}^{K}\frac{\partial \nu_{k'}(t')}{\partial \eta_l}\frac{\delta F_l(\{\eta\})}{\delta F_k(\{\nu\})} \\ =& \sum_{l=1}^{K} F_l(\{\eta\})\frac{\partial}{\partial \eta_l}\left(\frac{\delta\nu_{k'}(t')}{\delta F_k(\{\nu\})}\right) + \left(\prod_{l=1}^{K}\delta(\nu_l-\eta_l)\right)\frac{\partial \nu_{k'}(t')}{\partial \eta_k}. \end{aligned}$$

This may be evaluated in practice from the initial condition $\delta\nu_{k'}(0)/\delta F_k(\{\nu\}) = 0$, where $\partial\nu_{k'}(t')/\partial\eta_k$ is evaluated numerically by perturbing the initial conditions and integrating (B.18).

B.3.2 Well-Mixed Case: Lie Series Solution

A further alternative solution is provided by the use of Lie series [25]. The autonomous differential equation system (B.18) has formal solution:

$$\nu_{k'}(t') = \exp\Big(t' D\Big) \circ \eta_{k'} \tag{B.20}$$

for the operator

$$D \equiv \sum_{l=1}^{K} F_l(\{\nu\}) \frac{\partial}{\partial \nu_l}, \tag{B.21}$$

Using the notation

$$M_{k'}^{(n)}(\{\eta\}) \equiv (D^n \nu_{k'})\Big|_{\{\nu\}=\{\eta\}}. \tag{B.22}$$

where $\{\eta\} = \{\eta_{k'}\}_{k'=1}^{K}$, this is

$$\nu_{k'}(t') = \sum_{n=0}^{\infty} \frac{(t')^n}{n!} M_{k'}^{(n)}(\{\eta\}). \tag{B.23}$$

The terms $M_{k'}^{(n)}(\{\eta\})$ obey the recursion relationships

$$\begin{aligned} M_{k'}^{(0)}(\{\eta\}) &= \eta_{k'}, \\ M_{k'}^{(1)}(\{\eta\}) &= F_{k'}(\{\eta\}), \\ M_{k'}^{(n)}(\{\eta\}) &= \sum_{l=1}^{K} F_l(\{\eta\}) \frac{\partial M_{k'}^{(n-1)}(\{\eta\})}{\partial \eta_l} \quad \text{for} \quad n \geq 2. \end{aligned} \tag{B.24}$$

From (B.23), we have

$$\frac{\delta \nu_{k'}(t')}{\delta F_k(\{\nu\})} = \sum_{n=0}^{\infty} \frac{(t')^n}{n!} \frac{\delta M_{k'}^{(n)}(\{\eta\})}{\delta F_k(\{\nu\})}, \tag{B.25}$$

where

$$\begin{aligned} \frac{\delta M_{k'}^{(0)}(\{\eta\})}{\delta F_k(\{\nu\})} &= 0, \\ \frac{\delta M_{k'}^{(1)}(\{\eta\})}{\delta F_k(\{\nu\})} &= \delta_{k,k'} \prod_{l=1}^{K} \delta(\nu_l - \eta_l), \end{aligned} \tag{B.26}$$

and for $n \geq 2$:

$$\frac{\delta M_{k'}^{(n)}(\{\eta\})}{\delta F_k(\{\nu\})} = \left(\prod_{l=1}^{K} \delta(\nu_l - \eta_l)\right) \frac{\partial M_{k'}^{(n-1)}(\{\eta\})}{\partial \eta_k} + \sum_{l=1}^{K} F_l(\{\eta\}) \frac{\partial}{\partial \eta_l} \left(\frac{\delta M_{k'}^{(n-1)}(\{\eta\})}{\delta F_k(\{\nu\})} \right). \quad \text{(B.27)}$$

B.3.3 Spatially Heterogeneous Case: Alternate PDE Solution

Consider the spatially local model

$$\begin{aligned} \frac{d}{dt}\nu_k(\boldsymbol{\beta}, \boldsymbol{y}, t) =& F_k^{(0)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\}) \\ &+ \sum_{\lambda=1}^{k} \left(F_k^{(1,\lambda)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\}) \sum_{\langle i \rangle_\lambda^k} \sum_{m=1}^{\lambda} \left(\partial_m \nu_\lambda(\boldsymbol{\beta}_{\langle i \rangle_\lambda^k}, \boldsymbol{y}_{\langle i \rangle_\lambda^k}, t) \right)^2 \right. \\ &\left. + F_k^{(2,\lambda)}(\{\nu(\boldsymbol{\beta}, \boldsymbol{y}, t)\}) \sum_{\langle i \rangle_\lambda^k} \sum_{m=1}^{\lambda} \partial_m^2 \nu_\lambda(\boldsymbol{\beta}_{\langle i \rangle_\lambda^k}, \boldsymbol{y}_{\langle i \rangle_\lambda^k}, t) \right), \end{aligned}$$

with I.C.: $\nu_k(\boldsymbol{\beta}, \boldsymbol{y}, t = 0) = \eta_k(\boldsymbol{\beta}, \boldsymbol{y})$.

(B.28)

For brevity, we neglect the species label, which may be inserted trivially at the end. Use the chain rule:

$$\frac{d}{dt}\nu_{k'}(\boldsymbol{y}', t') = \sum_{l=1}^{K} \int d\boldsymbol{z} \frac{\partial \nu_{k'}(\boldsymbol{y}', t')}{\partial \eta_l(\boldsymbol{z})} \frac{d\eta_l(\boldsymbol{z})}{dt'}. \quad \text{(B.29)}$$

Now take the variational derivative $\delta / \delta F_k^{(\gamma)}(\{\nu(\boldsymbol{y})\})$ and use the product rule:

$$\begin{aligned} \frac{d}{dt}\left(\frac{\delta \nu_{k'}(\boldsymbol{y}', t')}{\delta F_k^{(\gamma)}(\{\nu(\boldsymbol{y})\})} \right) =& \sum_{l=1}^{K} \int d\boldsymbol{z} \frac{d\eta_l(\boldsymbol{z})}{dt'} \frac{\partial}{\partial \eta_l(\boldsymbol{z})} \left(\frac{\delta \nu_{k'}(\boldsymbol{y}', t')}{\delta F_k^{(\gamma)}(\{\nu(\boldsymbol{y})\})} \right) \\ &+ \int d\boldsymbol{z} \frac{\partial \nu_{k'}(\boldsymbol{y}', t')}{\partial \eta_k(\boldsymbol{z})} \frac{\delta}{\delta F_k^{(\gamma)}(\{\nu(\boldsymbol{y})\})} \frac{d\eta_k(\boldsymbol{z})}{dt'}. \end{aligned} \quad \text{(B.30)}$$

We can use (B.28) at $t = 0$ to evaluate $d\eta_l(\boldsymbol{z})/dt'$ in the first term. Further, taking the variational derivative to evaluate the second term:

$$\begin{aligned}\frac{\delta}{\delta F_k^{(\gamma)}(\{\nu(\boldsymbol{y})\})}\frac{d\eta_k(\boldsymbol{z})}{dt'} =&\delta(\{\nu(\boldsymbol{y})\}-\{\eta(\boldsymbol{z})\})\\ &\times\left(\delta_{(\gamma),(0)}+\sum_{\lambda=1}^{k}\left(\delta_{(\gamma),(1,\lambda)}\sum_{\langle i\rangle_\lambda^k}\sum_{m=1}^{\lambda}\left(\partial_m\eta_\lambda(\boldsymbol{z}_{\langle i\rangle_\lambda^k})\right)^2\right.\right.\\ &\left.\left.+\delta_{(\gamma),(2,\lambda)}\sum_{\langle i\rangle_\lambda^k}\sum_{m=1}^{\lambda}\left(\partial_m^2\eta_\lambda(\boldsymbol{z}_{\langle i\rangle_\lambda^k})\right)\right)\right),\end{aligned} \tag{B.31}$$

where as before ∂_m denotes the derivative with respect to the m-th component of $\boldsymbol{z}_{\langle i\rangle_\lambda^k}$, and we use the following notation for the multi-dimensional Dirac delta function

$$\delta(\{\nu(\boldsymbol{y})\}-\{\eta(\boldsymbol{z})\}) = \prod_{k'=1}^{k}\prod_{\langle i\rangle_{k'}^k}\delta\left(\nu_{k'}(\boldsymbol{y}_{\langle i\rangle_{k'}^k})-\eta_{k'}(\boldsymbol{z}_{\langle i\rangle_{k'}^k})\right). \tag{B.32}$$

It remains to numerically solve the PDE (B.30) using the initial condition:

$$\frac{\delta\nu_{k'}(\boldsymbol{y}', t'=0)}{\delta F_k^{(\gamma)}(\{\nu(\boldsymbol{y})\})} = 0 \tag{B.33}$$

everywhere for all $\boldsymbol{y}$.

Appendix C

Learning problem for spatial dynamic Boltzmann distributions

C.1 Formal solution for the adjoint system

The connection between (4.11) and (5.19) can be made more explicitly. A differential equation system for the perturbations $\delta\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t)$ in (4.11) can be derived by linearizing the differential equation around a particular solution [16, 56]. For the autonomous system (5.18), this leads to the linear ODE system:

$$\frac{d}{dt}\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) = \delta\boldsymbol{F}(\boldsymbol{\alpha}, \boldsymbol{x}, t) + G(\boldsymbol{\alpha}, \boldsymbol{x}, t)\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) \tag{C.1}$$

with some given initial condition $\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t_0) = \delta\boldsymbol{\eta}(\boldsymbol{\alpha}, \boldsymbol{x})$. Here we have used the vector notation introduced in Section 5.1.2.

Let the homogenous part of this system

$$\frac{d}{dt}\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) = G(\boldsymbol{\alpha}, \boldsymbol{x}, t)\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) \tag{C.2}$$

have solution given by the non-singular fundamental matrix $A(\boldsymbol{\alpha}, \boldsymbol{x}, t)$. Then (C.1) has as formal solution

$$\delta\boldsymbol{\nu}(\boldsymbol{\alpha}, \boldsymbol{x}, t) = A(\boldsymbol{\alpha}, \boldsymbol{x}, t)\left(\delta\boldsymbol{\eta}(\boldsymbol{\alpha}, \boldsymbol{x}) + \int_{t_0}^{t} dt'\, A^{-1}(\boldsymbol{\alpha}, \boldsymbol{x}, t')\delta\boldsymbol{F}(\boldsymbol{\alpha}, \boldsymbol{x}, t')\right), \tag{C.3}$$

which substituted into (4.11) gives:

$$\begin{aligned} 0 = \delta S = \int_{t_0}^{t_f} dt \sum_{n=0}^{\infty} \sum_{\boldsymbol{\alpha}} \int d\boldsymbol{x}\, \Delta\boldsymbol{\mu}^{\intercal}(\boldsymbol{\alpha}, \boldsymbol{x}, t) A(\boldsymbol{\alpha}, \boldsymbol{x}, t)\Bigg(\delta\boldsymbol{\eta}(\boldsymbol{\alpha}, \boldsymbol{x}) \\ + \int_{t_0}^{t} dt'\, A^{-1}(\boldsymbol{\alpha}, \boldsymbol{x}, t')\delta\boldsymbol{F}(\boldsymbol{\alpha}, \boldsymbol{x}, t')\Bigg), \end{aligned} \tag{C.4}$$

where $\Delta\boldsymbol{\mu}^{\intercal}(t)$ is the vector with components (5.8). Applying integration by parts on the term in parentheses to move the integral over time gives

$$\begin{aligned} &\left(\delta\boldsymbol{\eta}(\boldsymbol{\alpha}, \boldsymbol{x}) + \int_{t_0}^{t_f} dt'\, A^{-1}(\boldsymbol{\alpha}, \boldsymbol{x}, t')\delta\boldsymbol{F}(\boldsymbol{\alpha}, \boldsymbol{x}, t')\right)\left(\int_{t_0}^{t} dt'\, \Delta\boldsymbol{\mu}^{\intercal}(\boldsymbol{\alpha}, \boldsymbol{x}, t') A(\boldsymbol{\alpha}, \boldsymbol{x}, t')\right)\Bigg|_{t=t_0}^{t_f} \\ &- \int_{t_0}^{t_f} dt \int_{t_0}^{t} dt'\, \Delta\boldsymbol{\mu}^{\intercal}(\boldsymbol{\alpha}, \boldsymbol{x}, t') A(\boldsymbol{\alpha}, \boldsymbol{x}, t') A^{-1}(\boldsymbol{\alpha}, \boldsymbol{x}, t)\delta\boldsymbol{F}(\boldsymbol{\alpha}, \boldsymbol{x}, t), \end{aligned} \tag{C.5}$$

where the adjoint functions $\boldsymbol{\zeta}(t)$ can be identified as:

$$\boldsymbol{\zeta}^{\intercal}(\boldsymbol{\alpha}, \boldsymbol{x}, t) = \int_{t_0}^{t} dt'\, \Delta\boldsymbol{\mu}^{\intercal}(\boldsymbol{\alpha}, \boldsymbol{x}, t') A(\boldsymbol{\alpha}, \boldsymbol{x}, t') A^{-1}(\boldsymbol{\alpha}, \boldsymbol{x}, t). \tag{C.6}$$

By choosing the adjoint functions to satisfy the boundary condition $\boldsymbol{\zeta}(\boldsymbol{\alpha}, \boldsymbol{x}, t_f) = 0$, the boundary term in (C.5) vanishes and we obtain the previous result (4.14).

C.2 Derivation of moment equations from the chemical master equation

The moment equations (6.17,6.24) can be derived from the chemical master equation using the Doi-Peliti [20] formalism and its equivalent generating function representation. We demonstrate this for the Rössler system (6.24).

For notational convenience, we do not consider the single occupancy limit here. The state of the system is described by the $N \times M$ matrix $\boldsymbol{V'}$ with entries $v_{i,\alpha} \in \{0, 1, 2, \dots\}$, where $N = 10 \times 10 \times 10$ rows denote lattice sites, and $M = 3$ columns denote occupancies of species $\{A, B, C\}$.

Define the $N \times M$ single-entry *matrix* $\boldsymbol{e}_{ij}$ with entries zero everywhere except at index (i, j) where it is one. The creation and annihilation operators $\hat{a}_{i,\alpha}$ and $a_{i,\alpha}$ create and destroy particles of species α at unit i:

$$\begin{aligned} \hat{a}_{i,\alpha} \left| \boldsymbol{V'} \right\rangle &= \left| \boldsymbol{V'} + \boldsymbol{e}_{i,\alpha} \right\rangle, \\ a_{i,\alpha} \left| \boldsymbol{V'} \right\rangle &= v_{i,\alpha} \left| \boldsymbol{V'} - \boldsymbol{e}_{i,\alpha} \right\rangle. \end{aligned} \tag{C.7}$$

The operators corresponding to reactions in the Rössler system (excluding diffusion) are

then:

$$
\begin{aligned}
A \to 2A:\ & k_1 \sum_{i=1}^{N} \left(\hat{a}_{i,A} - 1\right) \hat{a}_{i,A} a_{i,A}, \\
2A \to A:\ & \kappa_1 \sum_{\langle ij \rangle} \left(1 - \hat{a}_{j,A}\right) \hat{a}_{i,A} a_{i,A} a_{j,A}, \\
A + B \to 2B:\ & \kappa_2 \sum_{\langle\langle ij \rangle\rangle} \left(\hat{a}_{i,B} - \hat{a}_{i,A}\right) \hat{a}_{j,B} a_{i,A} a_{j,B}, \\
A + C \to \varnothing:\ & \kappa_3 \sum_{\langle\langle ij \rangle\rangle} \left(1 - \hat{a}_{i,A} \hat{a}_{j,C}\right) a_{i,A} a_{j,C}, \\
B \to \varnothing:\ & k_2 \sum_{i=1}^{N} \left(1 - \hat{a}_{i,B}\right) a_{i,B}, \\
C \to 2C:\ & k_3 \sum_{i=1}^{N} \left(\hat{a}_{i,C} - 1\right) \hat{a}_{i,C} a_{i,C}, \\
2C \to C:\ & \kappa_4 \sum_{\langle ij \rangle} \left(1 - \hat{a}_{j,C}\right) \hat{a}_{i,C} a_{i,C} a_{j,C},
\end{aligned}
\tag{C.8}
$$

where $\sum_{\langle ij \rangle}$ sums over all neighboring sites without double counting, $\sum_{\langle\langle ij \rangle\rangle}$ sums over all neighboring sites with double counting, and we specify the species $\{A, B, C\}$ instead of an index $\alpha = 1, \ldots, M$ for clarity in the subscripts. Here we place new particles resulting from fission reactions with rates k_1 and k_3 at the same site - in the single occupancy limit, they must be placed at a neighboring site. For bimolecular reactions with rates κ_1 and κ_4, we make the in this case ambiguous choice to place new species at site i versus j. The time evolution operator W for the Rössler system is the sum of all terms in (C.8).

The system state and the ladder operators admit an equivalent generating function

representation:

$$\begin{aligned} \left|\boldsymbol{V}'\right\rangle &\to \prod_{i=1}^{N}\prod_{\alpha=1}^{M} z_{i,\alpha}^{v_{i,\alpha}}, \\ \hat{a}_{i,\alpha} &\to z_{i,\alpha}, \\ a_{i,\alpha} &\to \frac{\partial}{\partial z_{i,\alpha}}. \end{aligned} \tag{C.9}$$

An observable $\langle X\rangle$ with generating function representation X_z according to (C.9) evolves as:

$$\frac{d\langle X\rangle}{dt} = \left(X_z W \prod_{i=1}^{N}\prod_{\alpha=1}^{M} z_{i,\alpha}^{v_{i,\alpha}} \right)\Bigg|_{z=1}, \tag{C.10}$$

where W is now the sum of terms (C.8) in the generating function representation (C.9). From the number operator $\hat{a}_{k,\beta}a_{k,\beta}$ which counts the number of particles of species β at position k, the time evolution of the mean number of particles of species β is then

$$\frac{d\mu_\beta}{dt} = \left(\sum_{k=1}^{N} z_{k,\beta}\frac{\partial}{\partial z_{k,\beta}} W \prod_{i=1}^{N}\prod_{\alpha=1}^{M} z_{i,\alpha}^{v_{i,\alpha}} \right)\Bigg|_{z=1}, \tag{C.11}$$

which can be directly evaluated to give the moment equations (6.24). For a review on field theoretic methods for reaction-diffusion systems, we refer to Ref. [103]. The formalism can also describe systems in continuous space [20] where it has a similar generation function representation [16].

C.3 Learning diffusion constant derivations

C.3.1 Variable to fixed number of particles

A simple Gaussian distribution for n particles in 3D is

$$q(\boldsymbol{x},t) = (4\pi Dt)^{-3n/2} \exp\left[-\sum_{i=1}^{n} \frac{(x_i-\mu)^2}{4Dt}\right]. \tag{C.12}$$

To represent this as a spatial dynamic Boltzmann distribution, which allows for a *variable* number of particles m, we can consider a very tight Gaussian on the fixed n:

$$\begin{aligned} \tilde{p}(m,\boldsymbol{x},t) &= \frac{1}{Z(t)} \exp\left[-\sum_{i=1}^{m} \hat{\nu}_1(x_i,t) - m\overline{\nu}_1 - \binom{m}{2}\overline{\nu}_2\right], \\ Z(t) &= \sum_{m=0}^{\infty} \int d\boldsymbol{x}\ \exp\left[-\sum_{i=1}^{m} \hat{\nu}_1(x_i,t) - m\overline{\nu}_1 - \binom{m}{2}\overline{\nu}_2\right], \end{aligned} \tag{C.13}$$

where we make the following choices:

$$\begin{aligned} \overline{\nu}_1 &= \frac{1/2-n}{\sigma^2}, \\ \overline{\nu}_2 &= \frac{1}{\sigma^2}, \\ \hat{\nu}_1(x_i,t) &= \frac{(x_i-\mu)^2}{4Dt}. \end{aligned} \tag{C.14}$$

Note that with these choices:

$$\exp\left[-m\overline{\nu}_1 - \binom{m}{2}\overline{\nu}_2\right] \propto \exp\left[-\frac{(m-n)^2}{2\sigma^2}\right] \tag{C.15}$$

is a Gaussian about n. By choosing a very small σ^2, we have a distribution which essentially fixes the number of particles as constant n. In this case:

$$Z(t) \approx \int d\boldsymbol{x} \exp\left[-\sum_{i=1}^{n} \hat{\nu}_1(x_i,t)\right] = \left(\int dy\ \exp[-\hat{\nu}_1(y,t)]\right)^n, \tag{C.16}$$

therefore:

$$\tilde{p}(m,\boldsymbol{x},t) \approx \delta_{m,n} \frac{\exp[-\sum_{i=1}^{n} \hat{\nu}_1(x_i,t)]}{\left(\int dy\ \exp[-\hat{\nu}_1(y,t)]\right)^n}. \tag{C.17}$$

In this case, we may also write:

$$\begin{aligned} \tilde{p}(m,\boldsymbol{x},t) &\approx \delta_{m,n} \exp\left[-\sum_{i=1}^{n} \nu_1(x_i,t)\right], \\ \nu_1(x_i,t) &= \hat{\nu}_1(x_i,t) + \log\left(\int dy\ \exp[-\hat{\nu}_1(y,t)]\right). \end{aligned} \tag{C.18}$$

C.3.2 Derivation of diffusion equation for ν_1

Assume the form of the reduced model is:

$$\tilde{p}(\boldsymbol{x},t) = \exp\left[-\sum_{i=1}^{n} \nu_1(x_i,t)\right]. \tag{C.19}$$

Substituting $\tilde{p}$ into the diffusion equation, we find:

$$\partial_t \nu_1(x,t) = D\nabla^2 \nu_1(x,t) - D\left(\nabla \nu_1(x,t)\right)^2. \tag{C.20}$$

Note that $\hat{\nu}_1$ obeys:

$$-\partial_t \hat{\nu}_1(x_i,t) - \frac{\int dy\ \left(\partial_t \hat{\nu}_1(y,t)\right) \exp[-\hat{\nu}_1(y,t)]}{\int dy\ \exp[-\hat{\nu}_1(y,t)]} = D\left(\nabla_i \hat{\nu}_1(x_i,t)\right)^2 - D\nabla_i^2 \hat{\nu}_1(x_i,t). \tag{C.21}$$

If we start with an initial Gaussian distribution, then:

$$\begin{aligned} \hat{\nu}_1(x_i,t) &= \frac{(x_i-\mu)^2}{4Dt}, \\ \int dy\ \exp[-\hat{\nu}_1(y,t)] &= (4\pi Dt)^{3/2}, \\ \nu_1(x_i,t) &= \frac{(x_i-\mu)^2}{4Dt} + \frac{3}{2}\log(4\pi Dt). \end{aligned} \tag{C.22}$$

Only for the Gaussian distribution, we can evaluate the integral term in the differential equation (C.21) for $\hat{\nu}_1$:

$$\frac{\int dy\ (\partial_t \hat{\nu}_1(y,t))\exp[-\hat{\nu}_1(y,t)]}{\int dy\ \exp[-\hat{\nu}_1(y,t)]} = -\frac{1}{2t}. \tag{C.23}$$

C.3.3 Weak formulation of forward problem

Approximate the time derivative using backward Euler

$$\frac{\nu_1^{(n+1)}(x)-\nu_1^{(n)}(x)}{\Delta t} = D\nabla^2\nu_1^{(n+1)}(x) - D\left(\nabla\nu_1^{(n+1)}(x)\right)^2, \tag{C.24}$$

and rearrange to collect $n+1$ terms on one side:

$$\nu_1^{(n+1)}(x) - \Delta t D\nabla^2\nu_1^{(n+1)}(x) + \Delta t D\left(\nabla\nu_1^{(n+1)}(x)\right)^2 = \nu_1^{(n)}(x). \tag{C.25}$$

Introducing test function v and integrating over the domain Ω with differential element $d\omega$ gives

$$\begin{aligned} &\int_\Omega dx\ \nu_1^{(n+1)}(x)v(x) - \Delta t D\int_\Omega dx\ \nabla^2\nu_1^{(n+1)}(x)v(x) + \Delta t D\int_\Omega dx\ \left(\nabla\nu_1^{(n+1)}(x)\right)^2 v(x) \\ &= \int_\Omega dx\ \nu_1^{(n)}(x)v(x). \end{aligned} \tag{C.26}$$

Integrate by parts on the second derivative term:

$$\begin{aligned}\int_\Omega dx\, \nabla^2 \nu_1^{(n+1)}(x) v(x) &= \int_\Gamma dx\, \nabla \nu_1^{(n+1)}(x) \cdot \hat{n} v(x) - \int_\Omega dx\, \nabla \nu_1^{(n+1)}(x) \cdot \nabla v(x) \\ &= -\int_\Omega dx\, \nabla \nu_1^{(n+1)}(x) \cdot \nabla v(x).\end{aligned} \tag{C.27}$$

where Γ is the boundary and $\hat{n}$ the outward facing unit normal vector. The boundary term vanishes because the trial functions v are required to vanish where the solution is known.

The weak form can be written in the following form:

$$G(\nu_1^{(n+1)}(x), v) = 0. \tag{C.28}$$

Note that we don't write an explicit dependence on $\nu_1^{(n)}$ because it is already known (solution of the previous step). Specifically:

$$\begin{aligned}G(\nu_1^{(n+1)}(x), v) = &\int_\Omega dx\, \nu_1^{(n+1)}(x) v(x) + \Delta t D \int_\Omega dx\, \nabla \nu_1^{(n+1)}(x) \cdot \nabla v(x) \\ &+ \Delta t D \int_\Omega dx\, \left(\nabla \nu_1^{(n+1)}(x)\right)^2 v(x) - \int_\Omega dx\, \nu_1^{(n)}(x) v(x) = 0.\end{aligned} \tag{C.29}$$

Note that this is not linear due to the $(\nabla \nu_1)^2$ term. If it were linear, it would be advantageous instead to write it in the following form:

$$a(\nu_1^{(n+1)}(x), v) = L(v). \tag{C.30}$$

C.3.4 Derivation of the adjoint equation

Introduce an adjoint function $\xi(y,t)$ to enforce the differential equation constraint for the Lagrange multiplier $\nu_1(y,t)$ which controls the constraint for the mean number of particles at y:

$$\mu_{\tilde{p}}(y,t) = \int d\boldsymbol{x}\, \tilde{p}(\boldsymbol{x},t) \left(\sum_{i=1}^{n} \delta(x_i - y) \right), \tag{C.31}$$

Recall that ν_1 obeys differential equation (C.20):

$$\begin{aligned} \partial_t \nu_1(x,t) &= \mathcal{F}[\nu_1](x,t), \\ \mathcal{F}[\nu_1](x,t) &= D\nabla^2 \nu_1(x,t) - D\left(\nabla \nu_1(x,t)\right)^2. \end{aligned} \tag{C.32}$$

Define the Lagrangian and action:

$$\begin{aligned} \mathcal{L}[\nu_1,\xi](t) &= \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) + \int dx\, \xi(x,t)\left(\partial_t \nu_1(x,t) - \mathcal{F}[\nu_1](x,t)\right), \\ \mathcal{J}[\nu_1,\xi] &= \int_{t_0}^{t_f} dt\, \mathcal{L}[\nu_1,\xi](t) \end{aligned} \tag{C.33}$$

from which the variational derivative with respect to ν_1 should vanish:

$$\begin{aligned} 0 &= \lim_{\epsilon\to 0} \frac{\mathcal{J}[\nu_1+\epsilon\delta\nu_1,\xi] - \mathcal{J}[\nu_1,\xi]}{\epsilon} \\ &= \int_{t_0}^{t_f} dt \int dx\, \Delta\mu(x,t)[\delta\nu_1(x,t)] \\ &\quad + \int_{t_0}^{t_f} dt \int dx\, \xi(x,t)\left(\partial_t[\delta\nu_1(x,t)] - D\nabla^2[\delta\nu_1(x,t)] + 2D\nabla\nu_1(x,t)\cdot\nabla[\delta\nu_1(x,t)]\right), \end{aligned} \tag{C.34}$$

where

$$\Delta\mu(y,t) = \mu_{\tilde{p}}(y,t) - \mu_p(y,t). \tag{C.35}$$

Integrating by parts gives the following result:

$$\begin{aligned} &\int_{t_0}^{t_f} dt \int dx\, \xi(x,t)\partial_t[\delta\nu_1(x,t)] \\ &= \int dx\, \xi(x,t)\delta\nu_1(x,t)\Big|_{t=t_0}^{t=t_f} - \int_{t_0}^{t_f} dt \int dx\, \partial_t\xi(x,t)\delta\nu_1(x,t) \\ &= \int dx\, \xi(x,t_f)\delta\nu_1(x,t_f) - \int_{t_0}^{t_f} dt \int dx\, \partial_t\xi(x,t)\delta\nu_1(x,t), \end{aligned} \tag{C.36}$$

where one boundary vanished because $\delta\nu_1(x,t_0)=0$ because $\nu_1(x,t_0)$ is specified. Furthermore:

$$\begin{aligned}\int_{t_0}^{t_f} dt \int dx\, \xi(x,t)\nabla^2[\delta\nu_1(x,t)] =& \int_{t_0}^{t_f} dt \int_\Gamma dx\, \xi(x,t)\nabla[\delta\nu_1(x,t)]\cdot\hat{n} \\ &- \int_{t_0}^{t_f} dt \int dx\, \nabla\xi(x,t)\cdot\nabla[\delta\nu_1(x,t)] \\ =& \int_{t_0}^{t_f} dt \int_\Gamma dx\, \xi(x,t)\nabla[\delta\nu_1(x,t)]\cdot\hat{n} \\ &- \int_{t_0}^{t_f} dt \int_\Gamma dx\, (\nabla\xi(x,t)\cdot\hat{n})\,\delta\nu_1(x,t) \\ &+ \int_{t_0}^{t_f} dt \int dx\, \nabla^2\xi(x,t)\delta\nu_1(x,t),\end{aligned} \tag{C.37}$$

and

$$\begin{aligned}\int_{t_0}^{t_f} dt \int dx\, \xi(x,t)\nabla\nu_1(x,t)\cdot\nabla_i[\delta\nu_1(x,t)] =& \int_{t_0}^{t_f} dt \int_\Gamma dx\, \xi(x,t)\,(\nabla\nu_1(x,t)\cdot\hat{n})\,\delta\nu_1(x,t) \\ &- \int_{t_0}^{t_f} dt \int dx\, \xi(x,t)\nabla^2\nu_1(x,t)\delta\nu_1(x,t) \\ &- \int_{t_0}^{t_f} dt \int dx\, \nabla\xi(x,t)\cdot\nabla\nu_1(x,t)\delta\nu_1(x,t).\end{aligned} \tag{C.38}$$

Put together, this gives for the variational derivative (C.34):

$$\begin{aligned}0 =& \lim_{\epsilon\to 0}\frac{\mathcal{J}[\nu_1+\epsilon\delta\nu_1,\xi]-\mathcal{J}[\nu_1,\xi]}{\epsilon} \\ =& \int_{t_0}^{t_f} dt \int dx \Big\{\Delta\mu(x,t)-\partial_t\xi(x,t)-D\nabla^2\xi(x,t)-2D\xi(x,t)\nabla^2\nu_1(x,t) \\ &\qquad -2D\nabla\xi(x,t)\cdot\nabla\nu_1(x,t)\Big\}\delta\nu_1(x,t) \\ &+\int dx\,\xi(x,t_f)\delta\nu_1(x,t_f)-\int_{t_0}^{t_f} dt \int_\Gamma dx\, D\xi(x,t)\nabla[\delta\nu_1(x,t)]\cdot\hat{n} \\ &+\int_{t_0}^{t_f} dt \int_\Gamma dx \Big\{D\,(\nabla\xi(x,t)\cdot\hat{n})\,\delta\nu_1(x,t)+2D\xi(x,t)\,(\nabla\nu_1(x,t)\cdot\hat{n})\,\delta\nu_1(x,t)\Big\}.\end{aligned} \tag{C.39}$$

To proceed, we must include the boundary conditions.

With Dirichlet boundary conditions

Suppose we have Dirichlet conditions on the boundary $x \in \Gamma$ at all times $t \in [t_0, t_f]$:

$$\nu_1(x,t) = g(x,t). \tag{C.40}$$

We can introduce another Lagrange multiplier η to enforce this constraint:

$$\mathcal{J}[\nu_1,\xi] \to \mathcal{J}[\nu_1,\xi] + \int_{t_0}^{t_f} dt \int_\Gamma dx\, \eta(x,t)(\nu_1(x) - g(x,t)). \tag{C.41}$$

In the variational derivative, we pick up and extra term:

$$\begin{aligned}
0 =& \lim_{\epsilon\to 0} \frac{\mathcal{J}[\nu_1 + \epsilon\delta\nu_1, \xi] - \mathcal{J}[\nu_1,\xi]}{\epsilon} \\
=& \int_{t_0}^{t_f} dt \int dx \Big\{ \Delta\mu(x,t) - \partial_t\xi(x,t) - D\nabla^2\xi(x,t) - 2D\xi(x,t)\nabla^2\nu_1(x,t) \\
& \qquad - 2D\nabla\xi(x,t)\cdot\nabla\nu_1(x,t) \Big\}\delta\nu_1(x,t) \\
& + \int dx\, \xi(x,t_f)\delta\nu_1(x,t_f) - \int_{t_0}^{t_f} dt \int_\Gamma dx\, D\xi(x,t)\nabla[\delta\nu_1(x,t)]\cdot\hat{n} \\
& + \int_{t_0}^{t_f} dt \int_\Gamma dx \Big\{ D\left(\nabla\xi(x,t)\cdot\hat{n}\right) + 2D\xi(x,t)\left(\nabla\nu_1(x,t)\cdot\hat{n}\right) + \eta(x,t) \Big\}\delta\nu_1(x,t).
\end{aligned} \tag{C.42}$$

Here in particular see the discussion in Ref. [112] p. 20. Each term must vanish independently. For points in the interior $x \in \Omega$:

$$\begin{aligned}
&\partial_t\xi(x,t) = \Delta\mu(x,t) - D\nabla^2\xi(x,t) - 2D\xi(x,t)\nabla^2\nu_1(x,t) - 2D\nabla\xi(x,t)\cdot\nabla\nu_1(x,t), \\
&\xi(x,t_f) = 0,
\end{aligned} \tag{C.43}$$

and for points on the boundary $x \in \Gamma$:

$$\begin{aligned} \xi(x,t) &= 0, \\ D\left(\nabla\xi(x,t)\cdot\hat{n}\right) + 2D\xi(x,t)\left(\nabla\nu_1(x,t)\cdot\hat{n}\right) + \eta(x,t) &= 0. \end{aligned} \tag{C.44}$$

In practice, since the Lagrange multiplier η only appears in the final equation, it can be ignored unless we care about determining its value.

With Neumann boundary conditions

Suppose instead we have Neumann conditions on the boundary $x \in Gamma$ at all times $t \in [t_0, t_f]$:

$$\nabla_i \nu_1(x,t)\cdot\hat{n} = g(x,t). \tag{C.45}$$

We can introduce another Lagrange multiplier η to enforce this constraint:

$$\mathcal{J}[\nu_1,\xi] \to \mathcal{J}[\nu_1,\xi] + \int_{t_0}^{t_f} dt \int_\Gamma dx\, \eta(x,t)(\nabla\nu_1(x) - g(x,t)). \tag{C.46}$$

In the variational derivative, we pick up and extra term:

$$\begin{aligned} 0 &= \lim_{\epsilon\to 0} \frac{\mathcal{J}[\nu_1+\epsilon\delta\nu_1,\xi] - \mathcal{J}[\nu_1,\xi]}{\epsilon} \\ &= \int_{t_0}^{t_f} dt \int dx \Big\{ \Delta\mu(x,t) - \partial_t\xi(x,t) - D\nabla^2\xi(x,t) - 2D\xi(x,t)\nabla^2\nu_1(x,t) \\ &\qquad - 2D\nabla\xi(x,t)\cdot\nabla\nu_1(x,t) \Big\}\delta\nu_1(x,t) \\ &\quad + \int dx\, \xi(x,t_f)\delta\nu_1(x,t_f) + \int_{t_0}^{t_f} dt \int_\Gamma dx\, \left(-D\xi(x,t) + \eta(x,t)\right)\nabla[\delta\nu_1(x,t)]\cdot\hat{n} \\ &\quad + \int_{t_0}^{t_f} dt \int_\Gamma dx \Big\{ D\left(\nabla\xi(x,t)\cdot\hat{n}\right) + 2D\xi(x,t)\left(\nabla\nu_1(x,t)\cdot\hat{n}\right) \Big\}\delta\nu_1(x,t). \end{aligned} \tag{C.47}$$

Here in particular see the discussion in Ref. [112] p. 20. Each term must vanish independently. For points in the interior $x \in \Omega$:

$$\begin{aligned} \partial_t \xi(x,t) &= \Delta\mu(x,t) - D\nabla^2\xi(x,t) - 2D\xi(x,t)\nabla^2\nu_1(x,t) - 2D\nabla\xi(x,t)\cdot\nabla\nu_1(x,t), \\ \xi(x,t_f) &= 0, \end{aligned} \tag{C.48}$$

and on the boundary $x \in \Gamma$:

$$\begin{aligned} -D\xi(x,t) + \eta(x,t) &= 0, \\ \nabla\xi(x,t)\cdot\hat{n} + 2\xi(x,t)g(x,t) &= 0, \end{aligned} \tag{C.49}$$

where we have used the definition of $g(x,t)$ in the last equation. In practice, since the Lagrange multiplier η only appears in the final equation, it can be ignored unless we care about determining its value.

C.3.5 Weak formulation of the adjoint equations

In the forward problem, we solve from $t = t_0$ to $t = t_f$, we use a backward difference scheme, i.e. equation for $(n+1)$ given (n). In the adjoint problem, we solve from $t = t_f$ to $t = t_0$, we use a forward difference scheme, i.e. equation for (n) given $(n+1)$. Therefore, let:

$$\partial_t \xi(y,t) \approx \frac{\xi^{(n+1)}(y) - \xi^{(n)}(y)}{\Delta t}, \tag{C.50}$$

where:

$$\begin{aligned} &\frac{\xi^{(n+1)}(y) - \xi^{(n)}(y)}{\Delta t} \\ =&\Delta\mu^{(n)}(y) - D\nabla^2\xi^{(n)}(y) - 2D\nabla\xi^{(n)}(y)\cdot\nabla\nu_1^{(n)}(y) - 2D\xi^{(n)}(y)\nabla^2\nu_1^{(n)}(y). \end{aligned} \tag{C.51}$$

Collect (n) terms on one side:

$$\begin{aligned}&\frac{\xi^{(n)}(y)}{\Delta t} - D\nabla^2\xi^{(n)}(y) - 2D\nabla\xi^{(n)}(y)\cdot\nabla\nu_1^{(n)}(y) - 2D\xi^{(n)}(y)\nabla^2\nu_1^{(n)}(y)\\ =&\frac{\xi^{(n+1)}(y)}{\Delta t} - \Delta\mu^{(n)}(y).\end{aligned} \tag{C.52}$$

Introduce trial functions $v(y)$:

$$\begin{aligned}&\int dy\,\frac{\xi^{(n)}(y)}{\Delta t}v(y) - D\int dy\,\nabla^2\xi^{(n)}(y)v(y)\\ &-2D\int dy\,\nabla\xi^{(n)}(y)\cdot\nabla\nu_1^{(n)}(y)v(y) - 2D\int dy\,\xi^{(n)}(y)\nabla^2\nu_1^{(n)}(y)v(y)\\ =&\int dy\,\frac{\xi^{(n+1)}(y)}{\Delta t}v(y) - \int dy\,\Delta\mu^{(n)}(y)v(y).\end{aligned} \tag{C.53}$$

Integrating by parts gives:

$$\begin{aligned}\int dy\,\nabla^2\xi^{(n)}(y)v(y) =& \int_\Gamma dy\left(\nabla\xi^{(n)}(y)\cdot\hat{n}\right)v(y) - \int dy\,\nabla\xi^{(n)}(y)\cdot\nabla v(y),\\ \int dy\,\xi^{(n)}(y)\nabla^2\nu_1^{(n)}(y)v(y) =& \int_\Gamma dy\,\xi^{(n)}(y)v(y)\nabla\nu_1^{(n)}(y)\cdot\hat{n}\\ &-\int dy\,\nabla(\xi^{(n)}(y)v(y))\cdot\nabla\nu_1^{(n)}(y)\\ =&\int_\Gamma dy\,\xi^{(n)}(y)v(y)\nabla\nu_1^{(n)}(y)\cdot\hat{n}\\ &-\int dy\,\nabla\xi^{(n)}(y)\cdot\nabla\nu_1^{(n)}(y)v(y)\\ &-\int dy\,\xi^{(n)}(y)\nabla\nu_1^{(n)}(y)\cdot\nabla v(y).\end{aligned} \tag{C.54}$$

This gives the weak form after some terms cancel:

$$\begin{aligned}&\int dy\frac{\xi^{(n)}(y)}{\Delta t}v(y) - D\int_\Gamma dy\,\left(\nabla\xi^{(n)}(y)\cdot\hat{n}\right)v(y) + D\int dy\,\nabla\xi^{(n)}(y)\cdot\nabla v(y)\\ &+2D\int dy\,\xi^{(n)}(y)\nabla\nu_1^{(n)}(y)\cdot\nabla v(y) - 2D\int_\Gamma dy\,\xi^{(n)}(y)v(y)\nabla\nu_1^{(n)}(y)\cdot\hat{n}\\ =&\int dy\,\frac{\xi^{(n+1)}(y)}{\Delta t}v(y) - \int dy\,\Delta\mu^{(n)}(y)v(y).\end{aligned} \tag{C.55}$$

The boundary conditions for $v(y)$ are such that it vanishes where ν_1 is known. This is detailed for the two cases below.

Dirichlet boundary conditions

For Dirichlet boundary conditions, we have that $v(y) = 0$ on the boundary $y \in \Gamma$. Further, we had that $\xi(y) = 0$ on the boundary. Therefore both boundary conditions vanish, and we are left with:

$$\begin{aligned} &\int dy \frac{\xi^{(n)}(y)}{\Delta t} v(y) + D \int dy \nabla \xi^{(n)}(y) \cdot \nabla v(y) + 2D \int dy \xi^{(n)}(y) \nabla \nu_1^{(n)}(y) \cdot \nabla v(y) \\ &= \int dy \frac{\xi^{(n+1)}(y)}{\Delta t} v(y) - \int dy \Delta \mu^{(n)}(y) v(y). \end{aligned} \tag{C.56}$$

Neumann boundary conditions

For Neumann boundary conditions, we have $\nabla v(y) \cdot \hat{n} = 0$ on the boundary $y \in \Gamma$. We also have: $\nabla \nu_1(y) \cdot \hat{n} = g(y)$ by definition, and from the boundary equations $\nabla \xi(y) \cdot \hat{n} = -2\xi(y)g(y)$. Then:

$$\begin{aligned} &\int dy \, \frac{\xi^{(n)}(y)}{\Delta t} v(y) + 2D \int_\Gamma dy \, \xi^{(n)}(y) g(y) v(y) + D \int dy \, \nabla \xi^{(n)}(y) \cdot \nabla v(y) \\ &+ 2D \int dy \, \xi^{(n)}(y) \nabla \nu_1^{(n)}(y) \cdot \nabla v(y) - 2D \int_\Gamma dy \, \xi^{(n)}(y) v(y) g(y) \\ &= \int dy \, \frac{\xi^{(n+1)}(y)}{\Delta t} v(y) - \int dy \, \Delta \mu^{(n)}(y) v(y). \end{aligned} \tag{C.57}$$

Note that the boundary terms have cancelled, leaving the same result as for the Dirichlet boundary conditions.

C.3.6 Optimality condition

The optimality condition is for all boundary conditions:

$$\begin{aligned}\frac{dS}{dD} &= -\int_{t_0}^{t_f} dt \int dx \, \frac{\partial \mathcal{F}[\nu_1](x,t)}{\partial D} \xi(x,t) \\ &= \int_{t_0}^{t_f} dt \int dx \, \left(-\nabla^2 \nu_1(x,t) + \left(\nabla \nu_1(x,t)\right)^2 \right) \xi(x,t).\end{aligned} \tag{C.58}$$

We can use integration by parts to eliminate an undesired second derivative in the optimality condition:

$$\begin{aligned}\int_{t_0}^{t_f} dt \int dx \, \nabla^2 \nu_1(x,t) \xi(x,t) = &\int_{t_0}^{t_f} dt \int_{\Gamma} dx \, \left(\nabla_i \nu_1(x,t) \cdot \hat{n}\right) \xi(x,t) \\ &- \int_{t_0}^{t_f} dt \int dx \, \nabla \nu_1(x,t) \cdot \nabla \xi(x,t).\end{aligned} \tag{C.59}$$

To proceed further, we must consider the boundary conditions again.

Dirichlet boundary conditions

We have $\xi(x,t) = 0$ on the boundary $x \in \Gamma$, such that the boundary term vanishes:

$$\frac{dS}{dD} = \int_{t_0}^{t_f} dt \int dx \, \left(\nabla \nu_1(x,t) \cdot \nabla \xi(x,t) + \left(\nabla \nu_1(x,t)\right)^2 \xi(x,t) \right). \tag{C.60}$$

Neumann boundary conditions

We have specified $\nabla_i \nu_1(x,t) \hat{\dot{c}} n = g(x,t)$ on the boundary $x \in \Gamma$, such that:

$$\begin{aligned}\frac{dS}{dD} = &-\int_{t_0}^{t_f} dt \int_{\Gamma} dx \, g(x,t) \xi(x,t) \\ &+ \int_{t_0}^{t_f} dt \int dx \, \left(\nabla \nu_1(x,t) \cdot \nabla \xi(x,t) + \left(\nabla \nu_1(x,t)\right)^2 \xi(x,t) \right).\end{aligned} \tag{C.61}$$

C.4 Physics-based Gaussian graphical models

C.4.1 Convert between moment and interaction space

The covariance matrix Σ is divided into free $\boldsymbol{\sigma}^f$ and constrained terms $\boldsymbol{\sigma}^c$. The free terms evolve exactly according to the CME $\dot{\boldsymbol{\sigma}}^f$. The time evolution of the constrained terms can then be solved for. To simplify notation, introduce the vector notations: $\boldsymbol{\sigma}^f$ and $\boldsymbol{\sigma}^c$ of lengths n^f and n^c. Also define the functions to map indexes:

$$\begin{aligned} c_1(i), c_2(i) \ni \boldsymbol{\sigma}_i^c = \Sigma_{c_1(i),c_2(i)} = \Sigma^c_{c_1(i),c_2(i)}, \\ f_1(i), f_2(i) \ni \boldsymbol{\sigma}_i^f = \Sigma_{f_1(i),f_2(i)} = \Sigma^f_{f_1(i),f_2(i)}. \end{aligned} \tag{C.62}$$

Given the equations for $d\boldsymbol{\mu}/dt$ and $d\boldsymbol{\sigma}^f/dt$, the goal is to evaluate $d\boldsymbol{\nu}/dt$ at the current $\boldsymbol{\nu}$. Here the covariance matrix Σ is divided into free $\boldsymbol{\sigma}^f$ and constrained terms $\boldsymbol{\sigma}^c$. The result is:

$$\begin{aligned} \frac{dB}{dt} &= \frac{d\Sigma^{-1}}{dt} = -\Sigma^{-1}\frac{d\Sigma}{dt}\Sigma^{-1} = -B\frac{d\Sigma}{dt}B, \\ \frac{d\boldsymbol{\nu}_1}{dt} &= -\frac{dB}{dt}\boldsymbol{\mu} - B\frac{d\boldsymbol{\mu}}{dt} + \frac{1}{2}\mathrm{diag}\left(\frac{dB}{dt}\right). \end{aligned} \tag{C.63}$$

Note that we have to divide the terms of the covariance matrix Σ into free $\boldsymbol{\sigma}^f$ and constrained $\boldsymbol{\sigma}^c$ parts. The equations for the free part $\dot{\boldsymbol{\sigma}}^f$ are given by the CME, while the constrained ones are

$$\dot{\boldsymbol{\sigma}}^c = -(G^c)^{-1}G^f\dot{\boldsymbol{\sigma}}^f, \tag{C.64}$$

where

$$
\begin{aligned}
G^c &= B^{c_1,c_1} \circ B^{c_2,c_2} + \mathcal{D}^c \circ B^{c_1,c_2} \circ B^{c_2,c_1}, \\
G^f &= B^{c_1,f_1} \circ B^{c_2,f_2} + \mathcal{D}^f \circ B^{c_1,f_2} \circ B^{c_2,f_1}, \\
\mathcal{D}^c_{ik} &= (1 - \delta_{c_1(k),c_2(k)}), \\
\mathcal{D}^f_{ik} &= (1 - \delta_{f_1(k),f_2(k)}), \\
B^{c_1,c_1}_{ik} &= B_{c_1(i),c_1(k)} \text{ and similarly for the others.}
\end{aligned}
\tag{C.65}
$$

C.4.2 Sampling Gaussian distribution

The Gaussian distribution is:

$$
\tilde{p}(\boldsymbol{n}) = \frac{1}{\sqrt{(2\pi)^m |B^{-1}|}} \exp\left[-\frac{1}{2} (\boldsymbol{n} - \boldsymbol{a})^\intercal B (\boldsymbol{n} - \boldsymbol{a}) \right]. \tag{C.66}
$$

For the awake phase, divide the $\boldsymbol{n}$ into visible variables $\boldsymbol{n}_v$ and hidden $\boldsymbol{n}_h$ as:

$$
\boldsymbol{n} = \begin{pmatrix} \boldsymbol{n}_v \\ \boldsymbol{n}_h \end{pmatrix}. \tag{C.67}
$$

For a generic vector and symmetric matrix, let:

$$
\boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_v \\ \boldsymbol{x}_h \end{pmatrix} \quad \& \quad A = \begin{pmatrix} C_v & C_{vh}^\intercal \\ C_{vh} & C_h \end{pmatrix}. \tag{C.68}
$$

Under the awake phase, we must clamp the visible units to the data. The conditional

distribution for the hidden units is then:

$$\begin{aligned}\tilde{p}(\boldsymbol{n}_h|\boldsymbol{n}_v) =& \exp\left[-\frac{1}{2}(\boldsymbol{n}_h - \boldsymbol{a}_h + B_h^{-1}\, B_{vh}(\boldsymbol{n}_v - \boldsymbol{a}_v))^\intercal B_h(\boldsymbol{n}_h - \boldsymbol{a}_h + B_h^{-1}\, B_{vh}(\boldsymbol{n}_v - \boldsymbol{a}_v))\right] \\ &\times \frac{1}{\sqrt{(2\pi)^{m_h}|B_h^{-1}|}} \times \sqrt{\frac{|B_v^{-1}||B_h^{-1}|}{|B^{-1}|}} \exp\left[\frac{1}{2}(\boldsymbol{n}_v - \boldsymbol{a}_v)^\intercal B_{vh}^\intercal B_h^{-1} B_{vh}(\boldsymbol{n}_v - \boldsymbol{a}_v)\right].\end{aligned} \tag{C.69}$$

This is a Gaussian in n_h. The mean and variance are:

$$\begin{aligned}\langle \boldsymbol{n}_h \rangle &= \boldsymbol{a}_h - B_h^{-1}\, B_{vh}(\boldsymbol{n}_v - \boldsymbol{a}_v), \\ \Sigma &= B_h^{-1}.\end{aligned} \tag{C.70}$$

Note that the other terms cancel because they are just a normalizing factor when integrating over $\boldsymbol{n}_h$.

For the asleep phase, the moments follow directly from (C.66).

Appendix D

Deep learning moment closure approximations

D.1 Variational problem for dynamic Boltzmann distributions in continuous space

A general variational problem for the reduced model describing a reaction-diffusion systems in continuous space can be formulated as shown in [17]. Here we review the key results, and the show connection to the deep Boltzmann machine (DBM) notation used in this chapter.

D.1.1 Review of variational problem

The state of a reaction-diffusion system at time t is described by n particles of species $\boldsymbol{\alpha}$ at locations $\boldsymbol{x}$. Introduce k-particle interaction functions $\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n}, \boldsymbol{x}_{\langle i\rangle_k^n}, t)$ for $k = 1, \ldots, K$ (i.e. cutoff K), where $\langle i\rangle_k^n$ denotes an ordered subset of k indexes with each index in $\{1, \ldots, n\}$. For example, $\langle i\rangle_2^3$ denotes the possible subsets $\{1,2\}, \{1,3\}, \{2,3\}$,

and ν_2 therefore considers all pairwise interactions between three particles. The dynamic Boltzmann distribution is

$$\tilde{p}(n,\boldsymbol{\alpha},\boldsymbol{x},t;\{\nu\}) = \frac{1}{Z[\{\nu\}]}\exp\left[-\sum_{k=1}^{K}\sum_{\langle i\rangle_k^n}\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n},\boldsymbol{x}_{\langle i\rangle_k^n},t)\right]. \tag{D.1}$$

Introduce a general functional $\mathcal{F}_k$ to describe the time evolution of each interaction function

$$\frac{d}{dt}\nu_k(\boldsymbol{\alpha}_{\langle i\rangle_k^n},\boldsymbol{x}_{\langle i\rangle_k^n},t) = \mathcal{F}_k[\{\nu\},\{F_k\}](\boldsymbol{\alpha},\boldsymbol{x},t), \tag{D.2}$$

where $\mathcal{F}_k$ are functionals of the interactions $\{\nu\}$ and some unspecified set of ordinary functions $\{F_k\}$. This allows e.g. a PDE to be introduced as the reduced model.

If the form of the functionals $\mathcal{F}_k$ is chosen, a variational problem can be formulated for the functions $\{F_k\}$ appearing on the right hand side. Variational problems of this form for the functions appearing on the right hand side of a differential equation have been considered previously [56, 54, 55]. In the current framework the variational problem to extremize an action

$$S = \int_0^T dt\, \mathcal{D}_{\mathcal{KL}}(p||\tilde{p}) \tag{D.3}$$

is derived in [17] - the result is

$$\delta S = -\int_0^T dt\, \sum_{n=0}^{\infty}\sum_{\boldsymbol{\alpha}}\int d\boldsymbol{x}\, \sum_{k=1}^{K}\sum_{\langle i\rangle_k^n}\sum_{s=1}^{M_k}\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta F_k^{(s)}(\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\})}\delta F_k^{(s)}(\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\}) = 0, \tag{D.4}$$

where

$$\mathcal{J}[\{\nu\},\{\zeta\}](t) = \sum_{n'=0}^{\infty}\sum_{\boldsymbol{\alpha}'}\int d\boldsymbol{x}'\, \boldsymbol{\zeta}^{\intercal}(\boldsymbol{\alpha}',\boldsymbol{x}',t)\boldsymbol{\mathcal{F}}[\{\nu\}](\boldsymbol{\alpha}',\boldsymbol{x}',t), \tag{D.5}$$

and the adjoint system is

$$\frac{d\boldsymbol{\zeta}}{dt} = \Delta\boldsymbol{\mu}(\boldsymbol{\alpha},\boldsymbol{\beta},t) - \frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta\boldsymbol{\nu}(\boldsymbol{\alpha},\boldsymbol{x},t)}. \tag{D.6}$$

where $\Delta\boldsymbol{\mu}$ are the differences in the moments (by convention, the average under the model distribution minus the average under the data distribution) for which $\boldsymbol{\nu}$ are the Lagrange multipliers.

This is the variational calculus form of the sensitivity equation and adjoint system in Section 4.3. For further details we refer to [17].

D.1.2 Recover formalism for dynamic centered deep Boltzmann machines

To recover the DBM formalism of the current work, we first make the connection between k-particle interaction functions in continuous space and interaction parameters between discrete sites organized into layers. In this case, only self and pairwise interactions between neighboring layers exist, i.e. $K = 2$. Assign to every unit $s_{i,\alpha}^{(l)}$ a spatial location $x_i^{(l)}$ (including latent variables). First, consider the non-centered case $\mu_\alpha^{(l)} \to 0$, then the connection is:

$$\begin{aligned} \nu_1(\alpha, x, t) &\to -\sum_{l=0}^{L-1} \sum_{i=1}^{N^{(l)}} \delta_{x,x_i^{(l)}} a_\alpha^{(l)}(t), \\ \nu_2(\alpha, \beta, x, y, t) &\to -\sum_{l=0}^{L-2} \sum_{\langle ij \rangle} \delta_{x,x_i^{(l)}} \delta_{x,x_j^{(l+1)}} W_{\alpha\beta}^{(l,l+1)}(t), \end{aligned} \tag{D.7}$$

where $\delta_{x,y} = 1$ if $x = y$ and 0 otherwise is the Kronecker delta, and the negative sign accounts for the sign convention in the energy function (D.1). Similarly, the adjoint variables are

$$\begin{aligned} \zeta_1(\alpha, x, t) &\to \sum_{l=0}^{L-1} \sum_{i=1}^{N^{(l)}} \delta_{x,x_i^{(l)}} \phi_\alpha^{(l)}(t), \\ \zeta_2(\alpha, \beta, x, y, t) &\to \sum_{l=0}^{L-2} \sum_{\langle ij \rangle} \delta_{x,x_i^{(l)}} \delta_{x,x_j^{(l+1)}} \Lambda_{\alpha\beta}^{(l,l+1)}(t), \end{aligned} \tag{D.8}$$

Finally the constraint equations, originally formulated in terms of a functional $\mathcal{F}_k$ of a set of ordinary functions $\{F_k\}$, now become just an ordinary function:

$$\begin{aligned}\mathcal{F}_1[\{\nu\},\{F_1\}](\boldsymbol{\alpha},\boldsymbol{x},t) &\to -F^{(l)}_{a_\alpha}(\boldsymbol{\theta}(t);\boldsymbol{u}^{(l)}_{a_\alpha}),\\ \mathcal{F}_2[\{\nu\},\{F_2\}](\boldsymbol{\alpha},\boldsymbol{x},t) &\to -F^{(l,l+1)}_{W_{\alpha\beta}}(\boldsymbol{\theta}(t);\boldsymbol{u}^{(l,l+1)}_{W_{\alpha\beta}}).\end{aligned} \tag{D.9}$$

where the negative sign results from the convention for the energy function (D.1) as before.

To transform to the centered frame instead, the connection is:

$$\begin{aligned}\nu_1(\alpha,x,t) &\to -\sum_{l=0}^{L-1}\sum_{i=1}^{N^{(l)}}\delta_{x,x_i^{(l)}}\left(\tilde{a}^{(l)}_\alpha(t)-\sum_{\Delta l=\pm 1}q^{(l,l+\Delta l)}\sum_{\beta\in\mathcal{R}^{(l+\Delta l)}}\tilde{W}^{(l,l+\Delta l)}_{\alpha\beta}\mu^{(l+\Delta l)}_\beta\right),\\ \nu_2(\alpha,\beta,x,y,t) &\to -\sum_{l=0}^{L-2}\sum_{\langle ij\rangle}\delta_{x,x_i^{(l)}}\delta_{x,x_j^{(l+1)}}\tilde{W}^{(l,l+1)}_{\alpha\beta}(t),\end{aligned} \tag{D.10}$$

and the constraint equations describing the time evolution of the centered parameters now are:

$$\begin{aligned}&\mathcal{F}_1[\{\nu\},\{F_1\}](\boldsymbol{\alpha},\boldsymbol{x},t)\\ &\to -F^{(l)}_{a_\alpha}-\sum_{\Delta l=\pm 1}q^{(l,l+\Delta l)}\sum_{\beta\in\mathcal{R}^{(l+\Delta l)}}\left(F^{(l,l+\Delta l)}_{W_{\alpha\beta}}\mu^{(l+\Delta l)}_\beta+\tilde{W}^{(l,l+\Delta l)}_{\alpha\beta}\frac{d\mu^{(l+\Delta l)}_\beta}{dt}\right)\end{aligned} \tag{D.11}$$

and

$$\mathcal{F}_2[\{\nu\},\{F_2\}](\boldsymbol{\alpha},\boldsymbol{x},t)\to -F^{(l,l+1)}_{W_{\alpha\beta}}, \tag{D.12}$$

and the adjoint variables are unchanged.

The functional $\mathcal{J}$ in (D.5) becomes:

$$\mathcal{J}[\{\nu\},\{\zeta\}](t) \to -\sum_{l=0}^{L-2}\sum_{\alpha\in\mathcal{R}^{(l)}}\sum_{\beta\in\mathcal{R}^{(l+1)}}\Lambda_{\alpha\beta}^{(l,l+1)}F_{W_{\alpha\beta}}^{(l,l+1)} - \sum_{l=0}^{L-1}\sum_{\alpha\in\mathcal{R}^{(l)}}\phi_{\alpha}^{(l)}\times\left(F_{a_\alpha}^{(l)} + \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)}\sum_{\beta\in\mathcal{R}^{(l+\Delta l)}}\left(F_{W_{\alpha\beta}}^{(l,l+\Delta l)}\mu_\beta^{(l+\Delta l)} + \tilde{W}_{\alpha\beta}^{(l,l+\Delta l)}\frac{d\mu_\beta^{(l+\Delta l)}}{dt}\right)\right). \tag{D.13}$$

In the adjoint system (D.6), the variational derivatives become (see also Appendix D.4 for this result):

$$\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta\nu_1(\alpha,x,t)} \to -\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta\tilde{a}_\alpha^{(l)}} = \psi_{a_\alpha^{(l)}} \tag{D.14}$$

and

$$\begin{aligned}\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta\nu_2(\alpha,\beta,y,t)} \to -\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta\tilde{W}_{\alpha\beta}^{(l,l+1)}} =& -\psi_{W_{\alpha\beta}^{(l,l+1)}} + q^{(l,l+1)}\left(\mu_\alpha^{(l)}\psi_{a_\beta^{(l+1)}}\right.\\ &\left. + \mu_\beta^{(l+1)}\psi_{a_\alpha^{(l)}} - \phi_\beta^{(l+1)}\frac{d\mu_\alpha^{(l)}}{dt} - \phi_\alpha^{(l)}\frac{d\mu_\beta^{(l+1)}}{dt}\right),\end{aligned} \tag{D.15}$$

where ψ_θ from Section 4.3 is

$$\begin{aligned}\psi_\theta =& \sum_{m=0}^{L-2}\sum_{\zeta\in\mathcal{R}^{(m)}}\sum_{\eta\in\mathcal{R}^{(m+1)}}\Lambda_{\zeta\eta}^{(m,m+1)}\frac{\partial F_{W_{\zeta\eta}}^{(m,m+1)}}{\partial\theta} + \sum_{m=0}^{L-1}\sum_{\zeta\in\mathcal{R}^{(m)}}\phi_\zeta^{(m)}\times\\ &\left(\frac{\partial F_{a_\zeta}^{(m)}}{\partial\theta} + \sum_{\Delta m=\pm 1} q^{(m,m+\Delta m)}\sum_{\eta\in\mathcal{R}^{(m+\Delta m)}}\frac{\partial F_{W_{\zeta\eta}}^{(m,m+\Delta m)}}{\partial\theta}\mu_\eta^{(m+\Delta m)}\right).\end{aligned} \tag{D.16}$$

which substituted into (D.6) recovers the adjoint system in Section 4.3

$$\begin{aligned}
\frac{d}{dt}\phi_\alpha^{(l)} =& \sum_{i=1}^{N^{(l)}} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)}\right] - \psi_{a_\alpha^{(l)}}, \\
\frac{d}{dt}\Lambda_{\alpha\beta}^{(l,l+1)} =& \sum_{\langle ij\rangle} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)} s_{j,\beta}^{(l+1)}\right] - \psi_{W_{\alpha\beta}^{(l,l+1)}} \\
&+ q^{(l,l+1)}\left(\mu_\alpha^{(l)}\psi_{a_\beta^{(l+1)}} + \mu_\beta^{(l+1)}\psi_{a_\alpha^{(l)}} - \phi_\beta^{(l+1)}\frac{d\mu_\alpha^{(l)}}{dt} - \phi_\alpha^{(l)}\frac{d\mu_\beta^{(l+1)}}{dt}\right).
\end{aligned} \tag{D.17}$$

The condition for extremizing the action D.4 now gives the sensitivity equation - the variational derivatives become:

$$\begin{aligned}
\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta F_1^{(s)}(\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\})} \to -\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta F_{a_\alpha}^{(l)}} &= \phi_\alpha^{(l)}, \\
\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta F_2^{(s)}(\{\nu(\boldsymbol{\alpha},\boldsymbol{x},t)\})} \to -\frac{\delta\mathcal{J}[\{\nu\},\{\zeta\}](t)}{\delta F_{W_{\alpha\beta}}^{(l,l+1)}} &= \Lambda_{\alpha\beta}^{(l,l+1)} + q^{(l,l+1)}\phi_\alpha^{(l)}\mu_\beta^{(l+1)} + q^{(l,l+1)}\phi_\beta^{(l+1)}\mu_\alpha^{(l)},
\end{aligned} \tag{D.18}$$

which substituted into (D.4) recovers the update equations of Section 4.3:

$$\begin{aligned}
\frac{dS}{d\boldsymbol{u}_{a_\alpha}^{(l)}} =& -\int_0^T dt\ \phi_\alpha^{(l)}\frac{\partial F_{a_\alpha}^{(l)}}{\partial \boldsymbol{u}_{a_\alpha}^{(l)}}, \\
\frac{dS}{d\boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}} =& -\int_0^T dt\ \left(\Lambda_{\alpha\beta}^{(l,l+1)} + q^{(l,l+1)}\phi_\alpha^{(l)}\mu_\beta^{(l+1)} + q^{(l,l+1)}\phi_\beta^{(l+1)}\mu_\alpha^{(l)}\right)\frac{\partial F_{W_{\alpha\beta}}^{(l,l+1)}}{\partial \boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}}.
\end{aligned} \tag{D.19}$$

D.2 Derivation of the centered gradient for DBMs

Recall the setting of Section 3.3. The parameter transformations between regular DBM parameters $a_\alpha^{(l)}, W_{\alpha\beta}^{(l,l+1)}$ and centered DBM parameters $\tilde{a}_\alpha^{(l)}, \tilde{W}_{\alpha\beta}^{(l,l+1)}$ are:

$$\begin{aligned} \tilde{W}_{\alpha\beta}^{(l,l+1)} &= W_{\alpha\beta}^{(l,l+1)} \\ \tilde{a}_\alpha^{(l)} &= a_\alpha^{(l)} + \sum_{\Delta l = \pm 1} q^{(l,l+\Delta l)} \sum_{\beta \in \mathcal{R}^{(l+\Delta l)}} W_{\alpha\beta}^{(l,l+\Delta l)} \mu_\beta^{(l+\Delta l)}, \end{aligned} \tag{D.20}$$

where it is implicit that the sum should contain a delta function of the form $(1-\delta_{l,0}\delta_{\Delta l,-1}) \times (1-\delta_{l,L-1}\delta_{\Delta l,+1})$ since layer indexes are restricted to $l = 0, \ldots, L-1$.

After transforming the energy function to the centered parameters, taking the gradient with respect to centered parameters gives:

$$\begin{aligned} \Delta \tilde{W}_{\alpha\beta}^{(l,l+1)} &= \sum_{\langle ij \rangle} \Delta \mathbb{E} \left[(s_{i,\alpha}^{(l)} - \mu_\alpha^{(l)})(s_{j,\beta}^{(l+1)} - \mu_\beta^{(l+1)}) \right], \\ \Delta \tilde{a}_\alpha^{(l)} &= \sum_{i=1}^{N^{(l)}} \Delta \mathbb{E} \left[s_{i,\alpha}^{(l)} \right], \end{aligned} \tag{D.21}$$

where

$$\Delta \mathbb{E}[X] = \langle X \rangle^{(m)} - \langle X \rangle^{(d)} \tag{D.22}$$

is the *difference in expectation values*.

Parameters in the centered and regular DBMs are updated as:

$$\begin{aligned} \tilde{W}_{\alpha\beta}^{(\text{new},l,l+1)} = \tilde{W}_{\alpha\beta}^{(l,l+1)} - \eta \Delta \tilde{W}_{\alpha\beta}^{(l,l+1)} \quad \& \quad W_{\alpha\beta}^{(\text{new},l,l+1)} = W_{\alpha\beta}^{(l,l+1)} - \eta \Delta W_{\alpha\beta}^{(l,l+1)}, \\ \tilde{a}_\alpha^{(\text{new},l)} = \tilde{a}_\alpha^{(l)} - \eta \Delta \tilde{a}_\alpha^{(l)} \quad \& \quad a_\alpha^{(\text{new},l)} = a_\alpha^{(l)} - \eta \Delta a_\alpha^{(l)}. \end{aligned} \tag{D.23}$$

Transforming the new centered parameters back to identify the updates for the original

gives:

$$\begin{aligned} & \tilde{W}_{\alpha\beta}^{(\text{new},l,l+1)} = W_{\alpha\beta}^{(\text{new},l,l+1)}, \\ \Rightarrow \quad & \tilde{W}_{\alpha\beta}^{(l,l+1)} - \eta\Delta\tilde{W}_{\alpha\beta}^{(l,l+1)} = W_{\alpha\beta}^{(l,l+1)} - \eta\Delta W_{\alpha\beta}^{(l,l+1)}, \\ \Rightarrow \quad & \Delta W_{\alpha\beta}^{(l,l+1)} = \Delta\tilde{W}_{\alpha\beta}^{(l,l+1)}, \end{aligned} \tag{D.24}$$

and

$$\begin{aligned} & \tilde{a}_{\alpha}^{(\text{new},l)} = a_{\alpha}^{(\text{new},l)} + \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} W_{\alpha\beta}^{(\text{new},l,l+\Delta l)} \mu_{\beta}^{(l+\Delta l)}, \\ \Rightarrow \quad & \tilde{a}_{\alpha}^{(l)} - \eta\Delta\tilde{a}_{\alpha}^{(l)} = a_{\alpha}^{(l)} - \eta\Delta a_{\alpha}^{(l)} \\ & \qquad + \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} (W_{\alpha\beta}^{(l,l+\Delta l)} - \eta\Delta W_{\alpha\beta}^{(l,l+\Delta l)}) \mu_{\beta}^{(l+\Delta l)}, \\ \Rightarrow \quad & \Delta a_{\alpha}^{(l)} = \Delta\tilde{a}_{\alpha}^{(l)} - \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} \Delta W_{\alpha\beta}^{(l,l+\Delta l)} \mu_{\beta}^{(l+\Delta l)}, \end{aligned} \tag{D.25}$$

gives the update rules of Section 3.3:

$$\begin{aligned} \Delta W_{\alpha\beta}^{(l,l+1)} &= \sum_{\langle ij\rangle} \Delta\mathbb{E}\left[(s_{i,\alpha}^{(l)} - \mu_{\alpha}^{(l)})(s_{j,\beta}^{(l+1)} - \mu_{\beta}^{(l+1)}) \right], \\ \Delta a_{\alpha}^{(l)} &= \sum_{i=1}^{N^{(l)}} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)} \right] - \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} \Delta W_{\alpha\beta}^{(l,l+\Delta l)} \mu_{\beta}^{(l+\Delta l)}. \end{aligned} \tag{D.26}$$

D.3 Derivation of the moment closure approximation made by dynamic Boltzmann distributions

This clarifies the chain rule used to derive the key result of Section 4.2. For any observable $\langle X \rangle^{(m)}$, we have:

$$\begin{aligned}
\frac{d\langle X\rangle^{(m)}}{dt} &= \sum_{l=0}^{L-1} \sum_{\alpha \in \mathcal{R}^{(l)}} \frac{\partial \langle X\rangle^{(m)}}{\partial a_\alpha^{(l)}} \frac{\partial a_\alpha^{(l)}}{\partial t} + \sum_{l=0}^{L-2} \sum_{\alpha \in \mathcal{R}^{(l)}} \sum_{\beta \in \mathcal{R}^{(l+1)}} \frac{\partial \langle X\rangle^{(m)}}{\partial W_{\alpha\beta}^{(l,l+1)}} \frac{\partial W_{\alpha\beta}^{(l,l+1)}}{\partial t} \\
&= \sum_{l=0}^{L-1} \sum_{\alpha \in \mathcal{R}^{(l)}} \sum_{i=1}^{N^{(l)}} \mathrm{Cov}\left(X, s_{i,\alpha}^{(l)}\right) F_{a_\alpha}^{(l)} \\
&\quad + \sum_{l=0}^{L-2} \sum_{\alpha \in \mathcal{R}^{(l)}} \sum_{\beta \in \mathcal{R}^{(l+1)}} \sum_{\langle ij \rangle} \mathrm{Cov}\left(X, s_{i,\alpha}^{(l)} s_{j,\beta}^{(l+1)}\right) F_{W_{\alpha\beta}}^{(l,l+1)},
\end{aligned} \tag{D.27}$$

where $\mathrm{Cov}(X,Y) = \langle XY \rangle^{(m)} - \langle X \rangle^{(m)} \langle Y \rangle^{(m)}$.

D.4 Derivation of the centered gradient for dynamic Boltzmann distributions

This section derives the centered gradient [41] in the dynamic DBM formalism of Section 4.3.

D.4.1 Transforming the reduced model to the centered parameters

Taking a derivative in time of the parameter transformations (D.20) gives:

$$\begin{aligned}\frac{d}{dt}\tilde{W}_{\alpha\beta}^{(l,l+1)} =& F_{W_{\alpha\beta}}^{(l,l+1)},\\ \frac{d}{dt}\tilde{a}_{\alpha}^{(l)} =& F_{a_\alpha}^{(l)} + \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} \left(F_{W_{\alpha\beta}}^{(l,l+\Delta l)} \mu_\beta^{(l+\Delta l)} + W_{\alpha\beta}^{(l,l+\Delta l)} \frac{d\mu_\beta^{(l+\Delta l)}}{dt} \right).\end{aligned} \tag{D.28}$$

Note that we should not explicitly use the fact that we later substitute the centers for:

$$\mu_\alpha^{(l)} = \frac{1}{N^{(l)}} \sum_i \left\langle s_{i,\alpha}^{(l)} \right\rangle^{(d)}. \tag{D.29}$$

For now, we just consider them as arbitrary parameters.

D.4.2 Derivation of the adjoint system

Introduce adjoint variables $\phi_\alpha^{(l)}, \Lambda_{\alpha\beta}^{(l)}$ to $\tilde{a}_\alpha, \tilde{W}_{\alpha\beta}$. The Hamiltonian system is:

$$\begin{aligned}H =& -\mathcal{D}_{\mathcal{KL}} + \sum_{m=0}^{L-2} \sum_{\zeta\in\mathcal{R}^{(m)}} \sum_{\eta\in\mathcal{R}^{(m+1)}} \Lambda_{\zeta\eta}^{(m,m+1)} F_{W_{\zeta\eta}}^{(m,m+1)} + \sum_{m=0}^{L-1} \sum_{\zeta\in\mathcal{R}^{(m)}} \phi_\zeta^{(m)} \times \Bigg(F_{a_\zeta}^{(m)}\\ &+ \sum_{\Delta m=\pm 1} q^{(m,m+\Delta m)} \sum_{\eta\in\mathcal{R}^{(m+\Delta m)}} \left(F_{W_{\zeta\eta}}^{(m,m+\Delta m)} \mu_\eta^{(m+\Delta m)} + W_{\zeta\eta}^{(m,m+\Delta m)} \frac{d\mu_\eta^{(m+\Delta m)}}{dt} \right) \Bigg),\end{aligned} \tag{D.30}$$

from which the dynamical system is given by:

$$\begin{cases} \frac{d}{dt}\tilde{W}_{\alpha\beta}^{(l,l+1)} &= \frac{\partial H}{\partial \Lambda_{\alpha\beta}^{(l,l+1)}} \\ \frac{d}{dt}\tilde{a}_{\alpha}^{(l)} &= \frac{\partial H}{\partial \phi_{\alpha}^{(l)}} \end{cases} \quad \& \quad \begin{cases} \frac{d}{dt}\Lambda_{\alpha\beta}^{(l,l+1)} &= -\frac{\partial H}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} \\ \frac{d}{dt}\phi_{\alpha}^{(l)} &= -\frac{\partial H}{\partial \tilde{a}_{\alpha}^{(l)}} \end{cases}. \tag{D.31}$$

Calculating the adjoint equations gives:

$$\begin{aligned} \frac{d}{dt}\phi_{\alpha}^{(l)} =& \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \tilde{a}_{\alpha}^{(l)}} - \sum_{m=0}^{L-2} \sum_{\zeta \in \mathcal{R}^{(m)}} \sum_{\eta \in \mathcal{R}^{(m+1)}} \Lambda_{\zeta\eta}^{(m,m+1)} \frac{\partial F_{W_{\zeta\eta}}^{(m,m+1)}}{\partial \tilde{a}_{\alpha}^{(l)}} - \sum_{m=0}^{L-1} \sum_{\zeta \in \mathcal{R}^{(m)}} \phi_{\zeta}^{(m)} \times \\ & \left(\frac{\partial F_{a_\zeta}^{(m)}}{\partial \tilde{a}_{\alpha}^{(l)}} + \sum_{\Delta m = \pm 1} q^{(m,m+\Delta m)} \sum_{\eta \in \mathcal{R}^{(m+\Delta m)}} \frac{\partial F_{W_{\zeta\eta}}^{(m,m+\Delta m)}}{\partial \tilde{a}_{\alpha}^{(l)}} \mu_{\eta}^{(m+\Delta m)} \right), \\ \frac{d}{dt}\Lambda_{\alpha\beta}^{(l,l+1)} =& \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} - \sum_{m=0}^{L-2} \sum_{\zeta \in \mathcal{R}^{(m)}} \sum_{\eta \in \mathcal{R}^{(m+1)}} \Lambda_{\zeta\eta}^{(m,m+1)} \frac{\partial F_{W_{\zeta\eta}}^{(m,m+1)}}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} - \sum_{m=0}^{L-1} \sum_{\zeta \in \mathcal{R}^{(m)}} \phi_{\zeta}^{(m)} \times \\ & \left(\frac{\partial F_{a_\zeta}^{(m)}}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} + \sum_{\Delta m = \pm 1} q^{(m,m+\Delta m)} \sum_{\eta \in \mathcal{R}^{(m+\Delta m)}} \frac{\partial F_{W_{\zeta\eta}}^{(m,m+\Delta m)}}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} \mu_{\eta}^{(m+\Delta m)} \right) \\ & - \phi_{\beta}^{(l+1)} q^{(l,l+1)} \frac{d\mu_{\alpha}^{(l)}}{dt} - \phi_{\alpha}^{(l)} q^{(l,l+1)} \frac{d\mu_{\beta}^{(l+1)}}{dt}. \end{aligned} \tag{D.32}$$

The KL-divergence evaluate to:

$$\begin{aligned} \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \tilde{a}_{\alpha}^{(l)}} &= \sum_{i=1}^{N^{(l)}} \Delta \mathbb{E}\left[s_{i,\alpha}^{(l)} \right], \\ \frac{\partial \mathcal{D}_{\mathcal{KL}}}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} &= \sum_{\langle ij \rangle} \Delta \mathbb{E}\left[(s_{i,\alpha}^{(l)} - \mu_{\alpha}^{(l)})(s_{j,\beta}^{(l+1)} - \mu_{\beta}^{(l+1)}) \right]. \end{aligned} \tag{D.33}$$

We can simplify the remaining derivative terms as follows: for any F:

$$\begin{aligned}\frac{\partial F}{\partial \tilde{a}_\alpha^{(l)}} &= \sum_{\theta\in\boldsymbol{\theta}} \frac{\partial F}{\partial \theta}\frac{\partial \theta}{\partial \tilde{a}_\alpha^{(l)}},\\ \frac{\partial F}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} &= \sum_{\theta\in\boldsymbol{\theta}} \frac{\partial F}{\partial \theta}\frac{\partial \theta}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}},\end{aligned} \tag{D.34}$$

where we leave implicit that $\partial F/\partial\theta = 0$ if $\theta \notin \boldsymbol{\theta}$, i.e. if θ is not in the domain of F.

To evaluate $\partial\theta/\partial\tilde{\theta}$: first rearrange (D.20) to find the reverse transformations:

$$\begin{aligned}W_{\alpha\beta}^{(l,l+1)} &= \tilde{W}_{\alpha\beta}^{(l,l+1)},\\ a_\alpha^{(l)} &= \tilde{a}_\alpha^{(l)} - \sum_{\Delta l=\pm 1} q^{(l,l+\Delta l)} \sum_{\beta\in\mathcal{R}^{(l+\Delta l)}} \tilde{W}_{\alpha\beta}^{(l,l+\Delta l)}\mu_\beta^{(l+\Delta l)},\end{aligned} \tag{D.35}$$

and differentiate to find:

$$\begin{aligned}\frac{\partial W_{\zeta\eta}^{(m,m+1)}}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} &= \delta_{\zeta,\alpha}\delta_{\eta,\beta}\delta_{m,l},\\ \frac{\partial W_{\zeta\eta}^{(m,m+1)}}{\partial \tilde{a}_\alpha^{(l)}} &= 0,\\ \frac{\partial a_\zeta^{(m)}}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} &= -\delta_{m,l+1}\delta_{\zeta,\beta}q^{(l,l+1)}\mu_\alpha^{(l)} - \delta_{m,l}\delta_{\zeta,\alpha}q^{(l,l+1)}\mu_\beta^{(l+1)},\\ \frac{\partial a_\zeta^{(m)}}{\partial \tilde{a}_\alpha^{(l)}} &= \delta_{\zeta,\alpha}\delta_{m,l},\end{aligned} \tag{D.36}$$

which after substituting into (D.34) gives:

$$\begin{aligned}\frac{\partial F}{\partial \tilde{a}_\alpha^{(l)}} &= \frac{\partial F}{\partial a_\alpha^{(l)}},\\ \frac{\partial F}{\partial \tilde{W}_{\alpha\beta}^{(l,l+1)}} &= \frac{\partial F}{\partial W_{\alpha\beta}^{(l,l+1)}} - \frac{\partial F}{\partial a_\beta^{(l+1)}}q^{(l,l+1)}\mu_\alpha^{(l)} - \frac{\partial F}{\partial a_\alpha^{(l)}}q^{(l,l+1)}\mu_\beta^{(l+1)}.\end{aligned} \tag{D.37}$$

Returning to implement these simplifications in the adjoint system (D.32) gives the

result of Section 4.3:

$$\begin{aligned}
\frac{d}{dt}\phi_\alpha^{(l)} =& \sum_{i=1}^{N^{(l)}} \Delta\mathbb{E}\left[s_{i,\alpha}^{(l)}\right] - \psi_{a_\alpha^{(l)}}, \\
\frac{d}{dt}\Lambda_{\alpha\beta}^{(l,l+1)} =& \sum_{\langle ij\rangle} \Delta\mathbb{E}\left[(s_{i,\alpha}^{(l)} - \mu_\alpha^{(l)})(s_{j,\beta}^{(l+1)} - \mu_\beta^{(l+1)})\right] \\
& - \psi_{W_{\alpha\beta}^{(l,l+1)}} + q^{(l,l+1)}\left(\mu_\alpha^{(l)}\psi_{a_\beta^{(l+1)}} + \mu_\beta^{(l+1)}\psi_{a_\alpha^{(l)}} - \phi_\beta^{(l+1)}\frac{d\mu_\alpha^{(l)}}{dt} - \phi_\alpha^{(l)}\frac{d\mu_\beta^{(l+1)}}{dt}\right),
\end{aligned} \tag{D.38}$$

where we have defined for any parameter θ the following common derivative term:

$$\begin{aligned}
\psi_\theta = & \sum_{m=0}^{L-2}\sum_{\zeta\in\mathcal{R}^{(m)}}\sum_{\eta\in\mathcal{R}^{(m+1)}} \Lambda_{\zeta\eta}^{(m,m+1)}\frac{\partial F_{W_{\zeta\eta}}^{(m,m+1)}}{\partial\theta} + \sum_{m=0}^{L-1}\sum_{\zeta\in\mathcal{R}^{(m)}}\phi_\zeta^{(m)}\times \\
& \left(\frac{\partial F_{a_\zeta}^{(m)}}{\partial\theta} + \sum_{\Delta m=\pm 1} q^{(m,m+\Delta m)}\sum_{\eta\in\mathcal{R}^{(m+\Delta m)}}\frac{\partial F_{W_{\eta\zeta}}^{(m,m+\Delta m)}}{\partial\theta}\mu_\eta^{(m+\Delta m)}\right).
\end{aligned} \tag{D.39}$$

D.4.3 Derivation of the sensitivity equation

The sensitivity (update) equations can be obtained from the Lagrangian:

$$\begin{aligned}
\mathcal{L} =& \mathcal{D}_{\mathcal{K}\mathcal{L}} + \sum_{m=0}^{L-2}\sum_{\zeta\in\mathcal{R}^{(m)}}\sum_{\eta\in\mathcal{R}^{(m+1)}}\Lambda_{\zeta\eta}^{(m,m+1)}F_{W_{\zeta\eta}}^{(m,m+1)} + \sum_{m=0}^{L-1}\sum_{\zeta\in\mathcal{R}^{(m)}}\phi_\zeta^{(m)}\times\left(F_{a_\zeta}^{(m)}\right. \\
& \left. + \sum_{\Delta m=\pm 1} q^{(m,m+\Delta m)}\sum_{\eta\in\mathcal{R}^{(m+\Delta m)}}\left(F_{W_{\zeta\eta}}^{(m,m+\Delta m)}\mu_\eta^{(m+\Delta m)} + W_{\zeta\eta}^{(m,m+\Delta m)}\frac{d\mu_\eta^{(m+\Delta m)}}{dt}\right)\right),
\end{aligned} \tag{D.40}$$

from which the objective function is given by the usual relation $S = \int_0^T dt\,\mathcal{L}$. By differentiating with respect to the parameter vector $\boldsymbol{u}$ appearing in the differential equations F we

obtain the result of Section 4.3:

$$\begin{aligned}
\frac{dS}{d\boldsymbol{u}_{a_\alpha}^{(l)}} &= -\int_0^T dt\ \phi_\alpha^{(l)} \frac{\partial F_{a_\alpha}^{(l)}}{\partial \boldsymbol{u}_{a_\alpha}^{(l)}}, \\
\frac{dS}{d\boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}} &= -\int_0^T dt\ \left(\Lambda_{\alpha\beta}^{(l,l+1)} + q^{(l,l+1)}\phi_\alpha^{(l)}\mu_\beta^{(l+1)} + q^{(l,l+1)}\phi_\beta^{(l+1)}\mu_\alpha^{(l)}\right) \frac{\partial F_{W_{\alpha\beta}}^{(l,l+1)}}{\partial \boldsymbol{u}_{W_{\alpha\beta}}^{(l,l+1)}}.
\end{aligned} \tag{D.41}$$

In practice, we restrict the limits of integration to a window $[\tau, \tau + \Delta\tau]$, where we slide τ every few optimization steps to cover the full range $[0, T]$.

Appendix E

Physics-based machine learning for modeling stochastic IP3-dependent calcium dynamics

E.1 IP3 dependent calcium oscillations

E.1.1 Stochastic models

Figure 2 of the main textshows a schematic of the model of IP_3 dependent calcium oscillations in astrocytes. Clusters of IP_3 receptors (IP_3Rs) in the membrane of the endoplasmic reticulum are activated by cytosolic calcium and IP_3, allowing transport through the channel into the cytoplasm. The channel model for a single IP_3 receptor subunit is shown in Figure E.1. While the receptor is known to be composed of four subunits, the peak conductance is observed when only three are open. Hence, the original model of Ref. [102] considers only three subunits. The reactions in the receptor subunit

are:

$$
\begin{aligned}
S_{1k0} + \mathrm{Ca}_{\mathrm{Cyt}} &\underset{\beta_2}{\overset{\alpha_2}{\rightleftharpoons}} S_{1k1}, \\
S_{0k0} + \mathrm{Ca}_{\mathrm{Cyt}} &\underset{\beta_4}{\overset{\alpha_4}{\rightleftharpoons}} S_{0k1}, \\
S_{0k0} + \mathrm{IP}_3 &\underset{\beta_1}{\overset{\alpha_1}{\rightleftharpoons}} S_{1k0}, \\
S_{0k1} + \mathrm{IP}_3 &\underset{\beta_3}{\overset{\alpha_3}{\rightleftharpoons}} S_{1k1}, \\
S_{i0j} + \mathrm{Ca}_{\mathrm{Cyt}} &\underset{\beta_5}{\overset{\alpha_5}{\rightleftharpoons}} S_{i1j},
\end{aligned}
\tag{E.1}
$$

where the open state is S_{110}. Table E.1 gives the parameter values used for stochastic simulations, which are the same values used in the original Ref. [102] model. The molecular-based reaction rates are obtained from the concentration-based rates as:

$$
\begin{aligned}
\alpha_i &= \frac{a_i}{c_A \times V_{\mathrm{Cyt}}}, \\
\beta_i &= b_i,
\end{aligned}
\tag{E.2}
$$

as derived in Section E.1.3.

Transport through the channel is given by the reactions:

$$
3S_{110} + \mathrm{Ca}_{\mathrm{ER}} \underset{\gamma_b}{\overset{\gamma_f}{\rightleftharpoons}} 3S_{110} + \mathrm{Ca}_{\mathrm{Cyt}}, \tag{E.3}
$$

as also derived in Section E.1.3 from the differential equation model by Ref. [102].

The recovery of calcium from the cytoplasm is attributed to ATP-driven pumps such as SERCA pumps. In this model, since the density of these pumps is as high as $1000/\mu\mathrm{m}^2$ [3] and the ER is a highly folded structure with a large surface area, this process

is not modeled using stochastic particle-based methods, but rather by differential equations:

$$\begin{aligned}\frac{d[\mathrm{Ca_{Cyt}}]}{dt} &= J_1 - J_2,\\ J_1 &= c_1 v_2 (c_1^{-1}[\mathrm{Ca_{ER}}] - [\mathrm{Ca_{Cyt}}]),\\ J_2 &= \frac{v_3[\mathrm{Ca_{Cyt}}]^2}{[\mathrm{Ca_{Cyt}}]^2 + k_3^2},\end{aligned} \tag{E.4}$$

where J_1 is a leak current, and J_2 is the ATP-driven recovery of calcium back to the ER. See also Section E.1.3.

Examples of the stochastic simulations are shown in Figure E.2. The initial number of Ca^{2+} and IP_3 are sampled from the Gaussian distributions:

$$\begin{aligned}[\mathrm{Ca_{Cyt}}]_0 &= \mathcal{N}(\mu_0([\mathrm{Ca_{Cyt}}]), \sigma_0^2([\mathrm{Ca_{Cyt}}])),\\ [\mathrm{IP_3}]_0 &= \mathcal{N}(\mu_0([\mathrm{IP_3}]), \sigma_0^2([\mathrm{IP_3}])),\end{aligned} \tag{E.5}$$

where the parameter values are given in Table E.1.

After sampling the initial counts, an initial simulation is used to initialize the states of the IP_3 receptors. Let the numbers of particles corresponding to the concentrations (E.5) be $n_{\mathrm{Ca_{Cyt}},0}$ and $n_{\mathrm{IP_3},0}$, from which the number of calcium particles in the ER $n_{\mathrm{Ca_{ER}},0}$ can be calculated using c_0, c_1. The IP_3R are initialized to the specified and fixed number $n_{\mathrm{IP_3R}}$ of receptors, all in state S_{000}, and all other receptor states S_{ijk} with population zero. The initial simulation is run with only the IP_3R state reaction system (E.1), and where the number of $\mathrm{Ca_{Cyt}}, \mathrm{IP_3}, \mathrm{Ca_{ER}}$ is conserved, i.e. fixed to their initial values. The duration of the initial simulation is 10 s. From this, the initial states of the receptors are taken for the main simulation as the average state values over the last 4 s of the initial simulation.

The main simulations are run from $t = 0$ to $t = T_{\max}$, with the count of each species written out at intervals $\Delta t^{(\mathrm{write})}$. The currents from the differential equations (E.4) are updated at short time intervals $\Delta t^{(\mathrm{diff.\ eq.})}$, with all parameters as given in Table E.1.

Table E.1: Parameter values used for stochastic simulations.

Parameter	Value	Description
c_0	2 μM	Total [Ca] in terms of cytosolic volume
c_1	0.185	Ratio ER volume to cytosol volume
v_1	6 s^{-1}	Max Ca channel flux
v_2	0.11 s^{-1}	Ca leak flux constant
v_3	0.9 (μM$\times$ s)$^{-1}$	Max Ca uptake
k_3	0.1 μM	Activation constant for ATP-Ca pump
a_1	400 (μM$\times$ s)$^{-1}$	IP_3R reaction rate
a_2	0.2 (μM$\times$ s)$^{-1}$	IP_3R reaction rate
a_3	400 (μM$\times$ s)$^{-1}$	IP_3R reaction rate
a_4	0.2 (μM$\times$ s)$^{-1}$	IP_3R reaction rate
a_5	20 (μM$\times$ s)$^{-1}$	IP_3R reaction rate
d_1	0.13 μM	IP_3R reaction rate
d_2	1.049 μM	IP_3R reaction rate
d_3	943.4 $\times 10^{-3}$ μM	IP_3R reaction rate
d_4	144.5 $\times 10^{-3}$ μM	IP_3R reaction rate
d_5	82.34 $\times 10^{-3}$ μM	IP_3R reaction rate
$\mu_0([Ca_{Cyt}])$	0.25 μM	Initial mean Ca concentration
$\mu_0([IP_3])$	Varying	Initial mean IP_3 concentration
$\sigma_0([Ca_{Cyt}])$	10^{-3} μM	Initial standard deviation of Ca concentration
$\sigma_0([IP_3])$	10^{-3} μM	Initial standard deviation of IP_3 concentration
V_{Cyt}	10^{-12} or 10^{-14} L	Cytoplasm volume
$\Delta t^{(\text{write})}$	0.1 s	Writing interval
$\Delta t^{(\text{diff. eq.})}$	0.001 s	Integration step length for currents
$T_{\max}$	50 s	Maximum simulation time

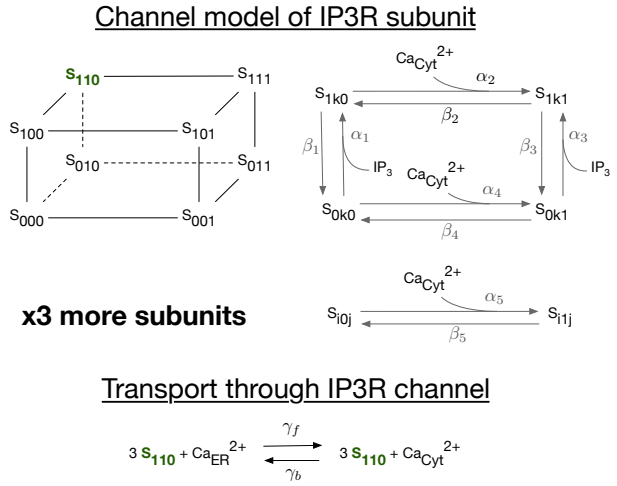

Figure E.1: Channel models for the IP_3R and transport through the channel, which is open when one calcium ion and one IP_3 are bound. A further calcium binding inhibits the channel. Peak conductance is observed when three subunits are open; a fourth that is physically observed is not modelled. The open state is S_{110}, indicated in green.

E.1.2 Range of oscillations

Figure E.1 shows the original bifurcation diagram by Ref. [102], and the range of oscillations in the stochastic model under consideration.

The stochastic curves should not be interpreted as a bifurcation diagram. Rather, at each concentration of IP_3:

- At each timepoint, average over stochastic simulations to obtain the mean $\mu(t)$ and standard deviation $\sigma(t)$.
- Calculate the upper and lower curves over time $c_\pm(t) = \mu(t) \pm \sigma(t)$.
- Take the min. and max.:

$$\begin{aligned} c_{-,\min} &= \min_t(c_-(t)) \\ c_{+,\max} &= \max_t(c_+(t)) \end{aligned} \tag{E.6}$$

over the last 40s of each of the 100s simulations.

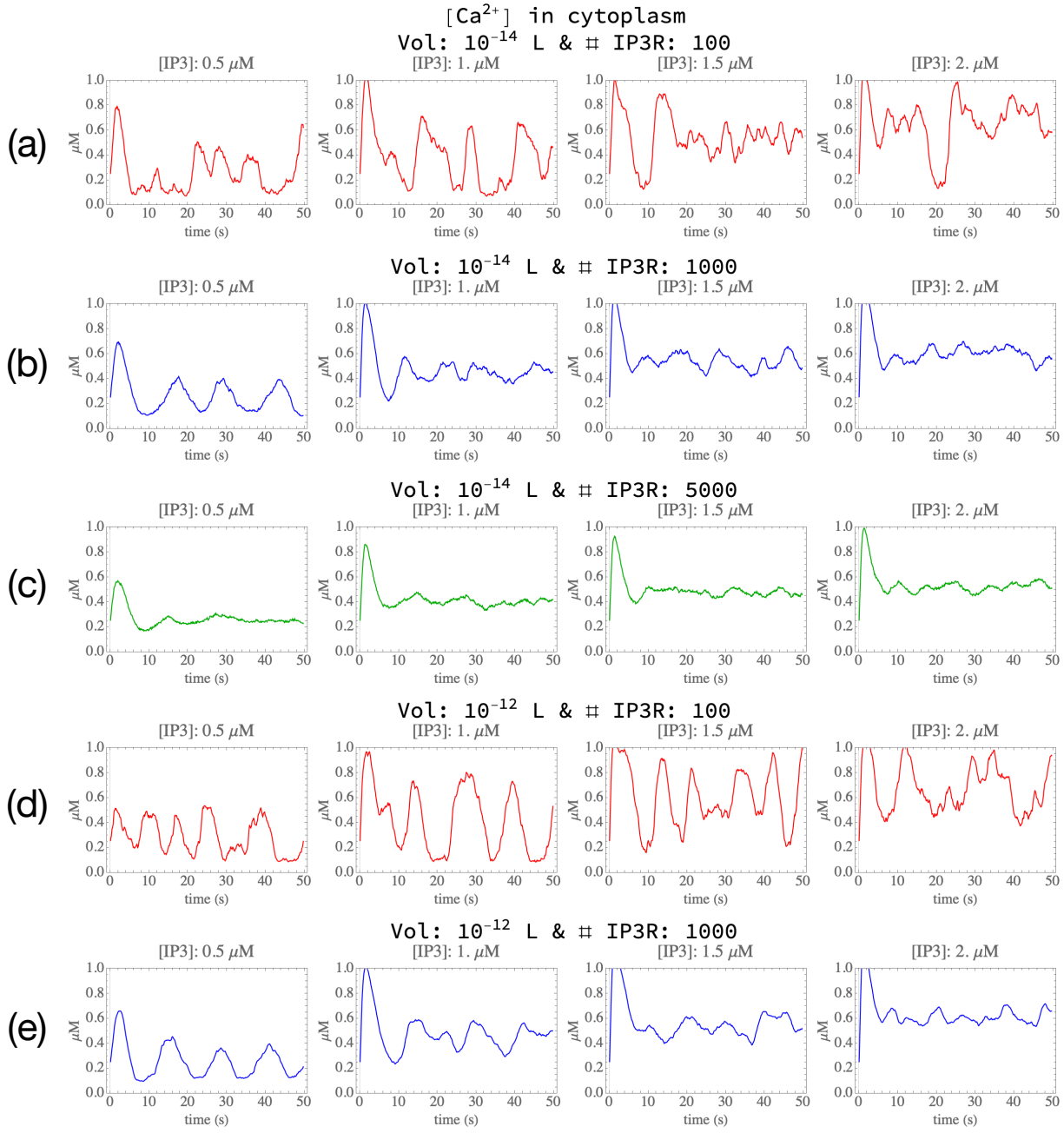

Figure E.2: Stochastic simulations of cytosolic calcium oscillations for (a) 100 IP_3R and (b) 1000 IP_3R at various IP_3 concentrations and cytoplasm volume of 10^{-14}L. Calcium spikes are observed at all concentrations, and are more pronounced at lower receptor number.

The resulting $c_{-,\min}, c_{+,\max}$ are plotted in Figure E.1 to indicate the range of oscillations. Error bars indicate 95% confidence levels.

E.1.3 Derivation of reaction model from differential equations

The original differential equations in Ref. [102] are:

$$\frac{d[\mathrm{Ca}^{2+}_{\mathrm{Cyt}}]_{\mathrm{Cyt}}}{dt} = J_1 - J_2 + J_3, \tag{E.7}$$

where

$$\begin{aligned} J_1 &= c_1 v_2 (c_1^{-1}[\mathrm{Ca}^{2+}_{\mathrm{ER}}]_{\mathrm{Cyt}} - [\mathrm{Ca}^{2+}_{\mathrm{Cyt}}]_{\mathrm{Cyt}}), \\ J_2 &= \frac{v_3 [\mathrm{Ca}^{2+}_{\mathrm{Cyt}}]^2_{\mathrm{Cyt}}}{[\mathrm{Ca}^{2+}_{\mathrm{Cyt}}]^2_{\mathrm{Cyt}} + k_3^2}, \\ J_3 &= c_1 v_1 x_{110}^3 (c_1^{-1}[\mathrm{Ca}^{2+}_{\mathrm{ER}}]_{\mathrm{Cyt}} - [\mathrm{Ca}^{2+}_{\mathrm{Cyt}}]_{\mathrm{Cyt}}), \end{aligned} \tag{E.8}$$

where x_{110} is the fraction of subunits in the open state S_{110}, and c_1, v_1, v_2, v_3, k_3 are constants given in Table E.1. The current J_1 is a leak current, J_2 is the flux out of cytoplasm due to an ATP-dependent Ca^{2+} pump, e.g. a SERCA pump, and J_3 is the transport of Ca^{2+} into cytoplasm through the open IP_3R. The notation $[X_{\mathrm{ER}}]_{\mathrm{Cyt}}$ is used to denote the number of particles of species X located in the ER, divided by the volume of Cyt to obtain a concentration (as opposed to the volume of the ER). This conversion is used to simplify the calculations by having to keep track of only a single volume.

The currents J_1, J_2 are kept as differential equations, while the transport J_3 is converted to an equivalent reaction system. For this transformation, *concentration-based* reaction rates must be transformed into *molecular-based* reaction rates.

Consider a general reaction of the form:

$$\sum_{i=1}^{R} m_i X_i \underset{\gamma}{\rightarrow} \dots. \tag{E.9}$$

Associated with this reaction is the stoichiometry vector $\boldsymbol{\nu}$ of length R, whose components ν_i describe the change in the number of particles of X_i. Here, γ will be referred to as the molecular-based reaction rate. The goal is to relate γ to the concentration-based reaction rate k appearing in mass action kinetics:

$$\frac{d[X_j]}{dt} = -\nu_j k \prod_{i=1}^{R} \left([X_i]\right)^{m_i} + \dots, \tag{E.10}$$

where the $\dots$ denote other possible reactions.

The propensity term for the reaction (E.9) in units of molecules per time is:

$$\rho = \gamma \prod_{i=1}^{R} \left(n_{X_i}\right)_{m_i}, \tag{E.11}$$

where n_{X_i} is the number of particles of species X_i and $(x)_n$ is the falling factorial. Note that from the binomial factor representing all possible combinations of particles, a factor $\prod_{i=1}^{R} m_i!$ has been absorbed into γ. At large particle numbers, this is commonly approximated by

$$\rho \approx \gamma \prod_{i=1}^{R} n_{X_i}^{m_i}. \tag{E.12}$$

In the mass action equation (E.10), the reaction-based rate of change in units of concentration per time is (without the stoichiometry vector):

$$k \prod_{i=1}^{R} \left([X_i]\right)^{m_i}. \tag{E.13}$$

Substitute the definition of the concentration $[X_i] = n_{X_i}/(c_A \times V)$ for n_{X_i} particles in

volume V where c_A is Avogadros constant, and convert to units of molecules per time by multiplying by $c_A \times V$:

$$kc_A V \prod_{i=1}^{R} \left(\frac{n_{X_i}}{c_A V} \right)^{m_i}. \tag{E.14}$$

Equating this with the approximation for the propensity (E.12) gives the relation:

$$\gamma = k(c_A V)^{1-\sum_{i=1}^{R} m_i}. \tag{E.15}$$

Using this relation, the transport current J_3 can be transformed by writing it in the equivalent form:

$$\begin{aligned} J_3 =& k_{1f}[S_{110}]^3_{\mathrm{Cyt}}[\mathrm{Ca}^{2+}_{\mathrm{ER}}]_{\mathrm{Cyt}} - k_{1b}[S_{110}]^3_{\mathrm{Cyt}}[\mathrm{Ca}^{2+}_{\mathrm{Cyt}}]_{\mathrm{Cyt}}, \\ k_{1f} =& v_1[\mathrm{IP_3R}]^{-3}_{\mathrm{Cyt}}, \\ k_{1b} =& c_1 v_1[\mathrm{IP_3R}]^{-3}_{\mathrm{Cyt}}. \end{aligned} \tag{E.16}$$

where S_{110} is the open state of the $\mathrm{IP_3R}$ in Figure E.1. From this the equivalent reaction can be identified:

$$3S_{110} + \mathrm{Ca}^{2+}_{\mathrm{ER}} \underset{\gamma_{1b}}{\overset{\gamma_{1f}}{\rightleftharpoons}} 3S_{110} + \mathrm{Ca}^{2+}_{\mathrm{Cyt}}, \tag{E.17}$$

where the molecular-based rates are given by (E.15):

$$\begin{aligned} \gamma_{1f} &= v_1 n^{-3}_{\mathrm{IP3R}}, \\ \gamma_{1b} &= c_1 v_1 n^{-3}_{\mathrm{IP3R}}, \end{aligned} \tag{E.18}$$

where n_{IP3R} is the number of $\mathrm{IP_3R}$.

E.1.4 Number of IP3 receptor subunits

The number of IP_3 receptors (IP_3R) depends on the density of channels and the surface area of the ER. The ER is a highly folded structure, and as such its surface area can vary significantly. In the text a large spread in the number of channels is explored. Here, an order of magnitude estimation is provided to justify their scale.

Starting with the volume of the cytoplasm V_{cyt}, the volume of the ER is $V_{\text{ER}} = c_1 \times V_{\text{cyt}}$ where $c_1 = 0.185$ is the ratio estimated in Ref. [102]. The ER has the smallest surface area if it is a sphere:

$$\text{SA}_{\text{ER}}^{\text{min}} = 4\pi \left(\frac{3}{4\pi} V_{\text{ER}} \right)^{2/3} \tag{E.19}$$

Let the actual surface area be some factor λ larger than the minimum:

$$\text{SA}_{\text{ER}} = \lambda \times \text{SA}_{\text{ER}}^{\text{min}} \tag{E.20}$$

IP_3R clusters are spread out over the ER with spacing $1 - 7\mu$m [105]. Assuming the IP3R clusters were are located in a grid with spacing Δx_{IP3R} gives:

$$n_{\text{IP3R clusters}} = \frac{\text{SA}_{\text{ER}}}{\Delta x^2_{\text{IP3R clusters}}} \tag{E.21}$$

Each cluster contains up to 15 channels [105]. Assuming 10 channels per cluster and with 4 subunits per channel gives:

$$n_{\text{IP3R subunits}} = 10 \times 4 \times n_{\text{IP3R clusters}} = \frac{160\pi\lambda}{\Delta x^2_{\text{IP3R clusters}}} \left(\frac{3}{4\pi} c_1 V_{\text{cyt}} \right)^{2/3} \tag{E.22}$$

Figure E.3 shows the number of IP_3R subunits for a range of spacings and surface area factors. For a highly folded ER with large λ and average cluster spacing $\Delta x_{\text{IP3R clusters}} \sim$

3μm, the estimates of $\mathcal{O}(100)$ subunits for $\text{Vol}_{\text{cyt}} = 10^{-14}L$ and $\mathcal{O}(1000)$ subunits for $\text{Vol}_{\text{cyt}} = 10^{-12}L$ are reasonable, which are the approximate magnitudes explored in the text.

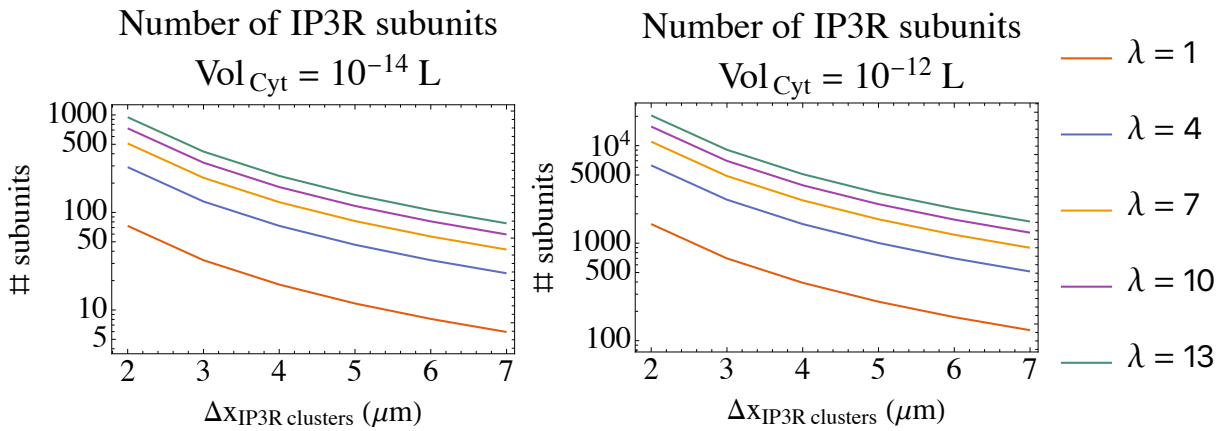

Figure E.3: Number of IP3R subunits given by (E.22) as a function of the cluster spacing $\Delta x_{\text{IP3R clusters}}$ in μm, and the dimensionless surface area factor λ, where $\lambda = 1$ corresponds to a sphere.

E.2 Training ML models

E.2.1 Data transformation

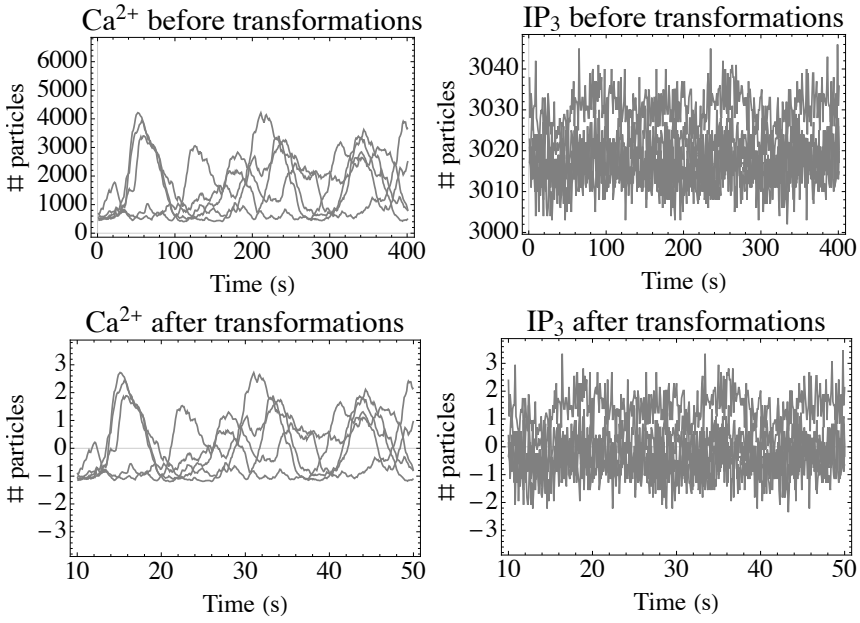

Figure E.4: Example standardizing transformation for the system studied in Figure 3 of the main textat $[\mathrm{IP}_3] = 0.5\mu$M. Five stochastic simulation trajectories are shown in gray. *Top row*: Number of particles of calcium and IP_3 before transformation. *Bottom row*: After transformation, oscillations occur around zero.

Let the stochastic simulation data be represented by the matrix $X(t)$ of size $M \times N_v$ where N_v is the dimension of the visible variables and M is the number of samples:

$$X(t) = \begin{pmatrix} \boldsymbol{x}_1^\intercal(t) \\ \boldsymbol{x}_2^\intercal(t) \\ \cdots \\ \boldsymbol{x}_M^\intercal(t) \end{pmatrix} \tag{E.23}$$

PCA applied to $X(t)$ leads to the parameters:

$$\hat{\boldsymbol{\theta}}_X(t) = \{\hat{\boldsymbol{b}}_X, \hat{W}_X, \sigma_X^2\}(t) \tag{E.24}$$

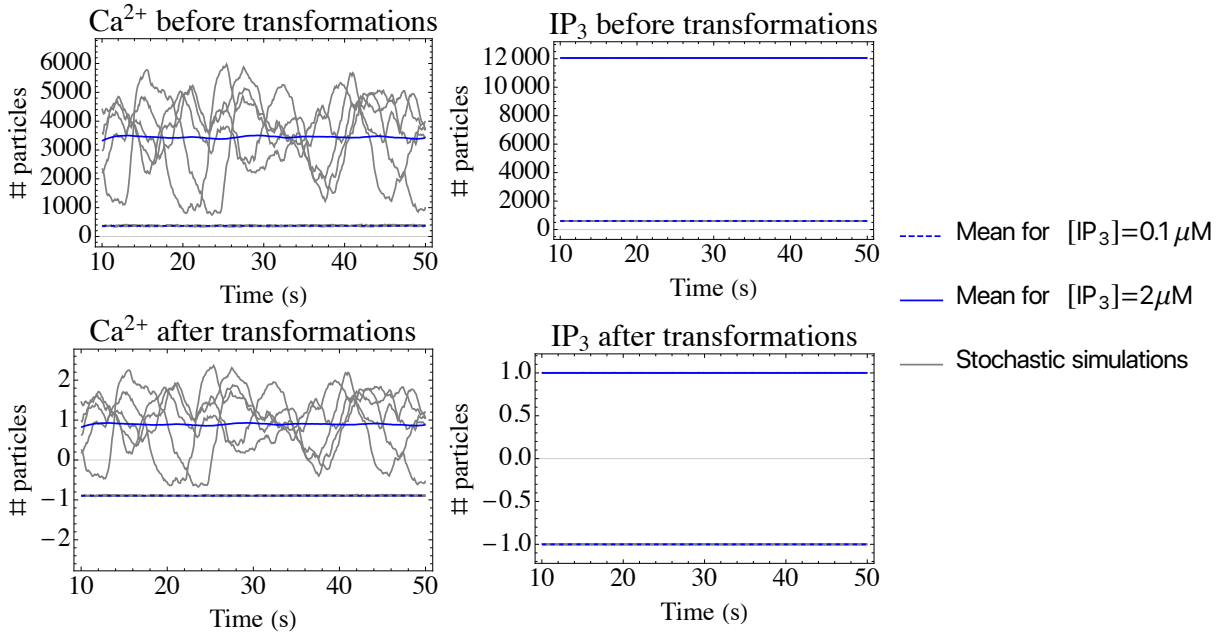

Figure E.5: Transformations for the IP_3 system studied in Figure 3 of the main text. The transformation is calculated for stochastic simulations at the upper and lower range of oscillations considered, i.e. $[IP_3] = 0.1\mu M$ and $[IP_3] = 2\mu M$.

From these parameters, approximations $\hat{\boldsymbol{F}}^{(\text{approx.})}_{\text{rxn.}}$ to $\hat{\boldsymbol{F}}$ under different reaction processes can be calculated.

When using the model to integrate the parameters $\hat{\boldsymbol{\theta}}_X(t)$, it traces out a trajectory in $\hat{D} = N_v + N_v \times N_h + 1$ dimensional space. In order for this integration to be stable, the inputs to the neural network $\hat{\boldsymbol{F}}^{(\text{approx.})}_{\text{rxn.}}$ must not be sensitive to small perturbations in $\hat{\boldsymbol{\theta}}_X(t)$. Note that the error in $\hat{\boldsymbol{\theta}}_X(t)$ is set by the error in the output of the neural network, and the integration drift that arises from integrating a noisy signal.

To ensure that the Jacobian $\partial \hat{\boldsymbol{F}}^{(\text{approx.})}_{\text{rxn.}} / \partial \hat{\boldsymbol{\theta}}_X(t)$ is small, introduce the following transformation. For a given trajectory $\boldsymbol{x}_i(t), i = 1, \ldots, M$, compute the mean and variance

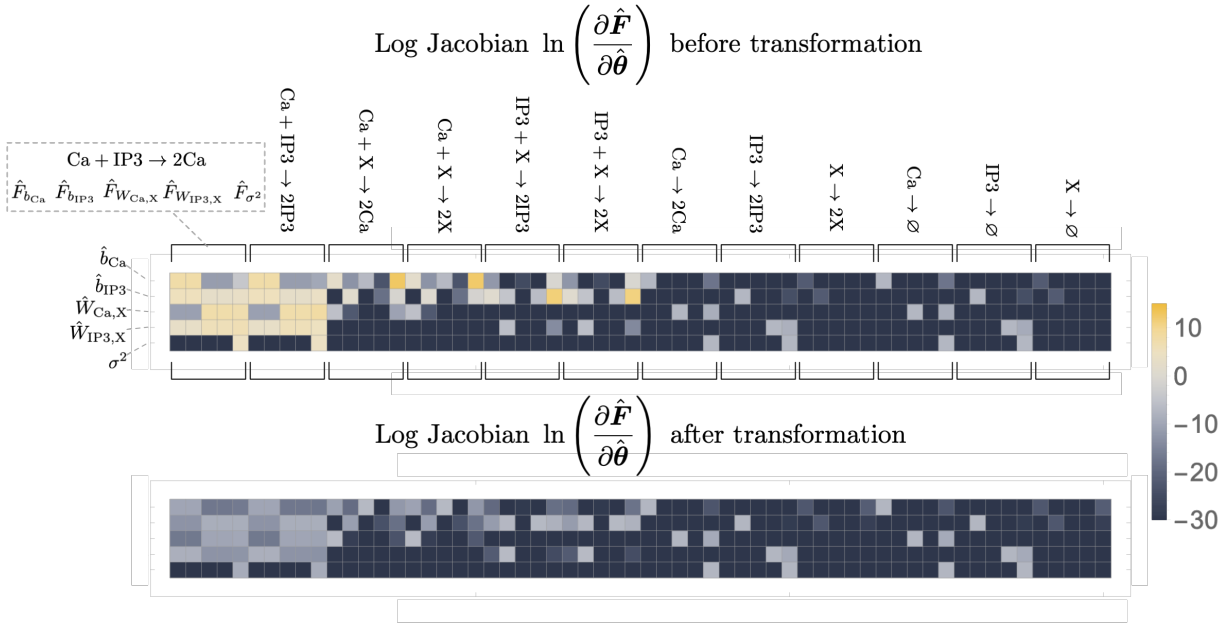

Figure E.6: Jacobian of reaction candidates with respect to standard parameters, plotted on a log scale. The reactions are those of the model studied in Figure 3 of the main text, and parameters $\hat{\boldsymbol{\theta}}$ are those at a single point in time $t = 50$ and $[\mathrm{IP}_3] = 0.7\mu\mathrm{M}$, with $\mu_h = 0$ and $\Sigma_h = I$. *Top:* Jacobian before the transformation (E.26). *Bottom:* Jacobian after the transformation. The transformation reduces the Jacobian, increasing the stability of the integration because small perturbations do not significantly alter the inputs to the neural network.

over time and samples:

$$\begin{aligned} \boldsymbol{m} &= \frac{1}{T}\sum_{t=1}^{T}\frac{1}{M}\sum_{i=1}^{M}\boldsymbol{x}_i(t), \\ \boldsymbol{v} &= \frac{1}{T}\sum_{t=1}^{T}\frac{1}{M}\sum_{i=1}^{M}[\boldsymbol{x}_i(t)-\boldsymbol{m}]^2, \end{aligned} \tag{E.25}$$

and use these parameters to define a transformation:

$$\boldsymbol{y}_i(t) = \frac{\boldsymbol{x}_i(t)-\boldsymbol{m}}{\sqrt{\boldsymbol{v}}}. \tag{E.26}$$

This leads to a new matrix $Y(t)$ of equal size $M \times N_v$. PCA applied to $Y(t)$ leads to a different set of parameters:

$$\hat{\boldsymbol{\theta}}_Y(t) = \{\hat{\boldsymbol{b}}_Y, \hat{W}_Y, \sigma_Y^2\}(t) \tag{E.27}$$

Figure E.4 shows how such a transformation standardizes oscillations. Figure E.5 shows the transformation for the system studied in Figure 3 of the main text. Here the averages in (E.25) are taken over IP_3 concentrations at the boundaries of the bifurcation diagram studied, i.e. at $[IP_3] = 0.1\mu M$ and $[IP_3] = 2\mu M$.

It is difficult to relate $\boldsymbol{\theta}_Y(t)$ to $\boldsymbol{\theta}_X(t)$ since it relates the eigendecomposition of a matrix product. Importantly, however, Figure E.6 shows that the Jacobian $\partial\hat{\boldsymbol{F}}_{\text{rxn.}}^{(\text{approx.})}/\partial\hat{\boldsymbol{\theta}}(t)$ has decreased for the example problem studied in Figure 3 of the main text.

E.2.2 Training inputs and targets

After transforming the data $X \to Y$, the ML parameters are identified from $Y(t)$ for $t = 1, \ldots, T$ using Equation (9) of the main text. For the models considered in this work in Figure 3 of the main textand Figure 4 of the main text, the data used are in the range 10s to 50s of the stochastic simulations, which are of length 50s. The first 10s are discarded to lessen the dependence of the oscillations on the chosen initial condition in

Table E.1. The parameters obtained are the standard parameters $\hat{\boldsymbol{\theta}}(t)$. Note that the sign of the eigenvectors is adjusted to be consistent by ensuring that for any eigenvector $\boldsymbol{u}$ we have $|\cos^{-1}(\boldsymbol{u}^\intercal \mathbf{1})| \leq \pi/2$.

The targets derivatives are calculated from $\hat{\boldsymbol{\theta}}(t)$ using total variation regularization (TVR) [104]. For a time series $\boldsymbol{z}$ with elements $z_1, z_2, \ldots, z_T$, the time derivative $\dot{\boldsymbol{z}}$ is obtained by solving the optimization problem:

$$\dot{\boldsymbol{z}} = \min_{\boldsymbol{u}} \left(\alpha ||\dot{\boldsymbol{u}}||_1 + \frac{1}{2} ||A\boldsymbol{u} - \boldsymbol{z}||_2 \right) \tag{E.28}$$

where A is the anti-differentiation matrix, and α is a regularization parameter. In this case, the A matrix is that of the Euler method. The optimization problem is solved using the lagged diffusivity method [104] for 10 optimization steps for every parameter $\boldsymbol{z}$ in $\hat{\boldsymbol{\theta}}$ with regularization parameter $\alpha = 100$. Additionally, after calculating the derivatives $d\hat{\boldsymbol{\theta}}/dt$, small derivative values with an absolute value below 10^{-5} are set to zero. The target outputs $d\hat{\boldsymbol{\theta}}/dt$ are therefore obtained:

$$\text{Target}(t) = \frac{d\hat{\boldsymbol{\theta}}(t)}{dt} = \begin{pmatrix} d\hat{\boldsymbol{b}}/dt \\ d\hat{W}/dt \\ d\sigma^2/dt \end{pmatrix}. \tag{E.29}$$

The inputs are obtained by integrating $d\hat{\boldsymbol{\theta}}/dt$ with the anti-differentiation matrix A to obtain smoothed inputs $\hat{\boldsymbol{\theta}}^{(\text{integrated})}$:

$$\text{Input}(t) = \hat{\boldsymbol{\theta}}^{(\text{integrated})}(t) = \begin{pmatrix} \hat{\boldsymbol{b}}(t) \\ \hat{W}(t) \\ \sigma^2(t) \end{pmatrix}. \tag{E.30}$$

E.2.3 Reaction approximations

The physics of the system is described by the chemical master equation (CME). The CME can be incorporated into the inference problem by using it to derive an approximation to the time evolution of parameters. Figure E.7 shows a schematic of how this derivation as follows.

For the distribution defined by the current parameters $\boldsymbol{\theta}(t) = \{\boldsymbol{b}, W, \sigma^2, \boldsymbol{\mu}_h, \Sigma_h\}(t)$ at an instant in time, Equation (7) of the main textgives the observables $\boldsymbol{\phi}(t) = \{\boldsymbol{\mu}, C\}(t)$. Next, consider for example the predator-prey reaction $H + P \to 2H$ with rate k for predators H and prey P. Under this reaction, the observables evolve in time according to:

$$\begin{aligned} \frac{d\langle n_P \rangle}{dt} &= -k\langle n_H n_P \rangle, \\ \frac{d\langle n_H \rangle}{dt} &= k\langle n_H n_P \rangle, \end{aligned} \tag{E.31}$$

and

$$\begin{aligned} \frac{d\left\langle n_P^2 \right\rangle}{dt} &= -2k\langle n_H n_P^2 \rangle + k\langle n_H n_P \rangle, \\ \frac{d\left\langle n_H^2 \right\rangle}{dt} &= 2k\langle n_H^2 n_P \rangle + k\langle n_H n_P \rangle, \\ \frac{d\langle n_H n_P \rangle}{dt} &= -k\langle n_H^2 n_P \rangle + k\langle n_H n_P^2 \rangle - k\langle n_H n_P \rangle, \end{aligned} \tag{E.32}$$

and for $X \notin \{H, P\}$

$$\begin{aligned} \frac{d\langle n_X n_P \rangle}{dt} &= -k\langle n_X n_H n_P \rangle, \\ \frac{d\langle n_X n_H \rangle}{dt} &= k\langle n_X n_H n_P \rangle, \end{aligned} \tag{E.33}$$

which are derived from the CME. These equations form the time evolution of the observables under this reaction $d\boldsymbol{\phi}^{(H+P\to 2H)}/dt$. In the derivation of the desired approximations, the reaction rates are set to unity $k = 1$. Ultimately, if a linear model is learned instead of a

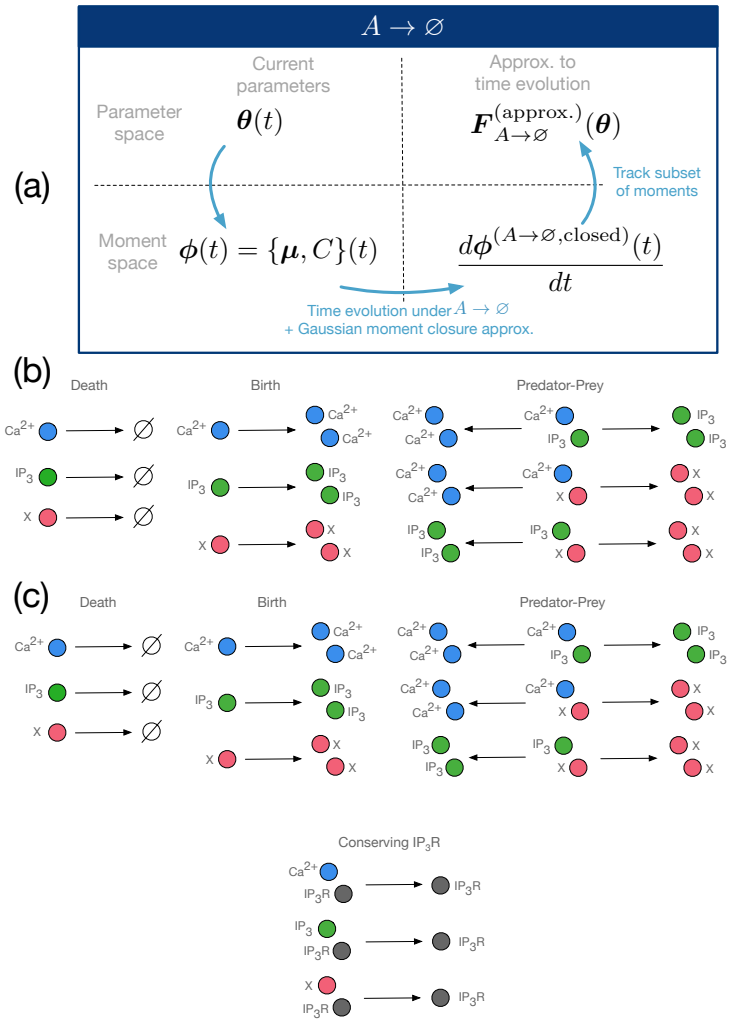

Figure E.7: (a) Schematic of how an approximation to the time evolution of parameters is derived for a single reaction pathway. The observables $\boldsymbol{\phi}(t)$ corresponding to the distribution defined by the current parameters $\boldsymbol{\theta}(t)$ at an instant in time are obtained from Equation (7) of the main text. The time evolution of each observable is calculated from the master equation, and closed using a Gaussian moment closure approximation (Equation (3) of the main text). By tracking exactly only a subset of moments $d\{\boldsymbol{\mu}_v, \Sigma_{vh}, \text{Tr}(\Sigma_v), \boldsymbol{\mu}_h, \Sigma_h\}/dt$ matching the dimensions of $\boldsymbol{\theta}(t) = \{\boldsymbol{b}, W, \sigma^2, \boldsymbol{\mu}_h, \Sigma_h\}(t)$, an approximation to the time evolution of the parameters is obtained. (b) The reactions used to construct the inputs to the sub-network models in Figure 3 of the main textgeneralizing in IP_3: death e.g. $H \to \varnothing$, birth e.g. $P \to 2P$, and a predator-prey interaction, e.g. $H + P \to 2H$. These reactions mimic the reaction scheme of Lotka-Volterra system. Each combination of $\{H, P\}$ from $\{Ca^{2+}, IP_3, X\}$ is included. (c) The reactions used in the models including IP_3R. The reactions are those of (b), including three extra reactions for IP_3R that explicitly conserve IP_3R.

neural network, the learned coefficients can be interpreted as the learned reaction rates.

These equations are not closed - higher order observables appear on the right hand side. In principle, any moment closure approximation can be used to derive an approximation, but the natural choice is to use Gaussian moment closure (Equation (3) of the main text) since the model is Gaussian. The space of candidates can also be increased by deriving approximations for more than one closure approximation. Under this approximation, for any species X, Y, Z:

$$\langle n_X n_Y n_Z \rangle \to -2\langle n_X \rangle \langle n_Y \rangle \langle n_Z \rangle + \langle n_X \rangle \langle n_Y n_Z \rangle + \langle n_Y \rangle \langle n_X n_Z \rangle + \langle n_Z \rangle \langle n_X n_Y \rangle. \tag{E.34}$$

This approximation gives the closed form for the observables $d\boldsymbol{\phi}^{(H+P\to 2H,\text{closed})}/dt$.

To convert back to the parameter frame, we note that the transformation $d\boldsymbol{\phi}/dt \to d\boldsymbol{\theta}/dt$ may not exist. Instead, we track only a certain number of observables equivalent to the number of parameters D in the model. While the choice of the observables is arbitrary, the obvious choice is those that match the dimensions of the parameters:

$$\frac{d}{dt}\left\{\boldsymbol{\mu}_v, C_{vh}, \mathrm{Tr}(C_v), \boldsymbol{\mu}_h, \Sigma_h\right\}, \tag{E.35}$$

which match the dimensions of $\boldsymbol{\theta}(t) = \{\boldsymbol{b}, W, \sigma^2, \boldsymbol{\mu}_h, \Sigma_h\}(t)$. The transformations are obtained by differentiating Equation (10) of the main text:

$$\begin{aligned}
\frac{dW^\intercal}{dt} =& -\Sigma_h^{-1}\frac{d\Sigma_h}{dt}\Sigma_h^{-1}C_{vh} + \Sigma_h^{-1}\frac{dC_{vh}}{dt}, \\
\frac{d\boldsymbol{b}}{dt} =& \frac{d\boldsymbol{\mu}_v}{dt} - \frac{dC_{vh}^\intercal}{dt}\Sigma_h^{-1}\boldsymbol{\mu}_h + C_{vh}^\intercal\Sigma_h^{-1}\frac{d\Sigma_h}{dt}\Sigma_h^{-1}\boldsymbol{\mu}_h - C_{vh}^\intercal\Sigma_h^{-1}\frac{d\boldsymbol{\mu}_h}{dt}, \\
\frac{d\sigma^2}{dt} =& \frac{d\mathrm{Tr}(C_v)}{dt} - \mathrm{Tr}\left(\left(\frac{dC_{vh}}{dt}\right)^\intercal\Sigma_h^{-1}\Sigma_{vh} + C_{vh}^\intercal\Sigma_h^{-1}\frac{dC_{vh}}{dt} - C_{vh}^\intercal\Sigma_h^{-1}\frac{d\Sigma_h}{dt}\Sigma_h^{-1}C_{vh}\right).
\end{aligned} \tag{E.36}$$

The resulting time evolution vector $\boldsymbol{F}^{(\text{approx.})}_{H+P\to 2H}(\boldsymbol{\theta}(t))$ is an approximation to the true time

evolution. It has been shown that the linearity of the CME in reaction operators extends this form of the approximation [16]. If a linear combination of such approximations is used instead of a neural network, the learned coefficients are directly the reaction rates associated with each process.

Finally, the differential equations can be transformed to the standard parameter space using the inverse of Equation (10) of the main textand its derivative:

$$\begin{aligned} \hat{\boldsymbol{b}} &= \boldsymbol{b} + W\boldsymbol{\mu}_h, \\ \hat{W} &= W\Sigma_h^{1/2}, \end{aligned} \tag{E.37}$$

and

$$\begin{aligned} \hat{F}_{\hat{\boldsymbol{b}}} &= F_{\boldsymbol{b}} + F_W\boldsymbol{\mu}_h + WF_{\boldsymbol{\mu}_h}, \\ \hat{F}_{\hat{W}} &= F_W\Sigma_h^{1/2} + \frac{1}{2}W\Sigma_h^{-1/2}F_{\Sigma_h}, \end{aligned} \tag{E.38}$$

where (E.38) only holds for diagonal latent covariance matrices Σ_h as parameterized in Equation (13) of the main textdue to the derivative of the matrix square root. In general, the derivative of the matrix square root may be expressed through a Kronecker sum as $d\Sigma_h^{1/2}/dt = (\Sigma_h^{1/2} \oplus \Sigma_h^{1/2})^{-1}$.

The result of the transformation is $\hat{\boldsymbol{F}}^{(\text{approx.})}_{H+P\to 2H}$, the approximation to the time evolution in the standard space.

E.2.4 Standardizing inputs / outputs of subnet

The inputs to the subnet of Figure 1 of the main textare standardized as is typical for neural networks, as well as the target outputs. From the training data $X(t)$ of size M samples by N_v visible variables, the matrix is transformed $X(t) \to Y(t)$ as discussed previously, and then the standard ML parameters $\hat{\boldsymbol{\theta}}_{\text{ML}}(t)$ are obtained.

The standardization for the outputs is straightforward. By differentiating $\hat{\boldsymbol{\theta}}(t)$ with total variation regularization, the target outputs of the subnet $d\hat{\boldsymbol{\theta}}_{\mathrm{ML}}/dt$ are obtained. The mean and standard deviation are calculated:

$$\begin{aligned}\boldsymbol{\mu}^{(\text{targets})} &= \frac{1}{T}\sum_{t=1}^{T}\frac{d\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t)}{dt},\\ \boldsymbol{\sigma}^{(\text{targets})} &= \left(\frac{1}{T}\sum_{t=1}^{T}\left(\frac{d\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t)}{dt}-\boldsymbol{\mu}^{(\text{targets})}\right)^2\right)^{1/2},\end{aligned} \tag{E.39}$$

and the targets are standardized:

$$\frac{d\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t)}{dt} \to \left(\frac{d\hat{\boldsymbol{\theta}}_{\mathrm{ML}}(t)}{dt}-\boldsymbol{\mu}^{(\text{targets})}\right)/\boldsymbol{\sigma}^{(\text{targets})} \tag{E.40}$$

Standardizing the inputs is not straightforward because they depend on the latent parameters $\boldsymbol{\mu}_h, \Sigma_h$, which are defined by their Fourier coefficients (Equation (13) of the main text), which are learned. The standardizing parameters may be learned with a normalizing layer, but this is challenging in practice. Instead, we bootstrap the inputs to estimate these parameters as follows. Keep only the highest frequency $f_{\text{max.}}$ in the Fourier expansion (Equation (13) of the main text), and set all corresponding coefficients $a = b = 1$. Then the ML parameters are bootstrapped with this choice for $\boldsymbol{\mu}_h, \Sigma_h$ to estimate the inputs $\hat{\boldsymbol{F}}^{(\text{approx.})}_{A+B\to 2B}$ for the different reactions, and the standardization proceeds for each reaction in the usual way:

$$\begin{aligned}\boldsymbol{\mu}^{(\text{inputs})}_{A+B\to 2B} &= \frac{1}{T}\sum_{t=1}^{T}\hat{\boldsymbol{F}}^{(\text{approx.})}_{A+B\to 2B}(\hat{\boldsymbol{\theta}}(t)),\\ \boldsymbol{\sigma}^{(\text{inputs})}_{A+B\to 2B} &= \left(\frac{1}{T}\sum_{t=1}^{T}\left(\hat{\boldsymbol{F}}^{(\text{approx.})}_{A+B\to 2B}(\hat{\boldsymbol{\theta}}(t))-\boldsymbol{\mu}^{(\text{inputs})}_{A+B\to 2B}\right)^2\right)^{1/2},\\ \hat{\boldsymbol{F}}^{(\text{approx.})}_{A+B\to 2B} &\to \left(\hat{\boldsymbol{F}}^{(\text{approx.})}_{A+B\to 2B}-\boldsymbol{\mu}^{(\text{inputs})}_{A+B\to 2B}\right)/\boldsymbol{\sigma}^{(\text{inputs})}_{A+B\to 2B}.\end{aligned} \tag{E.41}$$

E.3 Learned model of Calcium oscillations

E.3.1 Frequencies

The latent mean and variance are learned in parallel to the parameters in the differential equation model. By not fixing these parameters, the model is more easily able to find a non-intersecting trajectory in $\boldsymbol{\theta}$-space. The latent parameters μ_h, Σ_h are represented by the Fourier decomposition given in Equation (13) of the main text.

Figure E.8 shows the frequencies chosen for calcium oscillation models (Figure 3 of the main textand Figure 4 of the main text). The frequencies $f_n = \{1, 2, 3, 4, 5, 6\} \times 2\pi/40$ s^{-1} allow oscillations on the same period as the calcium oscillations over several seconds.

E.3.2 Learned latent representation

Figure E.9 shows the learned latent mean μ_h in this representation for the four models of Figure 3 of the main text: a deep & wide subnet compared to a shallow & thin subnet, with each a reaction-based model compared to parameter-only model without reactions. The learned frequencies for the reaction-based model are more coherent than for the parameter-only model as seen in Figure E.9, and similarly for Σ_h. This coherence suggests that the network uncovers an emergent order parameter.

E.3.3 Learned moment closure approximation

DBDs learn a moment closure approximation from data. The reduced model evolves in time as:

$$\frac{d\tilde{p}}{dt} = \sum_{\theta \in \boldsymbol{\theta}} \frac{\partial \tilde{p}}{\partial \theta} \times F_\theta, \tag{E.42}$$

where $F_\theta(\boldsymbol{\theta}) = d\theta/dt$ by definition. An observable $\langle X(\boldsymbol{n})\rangle$ where $X(\boldsymbol{n})$ is some scalar function evolves according to:

$$\frac{d\langle X(\boldsymbol{n})\rangle}{dt} = \sum_{\theta\in\boldsymbol{\theta}} \left(\sum_{\boldsymbol{n}} X(\boldsymbol{n}) \frac{\partial \tilde{p}}{\partial \theta} \right) \times F_\theta \tag{E.43}$$

For maximum-entropy distributions, this quantity is a covariance between $X(\boldsymbol{n})$ and the moments controlled by interactions in the energy function [18]. For example, for a restricted Boltzmann machine, correlations between visible units have been replaced by correlations with latent variables, whose activation is learned. For the Gaussian distribution of PCA, it is easiest to work out the equations numerically.

Figure E.10 plots the terms for the mean number of calcium $X(\boldsymbol{n}) = n_{\mathrm{Ca}}$ in the models corresponding to Figure 3 of the manuscript. The terms show that the F_{μ_h} term is learned to be a counter-phase variable to the $F_{b_{\mathrm{Ca}}}$ term. The oscillations for the reaction-based model are more coherent, have higher amplitude and frequency on the order of the calcium oscillations. On the other hand, the parameter-based model oscillates at a higher frequency and lower amplitude. This shows that in the comparison model, the inaccuracies in the learned parameters result partially from finding a flawed latent representation in $\boldsymbol{\mu}_h$ and Σ_h.

E.3.4 Comparison model without reaction approximations

Figure E.11 shows the architecture of the comparison model to Figure 1 of the main text. The network architecture is equivalent except that the reaction approximations are missing. Therefore, the network must learn the functional forms from scratch from the parameters $\boldsymbol{\theta}(t)$.

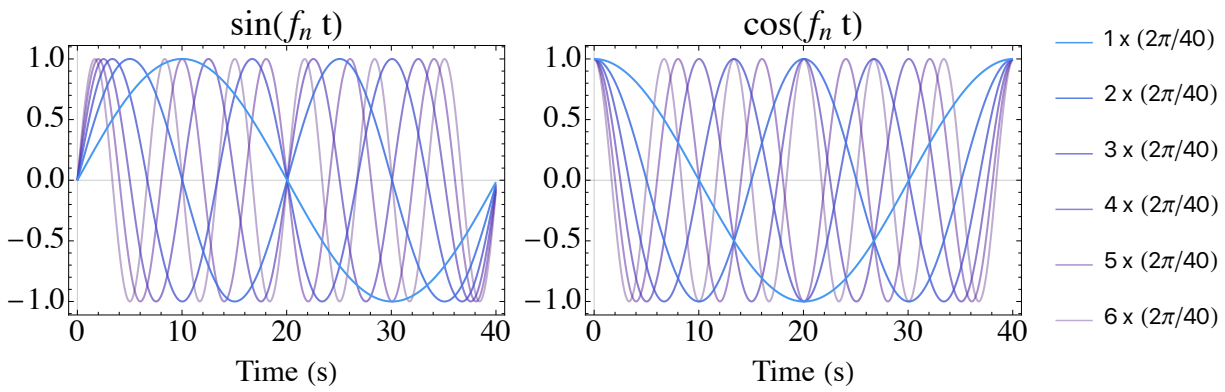

Figure E.8: Frequencies in Equation (13) of the main textused to train the DBD models. The frequencies are chosen to represent oscillations on the same order of magnitude as the calcium oscillations over several seconds.

E.3.5 Mean-squared error (MSE)

The mean-squared error (MSE) over parameters in learned models is shown in Figure 4 of the main text. Let $\hat{\boldsymbol{\theta}}_{\text{int}}(t;[\text{IP}_3])$ be the integrated parameters after learning the reduced model for a single IP_3 concentration, and $\hat{\boldsymbol{\theta}}_{\text{ML}}(t;[\text{IP}_3])$ the maximum likelihood parameters identified from the data. The MSE at a single IP_3 concentration is then given by:

$$\text{MSE}([\text{IP}_3]) = \frac{1}{T}\sum_{t=1}^{T} |\hat{\boldsymbol{\theta}}_{\text{int}}(t;[\text{IP}_3]) - \hat{\boldsymbol{\theta}}_{\text{ML}}(t;[\text{IP}_3])|^2. \tag{E.44}$$

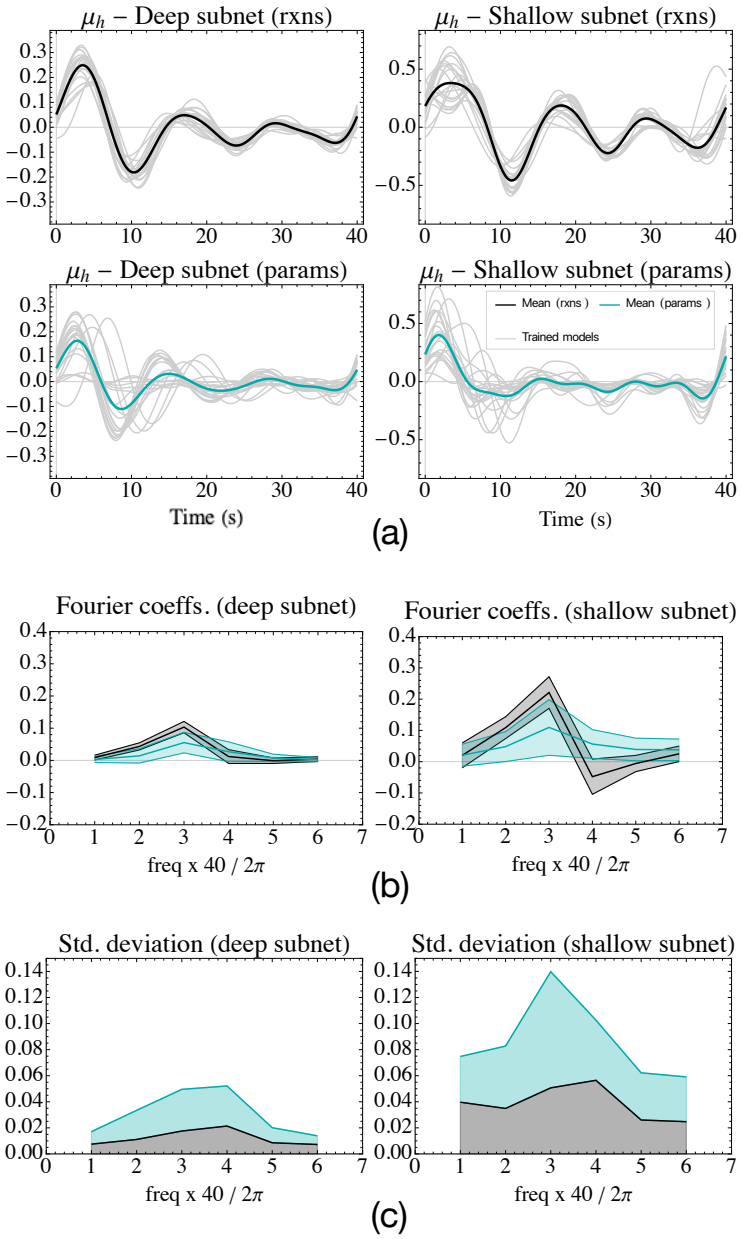

Figure E.9: (a) Learned μ_h time evolution according to the Fourier representation (Equation (13) of the main text) for 20 identical trained models and their mean. *Left column:* deep & wide subnet as shown in Figure 3 of the main text; *bottom row:* shallow & thin subnet. *Right column:* reactions based model as shown in Figure 1 of the main text; *right column:* comparison model without reaction approximations as shown in Figure E.11. (b) The learned Fourier coefficients for μ_h. (c) The standard deviation of the coefficients. The coefficients in the reaction based model are more coherent.

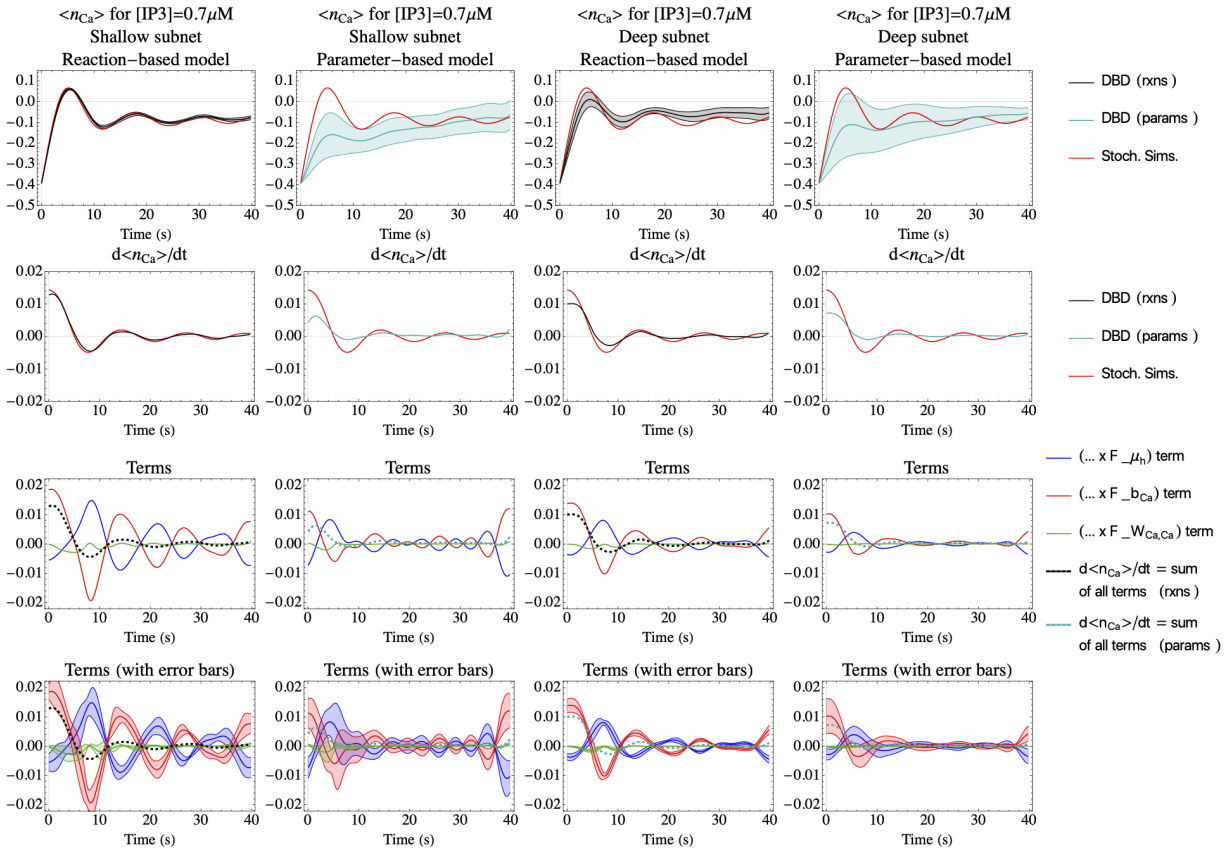

Figure E.10: Moment closure terms learned corresponding to the model of Figure 3 of the manuscript. *First column:* the shallow subnet model for the reaction-based framework. *Second column*: shallow subnet, parameter model. *Third column*: deep subnet, reaction model. *Fourth column*: deep subnet, parameter model. *First row:* At a single concentration of $\mathrm{IP}_3 = 0.7\mu\mathrm{M}$, the mean number of calcium $\langle n_{\mathrm{Ca}} \rangle$ is shown for DBD models, with ground truth from the stochastic simulations in red. *Second row:* The derivative in time of the mean calcium concentration: $d\langle n_{\mathrm{Ca}} \rangle / dt$. *Third row:* The terms in Equation (3) for the mean calcium $X(\boldsymbol{n}) = n_{\mathrm{Ca}}$. *Fourth row:* Same as third row with error bars from 10 optimization trials.

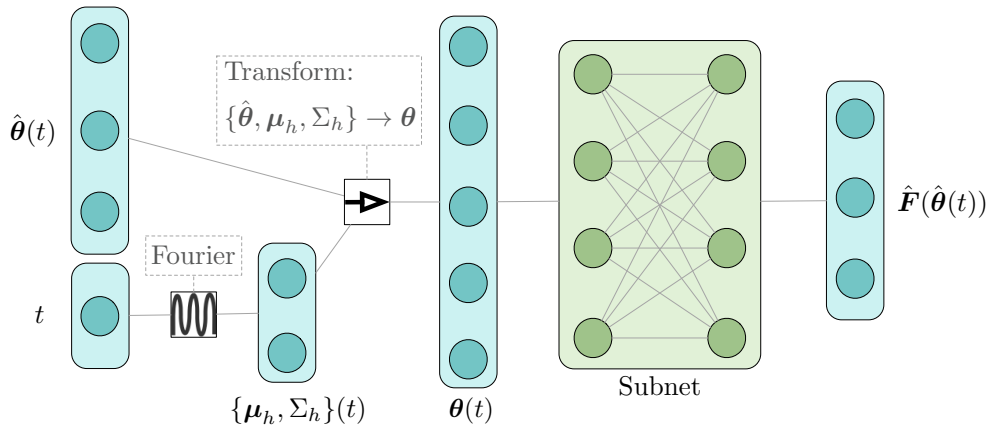

Figure E.11: Comparison architecture similar to Figure 1 of the main text, but missing the analytically derived reaction approximations. Instead, the inputs to the subnet model are the interaction parameters directly.

Bibliography

[1] E. G. GRAY, "Electron microscopy of synaptic contacts on dendrite spines of the cerebral cortex," *Nature*, vol. 183, no. 4675, pp. 1592–1593, 1959.

[2] L. H. HAMLYN, "The fine structure of the mossy fibre endings in the hippocampus of the rabbit.," *J Anat*, vol. 96, pp. 112–120, Jan 1962.

[3] T. M. Bartol, D. X. Keller, J. P. Kinney, C. L. Bajaj, K. M. Harris, T. J. Sejnowski, and M. B. Kennedy, "Computational reconstitution of spine calcium transients from individual proteins," *Frontiers in Synaptic Neuroscience*, vol. 7, p. 17, 2015.

[4] J. Bartol, Thomas M, C. Bromer, J. Kinney, M. A. Chirillo, J. N. Bourne, K. M. Harris, and T. J. Sejnowski, "Nanoconnectomic upper bound on the variability of synaptic plasticity," *eLife*, vol. 4, p. e10778, nov 2015.

[5] J. Stiles and T. Bartol, *Monte Carlo Methods for Simulating Realistic Synaptic Microphysiology Using MCell*. CRC Press, 2017/09/12 2000.

[6] R. A. Kerr, T. M. Bartol, B. Kaminsky, M. Dittrich, J.-C. J. Chang, S. B. Baden, T. Sejnowski, and J. R. Stiles, "Fast Monte Carlo simulation methods for biological reaction-diffusion systems in solution and on surfaces," *SIAM Journal on Scientific Computing : a publication of the Society for Industrial and Applied Mathematics*, vol. 30, pp. 3126–3126, 10 2008.

[7] C. Bromer, T. M. Bartol, J. B. Bowden, D. D. Hubbard, D. C. Hanka, P. V. Gonzalez, M. Kuwajima, J. M. Mendenhall, P. H. Parker, W. C. Abraham, T. J. Sejnowski, and K. M. Harris, "Long-term potentiation expands information content of hippocampal dentate gyrus synapses.," *Proc Natl Acad Sci U S A*, vol. 115, pp. E2410–E2418, Mar 2018.

[8] C. L. Lopreore, T. M. Bartol, J. S. Coggan, D. X. Keller, G. E. Sosinsky, M. H. Ellisman, and T. J. Sejnowski, "Computational modeling of three-dimensional electrodiffusion in biological systems: Application to the node of ranvier," *Biophysical Journal*, vol. 95, no. 6, pp. 2624–2635, 2008.

[9] N. T. Carnevale and M. L. Hines, *The NEURON book*. Cambridge University Press, 2006.

[10] T. Johnson, T. Bartol, T. Sejnowski, and E. Mjolsness, "Model reduction for stochastic CaMKII reaction kinetics in synapses by graph-constrained correlation dynamics," *Physical Biology*, vol. 12, pp. 045005–045005, 07 2015.

[11] I. T. Jolliffe, *Principal Components in Regression Analysis*, pp. 129–155. New York, NY: Springer New York, 1986.

[12] I. Goodfellow, *Deep learning*. Cambridge, Massachusetts: The MIT Press, 2016.

[13] 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.

[14] D. T. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *The Journal of Physical Chemistry*, vol. 81, no. 25, pp. 2340–2361, 1977.

[15] P. Wesseling, "Introduction to multigrid methods.," tech. rep., INSTITUTE FOR COMPUTER APPLICATIONS IN SCIENCE AND ENGINEERING HAMPTON VA, 1995.

[16] O. K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, "Learning dynamic Boltzmann distributions as reduced models of spatial chemical kinetics," *The Journal of Chemical Physics*, vol. 149, no. 3, p. 034107, 2018.

[17] O. K. Ernst, T. M. Bartol, T. J. Sejnowski, and E. Mjolsness, "Learning moment closure in reaction-diffusion systems with spatial dynamic boltzmann distributions," *Phys. Rev. E*, vol. 99, p. 063315, Jun 2019.

[18] O. K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, "Deep learning moment closure approximations using dynamic boltzmann distributions," 2019.

[19] I. J. Laurenzi, "An analytical solution of the stochastic master equation for reversible bimolecular reaction kinetics," *The Journal of Chemical Physics*, vol. 113, no. 8, pp. 3315–3322, 2000.

[20] M. Doi, "Second quantization representation for classical many-particle system," *Journal of Physics A: Mathematical and General*, vol. 9, no. 9, p. 1465, 1976.

[21] M. Doi, "Stochastic theory of diffusion-controlled reaction," *Journal of Physics A: Mathematical and General*, vol. 9, no. 9, p. 1479, 1976.

[22] Peliti, L., "Path integral approach to birth-death processes on a lattice," *J. Phys. France*, vol. 46, no. 9, pp. 1469–1483, 1985.

[23] J. Pausch and G. Pruessner, "Is actin filament and microtubule growth reaction- or diffusion-limited?," *bioRxiv*, 2018.

[24] E. Mjolsness, "Time-ordered product expansions for computational stochastic system biology," *Physical Biology*, vol. 10, p. 035009, jun 2013.

[25] P. Smadbeck and Y. N. Kaznessis, "A closure scheme for chemical master equations," *Proceedings of the National Academy of Sciences*, vol. 110, no. 35, pp. 14261–14265, 2013.

[26] C. Bishop, *Pattern recognition and machine learning*. New York: Springer, 2006.

[27] D. T. Gillespie, "Approximate accelerated stochastic simulation of chemically reacting systems," *The Journal of Chemical Physics*, vol. 115, no. 4, pp. 1716–1733, 2001.

[28] Y. Cao, D. T. Gillespie, and L. R. Petzold, "Efficient step size selection for the tau-leaping simulation method," *The Journal of Chemical Physics*, vol. 124, no. 4, p. 044109, 2006.

[29] A. Auger, P. Chatelain, and P. Koumoutsakos, "R-leaping: Accelerating the stochastic simulation algorithm by reaction leaps," *The Journal of Chemical Physics*, vol. 125, p. 084103, Aug 2006.

[30] E. Mjolsness, D. Orendorff, P. Chatelain, and P. Koumoutsakos, "An exact accelerated stochastic simulation algorithm," *The Journal of Chemical Physics*, vol. 130, no. 14, p. 144110, 2009.

[31] R. Salakhutdinov and G. Hinton, "Deep Boltzmann machines," in *Artificial intelligence and statistics*, pp. 448–455, 2009.

[32] H. Jang, H. Choi, Y. Yi, and J. Shin, "Adiabatic persistent contrastive divergence learning," in *2017 IEEE International Symposium on Information Theory (ISIT)*, pp. 3005–3009, IEEE, 2017.

[33] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[34] M. A. Carreira-Perpinan and G. E. Hinton, "On contrastive divergence learning.," in *Aistats*, vol. 10, pp. 33–40, Citeseer, 2005.

[35] M. A. Carreira-Perpinan and G. E. Hinton, "On contrastive divergence learning.," in *Aistats*, vol. 10, pp. 33–40, Citeseer, 2005.

[36] T. Tieleman, "Training restricted Boltzmann machines using approximations to the likelihood gradient," in *Proceedings of the 25th international conference on Machine learning*, pp. 1064–1071, ACM, 2008.

[37] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks," *Science*, vol. 355, no. 6325, pp. 602–606, 2017.

[38] J. Tubiana, S. Cocco, and R. Monasson, "Learning Compositional Representations of Interacting Systems with Restricted Boltzmann Machines: Comparative Study of Lattice Proteins," *Neural Computation*, vol. 31, pp. 1671–1717, 08 2019.

[39] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[40] G. Montavon and K.-R. Müller, "Deep Boltzmann machines and the centering trick," in *Neural Networks: Tricks of the Trade*, pp. 621–637, Springer, 2012.

[41] J. Melchior, A. Fischer, and L. Wiskott, "How to center deep Boltzmann machines," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 3387–3447, 2016.

[42] J. Tubiana and R. Monasson, "Emergence of compositional representations in restricted boltzmann machines," *Phys. Rev. Lett.*, vol. 118, p. 138301, Mar 2017.

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[44] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, no. 15, pp. 3932–3937, 2016.

[45] M. Griebel, "Sparse grids and related approximation schemes for higher dimensional problems," in *Proceedings of the Conference on Foundations of Computational Mathematics (FoCM05)*, (Santander), 2005.

[46] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence o(1/k2)," *Soviet Math. Dokl.*, vol. 27, no. 2, pp. 372 – 376, 1983.

[47] M. B. Giles and N. A. Pierce, "An introduction to the adjoint approach to design," *Flow, Turbulence and Combustion*, vol. 65, pp. 393–415, Dec 2000.

[48] L. Onsager, "Crystal Statistics. I. A two-dimensional model with an order-disorder transition," *Phys. Rev.*, vol. 65, pp. 117–149, Feb 1944.

[49] S. El-Showk, M. F. Paulos, D. Poland, S. Rychkov, D. Simmons-Duffin, and A. Vichi, "Solving the 3d ising model with the conformal bootstrap," *Phys. Rev. D*, vol. 86, p. 025022, Jul 2012.

[50] E. Mjolsness and U. Prasad, "Mathematics of small stochastic reaction networks: A boundary layer theory for eigenstate analysis," *Journal of Chemical Physics*, vol. 138, no. 10, p. 104111, 2013.

[51] E. Mjolsness, "Time-ordered product expansions for computational stochastic system biology," *Physical Biology*, vol. 10, p. 035009, June 2013.

[52] H. Takayasu and A. Y. Tretyakov, "Extinction, survival, and dynamical phase transition of branching annihilating random walk," *Phys. Rev. Lett.*, vol. 68, pp. 3060–3063, May 1992.

[53] J. Cardy and U. C. Täuber, "Theory of branching and annihilating random walks," *Phys. Rev. Lett.*, vol. 77, pp. 4780–4783, Dec 1996.

[54] R. V. Gamkrelidze and G. L. Kharatishvili, "Extremal problems in linear topological spaces. i.," *Mathematical systems theory*, vol. 1, pp. 229–256, Sep 1967.

[55] L. W. Neustadt, "Optimal control problems with operator equation restrictions," in *Symposium on Optimization* (A. V. Balakrishnan, M. Contensou, B. F. de Veubeke, P. Krée, J. L. Lions, and N. N. Moiseev, eds.), (Berlin, Heidelberg), pp. 292–306, Springer Berlin Heidelberg, 1970.

[56] R. V. Gamkrelidze, *Principles of Optimal Control Theory*. Boston, MA: Springer US, 1978.

[57] Y. Cao, S. Li, L. Petzold, and R. Serban, "Adjoint sensitivity analysis for differential-algebraic equations: The adjoint DAE system and its numerical solution," *SIAM Journal on Scientific Computing*, vol. 24, no. 3, pp. 1076–1089, 2003.

[58] P. J. Green and D. I. Hastie, "Reversible jump mcmc," *Genetics*, vol. 155, no. 3, pp. 1391–1403, 2009.

[59] B. G. VAN BLOEMEN WAANDERS, R. A. BARTLETT, K. R. LONG, P. T. BOGGS, and A. G. SALINGER, "Large scale non-linear programming for pde constrained optimization," tech. rep., Sandia National Labs., Albuquerque, NM (US); Sandia National Labs ..., 2002.

[60] R. Herzog and K. Kunisch, "Algorithms for pde-constrained optimization," *GAMM-Mitteilungen*, 2014.

[61] I. Murray and R. R. Salakhutdinov, "Evaluating probabilities under high-dimensional latent variable models," in *Advances in Neural Information Processing Systems 21* (D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, eds.), pp. 1137–1144, Curran Associates, Inc., 2009.

[62] A. S. Berahas, M. Jahani, P. Richtárik, and M. Takáč, "Quasi-newton methods for deep learning: Forget the past, just sample," *arXiv preprint arXiv:1901.09997*, 2019.

[63] A. S. Berahas and M. Takáč, "A robust multi-batch l-bfgs method for machine learning," *Optimization Methods and Software*, vol. 35, no. 1, pp. 191–219, 2020.

[64] J. Martens, "Deep learning via hessian-free optimization.," in *ICML*, vol. 27, pp. 735–742, 2010.

[65] M. S. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, and G. N. Wells, "The fenics project version 1.5," *Archive of Numerical Software*, vol. 3, no. 100, 2015.

[66] S. Wright and J. Nocedal, "Numerical optimization," *Springer Science*, vol. 35, no. 67-68, p. 7, 1999.

[67] N. N. Schraudolph, J. Yu, and S. Günter, "A stochastic quasi-newton method for online convex optimization," in *Artificial intelligence and statistics*, pp. 436–443, PMLR, 2007.

[68] M. J. D. Powell, "Algorithms for nonlinear constraints that use lagrangian functions," *Mathematical Programming*, vol. 14, no. 1, pp. 224–248, 1978.

[69] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[70] H. D. I. Abarbanel, *Predicting the future: Completing models of observed complex systems*. New York, NY: Springer, 2013.

[71] E. Mjolsness, "Prospects for declarative mathematical modeling of complex biological systems." arXiv:1804.11044, 2018.

[72] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," tech. rep., Colorado Univ. at Boulder Dept. of Computer Science, 1986.

[73] Z. Ghahramani, *Learning dynamic Bayesian networks*, pp. 168–197. Springer, 1998.

[74] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[75] R. Herzog, "Lectures notes algorithms and preconditioning in pde-constrained optimization," tech. rep., Chemnitz University of Technology, 2010.

[76] S. W. Funke and P. E. Farrell, "A framework for automated pde-constrained optimisation," *arXiv preprint arXiv:1302.3894*, 2013.

[77] T. Hughes, *The finite element method: linear static and dynamic finite element analysis*. Mineola, NY: Dover Publications, 2000.

[78] D. Arnold and A. Logg, "Periodic table of the finite elements," *SIAM News*, vol. 47, no. 9, 2014.

[79] L. Bronstein and H. Koeppl, "A variational approach to moment-closure approximations for the kinetics of biomolecular reaction networks," *The Journal of Chemical Physics*, vol. 148, no. 1, p. 014105, 2018.

[80] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[81] W. K. D. and R. O. E., "Irregular oscillations in a realistic abstract quadratic mass action system," *Zeitschrift für Naturforschung A*, vol. 35, p. 317, November 1980.

[82] G. Bellesia and B. B. Bales, "Population dynamics, information transfer, and spatial organization in a chemical reaction network under spatial confinement and crowding conditions," *Phys. Rev. E*, vol. 94, p. 042306, Oct 2016.

[83] V. Anishchenko, T. Vadivasova, G. Strelkova, and G. Okrokvertskhov, "Statistical properties of dynamical chaos.," *Mathematical Biosciences and Engineering: MBE*, vol. 1, no. 1, p. 161, 2004.

[84] O. K. Ernst, T. Bartol, T. Sejnowski, and E. Mjolsness, "Dynamic Boltzmann Distributions C++ Library v4.0," 2019.

[85] M. Ranzato, A. Krizhevsky, and G. Hinton, "Factored 3-way restricted Boltzmann machines for modeling natural images," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Y. W. Teh and M. Titterington, eds.), vol. 9 of *Proceedings of Machine Learning Research*, (Chia Laguna Resort, Sardinia, Italy), pp. 621–628, PMLR, 13–15 May 2010.

[86] K. Vincent, M. Gonzales, A. Gillette, C. Villongco, S. Pezzuto, J. Omens, M. Holst, and A. McCulloch, "High-order finite element methods for cardiac monodomain simulations," *Frontiers in Physiology*, vol. 6, p. 217, 2015.

[87] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Žídek, A. W. R. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. T. Jones, D. Silver, K. Kavukcuoglu, and D. Hassabis, "Improved protein structure prediction using potentials from deep learning," *Nature*, vol. 577, no. 7792, pp. 706–710, 2020.

[88] N. Baker, F. Alexander, T. Bremer, A. Hagberg, Y. Kevrekidis, H. Najm, M. Parashar, A. Patra, J. Sethian, S. Wild, K. Willcox, and S. Lee, "Workshop report on basic research needs for scientific machine learning: Core technologies for artificial intelligence," tech. rep., DOE Office of Science, 2 2019.

[89] E. Mjolsness and D. DeCoste, "Machine learning for science: State of the art and future prospects," *Science*, vol. 293, no. 5537, pp. 2051–2055, 2001.

[90] E. de Bézenac, A. Pajot, and P. Gallinari, "Deep learning for physical processes: incorporating prior scientific knowledge," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2019, p. 124009, dec 2019.

[91] D. Decoste and B. Schölkopf, "Training invariant support vector machines," *Mach. Learn.*, vol. 46, pp. 161–190, Mar. 2002.

[92] S. Branson, G. Van Horn, S. Belongie, and P. Perona, "Bird species categorization using pose normalized deep convolutional nets," *arXiv preprint arXiv:1406.2952*, 2014.

[93] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, "Fast and accurate modeling of molecular atomization energies with machine learning," *Phys. Rev. Lett.*, vol. 108, p. 058301, Jan 2012.

[94] E. Racah, C. Beckham, T. Maharaj, S. E. Kahou, Prabhat, and C. Pal, "Extreme weather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), pp. 3405–3416, Curran Associates Inc., 2017.

[95] T. Giordani, A. Suprano, E. Polino, F. Acanfora, L. Innocenti, A. Ferraro, M. Paternostro, N. Spagnolo, and F. Sciarrino, "Machine learning-based classification of vector vortex beams," *Phys. Rev. Lett.*, vol. 124, p. 160401, Apr 2020.

[96] R. Iten, T. Metger, H. Wilming, L. del Rio, and R. Renner, "Discovering physical concepts with neural networks," *Phys. Rev. Lett.*, vol. 124, p. 010508, Jan 2020.

[97] S. C. Leemann, S. Liu, A. Hexemer, M. A. Marcus, C. N. Melton, H. Nishimura, and C. Sun, "Demonstration of machine learning-based model-independent stabilization of source properties in synchrotron light sources," *Phys. Rev. Lett.*, vol. 123, p. 194801, Nov 2019.

[98] E. Magesan, J. M. Gambetta, A. D. Córcoles, and J. M. Chow, "Machine learning for discriminating quantum measurement trajectories and improving readout," *Phys. Rev. Lett.*, vol. 114, p. 200501, May 2015.

[99] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Multistep neural networks for data-driven discovery of nonlinear dynamical systems," *arXiv preprint arXiv:1801.01236*, 2018.

[100] T. N. Thiem, M. Kooshkbaghi, T. Bertalan, C. R. Laing, and I. G. Kevrekidis, "Emergent spaces for coupled oscillators," *Frontiers in computational neuroscience*, vol. 14, pp. 36–36, 05 2020.

[101] Z. Long, Y. Lu, X. Ma, and B. Dong, "PDE-net: Learning PDEs from data," in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 3208–3216, PMLR, 10–15 Jul 2018.

[102] G. W. De Young and J. Keizer, "A single-pool inositol 1,4,5-trisphosphate-receptor-based model for agonist-stimulated oscillations in ca2+ concentration.," *Proc Natl Acad Sci U S A*, vol. 89, pp. 9895–9899, Oct 1992.

[103] D. C. Mattis and M. L. Glasser, "The uses of quantum field theory in diffusion-limited reactions," *Rev. Mod. Phys.*, vol. 70, pp. 979–1001, Jul 1998.

[104] D. Xiao, L. Marin, and R. Chartrand, "Numerical differentiation of noisy, nonsmooth data," *ISRN Applied Mathematics*, vol. 2011, p. 164564, 2011.

[105] V. Voorsluijs, S. P. Dawson, Y. De Decker, and G. Dupont, "Deterministic limit of intracellular calcium spikes," *Phys. Rev. Lett.*, vol. 122, p. 088101, Feb 2019.

[106] T. Ohira and J. D. Cowan, *Stochastic Neurodynamics and the System Size Expansion*, pp. 290–294. Boston, MA: Springer US, 1997.

[107] D. Yu and J. Li, "Recent progresses in deep learning based acoustic models (updated)," 2018.

[108] J. Behler and M. Parrinello, "Generalized neural-network representation of high-dimensional potential-energy surfaces," *Phys. Rev. Lett.*, vol. 98, p. 146401, Apr 2007.

[109] A. Skupin, H. Kettenmann, U. Winkler, M. Wartenberg, H. Sauer, S. C. Tovey, C. W. Taylor, and M. Falcke, "How does intracellular ca^{2+} oscillate: By chance or by the clock?," *Biophysical Journal*, vol. 94, pp. 2404–2411, 2021/05/28 2008.

[110] E. Mjolsness, "Structural commutation relations for stochastic labelled graph grammar rule operators," *CoRR*, vol. abs/1909.04118, 2019.

[111] P. Lancaster, "On eigenvalues of matrices dependent on a parameter," *Numerische Mathematik*, vol. 6, no. 1, pp. 377–387, 1964.

[112] M. D. Gunzburger, *Perspectives in Flow Control and Optimization*. Society for Industrial and Applied Mathematics, 2002.