Implementing Bayesian Inference with Neural Networks

Von der Fakultät für Mathematik und Informatik
der Universität Leipzig
angenommene

DISSERTATION

zur Erlangung des akademischen Grades

DOCTOR RERUM NATURALIUM
(Dr.rer.nat.)

im Fachgebiet

Informatik

vorgelegt

von M.Sc. Sacha Sokoloski
geboren am 27.12.1982 in Toronto, Canada

Die Annahme der Dissertation wurde empfohlen von:

1. Professor Dr. Nihat Ay (MPI MIS Leipzig)
2. Professor Dr. Manfred Opper (Technische Universität Berlin)

Die Verleihung des akademischen Grades erfolgt mit Bestehen der Verteidigung am 20.05.2019 mit dem Gesamtprädikat magna cum laude.

Abstract

Embodied agents, be they animals or robots, acquire information about the world through their senses. Embodied agents, however, do not simply lose this information once it passes by, but rather they process it and store it for future use. The most general theory of how an agent can combine stored knowledge with new observations is Bayesian inference. In this dissertation I present a theory of how embodied agents can learn to implement Bayesian inference with neural networks.

By neural network I mean both artificial and biological neural networks, and in my dissertation I address both kinds. On one hand, I develop theory for implementing Bayesian inference in deep generative models, and I show how to train multilayer perceptrons to compute approximate predictions for Bayesian filtering. On the other hand, I show that several models in computational neuroscience are special cases of the general theory that I develop in this dissertation, and I use this theory to model and explain a range of phenomena which are found in the neuroscience literature. The key contributions of this dissertation can be summarized as follows:

- I develop a class of graphical model called nth-order harmoniums. An nth-order harmonium is an n-tuple of random variables, where the conditional distribution of each variable given all the others is always an element of the same exponential family. I show that harmoniums have a recursive structure which allows them to be analyzed at coarser and finer levels of detail.

- I define a class of harmoniums called rectified harmoniums, which are constrained to have priors which are conjugate to their posteriors. As a consequence of this, rectified harmoniums afford efficient sampling and learning.

- I develop deep harmoniums, which are harmoniums which can be represented by hierarchical, undirected graphs. I develop the theory of rectification for deep harmoniums, and develop a novel algorithm for training deep generative models.

- I show how to implement a variety of optimal and near-optimal Bayes filters by combining the solution to Bayes' rule provided by rectified harmoniums, with predictions computed by a recurrent neural network. I then show how to train a neural network to implement Bayesian filtering when the transition and emission distributions are unknown.

- I show how some well-established models of neural activity are special cases of the theory I present in this dissertation, and how these models can be generalized with the theory of rectification.

- I show how the theory that I present can model several neural phenomena including proprioception and gain-field modulation of tuning curves.

- I introduce a library for the programming language Haskell, within which I have implemented all the simulations presented in this dissertation. This library uses concepts from Riemannian geometry to provide a rigorous and efficient environment for implementing complex numerical simulations.

I also use the results presented in this dissertation to argue for the fundamental role of neural computation in embodied cognition. I argue, in other words, that before we will be able to build truly intelligent robots, we will need to truly understand biological brains.

Acknowledgements

I would like to thank Nihat Ay for seeing something in me and inviting me to pursue my doctorate at the Max Planck Institute for Mathematics in the Sciences. I would like to thank some of the members, past and present, of his lab, including Guido Montúfar, Keyan Ghazi-Zahedi, and Johannes Rauh, who patiently answered my questions, and supported me along the way.

I would like to thank director Jürgen Jost, and many of the members, past and present, of his lab, including Wiktor Młynarski, Eckehard Olbrich, and Nils Bertschinger, for their insightful discussions, and for providing the stimulating intellectual atmosphere that I experienced at the MIS. Finally, I would like to thank the entire staff of the MIS, in particular Antje Vandenberg and Heike Rackwitz, for their constant help, and for making German bureaucracy seem manageable.

I would like to thank Joseph Makin for his incredibly thorough reviews of my first paper, which lead eventually to my first publication. I would also like to thank Ruben Coen-Cagli, for providing me with support and a place to work after I moved to New York city, as I entered the final phase of my doctoral studies.

I would like to thank my family, for being there when I needed them. I would like to thank my wife Anna Erzberger, for supporting me in every way imaginable, and certainly more than I can list. I would also like to thank her parents and family, for their warmth, hospitality, and support. Finally I would like to thank my son Nikolai Armin Sokoloski, for adding a sense of purpose to my work. Thank you, Niko, for being so cooperative during my extended paternity leave. I would not have been so productive were it not for your permission.

Contents

Abstract **iii**

Acknowledgements **v**

Contents **vii**

List of Definitions **ix**

List of Theorems **xi**

List of Figures **xiii**

Notation **xv**

1 Introduction **1**
- 1.1 Inference in Deep Learning . . . 1
- 1.2 Inference in Computational Neuroscience . . . 3
- 1.3 Dissertation Outline . . . 4
- 1.4 Related Work . . . 6

2 Mathematical Background **9**
- 2.1 Measure Theory . . . 9
- 2.2 Statistical Models . . . 11
- 2.3 Exponential Families . . . 14
- 2.4 Graphical Models . . . 20
- 2.5 Stochastic Chains . . . 23
- 2.6 Bayesian Inference . . . 26

3 Families of Harmoniums **31**
- 3.1 Second-Order Harmoniums . . . 31
  - 3.1.1 Second-Order Conditional Specification . . . 32
  - 3.1.2 Properties of Harmoniums . . . 35
- 3.2 Higher-Order Harmoniums . . . 38
  - 3.2.1 Higher-Order Conditional Specification . . . 39
  - 3.2.2 Harmonium Factorization . . . 40
  - 3.2.3 Deep Harmoniums . . . 43

4 Harmonium Rectification **45**
- 4.1 Second-Order Harmonium Rectification . . . 45
  - 4.1.1 Rectified Second-Order Harmoniums . . . 46

    4.1.2 Exact Rectification 48
    4.1.3 Approximate Rectification 52
  4.2 Deep Harmonium Rectification 55
    4.2.1 Rectified Deep Harmoniums 55
    4.2.2 Fitting and Rectifying Deep Harmoniums 59

5 Bayesian Inference with Artificial Neural Networks 65
  5.1 Implementing Static Inference 65
    5.1.1 Implementing Bayes' Rule 65
    5.1.2 Encoding Beliefs 68
  5.2 Implementing Bayesian Filtering 70
    5.2.1 Implementing Bayesian Prediction 70
    5.2.2 Exponential Family Multilayer Perceptrons 73
    5.2.3 Modelling a Bayes Filter 76
    5.2.4 Training a Model Bayes Filter 79

6 Bayesian Inference with Biological Neural Networks 83
  6.1 Linear-Nonlinear Neurons 83
  6.2 Linear Probabilistic Population Codes 84
  6.3 LPPCs and Harmoniums 88
  6.4 Homogeneous LPPCs 90

7 Simulations 95
  7.1 Geometric Optimization Libraries 95
    7.1.1 Type Systems and Haskell 96
    7.1.2 Manifolds 99
    7.1.3 Differential Geometry 100
    7.1.4 Exponential Families 103
  7.2 Restricted Boltzmann Machines 104
  7.3 Biological Neural Circuits 106
    7.3.1 Training and Validation Procedure 106
    7.3.2 Colour Sequence Learning 108
    7.3.3 Self-Localization 109
    7.3.4 Proprioception 111
    7.3.5 Analysis of Results 112

8 Conclusion 115
  8.1 Brains, a Priori 115
  8.2 Outlook 117

A List of Exponential Families 119

Bibliography 123

List of Definitions

2.1 Definition (Absolute Continuity) 10
2.2 Definition (Minimal Sufficient Statistic) 16
2.3 Definition (Bayesian Network) 21
2.4 Definition (Markov Random Field) 21
2.5 Definition (Stationary Stochastic Process) 23
2.6 Definition (Equilibrium Distribution) 24
2.7 Definition (Harris Recurrence) 25
2.8 Definition (Aperiodic Markov Chain) 25
2.9 Definition (Ergodic Markov Chain) 25
2.10 Definition (Conjugate Prior) 27

3.1 Definition (Second-Order Harmonium) 32
3.2 Definition (Higher-Order Harmoniums) 39
3.3 Definition (Interaction) 41
3.4 Definition (Deep Harmonium) 43

4.1 Definition (Rectified Harmonium) 46
4.2 Definition (Rectified Likelihood) 48
4.3 Definition (Rectified Deep Harmonium) 55

5.1 Definition (Implementing Bayes' Rule 1) 66
5.2 Definition (Encoding) 68
5.3 Definition (Implementing Bayes' Rule 2) 68
5.4 Definition (Implementing Bayesian Prediction 1) 70
5.5 Definition (Implementing a Bayes Filter 1) 71
5.6 Definition (Neural Bayes Filter) 78

6.1 Definition (Homogeneous LPPC) 90

List of Theorems

2.1 Theorem (Radon-Nikodym Theorem) 10
2.2 Proposition (Gibbs' Inequality) 11
2.3 Theorem 14
2.4 Proposition 16
2.5 Corollary 16
2.6 Proposition 17
2.7 Theorem 17
2.8 Theorem 19
2.9 Theorem (The Hammersley-Clifford Theorem) 22
2.10 Theorem 25
2.11 Theorem (The Ergodic Theorem) 25

3.1 Theorem (Second-Order Conditional Specification) 33
3.2 Proposition 33
3.3 Proposition 36
3.4 Proposition 36
3.5 Proposition 37
3.6 Theorem (Higher-Order Conditional Specification) 39
3.7 Theorem (The Harmonium Factorization Theorem) 40
3.8 Lemma 42
3.9 Theorem 42

4.1 Theorem (The Rectification Theorem) 47
4.2 Corollary 47
4.3 Corollary 48
4.4 Corollary 48
4.5 Proposition 49
4.6 Theorem 51
4.7 Theorem 52
4.8 Theorem 56
4.9 Lemma 56
4.10 Theorem (The Deep Rectification Theorem) 57
4.11 Corollary 59
4.12 Theorem 60

5.1 Proposition 66
5.2 Proposition 69
5.3 Proposition 71

6.1 Proposition 89
6.2 Proposition 90
6.3 Proposition 91

List of Figures

1.1 Deep Generative Models 2
1.2 Neural Circuits of Proprioception 3

2.1 Isolines of the Relative Entropy 18
2.2 Natural Gradient Descent 19
2.3 A Directed Graphical Model 21
2.4 An Undirected Graphical Model 22
2.5 Markov Chains 24
2.6 Hidden Markov Models 26
2.7 Multisensory Bayesian Inference 28
2.8 Extended Kalman Filter 29

3.1 Second-Order Harmoniums as Two-Variable Graphs 32
3.2 Second-Order Harmoniums as Bipartite Graphs 34
3.3 Third-Order Harmoniums as Tripartite Graphs 39
3.4 Factorization of a Complex Harmonium 41
3.5 Deep Harmoniums as Hierarchical Graphs 44

4.1 Rectification 46
4.2 Mixture Models as Rectified Harmoniums 50
4.3 Rectified Deep Harmoniums 58
4.4 Comparison of Contrastive Divergence Algorithms 63

5.1 Affine Belief Encoding 69
5.2 An Exponential Family Multilayer Perceptron 74
5.3 Training on EFMLP on Toy Data 75
5.4 Graph of an EFMLP Bayes Filter 77
5.5 Graph of a Neural Bayes Filter 78

6.1 Bernoulli, Binomial, and Poisson Distributions 85
6.2 Convergence Properties of Collections of Tuning Curves 87
6.3 Bayesian Inference with LPPCs 88
6.4 Rectification in Homogeneous LPPCs 92

7.1 Coordinate Transforms in Goal 101
7.2 Fitting and Rectifying RBMs 105
7.3 Neural Hidden Markov Model 110
7.4 Neural Kalman Filtering 111
7.5 Optimizing Proprioception of the Human Arm 113

Notation

Calculus

$a, b, f, g \ldots$	Scalars, scalar valued functions
$\mathbf{a}, \mathbf{b}, \mathbf{f}, \mathbf{g} \ldots$	Vectors and vector-valued functions
$a_i, b_i, f_i, g_i \ldots$	ith element of a vector
$\mathbf{A}, \mathbf{B} \ldots$	Matrices
$\mathbf{A}^\top$	Matrix transpose
$\mathbf{a}_i, \mathbf{b}_i \ldots$	ith row of a matrix
$a_{i,j}, b_{i,j} \ldots$	(i, j)th element of a matrix
$a\mathbf{b}$	Scalar-multiplication of $\mathbf{b}$ by a
$\mathbf{a} \cdot \mathbf{b}$	Dot-product of the vectors $\mathbf{a}$ and $\mathbf{b}$
$\mathbf{a} \otimes \mathbf{b}$	Outer-product of the vectors $\mathbf{a}$ and $\mathbf{b}$
$\nabla_{\mathbf{a}} f$	Geometric gradient of $\mathbf{f}$ with respect to $\mathbf{a}$
$\partial_{\mathbf{a}} f$	Partial derivatives of f with respect to $\mathbf{a}$
$\partial_{\mathbf{a}} \mathbf{f}$	Jacobian of $\mathbf{f}$ with respect to $\mathbf{a}$
$\partial_{\mathbf{aa}} f$	Hessian of f with respect to $\mathbf{a}$
Ω	Sample space
$\mathcal{F}$	σ-algebra
$(\Omega, \mathcal{F})$	Measurable space
μ, ν	Measures
$\int_\Omega f\mu$	The integral of f over Ω with respect to μ

Probability Theory

Ω	Sample space
P	Probability measure
$Q(\boldsymbol{\theta})$	Probabilistic model (depends on parameters)
Λ	Space of probability measures
$X, Y, Z \ldots$	Random variables
Ω_X	State space of X
$\mathcal{F}_X$	σ-algebra over Ω_X
$X \in \Omega_X$	Declaration of the state space of X
P_X	Probability distribution of X
$P_{X\|Y}$	Conditional probability distribution of X given Y
p_X	Probability density of X
$p_{X\|Y}$	Conditional probability density of X given Y
$H(P)$	Entropy of P
$H(P, Q)$	Cross-Entropy of P and Q
$D(P \parallel Q)$	The relative entropy of P with respect to Q

$\mathbb{E}_P[f(X)]$	Integral of f over Ω_X with respect to P_X
$\mathbb{E}_P[f(X) \mid Y = \mathbf{y}]$	Integral of f with respect to $P_{X\mid Y=\mathbf{x}}$
$\mathbb{E}_P[\mathbb{E}_P[f(X) \mid Y = Z]]$	Integral of f with respect to $\int_{\Omega_Z} P_{X\mid Y=\mathbf{z}} P_Z(d\mathbf{z})$
$\mathbb{E}_P[\mathbb{E}_Q[f(X) \mid Y]]$	Integral of f with respect to $\int_{\Omega_Z} Q_{X\mid Y=\mathbf{y}} P_Y(d\mathbf{y})$
$\mathbf{C}_P(X, Y)$	Covariance of X and Y with respect to P

Exponential Families

$\mathcal{M}$	Exponential family (manifold)
$\mathbf{s}$	Sufficient Statistic
ψ	Log-partition function
ϕ	Negative Entropy
Θ	Natural parameter space
$\boldsymbol{\theta}$	Natural parameters
H	Mean parameter space
$\boldsymbol{\eta}$	Mean parameters
$\boldsymbol{\tau}$	Forward mapping (Natural to Mean Coordinates)
$\boldsymbol{\tau}^*$	Backward mapping (Mean to Natural Coordinates)
$\mathcal{M}_X$	Exponential family over measurable space $(\Omega_X, \mathcal{F}_X)$
$\mathbf{s}_X, \psi_X, \boldsymbol{\theta}_X \dots$	Sufficient statistic, etc. of $\mathcal{M}_X$
$\mathcal{M}_X \cdot \mathcal{M}_Z$	Product family of all $P_X \cdot P_Z$ in $\mathcal{M}_X$ and $\mathcal{M}_Z$
Q_X	Exponential family model in $\mathcal{M}_X$
$P_{X\mid Z} \in \mathcal{M}_X$	Short-hand for $P_{X\mid Z=\mathbf{z}} \in \mathcal{M}_X, \forall \mathbf{z} \in \Omega_Z$

Harmoniums and Neural Networks

X	Latent Variables
Z	Observable Variables
Y	Intermediate Variables
$\mathcal{H}_{XZ}$	A second-order harmonium family
$\mathcal{M}_X \otimes \mathcal{M}_Z$	An operator for constructing $\mathcal{H}_{XZ}$
$\mathcal{H}_{(X_i)_{i=1}^n}$	An nth-order harmonium family
$\mathcal{D}_{XYZ}$	A deep harmonium family
$\mathbf{s}_{XZ}, \psi_{XZ}, \boldsymbol{\theta}_{XZ} \dots$	Sufficient statistic, etc. of $\mathcal{H}_{XZ}$
$\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z$	Biases of $P_{XZ} \in \mathcal{H}_{XZ}$
$\boldsymbol{\Theta}_{XZ}$	Interaction parameters of $P_{XZ} \in \mathcal{H}_{XZ}$
$\boldsymbol{\rho}_X, \rho_0$	Rectification parameters of $P_{XZ} \in \mathcal{H}_{XZ}$
$\boldsymbol{\theta}_X^*$	Parameters of $P_X \in \mathcal{M}_X$ for rectified $P_{XZ} \in \mathcal{H}_{XZ}$
Q_X^*	Rectifier of Q_{XZ}
$\boldsymbol{\theta}_{X\mid Z}$	Exponential Family Multilayer Perceptron (EFMLP)
$\boldsymbol{\chi}$	Parameters of an EFMLP
$\mathcal{X}$	Parameter space of an EFMLP

Chapter 1

Introduction

Statistical inference is the process of transforming observations of an unknown quantity into an estimate of that quantity, and Bayesian inference is the most general form of statistical inference which is consistent with the laws of probability theory (Jaynes, 2003; Talbott, 2015). In the context of embodied cognition, Bayesian inference describes how to an agent should update its beliefs about the world based on observations, and provides a formal theory of perception (Knill and Richards, 1996; Kersten et al., 2004; Yuille and Kersten, 2006).

An embodied agent can implement Bayesian inference by learning a generative model of latent and observable variables; the prior beliefs of the agent are given by the marginal distribution of the generative model over the latent (unknown) variables, and the posterior beliefs of the agent are given by the model distribution of the latent variables given the observations. Yet the fact that a generative model can support Bayesian inference does not address how embodied agents can implement Bayesian inference in the brain, or learn the underlying generative model which defines it.

The subject of this dissertation is how embodied agents learn to implement Bayesian inference with neural networks. By “neural network” I mean both the artificial neural networks of deep learning, and the biological neural circuits studied in computational neuroscience. And whether we call it inferring the category of some unlabelled data, or inferring the stimulus of some population of neurons, inferring latent state is fundamental to both fields.

1.1 Inference in Deep Learning

Contemporary research into artificial neural networks falls under the banner of deep learning, yet modern deep learning sprung from the ashes of what used to be known as connectionism (Hinton, 1989). The field of connectionism is where both multilayer perceptrons (Rumelhart et al., 1986) and restricted Boltzmann machines (Ackley et al., 1985) were first developed and applied, before initial excitement in artificial neural networks was replaced by disappointment, and the field went into something of a dark age. The catalyst for the modern wave of research into artificial neural networks is arguably the work presented in Hinton et al. (2006), wherein the authors showed how to train a generative model known as a deep belief network.

As presented in Hinton et al. (2006), a deep belief network is trained in two steps. The first step is to train the bilayers of the network one after the other – a procedure known as “pre-training” – and the second step is to apply the wake-sleep algorithm

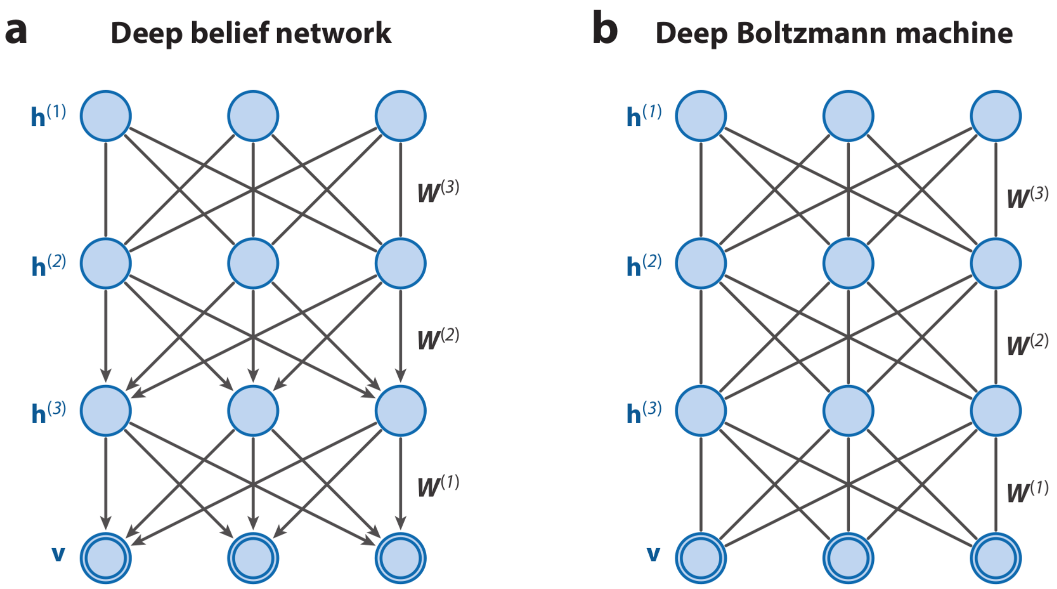

Figure 1.1: Graphical depictions of two forms of deep neural network capable of inference over the latent variables given observations. The above image was originally published in Salakhutdinov (2015). The observable variables in these models are indicated by $\mathbf{v}$, and the latent variables by $\mathbf{h}^{(i)}$, and the matrices $\mathbf{W}^{(i)}$ are the parameters of the model.

(Hinton et al., 1995) to train the network as a whole. Another generative model from the deep learning literature is the deep Boltzmann machine (Salakhutdinov and Hinton, 2009, 2012), which is represented by a fully undirected graph, in contrast to the hybrid graphical structure of the deep belief network (see figure 1.1).

Instead of applying the wake-sleep algorithm, a pre-trained network can instead be turned into a multilayer perceptron and trained with backpropagation (Rumelhart et al., 1986) to achieve state-of-the-art results on discriminative modelling problems. The resulting model, however, is no longer a generative model, and it cannot support Bayesian inference of the latent variables given observations. In spite of this, discriminative modelling currently dominates deep learning research, whereas deep generative models and inference of latent variables have diminished in relative importance. Nevertheless, some of the founding figures of deep learning expect generative modelling to return to prominence in the future (LeCun et al., 2015).

In the aforementioned generative models, inference is optimized by optimizing the parameters of the generative model – in so far as the generative model better explains the observations, the posterior over the latent variables given an observation should be more accurate as well. This focus on model learning in the deep learning community has resulted in algorithms which can successfully train highly complex models. The limitation of these models, however, is that they only support inferring the latent state given single observations – even in the case of temporal models, the corresponding sequence of observations must have a particular structure. In contrast, the models of computational neuroscience try to account for how animals can implement Bayesian inference in a more adaptive manner.

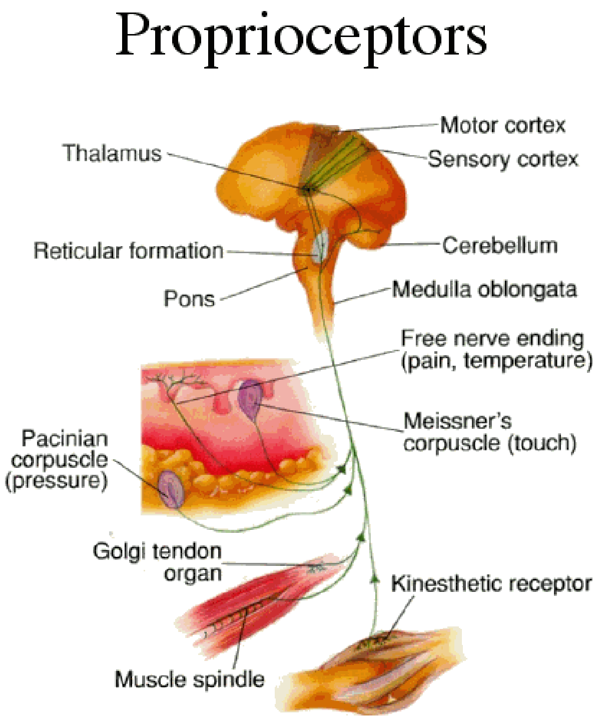
Proprioceptors
Motor cortex
Thalamus
Sensory cortex
Reticular formation
Cerebellum
Pons
Medulla oblongata
Free nerve ending (pain, temperature)
Meissner's corpuscle (touch)
Pacinian corpuscle (pressure)
Golgi tendon organ
Kinesthetic receptor
Muscle spindle

Figure 1.2: Proprioception is an example dynamic perception, and by extension Bayesian filtering. Proprioceptors provide observations of the state of the body to the brain. The cerebellum and motor cortex compute predictions of the bodies state based on previous beliefs. The observations and predictions are then combined with Bayes' rule to compute the posterior. The above image was copied from the slides available online at `https://msu.edu/course/kin/810/slds3.htm/`.

1.2 Inference in Computational Neuroscience

The Bayesian brain hypothesis is that the brain represents beliefs about the world with probability distributions, and updates these beliefs with Bayesian inference (Knill and Pouget, 2004; Doya, 2007). Bayesian approaches to modelling the brain are becoming increasingly popular at all levels of neuroscience, from neurons (Coen-Cagli et al., 2015) and neural populations (Funamizu et al., 2016) to behaviour (Ernst and Banks, 2002; Fischer and Peña, 2011). Similarly, neural circuit models of how animals implement Bayesian inference are a major part of computational neuroscience. These models can implement Bayes' rule given observations from multiple sensory modalities (Ma et al., 2006), and can implement Bayesian filtering (see figure 1.2) for sequences of observations of dynamic stimuli (Beck et al., 2011; Susemihl et al., 2014; Sokoloski, 2017).

A prior is conjugate to a posterior if the prior and posterior have the same parametric form, and the theoretical neural circuits of computational neuroscience often rely on conjugate priors for implementing Bayesian inference. If we think of Bayesian inference as the evaluation of a function which takes a prior as input and returns a posterior as output, then conjugate priors allow us to recursively apply this function to implement inference given multiple observations. Moreover, if we are given a set of such functions which implement different kinds of inference yet have inputs and outputs with identical forms – for example functions which implement inference for different sensory modalities, or which compute predictions for Bayesian filtering –

then we can compose these functions to form complex circuits for Bayesian inference.

In contrast with deep learning, learning the parameters of these neural circuits, and thereby learning to infer the latent state, has not received much attention in computational neuroscience. Nevertheless, the methods for learning and inference from these fields are naturally complimentary, and map readily onto an intuitive picture of the brain. On one hand, beliefs are encoded in the activity of neurons, and inference occurs on short timescales as a result of neural firing dynamics (Ma et al., 2006; Beck et al., 2011). On the other hand, parameters of the generative model are realized by neural connectivity, and learning is implemented with synaptic plasticity (Friston, 2003, 2010).

1.3 Dissertation Outline

The approach I take in this dissertation is to combine the tools of deep learning for training complex graphical models, with the circuits from computational neuroscience for implementing complex inference. I then analyze, generalize, and expand these theories toward developing a general theory of Bayesian inference with neural networks. I validate this theory with a large number of simulations. Moreover, I apply this theory to model biological neural circuits, and I find that I am able to explain several phenomena widely observed in the computational neural science literature.

Harmonium Families

In chapter 3 I develop harmoniums and harmonium families. Harmoniums are collections of random variables defined intrinsically by the fact that the conditional distribution of each random variable given all the others is always an element of the same exponential family. This approach to intrinsically defining a class of graphical model began with the work of Besag (1974), and the proof strategy developed in Besag (1974) has recently been extended in the work of Yang et al. (2013, 2015); Tansey et al. (2015) to include nth-order harmoniums.

In my dissertation I adopt the strategy of Arnold and Press (1989) to provide a more general derivation of the theory of nth-order harmoniums. This strategy emphasizes the recursive structure of harmoniums, in the sense that a harmonium composed of many random variables can be analyzed at finer and coarser levels of detail. In particular, I show that an n-th order harmonium family can be expressed as a second-order harmonium family of harmonium families, and this result is fundamental to my development of the theory of rectification.

Harmonium Rectification

In chapter 4 I develop the theory of rectification. To the best of my knowledge, the theory of rectification has no precedent in the deep learning literature, and only simple forms of it have been studied in the computational neuroscience literature. The concept of rectified harmoniums is inspired by a common feature of many neural population models from computational neuroscience, in which the sum of the tuning curves of the population is independent of the stimulus. This assumption ensures that there exists conjugate priors, which allows the posterior given a population response to be trivially calculated (Ma et al., 2006).

I generalize this property in two ways. Firstly, I generalize it from tuning curves which are independent of the stimulus, to tuning curves which depend on the stimulus through an exponential family sufficient statistic. Secondly, I generalize it from population code models to the general exponential family structure of the harmonium. I refer to harmoniums which satisfy this generalized constraint as rectified, and I show that rectified harmoniums support conjugate priors and can be trivially sampled. These properties suggest that rectified harmoniums are especially well-suited to supporting the dual tasks of inference and learning.

After deriving the general features of rectified harmoniums, I provide examples of classes of rectified harmoniums. I also extended this analysis to deep harmoniums, and find that when a deep harmonium is rectified at every level of its hierarchy, it inherits the computational advantages of rectification. Finally, I calculate derivatives and develop algorithms for rectified harmoniums and rectified deep harmoniums which allow them to be fit to data, while ensuring that they remain approximately rectified.

Bayesian Inference with Artificial Neural Networks

In chapter 5 I begin to develop the concept of implementation, by which a function implements some aspect of Bayesian inference. I then demonstrate how to encode the parameters of an exponential family distribution in an alternative space. This result provides a simple way of modelling how a population of neurons can encode the parameters of a probability distribution, and allows me to unify a number of approaches on Bayesian inference and filtering in biological (Ma et al., 2006; Beck et al., 2011) and artificial neural networks (Sutskever et al., 2009; Boulanger-Lewandowski et al., 2012).

I then show how to express multilayer perceptrons using the language of exponential families, and use them to construct two forms of recurrent neural network which implement Bayesian filtering. These filters are essentially extensions of the work of Sutskever et al. (2009) and Boulanger-Lewandowski et al. (2012), respectively. However, in those models, only single-layer neural networks were used, and so they lack the requisite complexity to support the established theory of universal approximation in multilayer perceptrons (Hornik, 1993), and are thus limited in their ability to implement Bayesian filtering. Moreover, the model of Boulanger-Lewandowski et al. (2012) fails to satisfy the requisite conditions for ensuring that Bayes' rule is optimally implemented by the network, and therefore tends to lose some of the information provided by the observations over time. I show that the neural networks which I propose avoid these problems.

Bayesian Inference with Biological Neural Networks

In chapter 6 I connect harmoniums and rectification with models from the computational neuroscience literature. I begin by reviewing linear-nonlinear neurons and linear probabilistic population codes (LPPCs). I then show how these models can be described as forms of harmonium, and find that LPPCs with Poisson neurons have several unique properties. In particular, I show that LPPCs with tuning curves that densely cover the stimulus space can always be rectified by modulating the gain of the tuning curves.

Simulations

In chapter 7 I introduce my library for the programming language Haskell, which I refer to as the Geometric Optimization Libraries (Goal). In my opinion (which is shared by many in the Haskell community) Haskell has the potential to provide an excellent and type-safe interface to numerical algorithms, but the right abstractions for describing these algorithms must first be found. Goal is my attempt to develop a type-safe framework for implementing numerical optimization algorithms by drawing inspiration from differential geometry. Nearly all of the figures which I present in this dissertation are generated in Goal, and so this dissertation serves as a demonstration of the effectiveness of my libraries.

I then present simulations to validate some of the algorithms which I develop in previous chapters. I begin with a simple simulation of training and rectifying restricted Boltzmann machines (RBMs) on the MNIST dataset, and find that my rectification-based algorithm results in an RBM which is both approximately rectified and a good model of the data. I then move on to modelling a number of neural circuits in the brain with neural Bayes filters. Each of these models is successful from the standpoint of optimization, but my model of proprioception proves especially effective as a model of the phenomenon itself.

In particular, my neural Bayes filter model of proprioception is trained to optimize its beliefs about the two-dimensional stimulus of joint angle and velocity. After training, the hidden layer of the EFMLP of the Bayes filter learns distinct tuning curves over the two variables. These tuning curves have the structure of gain-fields (Salinas and Thier, 2000), in the sense that they interact multiplicatively. Empirical work has indeed found that gain-fields exist in proprioceptive neurons in the cerebellum (Herzfeld et al., 2015), suggesting that my approach to modelling embodied perception in the brain captures some of the fundamental structure of neural computation.

Conclusion

The theoretical and empirical work of my dissertation demonstrates that embodied agents can implement Bayesian inference with neural networks. In chapter 8 I use these results to argue that embodied agents *should* implement Bayesian inference with neural networks. That is, I argue that if we lay out the necessary conditions for something to be an embodied agent, and build the most general theory we can of how such a system can efficiently solve the problems of embodied cognition, then we end up deriving something, a priori, which looks rather like a brain. I conclude my dissertation with a discussion of several of the ways I hope to extend my work in the future.

1.4 Related Work

In this section I highlight some of the literature which is related to my dissertation. This literature is a mixture of methods which compete with my own and theories which address important topics that I do not address in this dissertation. The reviewed material can be broken down into work on embodied cognition, work on machine learning, and work on computational neuroscience.

My focus in this dissertation is on embodied perception, however there are at least two other fundamental aspects of embodied cognition that I do not consider, namely action and morphology. Action, as formalized by control theory and reinforcement learning, closes the loop of the cognitive agent with the world. It is a vast topic, and there is no use in me trying to review it here; recent work, however, on casting optimal control as an inference problem is worth highlighting (Toussaint, 2009; Toussaint and Goerick, 2010; Friston et al., 2010; Rawlik et al., 2012). In particular, work on the framework known as linearly solvable MDPs or KL-control (Todorov, 2010; Theodorou and Todorov, 2012) allows control problems to be solved as graphical model inference problems (Kappen et al., 2012), and can be applied to solving control problems over latent variables (Matsubara et al., 2014). The effectiveness of the control-as-inference approach suggests that the work in this dissertation could be directly applied to solving control problems.

The importance of morphology to embodied cognition is exemplified by the passive dynamic walker, which exhibits natural walking down an incline without any actuators (McGeer, 1990). In general, morphological computation is the theory of how the form of the body of the embodied agent can help solve the problems of embodied cognition (Pfeifer and Gómez, 2009). Optimizing morphological computation involves matching the agent and its morphological and computational resources to the tasks that it must solve, and in recent years has become an active area of research (Zahedi and Ay, 2013; Montúfar et al., 2015). I believe that in the combination of these approaches of control as inference, morphological computation, and the approach to embodied perception that I develop in this dissertation, lies the beginning of a general theory of embodied cognition.

In deep learning there is a significant body of work on learning the parameters of neural networks with Bayesian methods (Neal, 2012; Ghahramani, 2015). These methods are applied to improve the learning of the parameters of the neural network, rather than inferring latent states. As such, in spite of involving Bayesian inference and neural networks, this work has no direct relationship with the work I present in this dissertation. At the same time, there is no reason these methods for improving learning cannot be applied to the deep models that I consider.

Although I presented the claim that generative models have take a secondary role in the deep learning community, there is still a large amount of work being done on this topic. A great deal of work has recently been done on understanding the representational power of different classes of generative models (Le Roux and Bengio, 2008; Montúfar and Ay, 2011; Montúfar and Rauh, 2016). Of particular relevance to this dissertation is the theory presented in Montúfar and Morton (2015a) on "Kronecker Product Models", which overlap with the harmonium families I develop in chapter 3. Furthermore, as an alternative to the established graphical model approaches, generative adversarial networks (Goodfellow et al., 2014; Radford et al., 2015) are a promising new approach to generative models which avoid some of the pitfalls of graphical models.

The methods I develop in section 5.2 for approximate Bayesian filtering are related to previous work on approximate filtering with restricted Boltzmann machines (Sutskever et al., 2009; Boulanger-Lewandowski et al., 2012). The work I present is nearly a special case of these models, yet both are deficient (in my estimation) because they fail to distinguish the predictions and posteriors of Bayesian filtering. On one hand, the RTRBM presented in Sutskever et al. (2009) has a fairly strict recurrent structure, which is too limited to approximate general prediction functions. On the other hand, the RNN-RBM model presented in Boulanger-Lewandowski et al. (2012) generalizes the RTRBM, but then loses the ability to exactly implement Bayes' rule. In the case of my model, I present conditions under which the predictions and posteriors of a Bayes filter can be exactly implemented, allowing me to describe optimal filters from the computational neuroscience literature (Beck and Pouget, 2007; Beck et al., 2011) as special cases of my general model.

Another interesting model for Bayesian filtering from the computational neuroscience literature is the recurrent exponential family harmonium (rEFH) presented in Makin et al. (2015). It has a markedly different structure from the aforementioned models, and ultimately learns an undirected dynamic model, rather then the directed, hidden Markov model-based approaches. This affords the rEFH interesting properties (see Makin et al., 2015, 2016), such as the ability to generate samples from the latent space backwards through time. On the other hand, this limits the complexity of the recurrent connectivity, and thereby also limits its generality in modelling Bayes filters.

In section 6.3 I show how to embed prior distributions in an LPPC by modulating the gain of the tuning curves. Similarly, both of the models developed by Ganguli and Simoncelli (2014) and Wei and Stocker (2015) can embed prior distributions in a neural population, and in their models they also consider how to modulate the location and scale of the tuning curves. These models, however, are restricted to 1-dimensional stimuli. In my approach, I can embed prior distributions over more or less arbitrary stimuli into the neural population.

Bayesian filtering techniques are typically applied to discrete-time problems, but especially in the context of neuroscience, continuous-time filtering is an important area of study. The brain can be understood to a reasonable degree as a massively parallel collection of doubly stochastic Poisson processes, and filtering on doubly stochastic Poisson processes was formally solved in (Snyder, 1972). Beck et al. (2011) provide an optimal solution to this problem for linear dynamical systems, and more recent advances have focused on how to approximate the (typically intractable) solutions of more general systems (Susemihl et al., 2013; Harel et al., 2015; Cseke et al., 2016). Although I do not consider continuous-time systems in this dissertation, in future work I hope to address how to train a point process model to learn to implement Bayesian filtering.

Chapter 2

Mathematical Background

In this chapter I present the necessary mathematics for understanding this dissertation. There is no new material in this chapter, and readers familiar with some or all of these topics may skip them without concern that they will miss details from my central argument. Nevertheless, I have done my best to make these reviews concise and enjoyable for even the experienced reader, and in any case, this chapter also serves as an introduction to the notation that I use throughout the dissertation.

The primary references for this chapter are Thrun et al. (2005); Athreya and Lahiri (2006); Amari and Nagaoka (2007); Wainwright and Jordan (2008); Kollar and Friedman (2009). In particular, section 2.1 is drawn from material in Athreya and Lahiri (2006); section 2.2 is drawn from material in Amari and Nagaoka (2007); section 2.3 is drawn from material in Amari and Nagaoka (2007); Wainwright and Jordan (2008); section 2.4 is drawn from material in Wainwright and Jordan (2008); Kollar and Friedman (2009); section 2.5 is drawn from material in Thrun et al. (2005); Meyn and Tweedie (2009); and section 2.6 is drawn from material in Thrun et al. (2005); Särkkä (2013).

2.1 Measure Theory

In this section I provide a quick sketch of basic measure theory. I do not provide a detailed account of all the properties of the various objects I introduce, which I may nevertheless use without remark over the course of this dissertation. For a thorough introduction to measure and probability theory, see Athreya and Lahiri (2006).

Let Ω be a set. A σ-algebra $\mathcal{F}$ on Ω is a set of subsets of Ω which includes the empty set $\emptyset$, is closed under complement, and is closed under countable unions. If $\mathcal{F}$ is a σ-algebra on Ω, then the pair $(\Omega, \mathcal{F})$ is known as a measurable space. If $(\Omega, \mathcal{F})$ and $(\Omega_f, \mathcal{F}_f)$, are measurable spaces, then $f\colon (\Omega, \mathcal{F}) \to (\Omega_f, \mathcal{F}_f)$ is a measurable function if for every $B \in \mathcal{F}_f$, the preimage $f^{-1}(B) \in \mathcal{F}$.

Given a measurable space $(\Omega, \mathcal{F})$, a measure μ on $\mathcal{F}$ is a non-negative function $\mu\colon \mathcal{F} \to \mathbb{R}^+$ which is 0 at $\emptyset$, and which is additive over disjoint unions of elements of $\mathcal{F}$, such that if the sets $A \in \mathcal{F}$ and $B \in \mathcal{F}$ are disjoint, then $\mu(A \cup B) = \mu(A) + \mu(B)$. Taken together, the triple $(\Omega, \mathcal{F}, \mu)$ is known as a measure space. Given the measure space $(\Omega, \mathcal{F}, \mu)$ and measurable space $(\Omega_f, \mathcal{F}_f)$, and given the measurable function $f\colon (\Omega, \mathcal{F}) \to (\Omega_f, \mathcal{F}_f)$, we denote the Lebesgue integral of f with respect to μ over the set $A \in \mathcal{F}$ by $\int_A f d\mu$, and we define the push-forward measure $\mu_f\colon \mathcal{F}_f \to \mathbb{R}^+$ as the function $\mu_f(B) \mapsto \mu(f^{-1}(B))$.

Where $(\Omega, \mathcal{F}, \mu)$ and $(\Omega, \mathcal{F}, \nu)$ are measure spaces, the Radon-Nikodym derivative of μ with respect to ν is the function $\frac{d\mu}{d\nu} : \Omega \to \mathbb{R}^+$ defined as the solution of the equation

$$\mu(A) = \int_A \frac{d\mu}{d\nu} d\nu, \forall A \in \mathcal{F}. \tag{2.1}$$

The Radon-Nikodym derivative allows us to understand a measure in terms of a function of elements of Ω rather than $\mathcal{F}$. In order to establish the existence of the derivative, we make use of the property known as absolute continuity.

Definition 2.1 (Absolute Continuity)**.** A measure μ is absolutely continuous with respect to ν, which we denote $\mu \ll \nu$, if and only if

$$\nu(A) = 0 \Rightarrow \mu(A) = 0, \forall A \in \mathcal{F}.$$

Theorem 2.1 (Radon-Nikodym Theorem)**.** *Given measures μ and ν, The Radon-Nikodym derivative $\frac{d\mu}{d\nu}$ exists if and only if $\mu \ll \nu$.*

Proof. Theorem 4.1.1, Athreya and Lahiri (2006). □

Given a measure space $(\Omega, \mathcal{F}, P)$, the measure P is a probability measure if $P(\Omega) = 1$. In this case, $(\Omega, \mathcal{F}, P)$ is known as a probability space, Ω is known as a sample space, and elements of $\mathcal{F}$ are known as events. For a given probability measure P, there is often a standard measure μ by which we form the Radon-Nikodym derivative $\frac{dP}{d\mu}$. In this case we refer to μ as the base measure and $\frac{dP}{d\mu}$ as the density of P, and we denote this density by p.

A measurable function $X\colon (\Omega, \mathcal{F}) \to (\Omega_X, \mathcal{F}_X)$ on a probability space is known as a random variable, and the codomain of the random variable Ω_X is known as the state space of that random variable. Given the random variable X, we refer to the push-forward measure P_X as the probability distribution of X, and the density p_X as the probability density of X. Finally, given a measurable function $f\colon \Omega_X \to \Omega_f$, we refer to the integral $\int_\Omega f dP_X$ as the expected value of $f(X)$, and denote it by $\mathbb{E}_P[f(X)]$.

Conditional dependence and statistical independence are relationships between random variables which are specific to probability theory. I define two random variables X and Y on $(\Omega, \mathcal{F}, P)$ as independent if $P_{XY} = P_X \cdot P_Y$. The general definition of conditional distributions is fairly involved, however when the distributions of the random variables in question are absolutely continuous with respect to some base measure, we may express it in a straightforward manner. This is sufficient for the purposes of this dissertation, and I therefore define the conditional density of X given Y as $p_{X|Y} = \frac{p_{XY}}{p_Y}$.

When working with random variables and conditional distributions at a high level, it is helpful to use notational conventions in order to avoid excessive reference to the realizations $\omega \in \Omega$. In this vein, the notation $X \in \Omega_X$ is shorthand for $X(\omega) \in \Omega_X, \forall \omega \in \Omega$. Similarly, although conditional distributions are not probability distributions until evaluated at particular conditions, where $\mathcal{M}$ is a set of probability distributions on the measurable space $(\Omega_X, \mathcal{F}_X)$, I write $P_{X|Y} \in \mathcal{M}$ to indicate that $P_{X|Y=Y(\omega)} \in \mathcal{M}, \forall \omega \in \Omega$. Finally, when considering iterated expectations involving multiple probability measures, I write $\mathbb{E}_P[\mathbb{E}_Q[f(X, Y) \mid Y]]$ as short hand for expressions of the form $\int_{\Omega_Y} \int_{\Omega_X} f(\mathbf{x}, \mathbf{y}) q_{X|Y}(\mathbf{x} \mid \mathbf{y}) \mu(d\mathbf{x}) P_Y(d\mathbf{y})$.

2.2 Statistical Models

Suppose that $(\Omega, \mathcal{F}, P)$ is a probability space, and let us define Λ as the set of all probability measures on $(\Omega, \mathcal{F})$. Given a d-dimensional set of parameters $\Theta \subset \mathbb{R}^d$, a statistical model is a function $Q\colon \Theta \to \Lambda$ (McCullagh, 2002). In statistics, "to train a model" or "to fit a model" or "to learn a model" is to find the parameters $\boldsymbol{\theta} \in \Theta$ for which $Q(\boldsymbol{\theta})$ is most similar to P. In this dissertation I use the letter Q exclusively to denote statistical models, and for the sake of notational implicitly, I typically suppress the dependence of Q on $\boldsymbol{\theta}$. That is, I use $Q \in \Lambda$ to denote a probability measure at some implicit set of parameters $\boldsymbol{\theta} \in \Theta$, and where necessary, I write $Q(A; \boldsymbol{\theta})$ to denote the measure $Q(\boldsymbol{\theta})$ of the event $A \in \mathcal{F}$.

In order to train a model, we require a measure of similarity. The relative entropy is the function

$$\begin{aligned} D\colon \Lambda \times \Lambda &\to \mathbb{R}^+ \\ (P \parallel Q) &\mapsto \int_\Omega \log \frac{dP}{dQ} dP. \end{aligned}$$

The relative entropy tells us how many nats of information are lost when Q is used to approximate P, and is in many ways the most fundamental measure of the similarity between two probability distributions (Kullback and Leibler, 1951; Shore and Johnson, 1980; Amari and Nagaoka, 2007). Regardless of the form of the distributions in question, it has the properties that we would like to associate with a similarity measure.

Proposition 2.2 (Gibbs' Inequality). *The relative entropy is non-negative, and is 0 if and only if its arguments are almost surely equal.*

Proof. Note that the second derivative of the function $f(x) \mapsto x \log x$ is $1/x$, and therefore that f is convex on $\mathbb{R}^+$. Since Radon-Nikodym derivatives are non-negative, we may apply Jensen's inequality (Athreya and Lahiri, 2006) to show that

$$\begin{aligned} \int_\Omega \log \frac{dP}{dQ} dP &= \int_\Omega \log \frac{dP}{dQ} \cdot \frac{dP}{dQ} dQ \\ &= \int_\Omega f \circ \frac{dP}{dQ} dQ \\ &\geq f(\int_\Omega \frac{dP}{dQ} dQ) \\ &= f(1) = 0. \end{aligned}$$

For the case of equality, note that $P \overset{a.s.}{=} Q$ implies that the solution of equation 2.1 for P and Q is $\frac{dP}{dQ} = 1$, which implies that $\int_\Omega \log \frac{dP}{dQ} dP = 0$. Conversely, since Jensen's inequality also implies that $\int_A \log \frac{dP}{dQ} dP \geq 0$ for any $A \in \mathcal{F}$, the linearity of integrals implies that if $\int_\Omega \log \frac{dP}{dQ} dP = 0$ then $\log \frac{dP}{dQ} \overset{a.s.}{=} 0$, which implies that $\frac{dP}{dQ} \overset{a.s.}{=} 1$, which implies that $P \overset{a.s.}{=} Q$. ☐

We often require an absolute as opposed to relative measure of uncertainty. Given some reference measure μ where $P \ll \mu$, we define the entropy as

$$\begin{aligned} H\colon \Lambda &\to \mathbb{R} \\ P &\mapsto -\int_\Omega \log \frac{dP}{d\mu} dP. \end{aligned} \tag{2.2}$$

The Shannon entropy is the entropy of a distribution over a discrete set. The Shannon entropy also has the property that it is non-negative, and forms the basis of many of our intuitions about information theory. In particular, the Shannon entropy can be viewed as the average optimal code length for some distribution of digital messages.

Given a reference measure μ where $Q \ll \mu$, we define the cross-entropy as

$$\begin{aligned} H\colon \Lambda \times \Lambda &\to \mathbb{R} \\ (P, Q) &\mapsto -\int_\Omega \log \frac{dQ}{d\mu} dP. \end{aligned} \tag{2.3}$$

In the discrete case, the cross entropy measures the average code length when the non-optimal distribution Q is used to encode messages distributed according to P.

The relative entropy, entropy, and cross-entropy are all intimately related. If $P \ll \mu$, $Q \ll \mu$, and $\mu \ll Q$, we may rewrite the relative entropy as

$$\begin{aligned} D(P \parallel Q) &= \int_\Omega \log \frac{dP}{dQ} dP \\ &= \int_\Omega \log(\frac{dP}{d\mu} \cdot \frac{d\mu}{dQ}) dP \\ &= \int_\Omega \log \frac{dP}{d\mu} dP - \int_\Omega \log \frac{dQ}{d\mu} dP \\ &= H(P, Q) - H(P). \end{aligned} \tag{2.4}$$

Since the entropy in this equation does not depend on Q, when P is fixed, minimizing the cross-entropy with respect to Q is equivalent to minimizing the relative entropy with respect to Q.

Let us now consider the random variable $X\colon (\Omega, \mathcal{F}) \to (\Omega_X, \mathcal{F}_X)$, and suppose that Q_X is a statistical model on $(\Omega_X, \mathcal{F}_X)$ parameterized by $\Theta \subset \mathbb{R}^d$. Since P is independent of the parameters Θ, we may minimize the relative entropy $D(P_X \parallel Q_X)$ – and thereby train Q_X – by gradient descent of the cross-entropy $\nabla_{\boldsymbol{\theta}} H(P_X, Q_X)$ with respect to the parameters $\boldsymbol{\theta} \in \Theta$. Where μ is the base measure of P_X and Q_X, the gradient may be expressed as

$$\nabla_{\boldsymbol{\theta}} D(P_X \parallel Q_X) = -\int_\Omega \nabla_{\boldsymbol{\theta}} \log q_X dP_X = -\mathbb{E}_P[\nabla_{\boldsymbol{\theta}} \log q_X(X)]. \tag{2.5}$$

Computing this gradient directly is often infeasible, and in any case, we often only have samples from P_X, rather than knowledge of the exact distribution itself. Given a set of n independent random variables $(X_i)_{i=1}^n$, where $P_{X_i} \overset{a.s.}{=} P_X$, let us therefore consider the empirical measure

$$\begin{aligned} P_X^n \colon \mathcal{F}_X &\to [0, 1] \\ A &\mapsto \frac{1}{n} \sum_{i=1}^n \delta_{X_i}(A) \end{aligned}$$

where δ_{X_i} is the Dirac measure at X_i. If we take the empirical measure to be our target distribution, then

$$\nabla_{\boldsymbol{\theta}} D(P_X^n \parallel Q_X) = -\frac{1}{n} \sum_{i=1}^n \nabla_{\boldsymbol{\theta}} \log q_X(X_i).$$

Since the law of large numbers implies that

$$\lim_{n\to\infty} -\frac{1}{n}\sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log q_X(X_i) = -\mathbb{E}_P[\nabla_{\boldsymbol{\theta}} \log q_X(X)],$$

we may approximate the true relative entropy gradient arbitrary well with samples from P_X. Minimizing the entropy of P_X relative to Q_X in this way is equivalent to maximum likelihood estimation.

Especially in the dynamic, embodied setting, we often wish to minimize the relative entropy with respect to one sample at a time rather than a complete set of samples. However, performing a complete minimization with respect to a single observation would ignore the prior history of learning, and so we instead take a small step down the gradient after every sample. That is, we assume an initial set of parameters $\boldsymbol{\theta}_0$, and then at every time $k+1$ we update our parameters with the equation

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \epsilon_k \nabla_{\boldsymbol{\theta}} \log q_X(X_{k+1}), \tag{2.6}$$

where ϵ_k is referred to as the step size or learning rate. This algorithm is known as stochastic gradient descent.

Stochastic gradient descent can be trivially motivated by considering the step sizes $\epsilon_k = \epsilon_0/n$, where ϵ_0 is an arbitrary constant. This implies that

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_0 - \frac{\epsilon_0}{n}\sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \log q_X(X_i) \approx \boldsymbol{\theta}_0 - \epsilon_0 \mathbb{E}_P[\nabla_{\boldsymbol{\theta}} \log q_X(X)],$$

which is simply a single step down the empirical gradient with step size ϵ_0. Of course, we usually do not wish to bind our step size to the number of samples, and what decades of analysis and experiment have shown is that if ϵ_k is chosen well, convergence can proceed much more quickly (Bottou, 2010).

Another class of algorithms for improving the rate of convergence of stochastic gradient descent are momentum algorithms. Imagine the parameters of a model being optimized as a point rolling down a surface due to gravity. In this context, the intuition behind momentum algorithms is to better model the effects of gravity by approximating second-order equations of motion. The classic momentum algorithm involves computing the momentum term

$$\boldsymbol{\beta}_{k+1} = \upsilon_k \boldsymbol{\beta}_k - \epsilon_k \nabla_{\boldsymbol{\theta}} \log q_X(X_{k+1}), \tag{2.7}$$

where $0 \leq \upsilon_k < 1$ determines the length of the "memory" of momentum. Given the momenta, the stochastic gradient descent procedure in equation 2.6 is replaced with

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \boldsymbol{\beta}_{k+1}.$$

There are many methods for choosing υ_k, and many more complicated forms of momentum-based algorithms. The Adam algorithm (Kingma and Ba, 2014), in particular, is the one which I make extensive use of in this dissertation.

Gradient descent algorithms are intuitively motivated as following the direction of steepest descent of the error in parameter space, but computing the direction of steepest descent formally requires knowing the geometry of what the parameter space describes. If the model Q_X is a differentiable, bijective function between Θ

and $\mathcal{M} = Q_X(\Theta)$, then $\mathcal{M}$ is a kind of differentiable manifold known as a statistical manifold. The metric tensor for statistical manifolds is the Fisher information metric (Amari and Nagaoka, 2007). At a given point $\boldsymbol{\theta}$, the metric tensor in Θ-coordinates is the matrix $\mathbf{G}_{\boldsymbol{\theta}} \in T^*_{\boldsymbol{\theta}}\mathcal{M} \otimes T^*_{\boldsymbol{\theta}}\mathcal{M}$ given by

$$g_{\boldsymbol{\theta},i,j} = \mathbb{E}_Q[\partial_{\theta_i}\partial_{\theta_j} \log q_X(X)],$$

where $T^*_{\boldsymbol{\theta}}\mathcal{M}$ is the cotangent space at $q_X(\boldsymbol{\theta}) \in \mathcal{M}$.

Accounting for the metric tensor, we may then write our stochastic gradient descent procedure as

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \epsilon_k \nabla_{\boldsymbol{\theta}} \log q_X(X_{k+1}) = \boldsymbol{\theta}_k - \epsilon_k \mathbf{G}^{-1}_{\boldsymbol{\theta}} \partial_{\boldsymbol{\theta}} \log q_X(X_{k+1}),$$

where $\partial_{\boldsymbol{\theta}}$ is the partial derivative operator. Unfortunately, computing the Fisher information metric is often very difficult. Moreover, since metric tensors are always positive semi-definite, the local minima of the gradient descent are not changed by the metric. As such, the metric is often ignored when computing the gradient, in the sense that we assume $\mathbf{G} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix. In the literature, the true gradient is often referred to as the natural gradient, and the approximate gradient is referred to as the vanilla gradient.

When the parameter space $\Theta \subset \mathbb{R}^d$ is bounded, an additional issue arises when simulating stochastic gradient descent. Taking discrete, approximate steps along the gradient in equation 2.6 or 2.7 runs the risk of inadvertently driving the simulated parameters out of the bounds of the parameter space. There is no straightforward solution to this problem, and therefore statistical manifolds with parameter spaces equal to a d-dimensional Euclidean space are very useful in practice[1].

2.3 Exponential Families

Suppose that $(\Omega, \mathcal{F}, P)$ is a probability space, $(\Omega_X, \mathcal{F}_X)$ is a measurable space, and $X\colon (\Omega, \mathcal{F}) \to (\Omega_X, \mathcal{F}_X)$ is a random variable. Let us refer to the measurable function $\mathbf{s}\colon (\Omega_X, \mathcal{F}_X) \to (\mathbb{R}^d, \mathbb{B})$ as the sufficient statistic, where $\mathbb{B}$ is the Borel σ-algebra, and let us define the random variables $(X_i)_{i=1}^n$ by $P_{X_i} \overset{a.s.}{=} P_X$. Suppose we wish to define a statistical model Q_X such that

$$\mathbb{E}_Q[\mathbf{s}(X)] = \sum_{i=1}^{n} \mathbf{s}(X_i), \tag{2.8}$$

and such that the entropy $H(Q_X)$ is maximized. These constraints ensure that Q_X is unbiased with respect to the true expectations of the sufficient statistic $\mathbb{E}_P[\mathbf{s}(X)]$, and that we otherwise build as few assumptions into our model as possible. Solving this variational problem results in an exponential family of distributions.

Theorem 2.3. *Given the base measure μ, the statistical model Q_X which maximizes the entropy (2.2) while satisfying constraint 2.8 has the form*

$$q_X(\mathbf{x}; \boldsymbol{\theta}) \propto e^{\boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x})}. \tag{2.9}$$

---

[1] See Bernigau (2015) for a review of and approach to this problem.

Proof. We begin by defining the Lagrangian

$$\mathcal{L}(q_X) = -\int_{\mathbb{X}} \log q_X \cdot q_X d\mu + \boldsymbol{\theta} \cdot (\int_{\mathbb{X}} \mathbf{s} \cdot q_X d\mu - \sum_{i=1}^{n} \mathbf{s}(X_i)) + \lambda(\int_{\mathbb{X}} q_X d\mu - 1),$$

where $\boldsymbol{\theta}$ and λ are the Lagrange multipliers. This Lagrangian incorporates constraint 2.8, the constraint on probability densities $\int_{\Omega_X} q_X d\mu = 1$, and the optimization goal of maximizing the entropy (2.2). By taking the functional derivative of $\mathcal{L}$ with respect to q_X, we may drop the constants in the Lagrangian, pull all terms into a single integral, and thereby express the derivative as

$$\begin{aligned} \frac{\delta \mathcal{L}(q_X)}{\delta q_X} &= \frac{\delta}{\delta q_X} \int_{\mathbb{X}} - \log q_X \cdot q_X + \boldsymbol{\theta} \cdot \mathbf{s} \cdot q_X + \lambda q_X d\mu \\ &= \frac{\delta}{\delta q_X} \int_{\mathbb{X}} L(\mathbf{x}, q_X(\mathbf{x})) \mu(d\mathbf{x}), \end{aligned}$$

where

$$L(\mathbf{x}, q_X(\mathbf{x})) \mapsto -q_X(\mathbf{x}) \log q_X(\mathbf{x}) + q_X(\mathbf{x}) \boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x}) + \lambda q_X(\mathbf{x}).$$

Since the functional L does not depend on derivative of q_X, the Euler-Lagrange equation implies that

$$\frac{\delta \mathcal{L}(q_X)}{\delta q_X} = \frac{\partial L}{\partial q_X}.$$

In turn,

$$\frac{\partial L(\mathbf{x}, q_X(\mathbf{x}))}{\partial q_X(\mathbf{x})} = -\log q_X(\mathbf{x}) - 1 + \boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x}) + \lambda.$$

By setting the derivative to 0 and solving for $q_X(\mathbf{x})$, we find that

$$\begin{aligned} & 0 = -\log q_X(\mathbf{x}) - 1 + \boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x}) + \lambda \\ \iff & \log q_X(\mathbf{x}) = -1 + \boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x}) + \lambda \\ \iff & q_X(\mathbf{x}) = e^{-1 + \boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x}) + \lambda} \\ \implies & q_X(\mathbf{x}) \propto e^{\boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x})}. \end{aligned}$$

Note that to ensure that q_X is a indeed a probability density, q_X must also satisfy the inequality constraint $q_X(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \Omega_X$. Luckily, this is satisfied in general, so we can avoid introducing the constraint in our calculations. □

With the form of the maximum entropy density in hand, we may satisfy the constraint on probability distributions $\int_{\Omega_X} dQ_X = 1$ by writing the probability distribution parameterized by $\boldsymbol{\theta}$ as

$$Q_X(A) = \int_A q_X d\mu = \int_A e^{\boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x}) - \psi(\boldsymbol{\theta})} \mu(d\mathbf{x}), \forall A \in \Omega_X$$

where we define the normalization term as

$$\begin{aligned} \psi \colon \mathbb{R}^d &\to \mathbb{R} \cup \{\infty\} \\ \boldsymbol{\theta} &\mapsto \log \int_{\Omega_X} e^{\boldsymbol{\theta} \cdot \mathbf{s}(\mathbf{x})} \mu(d\mathbf{x}). \end{aligned}$$

Statistical models with densities of the form of equation 2.9 are known as exponential family models, and the d-dimensional vector of Lagrange multipliers $\boldsymbol{\theta}$ are known as the natural parameters of the distribution. The normalizer ψ is known as the log-partition function, and the natural parameter space $\Theta \subseteq \mathbb{R}^d$ is equal to the set of all $\boldsymbol{\theta}$ such that the log-partition function is finite. Finally, the set of all distributions $\mathcal{M} = Q_X(\Theta)$ is known as a d-dimensional exponential family on the measurable space $(\Omega_X, \mathcal{F}_X)$, and the elements of $\mathcal{M}$ are known as exponential family distributions.

In order to show how to satisfy constraint 2.8 as a function of the natural parameters, we develop the relationship between exponential families and convex analysis.

Proposition 2.4. *Suppose that* $(\Omega, \mathcal{F}, P)$ *is a probability space, that* $(\Omega_X, \mathcal{F}_X)$ *is a measurable space, and that* $X\colon (\Omega, \mathcal{F}) \to (\Omega_X, \mathcal{F}_X)$ *is a random variable. In addition, suppose that* $\mathcal{M}$ *is an exponential family on* $(\Omega_X, \mathcal{F}_X)$ *defined by the base measure* μ *and sufficient statistic* $\mathbf{s}$*, and that* $P_X \in \mathcal{M}$ *with parameters* $\boldsymbol{\theta}$*. Then:*

1. *The derivative of the log-partition function* $\partial_{\boldsymbol{\theta}}\psi(\boldsymbol{\theta}) = \mathbb{E}_P[\mathbf{s}(X)]$ *is equal to the expected value of the sufficient statistic.*

2. *The Hessian of the log-partition function is the covariance matrix of the sufficient statistic, such that* $\partial_{\theta_i\theta_j}\psi(\boldsymbol{\theta}) = \mathbb{E}_P[s_i(X) \cdot s_j(X)]$.

3. *The log-partition function* ψ *is convex.*

Proof. Proposition 3.1, Wainwright and Jordan (2008). □

Because ψ is a convex function, the theory of convex analysis allows us to deduce many interesting properties of exponential families (Boyd and Vandenberghe, 2004; Wainwright and Jordan, 2008). To begin, the convex conjugate ϕ of ψ is defined as

$$\begin{aligned} \phi\colon \mathbb{R}^d &\to \mathbb{R} \cup \{\infty\} \\ \boldsymbol{\eta} &\mapsto \max_{\boldsymbol{\theta}\in\mathbb{R}^d}\{\boldsymbol{\theta}\cdot\boldsymbol{\eta} - \psi(\boldsymbol{\theta})\}. \end{aligned} \tag{2.10}$$

One way to evaluate the maximum in this definition is to compute the derivatives $\partial_{\boldsymbol{\theta}}(\boldsymbol{\theta}\cdot\boldsymbol{\eta} - \psi(\boldsymbol{\theta}))$ and set the result to 0, and then solve $\boldsymbol{\eta} = \partial_{\boldsymbol{\theta}}\psi(\boldsymbol{\theta})$. Unfortunately, there may be many solutions to this equation, because $\partial_{\boldsymbol{\theta}}\psi$ in general is not invertible. We many amend this, however, with a more careful choice of sufficient statistic.

Definition 2.2 (Minimal Sufficient Statistic)**.** The d-dimensional sufficient statistic $\mathbf{s}\colon \Omega_X \to \mathbb{R}^d$ is minimal if there exists no constant α_0 and non-zero vector $\boldsymbol{\alpha}$ such that, for any $\mathbf{x} \in \Omega_X$,

$$\mathbf{s}(\mathbf{x}) \cdot \boldsymbol{\alpha} = \alpha_0.$$

Note that this definition precludes a component of a minimal sufficient statistic from being a constant function.

Corollary 2.5. *Suppose that* $\mathcal{M}$ *is an exponential family with sufficient statistic* $\mathbf{s}$ *and base measure* μ*. If* $\mathbf{s}$ *is minimal, then* ψ *is strictly convex.*

Proof. Proposition 3.1b, Wainwright and Jordan (2008). □

Proposition 2.6. *Let $\mathcal{M}$ be an exponential family with minimal sufficient statistic* $\mathbf{s}$. *Then the derivatives of the log-partition function $\partial_{\boldsymbol{\theta}}\psi$ are invertible and equal to the derivatives of the convex conjugate ϕ, such that $(\partial_{\boldsymbol{\theta}}\psi)^{-1} = \partial_{\boldsymbol{\eta}}\phi$.*

Proof. Since ψ is strictly convex, $\max_{\boldsymbol{\theta}\in\mathbb{R}^d}\{\boldsymbol{\theta}\cdot\boldsymbol{\eta} - \psi(\boldsymbol{\theta})\}$ has the unique solution

$$\begin{aligned} \boldsymbol{\eta} &= \partial_{\boldsymbol{\theta}}\psi(\boldsymbol{\theta}) \\ \iff \boldsymbol{\theta} &= (\partial_{\boldsymbol{\theta}}\psi)^{-1}(\boldsymbol{\eta}). \end{aligned}$$

By substituting this solution into the definition of the convex conjugate (2.10) we may conclude that

$$\begin{aligned} \partial_{\boldsymbol{\eta}}\phi(\boldsymbol{\eta}) &= \partial_{\boldsymbol{\eta}}((\partial_{\boldsymbol{\theta}}\psi)^{-1}(\boldsymbol{\eta})\cdot\boldsymbol{\eta} - \psi((\partial_{\boldsymbol{\theta}}\psi)^{-1}(\boldsymbol{\eta}))) \\ &= (\partial_{\boldsymbol{\theta}}\psi)^{-1}(\boldsymbol{\eta}) + \boldsymbol{\eta}\cdot\partial_{\boldsymbol{\eta}}(\partial_{\boldsymbol{\theta}}\psi)^{-1}(\boldsymbol{\eta}) - \boldsymbol{\eta}\cdot\partial_{\boldsymbol{\eta}}(\partial_{\boldsymbol{\theta}}\psi)^{-1}(\boldsymbol{\eta}) \\ &= (\partial_{\boldsymbol{\theta}}\psi)^{-1}(\boldsymbol{\eta}). \end{aligned}$$

□

For simplicity, when $\mathcal{M}$ is an exponential family with a minimal sufficient statistic, we denote the derivatives of ψ and ϕ by $\boldsymbol{\tau}(\boldsymbol{\theta}) = \partial_{\boldsymbol{\theta}}\psi(\boldsymbol{\theta})$ and $\boldsymbol{\tau}^*(\boldsymbol{\eta}) = \partial_{\boldsymbol{\eta}}\phi(\boldsymbol{\eta})$, and refer to them as the forward mapping and backward mapping, respectively.

In general, we refer to the image of the forward mapping $\mathrm{H} = \boldsymbol{\tau}(\Theta)$ as the mean parameter space and we denote the elements of H by $\boldsymbol{\eta}$[2]. When the sufficient statistic is minimal, the isomorphism between Θ and H defined by the derivatives is bijective, such that H provides an alternative parameterization of Q_X (figure 2.1). Therefore, if $\mathcal{M}$ is an exponential family with a minimal sufficient statistic, we may ensure that the distribution $Q_X \in \mathcal{M}$ with parameters $\boldsymbol{\theta}$ satisfies equation 2.8 by setting $\boldsymbol{\eta} = \sum_{i=1}^{n}\mathbf{s}(X_i)$, and setting $\boldsymbol{\theta} = \boldsymbol{\tau}^*(\boldsymbol{\eta})$.

The convex conjugate itself is more than an abstract construction, and affords an intuitive interpretation.

Theorem 2.7. *Suppose that $\mathcal{M}$ is an exponential family, and that $P_X \in \mathcal{M}$ with mean parameters $\boldsymbol{\eta} \in \mathrm{H}$, and natural parameters $\boldsymbol{\theta} \in \Theta$. Then:*

1. *$\phi(\boldsymbol{\eta})$ is equal to the negative entropy of P_X.*
2. *ψ is the convex conjugate of ϕ, such that*

$$\psi(\boldsymbol{\theta}) = \max_{\boldsymbol{\eta}\in H}\{\boldsymbol{\theta}\cdot\boldsymbol{\eta} - \phi(\boldsymbol{\eta})\}.$$

3. *The maximizer of $\boldsymbol{\theta}\cdot\boldsymbol{\eta} - \phi(\boldsymbol{\eta})$ is given by $\boldsymbol{\eta} = \partial_{\boldsymbol{\theta}}(\psi(\boldsymbol{\theta}))$.*

Proof. Theorem 3.4, Wainwright and Jordan (2008). □

Having characterized the relationship between entropy and exponential families, let us return to the other information theoretic quantities developed in the previous section, namely the cross-entropy and relative entropy. Where X is a random variable,

---

[2] This definition avoids boundary issues which can arise in alternative definitions of H. In general we assume that both Θ and H are open sets. For a thorough analysis of these issues, see Wainwright and Jordan (2008).

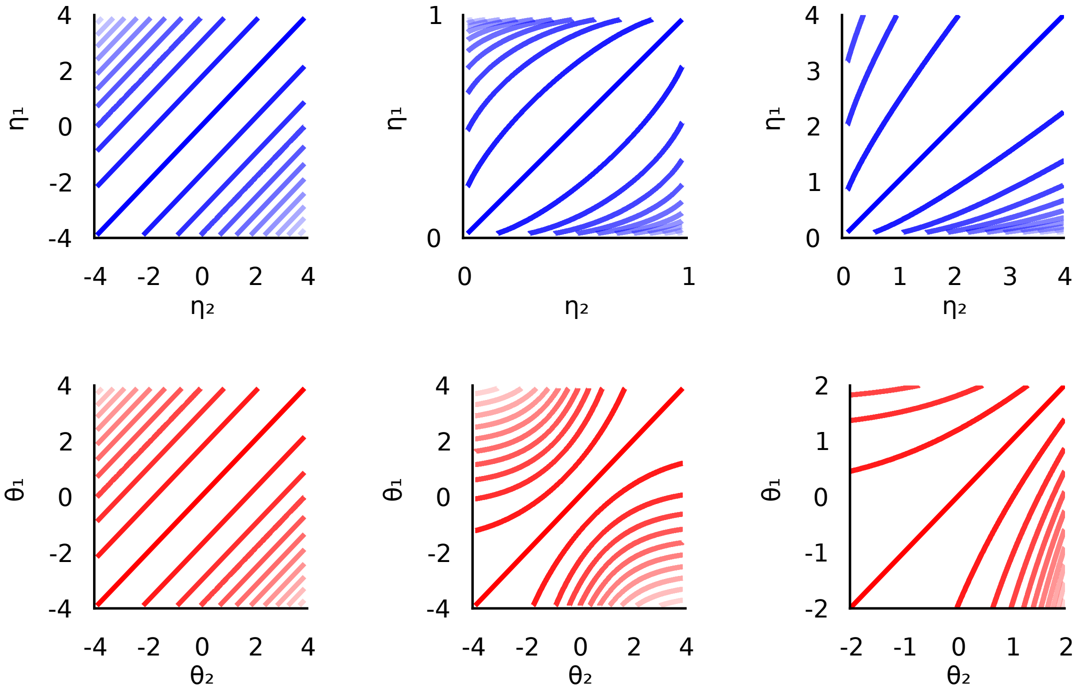

Figure 2.1: In this figure I present three pairs of plots of the isolines of the relative entropy between two elements of the same exponential family, in both mixture (blue) and natural coordinates (red). In each case the central diagonal line indicates a relative entropy of 0. *Left*: Here we consider the family of normal distributions with known variance $\sigma^2 = 1$. In this case the two coordinate systems are identical, and the relative entropy is simply the squared distance between the two parameters of the two distributions. The maximum relative entropy of the isolines is 28.9. *Middle*: Here we consider the family of Bernoulli distributions. The maximum relative entropy shown is 3.55 in mean coordinates, and 3.56 in natural coordinates. *Right*: Here we consider the family of Poisson distributions. The maximum relative entropy shown is 10.31 in mean coordinates, and 21.19 in natural coordinates.

P_X is its distribution, and Q_X is an exponential family model, the cross-entropy $H(P_X, Q_X)$ is given by

$$H(P_X, Q_X) = -\mathbb{E}_P[\log q_X(X)] = -\boldsymbol{\theta} \cdot \mathbb{E}_P[\mathbf{s}(X)] + \psi(\boldsymbol{\theta}). \tag{2.11}$$

The partial derivatives of the cross-entropy with respect to the natural parameters is then

$$\partial_{\boldsymbol{\theta}} H(P_X, Q_X) = \mathbb{E}_Q[\mathbf{s}(X)] - \mathbb{E}_P[\mathbf{s}(X)]. \tag{2.12}$$

Since the backward mapping allows us to compute the unique natural parameters for a given expected value of the sufficient statistic, we may directly compute the optimal parameters $\boldsymbol{\theta}$ of the statistical model Q_X by computing $\boldsymbol{\theta} = \boldsymbol{\tau}^*(\boldsymbol{\eta})$, where $\boldsymbol{\eta} = \mathbb{E}_P[\mathbf{s}(X)]$. Moreover, when $P_X = P_X^n$ is the empirical distribution, this computation is equivalent to computing the maximum likelihood estimator. As such, in the context of exponential families, the principle of maximum entropy is equivalent to the principle of maximum likelihood.

If we denote the natural parameters of Q_X by $\boldsymbol{\theta}_Q$, and assume that P_X is an element of the same exponential family as Q_X with mean parameters $\boldsymbol{\eta}_P$, then the relative entropy has a particularly simple expression. Because the relative entropy is equal to the cross-entropy minus the entropy, we may combine equation 2.11 and

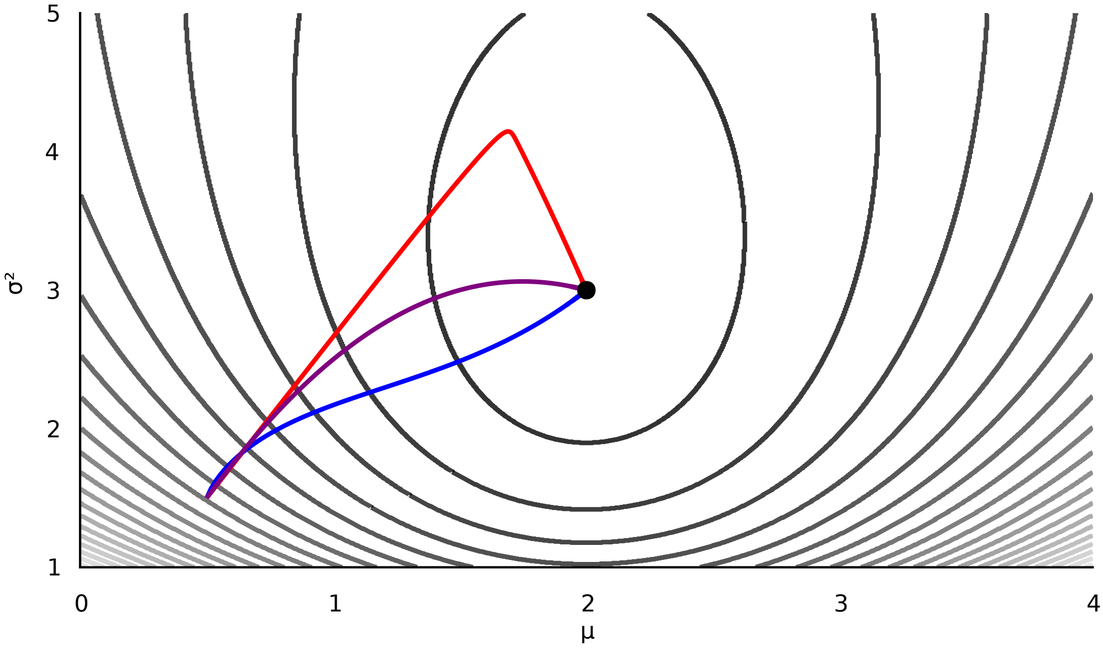

Figure 2.2: In this figure I present three instances of gradient descent of the relative entropy $D(P_X \parallel Q_X)$, where the target distribution P_X is a normal distribution with mean $\mu = 2$ and variance $\sigma^2 = 3$, and Q_X is the distribution we optimize. I depict the isolines of the relative-entropy as a function of mean and variance of Q_X (black lines), where the central point is equal to 0 and the highest valued isoline has a relative entropy of 2.39 (most transparent line). I depict the descent of the natural parameters (red line), the descent of the mean parameters (blue line), and the coordinate independent natural gradient descent (purple line). Observe how the natural gradient descent is perpendicular to the isolines of the cross-entropy.

theorem 2.7 to conclude that

$$D(P_X \parallel Q_X) = \phi(\boldsymbol{\eta}_P) + \psi(\boldsymbol{\theta}_Q) - \boldsymbol{\eta}_P \cdot \boldsymbol{\theta}_Q.$$

When P is equal to Q, $D(P_X \parallel Q_X)$ is 0, and we can recover the definitions of ψ and ϕ as they are developed in convex analysis.

Given an exponential family $\mathcal{M}$ with a minimal sufficient statistic, we know how to derive a pair of coordinate systems Θ and H for identifying elements of the exponential family, and how to compute coordinate transformations with the derivatives of ψ and ϕ. In fact, ψ and ϕ may be also be used to describe the metric tensor of $\mathcal{M}$ in each of their respective coordinate systems, allowing us to perform gradient descent in either system (figure 2.2).

Theorem 2.8. *Suppose that $\mathcal{M}$ is an exponential family on the measurable space $(\Omega_X, \mathcal{F}_X)$ defined by the base measure μ and minimal sufficient statistic $\mathbf{s}$, and that $P_X \in \mathcal{M}$ with natural parameters $\boldsymbol{\theta}$, and mean parameters $\boldsymbol{\eta}$. Then:*

1. *The Fisher information metric in natural coordinates is the Hessian of the log-partition function, such that $\mathbf{G}_{\boldsymbol{\theta}} = \partial_{\boldsymbol{\theta}\boldsymbol{\theta}}\psi(\boldsymbol{\theta})$.*
2. *The Fisher information metric in mean coordinates is the Hessian of the negative entropy, such that $\mathbf{G}_{\boldsymbol{\eta}} = \partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta})$.*
3. $\mathbf{G}_{\boldsymbol{\theta}} = \mathbf{G}^{-1}_{\boldsymbol{\tau}(\boldsymbol{\theta})}$.

Proof. On one hand, the Fisher information metric in natural coordinates is given by

$$\begin{aligned} g_{\boldsymbol{\theta},i,j} =& \mathbb{E}_P[\partial_{\theta_i\theta_j} \log p_X(X)] \\ & \mathbb{E}_P[\partial_{\theta_i\theta_j}(\boldsymbol{\theta} \cdot \mathbf{s}(X) - \psi(\boldsymbol{\theta}))] \\ & \mathbb{E}_P[\partial_{\theta_i\theta_j}\psi(\boldsymbol{\theta})] = \partial_{\theta_i\theta_j}\psi(\boldsymbol{\theta}). \end{aligned}$$

Note that according to proposition 2.4, the Fisher information metric in natural coordinates is equal to the covariance matrix of the sufficient statistic.

Because $\boldsymbol{\tau}^*$ is the inverse of $\boldsymbol{\tau}$, the inverse function theorem implies that the Hessian $\partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta})$ is the inverse of the Hessian $\partial_{\boldsymbol{\tau}\boldsymbol{\tau}}(\boldsymbol{\tau}(\boldsymbol{\eta}))$. Therefore, by the tensor transformation laws, and because Hessians are always symmetric, we may conclude that

$$\begin{aligned} \mathbf{G}_{\boldsymbol{\eta}} &= \partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta}) \cdot \mathbf{G}_{\boldsymbol{\tau}(\boldsymbol{\eta})} \cdot \partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta}) \\ &= \partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta}) \cdot (\partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta}))^{-1} \cdot \partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta}) \\ &= \partial_{\boldsymbol{\eta}\boldsymbol{\eta}}\phi(\boldsymbol{\eta}) = \mathbf{G}^{-1}_{\boldsymbol{\tau}(\boldsymbol{\eta})}. \end{aligned}$$

□

Optimizing exponential family models can ultimately be reduced to evaluating functions of the forward and backward mappings of the corresponding exponential family. For many exponential families, however, there are no closed-form expressions for these mappings. Nevertheless, both mappings can be approximated if we can generate samples from distributions in the exponential family in question.

On one hand, suppose that the natural parameters of the model Q_X are $\boldsymbol{\theta}$, and that we wish to compute the mean parameters $\boldsymbol{\eta} = \boldsymbol{\tau}(\boldsymbol{\theta})$. Since $\boldsymbol{\eta} = \mathbb{E}_Q[\mathbf{s}(X)]$, if we can generate samples $(X_i)_{i=1}^n$ from Q_X, then we may apply the law of large numbers to compute the approximate mean parameters $\boldsymbol{\eta} \approx \sum_{i=1}^n \mathbf{s}(X_i)$. On the other hand, suppose we know the mean parameters $\boldsymbol{\eta} = \mathbb{E}_P[\mathbf{s}(X)]$, and that we wish to compute the natural parameters $\boldsymbol{\theta} = \boldsymbol{\tau}^*(\boldsymbol{\eta})$. Since $\boldsymbol{\theta}$ are the natural parameters which minimize the cross-entropy in equation 2.11, we may find $\boldsymbol{\tau}^*(\boldsymbol{\eta})$ by choosing some initial natural parameters $\boldsymbol{\theta}_0$, and then generating the sequence of natural parameters $\boldsymbol{\theta}_k$ by combining the stochastic gradient descent procedure described by equation 2.6 with the expression for cross-entropy gradient in equation 2.12. In order to approximate the expectations $\mathbb{E}_Q[\mathbf{s}(X)]$ in the cross-entropy gradient, we may again generate samples from the distribution $Q_X(\boldsymbol{\theta}_k)$ at every step of the gradient descent.

2.4 Graphical Models

So far we have concerned ourselves primarily with single random variables and their distributions. In practice of course, we may wish to consider many different random variables and the statistical dependencies between them. In general, we refer to a statistical model of the joint distribution of multiple random variables as a generative model. Given a collection of random variables $(X_i)_{i=1}^n$ on the probability space $(\Omega, \mathcal{F}, P)$, a graphical model is a generative model $Q_{(X_i)_{i=1}^n}$ of the random variables, combined with a graphical representation of the statistical dependencies amongst the random variables described by the generative model.

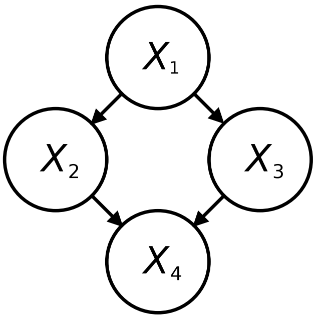

Figure 2.3: A directed graphical model over four random variables. The graph implies that the generative model may be factorized as $Q_{X_1X_2X_3X_4} = Q_{X_1} \cdot Q_{X_2|X_1} \cdot Q_{X_3|X_1} \cdot Q_{X_4|X_2,X_3}$. Sampling from directed graphical models is typically straightforward. Inference, however, is typically more complicated, and directed graphical models cannot represent cyclic dependencies amongst the random variables.

A graph is a pair (V, E) of sets, where V is a set of vertices, and E is a set of edges. Formally, we define a vertex as an index, the directed edge from vertex i to vertex j as (i, j), and the undirected edge between i and j as $\{i, j\}$. In a graphical model, vertices represent random variables, directed edges represent conditional dependencies, and undirected edges represent joint dependencies. When modelling conditional dependencies, we generally assume that the directed edges form no cycles, such that there is no set of edges $\{(i_0, i_1), (i_1, i_2) \ldots, (i_{k-1}, i_k)\} \subset E$ such that $i_0 = i_k$.

A Bayesian network is a joint probability distribution which can be represented by a directed, acyclic graph. The semantics of Bayesian networks are straightforward.

Definition 2.3 (Bayesian Network)**.** Let (V, E) be a directed, acyclic graph where $V = \{1, \ldots, n\}$, $E = \{(j, i) \colon i \in V, j \in A_i\}$, and where $(A_i)_{i=1}^n$ is a collection of subsets of V. Then the collection of random variables $(X_i)_{i=1}^n$ is a Bayesian network represented by (V, E) if and only if

$$P_{(X_i)_{i=1}^n} = \prod_{i=1}^{n} P_{X_i|(X_j)_{j \in A_i}}. \tag{2.13}$$

A generative model $Q_{(X_i)_{i=1}^n}$ is then a directed graphical model if $Q_{(X_i)_{i=1}^n}(\boldsymbol{\theta})$ is a Bayesian network represented by the directed, acyclic graph (V, E), for any parameters $\boldsymbol{\theta} \in \Theta$ (see figure 2.3).

A Markov random field is a joint distribution which can be represented by an undirected graph. The semantics of undirected graphical models are somewhat more complicated than in the directed case.

Definition 2.4 (Markov Random Field)**.** Let (V, E) be an undirected graph, and for every $i \in V$, let $B_i = \{j \colon \{i, j\} \in E\}$. Then the collection of random variables $(X_i)_{i=1}^n$ is a Markov random field represented by (V, E) if and only if, for any $i \in V$, and $j \in V/(\{i\} \cup B_i)$,

$$P_{X_iX_j|(X_k)_{k \in B_i}} = P_{X_j|(X_k)_{k \in B_i}} \cdot P_{X_j|(X_k)_{k \in B_i}}.$$

An undirected graphical model is then a generative model $Q_{(X_i)_{i=1}^n}$ if $Q_{(X_i)_{i=1}^n}(\boldsymbol{\theta})$ is a Markov random field represented by the undirected graph (V, E), for any parameters $\boldsymbol{\theta} \in \Theta$ (see figure 2.4).

Intuitively, a joint distribution $P_{(X_i)_{i=1}^n}$ is a Markov random field if a given random variable X_i is independent of the rest of the random variables in the collection given its immediate neighbours in the graph. In contrast with the definition of a Bayesian network, however, this property is local in definition, and knowing the graph of a Markov random field does not immediately tell us how we might factorize the complete joint distribution.

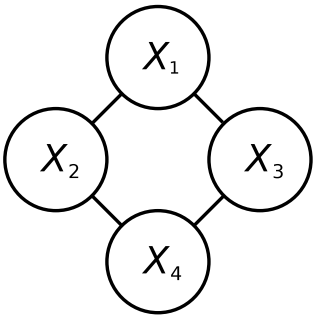

Figure 2.4: An undirected graphical model over four random variables. This graph implies that $P_{X_1 X_4|X_2,X_3} = P_{X_1|X_2,X_3} \cdot P_{X_4|X_2,X_3}$, and that $P_{X_2 X_3|X_1,X_4} = P_{X_2|X_1,X_4} \cdot P_{X_3|X_1,X_4}$ This graph represents cyclic dependencies amongst the random variables which cannot be represented by a directed acyclic graph. Generating samples from this graphical model, however, may be very challenging.

To describe factorization in Markov random fields we must first compute the cliques of a graph. A clique is a subset of fully interconnected vertices $C \subseteq V$ such that $\forall i, j \in C, \{(i,j),(j,i)\} \in E$. Moreover, a clique is maximal if there not exists no clique C' such that $C \subset C'$. The Hammersley-Clifford theorem provides a way of factoring Markov random fields by relating the joint distribution with the maximal cliques of the graph which represents it.

Theorem 2.9 (The Hammersley-Clifford Theorem)**.** *Let $X_1, \dots X_n$ be a collection of random variables, let (V,E) be an undirected graph, and let $\mathcal{C} = (C_i)_{i=1}^k$ be the set of all maximal cliques of (V,E). Then (V,E) represents the joint distribution $P_{(X_i)_{i=1}^n}$ with a strictly positive density $p_{(X_i)_{i=1}^n}$ if and only if there exists a set of functions $(f_i)_{i=1}^k$ such that*

$$p_{(X_i)_{i=1}^n}((\mathbf{x}_i)_{i=1}^n) \propto \prod_{i=1}^k e^{f_i(\mathbf{x}_{C_i})},$$

where $\mathbf{x}_C$ is the subsequence of $(\mathbf{x}_i)_{i=1}^n$ with indices in C.

Proof. Kollar and Friedman (2009). ☐

Note that since exponential family densities are always positive, we may always apply the Hammersley-Clifford theorem to identify the dependency structure in exponential family joint distributions. Moreover, when the exponential families have minimal sufficient statistics, identifying the factors in theorem 2.9 reduces to identifying appropriate sets of terms in the dot product $\mathbf{s}(\mathbf{x}) \cdot \boldsymbol{\theta}$ (see Altun et al., 2004).

In statistical applications of graphical models, we do not typically observe all the random variables in the collection, but rather only a certain subset of them. Let us therefore separate the random variables of a graphical model into latent variables $X = (X_i)_{i=1}^{n_X}$ with state space $(\Omega_X, \mathcal{F}_X)$ and observable variables $Z = (Z_i)_{i=1}^{n_Z}$ with state space $(\Omega_Z, \mathcal{F}_Z)$. Let us then assume that generative model Q_{XZ} over the observable and latent variables is an exponential family with sufficient statistic $\mathbf{s}$ and base measure μ. Let us also assume that the base measure $\mu = \mu_Z \times \mu_X$ is a product measure which evaluates pairs of subsets of Ω_Z and Ω_X. Given all this, we may maximize the likelihood of the parameters of Q_{XZ} given samples from the true observable density P_Z, by maximizing the likelihood with respect to the marginal density

$$q_Z(\mathbf{z}) = \int_{\Omega_X} e^{\mathbf{s}(\mathbf{x},\mathbf{z})\cdot\boldsymbol{\theta} - \psi(\boldsymbol{\theta})} \mu_X(d\mathbf{x}) = e^{\psi_{X|Z}(\boldsymbol{\theta},\mathbf{z}) - \psi(\boldsymbol{\theta})}, \tag{2.14}$$

where we refer to

$$\begin{aligned} \psi_{X|Z} \colon \Theta \times \Omega_Z &\to \mathbb{R} \\ (\boldsymbol{\theta}, \mathbf{z}) &\mapsto \log \int_{\Omega_X} e^{\mathbf{s}(\mathbf{x},\mathbf{z})\cdot\boldsymbol{\theta}} \mu_X(d\mathbf{x}). \end{aligned} \tag{2.15}$$

as the conditional log-partition function.

Although we defined the conditional log-partition function in order to express a marginal distribution of Q_{XZ}, we can interpret it more readily by considering the conditional density

$$q_{X|Z}(\mathbf{x} \mid \mathbf{z}) = \frac{q_{XZ}(\mathbf{x}, \mathbf{z})}{q_Z(\mathbf{z})} = \frac{e^{\mathbf{s}(\mathbf{x},\mathbf{z}) \cdot \boldsymbol{\theta} - \psi(\boldsymbol{\theta})}}{e^{\psi_{X|Z}(\boldsymbol{\theta},\mathbf{z}) - \psi(\boldsymbol{\theta})}} = e^{\mathbf{s}(\mathbf{x},\mathbf{z}) \cdot \boldsymbol{\theta} - \psi_{X|Z}(\boldsymbol{\theta},\mathbf{z})}. \tag{2.16}$$

Indeed, at a particular observation $\mathbf{z}$, the conditional distribution $Q_{X|Z}$ is an element of the exponential family with base measure μ_X and sufficient statistic $\mathbf{s}(\mathbf{z}, \cdot)$, with log-partition function $\psi_{X|Z}(\boldsymbol{\theta}, \mathbf{z})$. By simply integrating over the observable rather than latent variables, we may similarly express $Q_{Z|X}$ in terms of the conditional log-partition function $\psi_{Z|X}(\boldsymbol{\theta}, \mathbf{x})$.

In order to optimize the parameters of Q_{XZ}, we minimize the relative entropy of the marginal distribution Q_Z with respect to the target distribution P_Z. We therefore consider the cross-entropy

$$\begin{aligned} H(P_Z, Q_Z) &= \int_{\Omega_Z} -\psi_{X|Z}(\boldsymbol{\theta}, \mathbf{z}) + \psi(\boldsymbol{\theta}) P(d\mathbf{z}) \\ &= \psi(\boldsymbol{\theta}) - \mathbb{E}_P[\psi_{X|Z}(\boldsymbol{\theta}, Z)], \end{aligned} \tag{2.17}$$

which yields the derivatives of the relative entropy gradient

$$\partial_{\boldsymbol{\theta}} D(P_Z \parallel Q_Z) = \boldsymbol{\eta} - \mathbb{E}_P[\partial_{\boldsymbol{\theta}} \psi_{X|Z}(\boldsymbol{\theta}, Z)].$$

Because $\partial_{\boldsymbol{\theta}} \psi_{X|Z}$ at $\mathbf{z}$ is the log-partition function of an exponential family, proposition 2.4 implies that $\partial_{\boldsymbol{\theta}} \psi_{X|Z}(\boldsymbol{\theta}, \mathbf{z}) = \mathbb{E}_Q[\mathbf{s}(\mathbf{z}, X) \mid Z = \mathbf{z}]$. We may thus express the relative entropy gradient as

$$\partial_{\boldsymbol{\theta}} D(P_Z \parallel Q_Z) = \mathbb{E}_Q[\mathbf{s}(Z, X)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}(Z, X) \mid Z]]. \tag{2.18}$$

Even though our problem is now more sophisticated, its form remains the same as in the simple exponential family case: we wish to match the sufficient statistics of our model to the statistics of what we observe.

2.5 Stochastic Chains

Given the probability space $(\Omega, \mathcal{F}, P)$, a discrete-time stochastic process, or a stochastic chain, is a sequence of random variables $(X_k)_{k \in \mathbb{N}}$ on $(\Omega, \mathcal{F}, P)$, where each X_k has the same state space $(\Omega_X, \mathcal{F}_X)$. In this dissertation I only consider chains which are invariant to shifts in time, a property which is known as stationarity.

Definition 2.5 (Stationary Stochastic Process). A stochastic chain $(X_k)_{k \in \mathbb{N}}$ on the probability space $(\Omega, \mathcal{F}, P)$ is stationary if for any $K \subset \mathbb{N}$, and $l \in \mathbb{N}$,

$$P_{(X_k)_{k \in K}} = P_{(X_{k+l})_{k \in K}}.$$

A Markov chain is a stochastic chain which satisfies the (temporal) Markov property[3]

$$P_{X_{k+1} | (X_i)_{i=0}^{k}} = P_{X_{k+1} | X_k}. \tag{2.19}$$

---

[3] The term "Markov chain" is often reserved for processes with discrete state spaces, however in this dissertation I do not make this distinction.

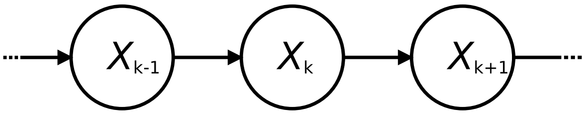

Figure 2.5: A Markov chain is a form of dynamical system where each state depends only on the previous state. When discretized in time, stochastic dynamical systems can often be modelled as Markov chains.

The conditional distributions $P_{X_{k+1}|X_k}$ of the Markov chain for all $k > 0$ are known as transition distributions, and the distribution P_{X_0} is known as the initial distribution. Since the Markov chains I consider are stationary, I generally assume the existence of the auxiliary random variables $X' = X_{k+1}$ and $X = X_k$ for an arbitrary k, such that I may denote any transition distribution by $P_{X'|X} = P_{X_{k+1}|X_k}$; I refer to this conditional distribution as *the* transition distribution.

Note that equation 2.19 is simply a form of equation 2.13 for describing directed graphical models. The Markov property of Markov chains therefore affords a simple directed graph representation, as depicted in figure 2.5.

Given the initial distribution and transition distribution, the marginal distribution of the Markov chain at any time is given recursively by

$$P_{X_{k+1}}(A) = \int_{\Omega_X} P_{X'|X}(A \mid \mathbf{x}_k) P_{X_k}(d\mathbf{x}_k), \tag{2.20}$$

for any $A \in \mathcal{F}_X$. Of particular interest in the study of Markov chains is the existence, uniqueness, and stability of the limiting distribution $\lim_{k\to\infty} P_{X_k}$; this is the province of ergodic theory. Intuitively, ergodic theory tells us how to compute spatial averages from temporal averages. In the context of Markov chains on general state spaces, there are a number of ways of formalizing ergodicity depending on the exact nature of the problem being addressed. For the purposes of this dissertation, we may distill ergodic theory down to a few key definitions and theorems.

To begin, the limiting distribution $\lim_{k\to\infty} P_{X_k}$ is necessarily a fixed point of equation 2.20, and is referred to as an equilibrium distribution.

Definition 2.6 (Equilibrium Distribution)**.** A probability distribution Π_X is an equilibrium distribution of the transition distribution $P_{X'|X}$ if it satisfies

$$\Pi_X(A) = \int_{\Omega_X} P_{X'|X}(A \mid \mathbf{x}) \Pi_{X_k}(d\mathbf{x}),$$

for any $A \in \mathcal{F}_X$.

The equilibrium distribution is the basis for the notion of "spatial averages" mentioned above. "Temporal averages", on the other hand, are averages over the realization of a Markov chain.

Now, we can only compute the spatial average based on a temporal average if the Markov chain is guaranteed to repeatedly visit every part of the state space in question. One way of formalizing this is with the notion of Harris recurrence, which in turn is formalized with the random variables known as first entrance times. Given the Markov chain $(X_k)_{k\in\mathbb{N}}$ on the probability space $(\Omega, \mathcal{F}, P)$ with state space $(\Omega_X, \mathcal{F}_X)$, the first entrance time of a set $A \in \mathcal{F}_X$ is the random variable

$$T_A(\omega) \mapsto \min\{k \colon k \geq 1, X_k(\omega) \in A\},$$

where $T_A(\omega) = \infty$ if the Markov chain realized by ω never enters A.

Definition 2.7 (Harris Recurrence)**.** Given a measure μ on $\mathcal{F}_X$, the Markov chain $(X_k)_{k\in\mathbb{N}}$ is Harris recurrent with base measure μ if for any $\mathbf{x} \in \Omega_X$ and $A \in \mathcal{F}_X$,

$$\mu(A) > 0 \implies P(T_A < \infty \mid X_0 = \mathbf{x}) = 1.$$

Intuitively, a Markov chain is Harris recurrent if it returns an infinite number of times to any set in the support of μ with probability 1. Strictly speaking, the base measure in the definition of Harris recurrence need not be the same as the base measure used to define its transition density, however it is generally safe to assume as much.

Even when a Markov chain is Harris recurrent, we cannot rule out that it is periodic. A periodic Markov chain is one for which the limiting distribution of the Markov chain cycles through a collection of distributions. Ergodic theory can also be applied to periodic Markov chains, however I do not make use of periodic Markov chains in this dissertation. The general definition of aperiodicity is rather technical, and so I provide a simpler definition which is sufficient for my purposes.

Definition 2.8 (Aperiodic Markov Chain)**.** The Harris recurrent Markov chain $(X_k)_{k\in\mathbb{N}}$ with base measure μ is aperiodic if for any $A \in \mathcal{F}_X$ such that $\mu(A) > 0$, $p_{X'|X}(\mathbf{x}' \mid \mathbf{x}) > 0$ for any $\mathbf{x} \in \Omega_X$ and $\mathbf{x}' \in A$.

We refer to a Markov chain as ergodic if it converges to a unique equilibrium distribution regardless of its initial distribution. Convergence of distributions can be defined in a number of ways, but the approach taken in Meyn and Tweedie (2009) makes use of the total variation norm

$$\parallel \mu \parallel_{TV} \mapsto \max_{A\in\mathcal{F}} \mu(A) - \min_{A\in\mathcal{F}} \mu(A),$$

where μ is any measure on the σ-algebra $\mathcal{F}$.

Definition 2.9 (Ergodic Markov Chain)**.** The Markov chain $(X_k)_{k\in\mathbb{N}}$ is ergodic if it has a unique equilibrium distribution Π_X, and

$$\lim_{k\to\infty} \parallel P_{X_k} - \Pi_X \parallel_{TV} = 0,$$

for any initial distribution P_{X_0}.

Based on these definitions, we may finally establish the necessary ergodic theory.

Theorem 2.10. *Suppose that there exists an equilibrium distribution for the aperiodic, Harris recurrent Markov chain* $(X_k)_{k\in\mathbb{N}}$*. Then* $(X_k)_{k\in\mathbb{N}}$ *is ergodic.*

Proof. Theorem 13.3.3 in Meyn and Tweedie (2009). □

As suggested by the name, we can compute spatial averages of ergodic Markov chains by computing temporal averages.

Theorem 2.11 (The Ergodic Theorem)**.** *Suppose that the Markov chain* $(X_k)_{k\in\mathbb{N}}$ *is aperiodic and Harris recurrent with equilibrium distribution* Π*. Then for any function* $f\colon \Omega_X \to \mathbb{R}$ *which satisfies* $\int_{\Omega_X} |f| d\Pi < \infty$*,*

$$\lim_{k\to\infty} \sum_{i=0}^{k} \frac{f(X_k)}{k} = \int_{\Omega_X} f d\Pi.$$

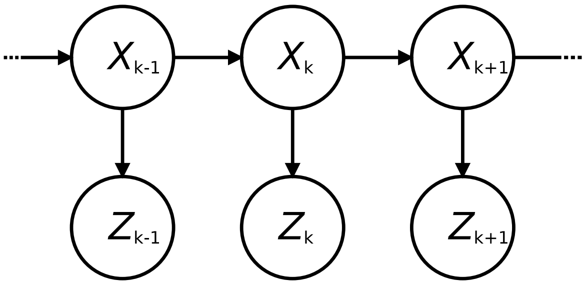

Figure 2.6: Hidden Markov models are used to model dynamical systems which cannot be directly observed.

Proof. This is a consequence of theorems 13.0.1 and 17.3.2 in Meyn and Tweedie (2009). ☐

This is all the ergodic theory that I require in this dissertation, however there is one other form of stochastic chain that I consider which is no less essential to this dissertation than the Markov chain. A hidden Markov chain is a sequence of pairs of random variables $(X_k, Z_k)_{k\in\mathbb{N}}$, where X_k is the latent random variable at time k, and Z_k is the observable random variable at time k. The dynamic latent state $(X_k)_{k\in\mathbb{N}}$ of a hidden Markov chain is a Markov chain, and any latent variable X_k is also independent of Z_k given X_{k-1}. Each observable random variable Z_k is conditionally independent of all the other random variables in the model given the simultaneous response X_k, and the distributions $P_{Z_k|X_k}$ are known as the emission distributions. When all the emission distributions are equal, I speak of *the* emission distribution $P_{Z|X}$ given the auxiliary random variables $X = X_k$ and $Z = Z_k$.

I do not introduce any new theorems for hidden Markov chains. However, as we will see, hidden Markov chains are fundamental to dynamic Bayesian inference. Hidden Markov chains are more widely known as hidden Markov models, however in this dissertation I reserve the term hidden Markov model for generative models $Q_{(X_k,Z_k)_{k\in\mathbb{N}}}$ which model hidden Markov chains. I depict the graphical representation of a hidden Markov model in figure 2.6.

2.6 Bayesian Inference

Consider the joint distribution P_{XZ} of the latent random variable X and the observable random variable Z on the probability space $(\Omega, \mathcal{F}, P)$. In its most general form, the Bayesian approach to statistics reduces statistical inference on the observation $\mathbf{z}$ to the evaluation of the conditional distribution $P_{X|Z=\mathbf{z}}$. What makes the theory of Bayesian inference more than trivial is that we may often express this conditional distribution in a more articulated form, especially when we make use of the dependency structure of the distribution in question.

Recall that the marginal density over X is $p_X(\mathbf{x}) = \int_{\Omega_Z} p_{XZ}(\mathbf{x}, \mathbf{z})\mu_Z(d\mathbf{z})$ and that the conditional density of X given Z is $p_{X|Z} = p_{XZ}/p_Z$, and similarly for p_Z and $p_{Z|X}$. Based on these definitions we may derive Bayes' rule

$$p_{X|Z}(\mathbf{x} \mid \mathbf{z}) = \frac{p_{XZ}(\mathbf{x}, \mathbf{z})p_X(\mathbf{x})}{p_X(\mathbf{x})p_Z(\mathbf{z})} = \frac{p_{Z|X}(\mathbf{z} \mid \mathbf{x})p_X(\mathbf{x})}{p_Z(\mathbf{z})}.$$

In Bayesian inference, the various distributions involved in Bayes' rule have distinct interpretations. In particular, P_X is the prior, which represents our current beliefs about the latent variable; $P_{Z|X}$ is the likelihood, which describes how to generate an observation given the latent state; finally, $P_{X|Z}$ is the posterior, which represents our new beliefs about the latent variables after being given a particular observation.

For a given observation $\mathbf{z}$, $p_Z(\mathbf{z})$ is constant. Since $p_{X|Z=\mathbf{z}}$ is a probability density and must therefore integrate to 1, knowing the posterior density up to a constant factor is sufficient for determining the posterior. Bayes' rule is therefore often written

$$p_{X|Z}(\mathbf{x} \mid \mathbf{z}) \propto p_{Z|X}(\mathbf{z} \mid \mathbf{x}) p_X(\mathbf{x}), \tag{2.21}$$

which emphasizes that given an observation, knowing the prior and the likelihood is sufficient for determining the posterior.

Let us now consider the sequence of observable variables $(Z_i)_{i=1}^n$ which are conditionally independent given the latent variable, such that the conditional density $p_{(Z_i)_{i=1}^n|X} = \prod_{i=1}^n p_{Z_i|X}$. We can intuitively think of this likelihood as a "multisensory" likelihood in the sense that each random variable Z_i represents a different sensory modality which responds to some unknown variable. By applying Bayes' rule to a sequence of observations $\mathbf{z}_1, \dots, \mathbf{z}_n$, we may express the posterior given these observations as

$$\begin{aligned} p_{X|(Z_i)_{i=1}^n}(\mathbf{x} \mid \mathbf{z}_1, \dots, \mathbf{z}_n) &= \frac{p_{(Z_i)_{i=1}^n|X}(\mathbf{z}_1, \dots, \mathbf{z}_n \mid \mathbf{x}) p_X(\mathbf{x})}{p_{(Z_i)_{i=1}^n}(\mathbf{z}_1, \dots, \mathbf{z}_n)} \\ &= \frac{\prod_{i=1}^n p_{Z_i|X}(\mathbf{z}_i \mid \mathbf{x}) p_X(\mathbf{x})}{p_{Z_n|(Z_i)_{i=1}^{n-1}}(\mathbf{z}_n \mid \mathbf{z}_1, \dots, \mathbf{z}_{n-1}) p_{(Z_i)_{i=1}^{n-1}}(\mathbf{z}_1, \dots, \mathbf{z}_{n-1})} \\ &= \frac{p_{Z_n|X}(\mathbf{z}_n \mid \mathbf{x}) p_{X|(Z_i)_{i=1}^{n-1}}(\mathbf{x} \mid \mathbf{z}_1, \dots, \mathbf{z}_{n-1})}{p_{Z_n|(Z_i)_{i=1}^{n-1}}(\mathbf{z}_n \mid \mathbf{z}_1, \dots, \mathbf{z}_{n-1})}. \end{aligned} \tag{2.22}$$

Since $p_{Z_k|(Z_i)_{i=1}^n}$ does not depend on $\mathbf{x}$, we may again absorb this factor into a constant of proportionality and write

$$p_{X|(Z_i)_{i=1}^n}(\mathbf{x} \mid \mathbf{z}_1, \dots, \mathbf{z}_n) \propto p_{Z_n|X}(\mathbf{z}_n \mid \mathbf{x}) p_{X|(Z_i)_{i=1}^{n-1}}(\mathbf{x} \mid \mathbf{z}_1, \dots, \mathbf{z}_{n-1}). \tag{2.23}$$

Observe how this relation is a single application of Bayes' rule (2.21) given the response $\mathbf{z}_n$, where the prior is the posterior $P_{X|(Z_i)_{i=1}^{n-1}}$ given the observations $\mathbf{z}_1, \dots, \mathbf{z}_{n-1}$. We may therefore calculate the posterior for an entire sequence of observations by recursively applying Bayes' rule to calculate a sequence of posteriors until we reach the prior density p_X.

Although equation 2.23 reduces to a recursive application of Bayes' rule, evaluating this posterior is often intractable, as each application of Bayes' rule tends to increase the complexity of the beliefs of the agent. A powerful technique for ameliorating this problem is to make use of conjugate priors.

Definition 2.10 (Conjugate Prior)**.** Let X be a latent random variable and Z an observable random variable on $(\Omega, \mathcal{F}, P)$, and let $\mathcal{M}$ be a manifold of probability distributions. Then the prior P_X is conjugate to the posterior $P_{X|Z}$ if $P_X \in \mathcal{M}$ and $P_{X|Z} \in \mathcal{M}$.

Conjugate priors ensure that in recursive applications of Bayes' rule, the complexity of the posterior does not increase beyond the dimension of $\mathcal{M}$. I depict an example of Bayesian inference with conjugate priors in figure 2.7.

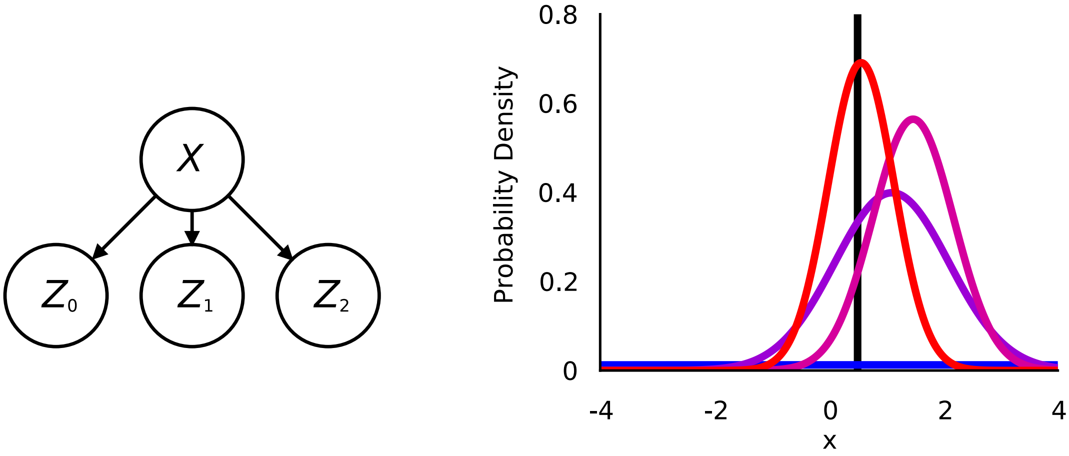

Figure 2.7: In these two figures I depict an application of multisensory Bayesian inference on 1-dimensional random variables. The prior and posteriors in this application are normal distributions. *Left*: A graphical representation of the multisensory inference problem; we aim to use the three observations to infer the latent variable X. In this model, the prior density p_X is approximately flat over the region of interest, and the likelihoods $p_{Z_i|X=x}$ are normal distributions with mean x and variance 1. *Right*: The beliefs of the agent as it integrates the sequence observations $z_0 = 1.10$, $z_1 = 1.84$, $z_2 = -1.25$ of the true stimulus $x = 0.5$ (black line). The prior is approximately uniform over the state space (blue), and the sequence of posteriors (purple, magenta, red) are the result of recursively applying Bayes' rule to the sequence of observations.

This recursive approach to Bayesian inference can also be extended to dynamic latent variables. Bayesian filtering is the Bayesian approach to computing posterior beliefs about the latent state of a hidden Markov chain $(X_k, Z_k)_{k\in\mathbb{N}}$, and can be thought of as an extension of recursive relation 2.23 to the case where the underlying stimulus is dynamic. A Bayes filter is the algorithm for computing these posteriors, and is defined by two equations. The first is derived by following the derivation presented in equations 2.22, which results in

$$p_{X_k|(Z_i)_{i=0}^{k}}(\mathbf{x}_k \mid \mathbf{z}_0,\dots,\mathbf{z}_k) \propto p_{Z_k|X_k}(\mathbf{z}_k \mid \mathbf{x}_k)\, p_{X_k|(Z_i)_{i=0}^{k-1}}(\mathbf{x}_k \mid \mathbf{z}_0,\dots,\mathbf{z}_{k-1}). \tag{2.24}$$

When normalized, this relation is known as the update equation. It reduces the computation of the posterior beliefs at time k to applying Bayes' rule to the likelihood of $\mathbf{z}_k$ and the prior $P_{X_k|(Z_i)_{i=0}^{k-1}}$ at $\mathbf{z}_0,\dots,\mathbf{z}_{k-1}$.

In contrast to relation 2.23, this prior is not simply the posterior at the previous time, but rather the prior beliefs about the stimulus at time k given only the sequence of responses up to time $k-1$. Based on the graphical structure of the hidden Markov chain, we may express this distribution by

$$\begin{aligned}
&p_{X_k|(Z_i)_{i=0}^{k-1}}(\mathbf{x}_k \mid \mathbf{z}_0,\dots,\mathbf{z}_{k-1})\\
&= \int_{\Omega_X} p_{X_{k-1}X_k|(Z_i)_{i=0}^{k-1}}(\mathbf{x}_{k-1},\mathbf{x}_k \mid \mathbf{z}_0,\dots,\mathbf{z}_{k-1})\mu_X(d\mathbf{x}_{k-1})\\
&= \int_{\Omega_X} p_{X_k|X_{k-1}(Z_i)_{i=0}^{k-1}}(\mathbf{x}_k \mid \mathbf{x}_{k-1},\mathbf{z}_0,\dots,\mathbf{z}_{k-1})\\
&\qquad\qquad p_{X_{k-1}|(Z_i)_{i=0}^{k-1}}(\mathbf{x}_{k-1} \mid \mathbf{z}_0,\dots,\mathbf{z}_{k-1})\mu_X(d\mathbf{x}_{k-1})\\
&= \int_{\Omega_X} p_{X'|X}(\mathbf{x}_k \mid \mathbf{x}_{k-1})\, p_{X_{k-1}|(Z_i)_{i=0}^{k-1}}(\mathbf{x}_{k-1} \mid \mathbf{z}_0,\dots,\mathbf{z}_{k-1})\mu_X(d\mathbf{x}_{k-1}).
\end{aligned} \tag{2.25}$$

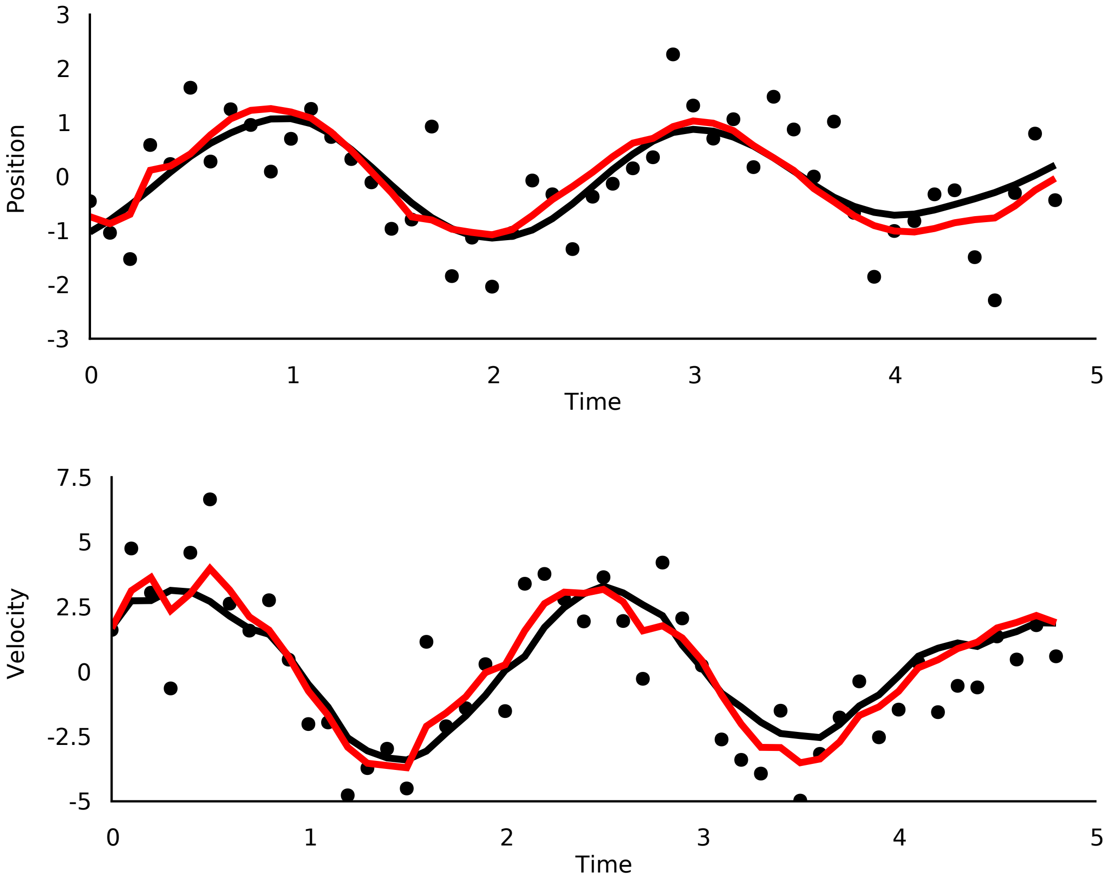

Figure 2.8: An extended Kalman filter applied to estimating the state of a stochastic pendulum. I depict the true position and velocity of the pendulum over time (black lines), the noisy observations of this state (black dots), and the mean of the approximate beliefs computed by the extended Kalman filter (red lines).

Because these prior beliefs transform available information into information about the future, these prior beliefs are known as predictions, and this equation is known as the prediction equation.

The posterior beliefs at time k are a function of the emission distribution and the predictions at time k, and the predictions at time k are a function of the transition distribution and the posterior beliefs at time $k-1$. Therefore, we may recursively compute the posterior beliefs at any time k given the sequence of observations $\mathbf{z}_0, \ldots, \mathbf{z}_k$ by using the update equation to calculate the posterior beliefs as a function of the predictions, and by using the prediction equation to compute the predictions as a function of the previous posterior beliefs. This recursion ultimately completes at the prior P_{X_0} in the same manner as relation 2.23. This two-step, recursive algorithm is known as a Bayes filter.

Unsurprisingly, evaluating a Bayes filter brings additional computational challenges, namely, evaluating the prediction equation (2.25). Conjugate priors can still help us manage the complexity of the posterior, provided that the prediction distribution is conjugate to the posterior. This in no why implies, however, that the prediction equation may be tractably computed. Thankfully, in some cases we may indeed solve the prediction equation in closed-form, and there exists many approaches to finding approximate solutions.

In particular, when Ω_X is a finite set, the update and prediction equations can be evaluated brute-force. On the other hand, the Kalman filter is the form of Bayes filter for the case where the transition and emission distributions are given by linear transformations of the latent state with additive Gaussian noise, and in this case the update and prediction equations also afford a closed-form solution. These are the two most well-known solutions to Bayesian filtering, and in both cases, the predictions are conjugate to the posterior, and the complexity of the posterior remains constant over time.

When the transition and emission distributions of a hidden Markov chain are given by *nonlinear* transformations and additive Gaussian noise, the extended Kalman filter may be applied to compute approximate beliefs about the latent state. Intuitively, an extended Kalman filter relies on a local linearization of the transition and emission distributions, and propagates previous beliefs through these approximations. The resulting predictions and posterior beliefs are normally distributed, and the complexity of the beliefs remains constant over time. I present an example of extended Kalman filtering in figure 2.8. I will treat all of these approaches to filtering in more detail in later chapters.

Chapter 3

Families of Harmoniums

Bayesian inference begins with a joint distribution over latent and observable variables. The class of joint distributions which form the basis of my work are the joint distributions of particular tuples of random variables, which I refer to as harmoniums[4]. A harmonium is defined by the fact that the conditional distribution of each random variable in the harmonium is always an element of the same exponential family.

In the following sections I repeat a particular strategy for defining different kinds of harmoniums. That is, I begin by defining a kind of harmonium as a collection of random variables with a particular set of properties. I then construct a corresponding exponential family – a harmonium family – which contains exactly all the distributions of the harmoniums of a particular kind. I then continue by deriving several properties of these various kinds of harmonium.

Beginning with this chapter, and throughout the rest of this dissertation, I make use of a set of conventions and assumptions in order to simplify the presentation of the material herein. Firstly, I assume the existence of a probability space $(\Omega, \mathcal{F}, P)$. Random variables, e.g. X, are written without stating the probability space or state space, where the state space is inferred from context, and when necessary, denoted Ω_X.

Given a random variable X, exponential families are written with a subscript as $\mathcal{M}_X$, which indicates that the exponential family $\mathcal{M}_X$ is a manifold of distributions over Ω_X. Similarly, the sufficient statistic and base measure of an exponential family are subscripted with the same random variable as the family itself, e.g. $\mathcal{M}_X$ is defined by the sufficient statistic $\mathbf{s}_X$ and base measure μ_X, and I assume that they are implicitly declared when the exponential family $\mathcal{M}_X$ is declared. Finally, the exponential family model Q_X parameterized by the natural parameter space Θ_X of $\mathcal{M}_X$ is the model for which $\mathcal{M}_X = Q_X(\Theta_X)$.

3.1 Second-Order Harmoniums

In this section I define and develop second-order harmoniums. A second-order harmonium is a pair of random variables, where the conditional distribution of each random variable given the other is always an element of the same exponential family. Based on the work of Arnold and Press (1989), I show that the set of all second-order

---

[4] See Smolensky (1986) for the origin of the name.

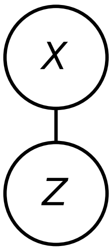

Figure 3.1: A harmonium is a pair of of random variables (X, Z) where the conditional distributions $P_{X|Z}$ and $P_{Z|X}$ are always elements of the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$, respectively. The undirected graph which represents a harmonium is simply a pair of connected random variables.

harmoniums defined by given pairs of exponential families is itself an exponential family with a particular structure. I then derive a number of basic properties of second-order harmoniums.

3.1.1 Second-Order Conditional Specification

In section 2.4 we considered joint distributions P_{XZ} over latent variables X and observable variables Z such that P_{XZ} is an element of some exponential family defined by the sufficient statistic $\mathbf{s}$ and base measure $\mu = \mu_X \times \mu_Z$. We found that the conditional distribution $P_{X|Z=\mathbf{z}}$ at a particular observation $\mathbf{z}$ is an element of the exponential family defined by the sufficient statistic $\mathbf{s}(\mathbf{z}, \cdot)$ and base measure μ_X, and similarly for $P_{Z|X}$. Although this fact is theoretically interesting, it is only of limited practical value. On one hand, the resulting exponential families may be intractable. On the other, the exponential family itself depends on $\mathbf{z}$, which makes it difficult to define general purpose inference and sampling algorithms which depend on the family of these conditional distributions. In order to avoid these problems, I consider pairs of random variables with conditional distributions which are constrained to always be elements of single exponential families (figure 3.1).

Definition 3.1 (Second-Order Harmonium)**.** The pair of random variables (X, Z) is a second-order harmonium defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$ if $P_{X|Z} \in \mathcal{M}_X$ and $P_{Z|X} \in \mathcal{M}_Z$.

This restriction on the conditional distribution of a harmonium imposes a great deal of structure on its joint distribution. In particular, we may show that the set of all second-order harmonium distributions defined by a given pair of exponential families is itself an exponential family.

The second-order harmonium family $\mathcal{H}_{XZ}$ defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$ is the $d_{XZ} = d_X + d_Z + d_X d_Z$ dimensional exponential family with base measure $\mu_{XZ} = \mu_X \times \mu_Z$ and sufficient statistic $\mathbf{s}_{XZ} \colon \Omega_X \times \Omega_Z \to \mathbb{R}^{d_{XZ}}$ with component functions $s_{XZ,i}$ given by

$$\begin{aligned} s_{XZ,i}(\mathbf{x}, \mathbf{z}) =& s_{X,i}(\mathbf{x}), \\ s_{XZ,d_X+j}(\mathbf{x}, \mathbf{z}) =& s_{Z,j}(\mathbf{z}), \\ s_{XZ,d_Z+jd_X+i}(\mathbf{x}, \mathbf{z}) =& s_{X,i}(\mathbf{x}) \cdot s_{Z,j}(\mathbf{z}), \end{aligned}$$

for $i \in [1, \dots, d_X]$ and $j \in [1, \dots, d_Z]$[5].

---

[5] Note that discrete-space exponential families with this form are also studied under the name "Kronecker product models". For a thorough analysis of the representational power of harmoniums/Kronecker product models see Montúfar and Morton (2015a).

Given a distribution $P_{XZ} \in \mathcal{H}_{XZ}$ with parameters $\boldsymbol{\theta}_{XZ}$, the dot product of the sufficient statistic $\mathbf{s}_{XZ}$ may be alternatively expressed as

$$\mathbf{s}_{XZ}(\mathbf{x}, \mathbf{z}) \cdot \boldsymbol{\theta}_{XZ} = \boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}), \tag{3.1}$$

where $\boldsymbol{\theta}_{XZ} = (\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$ and the matrix $\boldsymbol{\Theta}_{XZ}$ is expressed as a vector in row-major form. Expressing the parameters P_{XZ} in terms of the parameters $\boldsymbol{\theta}_X$, $\boldsymbol{\theta}_Z$, and $\boldsymbol{\Theta}_{XZ}$ is often more convenient and intuitive then expressing it in the vector form $\boldsymbol{\theta}_{XZ}$. In general, we refer to the natural parameters $\boldsymbol{\theta}_X$ and $\boldsymbol{\theta}_Z$ as biases, the matrix $\boldsymbol{\Theta}_{XZ}$ as the interaction matrix, and we refer to elements of the interaction matrix $\theta_{XZ,i,j}$ as interactions.

Theorem 3.1 (Second-Order Conditional Specification)**.** *Let $\mathcal{H}_{XZ}$ be the second-order harmonium family defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$ with minimal sufficient statistics. Then the pair of random variables (X, Z) is a second-order harmonium defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$ if and only if $P_{XZ} \in \mathcal{H}_{XZ}$.*

Proof. According to theorem 3, Arnold et al. (2001), the density p_{XZ} of a second-order harmonium defined by exponential families with minimal sufficient statistics may be expressed as

$$\log p_{XZ}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{d_X+1} \sum_{j=1}^{d_Z+1} \bar{s}_{X,i}(\mathbf{x}) \bar{s}_{Z,j}(\mathbf{z}) \cdot \bar{\theta}_{XZ,i,j},$$

where $\bar{\theta}_{XZ,i,j}$ are parameters, where $\bar{s}_{X,i+1}(\mathbf{x}) = s_{X,i}(\mathbf{x})$ and $\bar{s}_{Z,j+1}(\mathbf{z}) = s_{Z,j}(\mathbf{z})$ for $i, j < 1$, and where $s_{X,1}(\mathbf{x}) = s_{Z,1}(\mathbf{z}) = 1$.

Therefore if P_{XZ} is a harmonium defined by exponential families with minimal sufficient statistics, then

$$p_{XZ}(\mathbf{x}, \mathbf{z}) = e^{\sum_{i=1}^{d_X+1} \sum_{j=1}^{d_Z+1} \bar{s}_{X,i}(\mathbf{x}) \bar{s}_{Z,j}(\mathbf{z}) \bar{\theta}_{XZ,i,j}} \propto e^{\mathbf{s}_{XZ}(\mathbf{x},\mathbf{z}) \cdot \boldsymbol{\theta}_{XZ}}$$

for some parameters $\boldsymbol{\theta}_{XZ}$, which implies $P_{XZ} \in \mathcal{H}_{XZ}$.

Conversely, if $P_{XZ} \in \mathcal{H}_{XZ}$, then as we later show in equations 3.5 and 3.6, the conditional distributions satisfy $P_{X|Z} \in \mathcal{M}_X$ and $P_{Z|X} \in \mathcal{M}_Z$, which implies that P_{XZ} is the distribution of a harmonium. □

Graphical models are often composed of a very large number of random variables, and so it may appear as though the previous statements are limited in scope. However, since we define the distributions of our random variables as arbitrary exponential families, we may greatly increase the number of nodes in our graphical model by applying the following proposition.

Proposition 3.2. *Let $(X_i)_{i=1}^n$ be a set of mutually independent random variables, and let $X = (X_i)_{i=1}^n$. Let $(\mathcal{M}_{X_i})_{i=1}^n$ be a sequence of exponential families with base measures $\mu_{X_1}, \dots, \mu_{X_n}$ and sufficient statistics $\mathbf{s}_{X_1}, \dots, \mathbf{s}_{X_n}$, and let $\mathcal{M}_X$ be the exponential family with base measure $\mu_X = \mu_{X_1} \times \cdots \times \mu_{X_n}$ and sufficient statistic $\mathbf{s}_X(\mathbf{x}) = (\mathbf{s}_{X_1}(\mathbf{x}_1), \dots, \mathbf{s}_{X_n}(\mathbf{x}_n))$. Then $P_X \in \mathcal{M}_X$ if and only if $P_{X_i} \in \mathcal{M}_{X_i}$ for every i.*

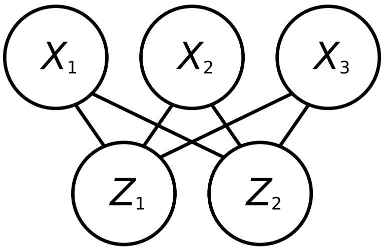

Figure 3.2: Since the distribution of a collection of independent, exponential family random variables is itself an exponential family distribution, second-order harmoniums are often represented by bipartite graphs.

Proof. Suppose that $P_{X_i} \in \mathcal{M}_{X_i}$ with parameters $\boldsymbol{\theta}_{X_i} \in \Theta_{X_i}$ for all i. Because $(X_i)_{i=1}^n$ are mutually independent, the density of the joint vector X is the product density $p_X = \prod_{i=1}^n p_{X_i}$. Therefore

$$p_X(\mathbf{x}) = \prod_{i=1}^{n} p_{X_i}(\mathbf{x}_i) \propto e^{\sum_{i=1}^n \mathbf{s}_{X_i}(\mathbf{x}_i)\cdot\boldsymbol{\theta}_{X_i}} = e^{\mathbf{s}(\mathbf{x})\cdot\boldsymbol{\theta}},$$

which implies that $P_X \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_{X_1}, \dots, \boldsymbol{\theta}_{X_n})$.

Now suppose that $P_X \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_{X_1}, \dots, \boldsymbol{\theta}_{X_n})$. Then for any i,

$$\begin{aligned}
p_{X_i}(\mathbf{x}_i) &= \int_{\{\Omega_{X_j}\}_{j\neq i}} p_X(\mathbf{x}_1, \dots, \mathbf{x}_n)\mu(d\mathbf{x}_1, \dots d\mathbf{x}_{i-1}, d\mathbf{x}_{i+1}, \dots, \mathbf{x}_n) \\
&\propto \int_{\{\Omega_{X_j}\}_{j\neq i}} e^{\mathbf{s}_X(\mathbf{x})\cdot\boldsymbol{\theta}}\mu(d\mathbf{x}_1, \dots d\mathbf{x}_{i-1}, d\mathbf{x}_{i+1}, \dots, \mathbf{x}_n) \\
&= e^{\mathbf{s}_{X_i}(\mathbf{x}_i)\cdot\boldsymbol{\theta}_{X_i}} \int_{\{\Omega_{X_j}\}_{j\neq i}} e^{\sum_{j\neq i}\mathbf{s}_{X_j}(\mathbf{x}_j)\cdot\boldsymbol{\theta}}\mu(d\mathbf{x}_1, \dots d\mathbf{x}_{i-1}, d\mathbf{x}_{i+1}, \dots, \mathbf{x}_n) \\
&\propto e^{\mathbf{s}_{X_i}(\mathbf{x}_i)\cdot\boldsymbol{\theta}_{X_i}},
\end{aligned}$$

which implies that $P_{X_i} \in \mathcal{M}_{X_i}$ with parameters $\boldsymbol{\theta}_{X_i}$. ☐

This proposition effectively states that products of independent exponential family distributions are themselves exponential family distributions. As first presented by Welling et al. (2005), the exponential family harmonium is a graphical model which can be represented by a bipartite graph. By combining proposition 3.2 with theorem 3.1, we may recover the exponential family harmonium as the most general model whose conditional distributions are given by products of exponential family distributions.

It is also interesting to note that theorem 3.1 and proposition 3.2 essentially define two forms of product manifold, and it will sometimes be helpful to formalize these products as operators. On one hand, a harmonium family is the result of a tensor product of the sufficient statistic of the constituent exponential families, and I will sometimes denote the harmonium family $\mathcal{H}_{XZ}$ defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$ by $\mathcal{H}_{XZ} = \mathcal{M}_X \otimes \mathcal{M}_Z$. On the other hand, the exponential family $\mathcal{M}_X$ with sufficient statistic $\mathbf{s}_X = (\mathbf{s}_{X_1}, \mathbf{s}_{X_2})$ and base measure $\mu_X = \mu_{X_1} \times \mu_{X_2}$ is the set of all product distributions $P_{X_1} \cdot P_{X_2}$ where $P_{X_1} \in \mathcal{M}_{X_1}$ and $P_{X_2} \in \mathcal{M}_{X_2}$. I will refer to these exponential families as product exponential families, and sometimes denote them by $\mathcal{M}_X = \mathcal{M}_{X_1} \cdot \mathcal{M}_{X_2}$.

3.1.2 Properties of Harmoniums

Let us consider the second-order harmonium family $\mathcal{H}_{XZ}$ defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$. In order to apply the theory of exponential family graphical models developed in section 2.4 to harmonium families, we must derive the conditional log-partition function of harmonium families (2.15), and thereby express the marginal (2.14) and conditional (2.16) distributions of harmoniums.

Based on equation 3.1, we may express the conditional log-partition function $\psi_{X|Z}$ at $\boldsymbol{\theta}_{XZ} \in \Theta_{XZ}$ and $\mathbf{z} \in \Omega_Z$ as

$$\begin{aligned}\psi_{X|Z}(\boldsymbol{\theta}_{XZ}, \mathbf{z}) &= \log \int_{\Omega_X} e^{\boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})} \mu_X(d\mathbf{x}) \\ &= \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \log \int_Z e^{(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) \cdot \mathbf{s}_Z(\mathbf{z})} \mu_X(d\mathbf{x}) \\ &= \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \psi_X(\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})). \end{aligned} \tag{3.2}$$

By inserting this equation into the definition of the marginal density of an exponential family graphical model (2.14), we may conclude that the marginal density of the observable variables of the harmonium is given by

$$\log p_Z(\mathbf{z}) = \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \psi_X(\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})) - \psi_{XZ}(\boldsymbol{\theta}_{XZ}). \tag{3.3}$$

Conversely, the marginal density of the latent variables is given by

$$\log p_X(\mathbf{x}) = \boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) - \psi_{XZ}(\boldsymbol{\theta}_{XZ}). \tag{3.4}$$

In the language of Bayesian inference, P_X is the prior distribution of the harmonium (X, Z).

By inserting equation 3.2 into equation 2.16, the density of the latent variables given the observable variables of the harmonium is given by

$$\begin{aligned}&\log p_{X|Z}(\mathbf{x} \mid \mathbf{z}) \\ &\quad = \mathbf{s}_{XZ}(\mathbf{x}, \mathbf{z}) \cdot \boldsymbol{\theta}_{XZ} - \psi_{X|Z}(\boldsymbol{\theta}_{XZ}, \mathbf{z}) \\ &\quad = \boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}) \\ &\qquad\quad - \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) - \psi_X(\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})) \\ &\quad = \mathbf{s}_X(\mathbf{x}) \cdot (\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})) - \psi_X(\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})), \end{aligned} \tag{3.5}$$

which is the exponential family distribution $P_{X|Z=\mathbf{z}} \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})$. Again, in Bayesian language this is the posterior of the harmonium. Conversely, the likelihood of the harmonium is the conditional distribution $P_{Z|X}$ given by

$$\log p_{Z|X}(\mathbf{z} \mid \mathbf{x}) = \mathbf{s}_Z(\mathbf{z}) \cdot (\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) - \psi_Z(\boldsymbol{\theta}_X + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}). \tag{3.6}$$

Observe how the likelihood and posterior distributions of a any harmonium always have a log-linear form. That the posterior of a harmonium can be tractably computed is an important advantage of harmoniums in the context of Bayesian inference.

Since we will often work with product exponential families, it helps to consider their corresponding log-partition functions. Where $(\mathcal{M}_{X_i})_{i=1}^n$ is a sequence of exponential families with base measures $\mu_{X_1}, \ldots, \mu_{X_n}$ and sufficient statistics $\mathbf{s}_{X_1}, \ldots, \mathbf{s}_{X_n}$, and where $\mathcal{M}_X$ is the exponential family with base measure $\mu_{X_1} \times \cdots \times \mu_{X_n}$ and sufficient statistic defined by $\mathbf{s}_X(\mathbf{x}) = (\mathbf{s}_{X_1}(\mathbf{x}_1), \dots, \mathbf{s}_{X_n}(\mathbf{x}_n))$, the log-partition function ψ_X may be expressed as

$$\begin{aligned}\psi_X(\boldsymbol{\theta}_X) &= \log \int_X e^{\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\theta}_X} \mu(d\mathbf{x}) \\ &= \sum_{i=1}^{n} \log \int_{X_i} e^{\mathbf{s}_{X_i}(\mathbf{x}_i) \cdot \boldsymbol{\theta}_{X_i}} \mu_{X_i}(d\mathbf{x}_i) \\ &= \sum_{i=1}^{n} \psi_{X_i}(\boldsymbol{\theta}_{X_i}). \end{aligned} \tag{3.7}$$

In later sections we make use of the fact that a harmonium family is an exponential family. In particular, when a second-order harmonium family is defined by a pair of exponential families with minimal sufficient statistics, then the sufficient statistic of the second-order harmonium family is itself minimal.

Proposition 3.3. *Let $\mathcal{H}_{XZ}$ be a second-order harmonium family defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$ with minimal sufficient statistics. Then the sufficient statistic of $\mathcal{H}_{XZ}$ is minimal.*

Proof. By way of contradiction, suppose that $\mathcal{M}_X$ and $\mathcal{M}_Z$ have minimal sufficient statistics, but that $\mathcal{H}_{XZ}$ does not. Then for some α_0 and non-zero $\boldsymbol{\alpha}_{XZ} = (\boldsymbol{\alpha}_X, \boldsymbol{\alpha}_Z, \mathbf{A}_{XZ})$,

$$\begin{aligned} & \alpha_0 = \boldsymbol{\alpha}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\alpha}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{s}_X(\mathbf{x}) \cdot \mathbf{A}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}) \\ \iff \quad & \boldsymbol{\alpha}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\alpha}_Z \cdot \mathbf{s}_Z(\mathbf{z}) - \alpha_0 = -\mathbf{s}_X(\mathbf{x}) \cdot \mathbf{A}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}). \end{aligned}$$

Note that the right hand side is a bilinear function of $\mathbf{s}_X(\mathbf{x})$ and $\mathbf{s}_Z(\mathbf{z})$, which implies that the left hand side is a bilinear function of $\mathbf{s}_X(\mathbf{x})$ and $\mathbf{s}_Z(\mathbf{z})$. Therefore

$$\begin{aligned} & \boldsymbol{\alpha}_X \cdot (\mathbf{s}_X(\mathbf{x}) + \mathbf{s}_X(\mathbf{x})) + \boldsymbol{\alpha}_Z \cdot \mathbf{s}_Z(\mathbf{z}) - \alpha_0 \\ & = \boldsymbol{\alpha}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\alpha}_X \cdot \mathbf{s}_X(\mathbf{x}) + 2\boldsymbol{\alpha}_Z \cdot \mathbf{s}_Z(\mathbf{z}) - 2\alpha_0 \\ \iff & \boldsymbol{\alpha}_Z \cdot \mathbf{s}_Z(\mathbf{z}) = \alpha_0. \end{aligned}$$

Because $\mathbf{s}_Z$ is minimal, this implies that $\boldsymbol{\alpha}_Z = \mathbf{0}$, and therefore that $\alpha_0 = 0$. Similarly, because $\mathbf{s}_X$ is minimal, we may conclude that $\boldsymbol{\alpha}_X = \mathbf{0}$, and therefore that $\mathbf{s}_X(\mathbf{x}) \cdot \mathbf{A}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}) = \mathbf{0}$.

Finally, because $\mathbf{s}_X(\mathbf{x})$ is minimal, $\mathbf{s}_X(\mathbf{x}) \cdot (\mathbf{A}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})) = 0$ if and only if $\mathbf{A}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}) = \mathbf{0}$. Because $\mathbf{s}_Z(\mathbf{z})$ is minimal, this equation is true if and only $\mathbf{A}_{XZ} = \mathbf{0}$. This implies that $\boldsymbol{\alpha}_{XZ} = (\boldsymbol{\alpha}_X, \boldsymbol{\alpha}_Z, \mathbf{A}_{XZ}) = \mathbf{0}$, which contradicts our assumption. □

Within the context of harmonium families, it is also important to be able to establish the conditions under which the random variables in question are independent. In the case of harmonium families with minimal sufficient statistics, this is simply the case when all the interaction parameters are zero.

Proposition 3.4. *Let $\mathcal{H}_{XZ}$ be a second-order harmonium family defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$ with minimal sufficient statistics, and suppose that (X, Z) is a harmonium with distribution $P_{XZ} \in \mathcal{H}_{XZ}$ and parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$. Then X and Z are independent if and only if $\boldsymbol{\Theta}_{XZ} = \mathbf{0}$.*

Proof. First note that $\Leftarrow$ is trivial, and so we only need to prove $\Rightarrow$. Based on equations 3.1, 3.4, and 3.3, X and Z are independent if and only if

$$\begin{aligned}
&p_{XZ}(\mathbf{x}, \mathbf{z}) = p_X(\mathbf{x}) p_Z(\mathbf{z}) \\
\iff &e^{\boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})} \\
&\quad \propto e^{\boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) + \psi_X(\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}))} \\
\iff &\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}) \\
&\quad = \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) + \psi_X(\boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})) + c.
\end{aligned}$$

The left hand side of this equation is a bilinear form in $\mathbf{s}_X(\mathbf{x})$ and $\mathbf{s}_Z(\mathbf{z})$, whereas the right hand side is not unless it is equal to zero. Because $\mathcal{M}_X$ and $\mathcal{M}_Z$ have minimal sufficient statistics, proposition 3.3 therefore implies that $\boldsymbol{\Theta}_{XZ} = \mathbf{0}$. ☐

In order to minimize the relative entropy (2.17) of some target distribution with respect to the marginal distribution of the harmonium model Q_Z we follow the gradient in equation 2.18. The derivatives of this relative entropy with respect to the parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$, are given by

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_X} D(P_Z \parallel Q_Z) &= \mathbb{E}_Q[\mathbf{s}_X(X)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_X(X) \mid Z]], \\
\partial_{\boldsymbol{\theta}_Z} D(P_Z \parallel Q_Z) &= \mathbb{E}_Q[\mathbf{s}_Z(Z)] - \mathbb{E}_P[\mathbf{s}_Z(Z)], \\
\partial_{\boldsymbol{\Theta}_{XZ}} D(P_Z \parallel Q_Z) &= \\
&\mathbb{E}_Q[\mathbf{s}_X(X) \otimes \mathbf{s}_Z(Z)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_X(X) \otimes \mathbf{s}_Z(Z) \mid Z]], \qquad (3.8)
\end{aligned}$$

where $\mathbf{s}_Z(Z) \otimes \mathbf{s}_Y(Y)$ is the outer product matrix defined by

$$(\mathbf{s}_Z(Z) \otimes \mathbf{s}_Y(Y))_{i,j} = s_{X,i}(X) s_{Z,j}(Z).$$

The expectations which define these derivatives can rarely be evaluated exactly, however we can approximate them with Markov chain Monte Carlo (MCMC). MCMC methods generate samples from a complicated target distribution by generating samples from a more tractable Markov chain, the equilibrium distribution of which is equal to the target distribution. Gibbs sampling is a form of MCMC where the transition distribution of the Markov chain is given by the conditional distributions of the target distribution (Geman and Geman, 1984; Casella and George, 1992).

Consider the harmonium (X, Z) with distribution $P_{XZ} \in \mathcal{H}_{XZ}$. We define the Gibbs sampler for (X, Z) as the Markov chain $(X_k, Z_k)_{k \in \mathbb{N}}$ with transition distribution

$$p_{X'Z'|X,Z}(\mathbf{x}', \mathbf{z}' \mid \mathbf{x}, \mathbf{z}) = p_{X|Z}(\mathbf{x}' \mid \mathbf{z}') p_{Z|X}(\mathbf{z}' \mid \mathbf{x}).$$

Based on the well-developed theory of Gibbs sampling, it is easy to apply the ergodic theory of section 2.5 to this Markov chain, and thereby estimate the expectations in equations 3.8.

Proposition 3.5. *The Gibbs sampler $(X_k, Z_k)_{k \in \mathbb{N}}$ for the harmonium (X, Z) is aperiodic and Harris recurrent, with equilibrium distribution $\Pi_{XZ} = P_{XZ}$.*

Proof. The aperiodicity of the sampler follows directly from definition 2.8. Harris recurrence follows from theorem 9.4 of Robert and Casella (2004) as a result of the positivity condition.

As defined in Robert and Casella (2004), the harmonium (X, Z) satisfies the positivity condition if $p_X(\mathbf{x}) > 0 \wedge p_Z(\mathbf{z}) > 0 \implies p_{XZ}(\mathbf{x}, \mathbf{z}) > 0$. This is trivially true for harmoniums since p_X, p_Z, and p_{XZ} are all strictly positive on their domains Ω_X,Ω_Z, and $\Omega_X \times \Omega_Z$. $\square$

Note that because a Gibbs sampler is defined by the conditional distributions $P_{X|Z}$ and $P_{Z|X}$, and the unique equilibrium distribution of the sampler is P_{XZ}, this implies that $P_{X|Z}$ and $P_{Z|X}$ are sufficient for uniquely determining the corresponding joint distribution P_{XZ}.

Although Gibbs sampling can estimate the expectations in equation 3.8, it can take a long time for the Gibbs sampler to converge. This can be problematic when training a model, which often requires evaluating millions of steps of stochastic gradient descent. Contrastive divergence minimization is an MCMC approach to fitting second-order harmoniums to data based on an alternative objective (Hinton, 2002). The n-step contrastive divergence is

$$D_n(P_Z \parallel Q_Z) = D(P_Z \parallel Q_Z) - D(P_{Z_n} \parallel Q_Z),$$

where (X_n, Z_n) is the nth step of the Gibbs sampler with transition density

$$p_{X'Z'|X,Z}(\mathbf{x}', \mathbf{z}' \mid \mathbf{x}, \mathbf{z}) = q_{X|Z}(\mathbf{x}' \mid \mathbf{z}') q_{Z|X}(\mathbf{z}' \mid \mathbf{x}),$$

and initial distribution $P_{X_0 Z_0}$ with density $p_{X_0 Z_0} = q_{X|Z} \cdot p_Z$. It can be shown that this objective can be approximately minimized by following the derivatives

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_X} D_n(P_Z \parallel Q_Z) &= \mathbb{E}_P[\mathbf{s}_X(X_n)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_X(X) \mid Z]], \\
\partial_{\boldsymbol{\theta}_Z} D_n(P_Z \parallel Q_Z) &= \mathbb{E}_P[\mathbf{s}_Z(Z_n)] - \mathbb{E}_P[\mathbf{s}_Z(Z)], \\
\partial_{\boldsymbol{\Theta}_{XZ}} D_n(P_Z \parallel Q_Z) &= \\
&\mathbb{E}_P[\mathbf{s}_X(X_n) \otimes \mathbf{s}_Z(Z_n)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_X(X) \otimes \mathbf{s}_Z(Z) \mid Z]],
\end{aligned}$$

which can be estimated with only a n steps of the Gibbs sampler. Contrastive divergence minimization has the attractive feature that the contrastive divergence objective converges to the relative entropy objective in the limit as n goes to infinity. In practice, contrastive divergence has been highly effective for training harmonium models even for $n = 1$.

3.2 Higher-Order Harmoniums

In this section I introduce nth-order harmoniums, which are distributions over n random variables, where the conditional distribution of each random variable given all the others is always an element of the same exponential family. I show that the set of all nth-order harmoniums of a particular kind is an exponential family with a particular tensor structure. I then show that an nth-order harmonium can be interpreted as a second-order harmonium of harmoniums of order i and j, where $n = i + j$, in a manner which preserves the graphical model structure of the harmonium family in question. This allows us to reduce much of the theory of nth-order harmoniums to the second-order case. Finally, I introduce deep harmoniums, which are a form of harmonium family which can be represented by a hierarchical graph.

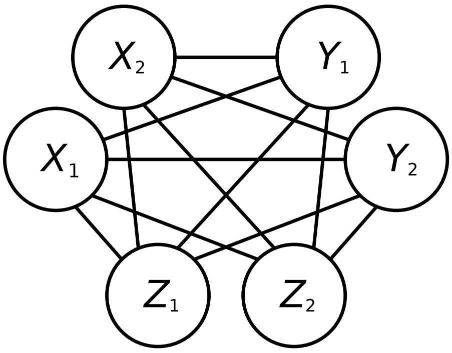

Figure 3.3: A general third-order harmonium can be represented by a tripartite graph.

3.2.1 Higher-Order Conditional Specification

Let us now move beyond the two random variable case, and consider a tuple of n random variables, where the conditional distribution of each random variable given all the others is always a member of the same exponential family.

Definition 3.2 (Higher-Order Harmoniums)**.** The n-tuple of random variables $(X_i)_{i=1}^n$ is an nth-order harmonium defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$, if for every i, $1 \leq i \leq n$, the conditional distribution satisfies $P_{X_i|(X_j)_{j \neq i}} \in \mathcal{M}_{X_i}$.

As in section 3.1.1, I define an exponential family which is the set of all nth-order harmoniums. In this section, however, I define the corresponding exponential family recursively. In particular, the nth-order harmonium family $\mathcal{H}_{(X_i)_{i=1}^n}$ defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$ is the second-order harmonium family defined by the exponential families $\mathcal{H}_{(X_i)_{i=1}^{n-1}}$ and $\mathcal{M}_{X_n}$, where $\mathcal{H}_{X_1} = \mathcal{M}_{X_1}$. Based on this definition, we may generalize theorem 3.1 to the case of higher-order harmoniums.

Theorem 3.6 (Higher-Order Conditional Specification)**.** *Let $\mathcal{H}_{(X_i)_{i=1}^n}$ be the nth-order harmonium family defined by the sequence of exponential families $(\mathcal{M}_{X_i})_{i=1}^n$ with minimal sufficient statistics. Then $(X_i)_{i=1}^n$ is an nth-order harmonium defined by $(\mathcal{M}_{X_i})_{i=1}^n$ if and only if $P_{(X_i)_{i=1}^n} \in \mathcal{H}_{(X_i)_{i=1}^n}$.*

Proof. To show $\Rightarrow$, suppose that $(X_i)_{i=1}^n$ is an nth-order harmonium defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$. Then by theorem 3.1, the conditional distribution $P_{X_1X_2|(X_i)_{i\neq 1,2}} \in \mathcal{H}_{X_1X_2}$, and by proposition 3.3, the sufficient statistic of the harmonium family $\mathcal{H}_{X_1X_2}$ is minimal. Since $P_{X_3|(X_i)_{i\neq 3}} \in \mathcal{M}_{X_3}$ and harmonium families are exponential families, again by theorem 3.1, $P_{(X_1X_2)X_3|(X_i)_{i\neq 1,2,3}} \in \mathcal{H}_{(X_1X_2)X_3}$, and the sufficient statistic of $\mathcal{H}_{(X_1X_2)X_3}$ is minimal. By repeated application of theorem 3.1 we find that that $P_{(X_i)_{i=1}^{n-1}X_n} \in \mathcal{H}_{(X_i)_{i=1}^{n-1}X_n}$, which is to say that $P_{(X_i)_{i=1}^n} \in \mathcal{H}_{(X_i)_{i=1}^n}$.

To show $\Leftarrow$, suppose that $\mathcal{H}_{(X_i)_{i=1}^n}$ is the nth-order harmonium family defined by the sequence of exponential families $(\mathcal{M}_{X_i})_{i=1}^n$. Then for any $P_{(X_i)_{i=1}^n} \in \mathcal{H}_{(X_i)_{i=1}^n}$, $P_{X_n|(X_i)_{i=1}^{n-1}} \in \mathcal{M}_{X_n}$, and $P_{(X_i)_{i=1}^{n-1}|X_n} \in \mathcal{H}_{(X_i)_{i=1}^{n-1}}$. Because $P_{(X_i)_{i=1}^{n-1}|X_n}$ is in the $(n-1)$th-order harmonium family $\mathcal{H}_{(X_i)_{i=1}^{n-1}}$, this implies that $P_{X_{n-1}|(X_i)_{i=1}^{n-2}X_n} \in \mathcal{H}_{X_{n-1}}$, and $P_{(X_i)_{i=1}^{n-2}|X_{n-1}X_n} \in \mathcal{H}_{(X_i)_{i=1}^{n-2}}$. By recursion to the second-order case, $P_{X_i|(X_j)_{j\neq i}} \in \mathcal{M}_{X_i}$ for any i, and therefore any element of $\mathcal{H}_{(X_i)_{i=1}^n}$ is the distribution of an nth-order harmonium defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$. $\square$

In figure 3.3 I depict the graph of a third-order harmonium defined by three product exponential families.

3.2.2 Harmonium Factorization

We have defined nth-order harmonium families as second-order harmonium families over a lower-order harmonium family and an exponential family. Because harmonium families are exponential families, this suggests that we may partition an nth-order harmonium into a harmonium of harmoniums. Indeed, with the next theorem I show that any nth-order harmonium can ultimately be understood as a second-order harmonium of harmoniums of orders i and j, where $n = i + j$, and that this factorization procedure preserves the graphical model structure of the harmonium family in question.

Establishing this result requires the notion of a partition of a set I, which I define as a pair of sets $J, K \subset I$, which satisfy $J \cap K = \emptyset$ and $J \cup K = I$. To simplify the notation in this subsection, I will denote indexed subsequences of random variables by $X_I = (X_i)_{i \in I}$. Moreover, to describe the restriction of a harmonium family to the undirected graph $G = (I, E)$ with vertices $I = \{1, \dots, n\}$, I use the notation

$$\mathcal{H}^G_{(X_i)_{i=1}^n} = \{P_{(X_i)_{i=1}^n} \in \mathcal{H}_{(X_i)_{i=1}^n} \mid (X_i)_{i=1}^n \text{is represented by G}\},$$

to indicate the subset of the harmonium family $\mathcal{H}_{(X_i)_{i=1}^n}$ which contains all Markov random fields represented by G.

Theorem 3.7 (The Harmonium Factorization Theorem)**.** *Let $(X_i)_{i=1}^n$ be a sequence of random variables, let $(\mathcal{M}_{X_i})_{i=1}^n$ be a sequence of exponential families with minimal sufficient statistics, and let $J, K \subset I$ be any partition of the indices $I = \{1, \dots, n\}$. Moreover, let $G = (I, E)$ be an undirected graph, and let F and H be the subgraphs of G with vertices J and K. Then*

1. *$P_{X_I} \in \mathcal{H}_{X_I}$ if and only if $P_{X_J X_K} \in \mathcal{H}_{X_J} \otimes \mathcal{H}_{X_K}$.*
2. *If $P_{X_I} \in \mathcal{H}^G_{X_I}$, then $P_{X_J X_K} \in \mathcal{H}^F_{X_J} \otimes \mathcal{H}^H_{X_K}$.*

Proof. 1. First, to prove $\Rightarrow$, suppose that $(X_i)_{i=1}^n$ is an nth-order harmonium, and let J and K be an arbitrary partition of the indices I. Because X_I is an nth-order harmonium, theorem 3.6 implies that $P_{X_J|X_K}$ is in the harmonium family $\mathcal{H}_{X_J}$ defined by the exponential families $(\mathcal{M}_{X_j})_{j \in J}$, and similarly that $P_{X_K|X_J}$ is in the harmonium family $\mathcal{H}_{X_K}$. Since harmonium families are also exponential families, the conditional distributions of X_J given X_K and X_K given X_J are always in the exponential families $\mathcal{H}_{X_J}$ and $\mathcal{H}_{X_K}$, which is to say that (X_J, X_K) is a second-order harmonium defined by $\mathcal{H}_{X_J}$ and $\mathcal{H}_{X_K}$.

To prove $\Leftarrow$, suppose that (X_J, X_K) is a second-order harmonium defined by the exponential families $\mathcal{H}_{X_J}$ and $\mathcal{H}_{X_K}$. For any $j \in J$, the conditional distribution $P_{X_j|(X_l)_{l \neq j}}$ is a conditional distribution of the conditional distribution $P_{X_J|X_K}$, and similarly $P_{X_k|(X_l)_{k \neq l}}$ is a conditional distribution of $P_{X_K|X_J}$ for $k \in K$. Moreover, since $P_{X_J|X_K} \in \mathcal{H}_{X_J}$ and $P_{X_K|X_J} \in \mathcal{H}_{X_K}$, $P_{X_J|X_K}$ and $P_{X_K|X_J}$ are higher-order harmonium distributions, and therefore the conditional distributions $P_{X_j|(X_k)_{k \neq j}}$ and $P_{X_k|(X_j)_{j \neq k}}$ are always elements of $\mathcal{M}_{X_j}$ and $\mathcal{M}_{X_k}$ for any $j \in J$ or $k \in K$. Thus for any $1 \leq i \leq n$, $P_{X_i|(X_l)_{l \neq i}} \in \mathcal{M}_{X_i}$, which is to say that $(X_i)_{i=1}^n$ is an nth-order harmonium defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$, and therefore in $\mathcal{H}_{X_I}$.

2. If $P_{X_I} \in \mathcal{H}^G_{X_I}$, then X_I is a Markov random field represented by G, and *1.* implies that $P_{X_J|X_K} \in \mathcal{H}_{X_J}$ and $P_{X_K|X_J} \in \mathcal{H}_{X_K}$. Let us assume that l and m are in J. If X_l is conditionally independent of X_m given $(X_j)_{j \neq l,m}$, then every path in G

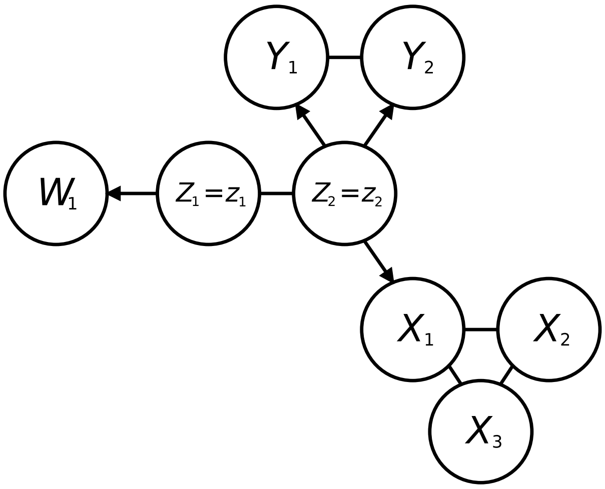

Figure 3.4: Suppose that the random variables W_1, X_1, X_2, X_3, Y_1, Y_2, Z_1, and Z_2 depicted in this graph form an 8th-order harmonium. Then according to the harmonium factorization theorem (3.7) and equation 3.6, X_1, X_2, X_3, Y_1, Y_2, and W_1 conditioned on Z_1 and Z_2 is a 6th-order harmonium. Since there are no edges across the lettered groups of unconditioned nodes, this 6th-order harmonium is in turn the product of three independent harmoniums, (X_1, X_2, X_3), (Y_1, Y_2), and W_1, of order 3, 2, and 1, respectively.

between vertex l and m passes through at least one additional vertex $j \neq l, m$. Since a subgraph cannot have a path length between two vertices which is shorter than the path length between the vertices in the total graph, X_l and X_m remain conditionally independent according to the representation F, and therefore $P_{X_J|X_K} \in \mathcal{H}^F_{X_J}$. *Mutatis mutandis*, this same argument implies that $P_{X_K|X_J} \in \mathcal{H}^H_{X_K}$. □

Note that in the second part of this theorem, $P_{X_J X_K} \in \mathcal{H}^F_{X_J} \otimes \mathcal{H}^H_{X_K}$ does not imply $P_{X_I} \in \mathcal{H}^G_{X_I}$, because the graphical representation of $\mathcal{H}^F_{X_J} \otimes \mathcal{H}^H_{X_K}$ may contain more edges than $\mathcal{H}^G_{X_I}$.

The first part of this theorem effectively states that the operator $\otimes$ is associative, such that

$$\mathcal{M}_X \otimes \mathcal{M}_Y \otimes \mathcal{M}_Z = (\mathcal{M}_X \otimes \mathcal{M}_Y) \otimes \mathcal{M}_Z = \mathcal{M}_X \otimes (\mathcal{M}_Y \otimes \mathcal{M}_Z).$$

For example, the third-order harmonium (X, Y, Z) defined by the exponential families $\mathcal{M}_X$, $\mathcal{M}_Y$, and $\mathcal{M}_Z$ can be factorized into either the second-order harmonium $((X, Y), Z)$ defined by $\mathcal{M}_X \otimes \mathcal{M}_Y$ and $\mathcal{M}_X$, or into the second-order harmonium $(X, (Y, Z))$ defined by $\mathcal{M}_X$ and $\mathcal{M}_Y \otimes \mathcal{M}_Z$.

Now, marginalization and conditionalization are arguably the two most fundamental operations in probability theory, and both involve partitioning collections of random variables into two groups. Because any partition of the random variables of an nth-order harmonium results in a second-order harmonium of harmoniums, we may use the properties derived for second-order harmoniums in section 3.1.2 to express any marginal or conditional distribution of an nth-order harmonium. I depict an example of conditionalization in a complex harmonium in figure 3.4.

The harmonium factorization theorem establishes properties about the abstract distributions of harmoniums and their factorizations, but practically we need to relate these properties to the natural parameters of the harmoniums and their families. In order to describe this, we require a general notion of an interaction.

Definition 3.3 (Interaction)**.** Let $(X_i)_{i=1}^n$ be an nth-order harmonium defined by $(X_i)_{i=1}^n$, with distribution $P_{(X_i)_{i=1}^n} \in \mathcal{H}_{(X_i)_{i=1}^n}$ and natural parameters $\boldsymbol{\theta}_{(X_i)_{i=1}^n}$. Then the ith natural parameter $\theta_{(X_i)_{i=1}^n,i}$ is an interaction between X_j and X_k if the ith summand in the dot product $\boldsymbol{\theta}_{(X_i)_{i=1}^n} \cdot \mathbf{s}_{(X_i)_{i=1}^n}$ contains both $\mathbf{s}_{X_j}$ and $\mathbf{s}_{X_k}$. A natural parameter that is not an interaction is a bias.

Given this definition, we may conclude that any interaction between two non-adjacent random variables in a harmonium Markov random field is 0.

Lemma 3.8. *Let $\mathcal{H}_{(X_i)_{i=1}^n}$ be an nth-order harmonium family with a minimal sufficient statistic, and suppose that $(X_i)_{i=1}^n$ is an nth-order harmonium with distribution $P_{(X_i)_{i\in n}} \in \mathcal{H}_{(X_i)_{i=1}^n}$ and parameters $\boldsymbol{\theta}_{(X_i)_{i=1}^n}$. Then X_i and X_j are conditionally independent given the remaining random variables $(X_k)_{k\neq i,k}$ if and only if any interaction between X_i and X_j is 0.*

Proof. Note that $\Leftarrow$ follows directly from the Hammersley-Clifford theorem, and so we only need to prove $\Rightarrow$.

Suppose that $P_{X_iX_j|(X_k)_{k\neq i,j}} = P_{X_i|(X_k)_{k\neq i,j}} \cdot P_{X_j|(X_k)_{k\neq i,j}}$, and consider the factorization $((X_i, X_j), (X_k)_{k\neq i,j})$. By theorem 3.7, the conditional distribution $P_{X_iX_j|(X_k)_{k\neq i,j}}$ at $(\mathbf{x}_k)_{k\neq i,j}$ is an element of the second-order harmonium family $\mathcal{H}_{X_iX_j}$ with natural parameters

$$\boldsymbol{\theta}^*_{X_iX_j} = \boldsymbol{\theta}_{X_iX_j} + \boldsymbol{\Theta}_{((X_i,X_j),(X_k)_{k\neq i,j})} \cdot \mathbf{s}_{(X_k)_{k\neq i,j}}((\mathbf{x}_k)_{k\neq i,j}).$$

According to equation 3.1, we may express $\boldsymbol{\theta}^*_{X_iX_k}$ as $(\boldsymbol{\theta}^*_{X_i}, \boldsymbol{\theta}^*_{X_j}, \boldsymbol{\Theta}^*_{X_iX_j})$. Note that by construction, $\boldsymbol{\Theta}^*_{X_iX_j}$ contains all the interaction parameters between X_i and X_j. Because X_i and X_j are conditionally independent, proposition 3.4 implies that $\boldsymbol{\Theta}_{X_iX_k} = \mathbf{0}$, which proves the lemma. ☐

This lemma is essentially a version of the Hammersley-Clifford theorem specific to the exponential family structure of harmoniums. Although this lemma tells us about the graphical structure of a single harmonium, it does not tell us the relationship between a graphical model and a harmonium family. In order to describe this relationship, I use $\boldsymbol{\theta}^G_{(X_i)_{i=1}^n}$ and $\mathbf{s}^G_{(X_i)_{i=1}^n}$ to denote the natural parameters and sufficient statistic which contain all the interactions and corresponding statistics between random variables which are elements of a shared maximal clique. Formally, where $\mathcal{C}$ is the set of maximal cliques of G, I define the subindices J such that $j \in J$ if and only if $\theta_{(X_i)_{i=1}^n,j}$ is a bias, or for any $C \in \mathcal{C}$ and any $k, l \in C$, $\theta_{(X_i)_{i=1}^n,j}$ is an interaction between X_k and X_l. Given J, I then define $\boldsymbol{\theta}^G_{(X_i)_{i=1}^n} = (\theta_{(X_i)_{i=1}^n,j})_{j\in J}$ and $\mathbf{s}^G_{(X_i)_{i=1}^n} = (s_{(X_i)_{i=1}^n,j})_{j\in J}$.

Theorem 3.9. *Suppose that $\mathcal{M}_{(X_i)_{i=1}^n}$ is the nth-order harmonium defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$ with minimal sufficient statistics, and let $G = (I, E)$ be an undirected graph. Then the subset $\mathcal{H}^G_{(X_i)_{i=1}^n}$ of $\mathcal{H}_{(X_i)_{i=1}^n}$ is the exponential family with base measure $\mu_{(X_i)_{i=1}^n}$ and minimal sufficient statistic $\mathbf{s}^G_{(X_i)_{i=1}^n}$.*

Proof. $P_{(X_i)_{i=1}^n}$ is in the exponential family with base measure $\mu_{(X_i)_{i=1}^n}$ and sufficient statistic $\mathbf{s}^G_{(X_i)_{i=1}^n}$ if and only if

$$p_{(X_i)_{i=1}^n}(\mathbf{x}_1, \dots, \mathbf{x}_n) \propto e^{\mathbf{s}^G_{(X_i)_{i=1}^n}(\mathbf{x}_1,\dots,\mathbf{x}_n)\cdot\boldsymbol{\theta}^G_{(X_i)_{i=1}^n}} = e^{\mathbf{s}_{(X_i)_{i=1}^n}(\mathbf{x}_1,\dots,\mathbf{x}_n)\cdot\boldsymbol{\theta}_{(X_i)_{i=1}^n}},$$

which is true if an only if $P_{(X_i)_{i=1}^n} \in \mathcal{H}_{(X_i)_{i=1}^n}$, where $\boldsymbol{\theta}_{(X_i)_{i=1}^n}$ is 0 at every interaction not included in $\boldsymbol{\theta}^G_{(X_i)_{i=1}^n}$. By lemma 3.8 and the definition of a Markov random field, this is true if and only if $P_{(X_i)_{i=1}^n} \in \mathcal{H}^G_{(X_i)_{i=1}^n}$.

Moreover, because $\mathbf{s}_{(X_i)_{i=1}^n}$ is minimal, any subsequence of $\mathbf{s}_{(X_i)_{i=1}^n}$ is minimal, and therefore so is $\mathbf{s}^G_{(X_i)_{i=1}^n}$. □

3.2.3 Deep Harmoniums

Hierarchical graphical models model dependencies between large numbers of random variables, while keeping the maximum clique size of the corresponding graphical representation low. A deep harmonium, in particular, is a graphical model over a collection of random variables with only second-order interactions.

Definition 3.4 (Deep Harmonium)**.** The sequence of random variables $(X_i)_{i=1}^n$ is a deep harmonium if $(X_i)_{i=1}^n$ is a nth-order harmonium, and for any $0 < i < j < k \leq n$, X_i and X_k are conditionally independent given X_j.

I depict the graph of a deep harmonium with three variables in figure 3.5. The family of deep harmoniums is a direct result of theorem 3.9, but for notational simplicity I will denote the deep harmonium family defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$ by $\mathcal{D}_{(X_i)_{i=1}^n}$. In the case of the three-layer deep harmonium family $\mathcal{D}_{XYZ}$ defined by $\mathcal{M}_X$, $\mathcal{M}_Y$, and $\mathcal{M}_Z$, the sufficient statistic $\mathbf{s}_{XYZ}$ of $\mathcal{D}_{XYZ}$ is given by

$$\begin{aligned}
s_{XYZ,i}(\mathbf{x},\mathbf{y},\mathbf{z}) =& s_{X,i}(\mathbf{x}),\\
s_{XYZ,d_X+j}(\mathbf{x},\mathbf{y},\mathbf{z}) =& s_{Y,j}(\mathbf{y}),\\
s_{XYZ,d_X+d_Y+k}(\mathbf{x},\mathbf{y},\mathbf{z}) =& s_{Z,k}(\mathbf{z}),\\
s_{XYZ,jd_X+i+d_Y+d_Z}(\mathbf{x},\mathbf{y},\mathbf{z}) =& s_{X,i}(\mathbf{x}) \cdot s_{Y,j}(\mathbf{y}),\\
s_{XYZ,d_Xd_Y+d_X+kd_Y+j+d_Z}(\mathbf{x},\mathbf{y},\mathbf{z}) =& s_{Y,j}(\mathbf{y}) \cdot s_{Z,k}(\mathbf{z}),
\end{aligned}$$

for $i \in [1,\dots,d_X]$, $j \in [1,\dots,d_Y]$, and $k \in [1,\dots,d_Z]$. In this case, we may intuitively express the dot product of $\mathbf{s}_{XYZ}$ with the natural parameters $\boldsymbol{\theta}_{XYZ} = (\boldsymbol{\theta}_X, \boldsymbol{\theta}_Y, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XY}, \boldsymbol{\Theta}_{YZ}) \in \Theta_{XYZ}$ as

$$\begin{aligned}
\mathbf{s}_{XYZ}(\mathbf{x},\mathbf{y},\mathbf{z}) \cdot \boldsymbol{\theta}_{XYZ} = {}& \boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\theta}_Y \cdot \mathbf{s}_Y(\mathbf{y}) + \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z})\\
&+ \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XY} \cdot \mathbf{s}_Y(\mathbf{y}) + \mathbf{s}_Y(\mathbf{y}) \cdot \boldsymbol{\Theta}_{YZ} \cdot \mathbf{s}_Z(\mathbf{z}).
\end{aligned}$$

When developing examples of deep harmoniums, I will typically consider three-layer families.

Suppose we wish to train the deep harmonium generative model Q_{XYZ} on random observations drawn from a target distribution P_Z. This entails minimizing the entropy of P_Z relative to the marginal Q_Z of the deep harmonium model. This is equivalent to minimizing the entropy of P_Z relative to the marginal Q_Z of the harmonium factorization $((X,Y),Z)$, the derivatives of which are given by equations 3.8. By expanding these equations and applying theorem 3.9 we may express the relative entropy derivatives of a deep harmonium model as

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_X} D(P_Z \parallel Q_Z) &= \mathbb{E}_Q[\mathbf{s}_X(X)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_X(X) \mid Z]],\\
\partial_{\boldsymbol{\theta}_Y} D(P_Z \parallel Q_Z) &= \mathbb{E}_Q[\mathbf{s}_Y(Y)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_Y(Y) \mid Z]],\\
\partial_{\boldsymbol{\theta}_Z} D(P_Z \parallel Q_Z) &= \mathbb{E}_Q[\mathbf{s}_Z(Z)] - \mathbb{E}_P[\mathbf{s}_Z(Z)],\\
\partial_{\boldsymbol{\Theta}_{XY}} D(P_Z \parallel Q_Z) &= \mathbb{E}_Q[\mathbf{s}_X(X) \otimes \mathbf{s}_Y(Y)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_X(X) \otimes \mathbf{s}_Y(Y) \mid Z]],\\
\partial_{\boldsymbol{\Theta}_{YZ}} D(P_Z \parallel Q_Z) &= \mathbb{E}_Q[\mathbf{s}_Y(Y) \otimes \mathbf{s}_Z(Z)] - \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_Y(Y) \otimes \mathbf{s}_Z(Z) \mid Z]].
\end{aligned}$$

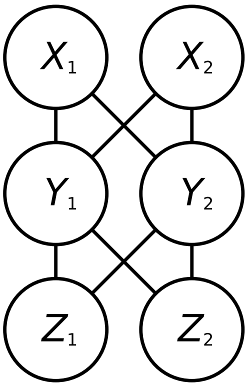

Figure 3.5: A deep harmonium family is a higher-order harmonium family which can be represented by a graph with a clique size of at most 2.

Approximating the expectations in these derivatives can also be done with Gibbs sampling, as discussed in section 3.1.2. Gibbs sampling, however, is even more costly in deep harmoniums then in second-order harmoniums. Moreover, contrastive divergence minimization cannot be applied to completely avoid these complexity issues, because the posterior can no longer be sampled directly. In section 4.2.2 I present a novel algorithm for relative entropy minimization in deep harmoniums which addresses some of these problems.

Chapter 4

Harmonium Rectification

In this chapter I introduce and develop the concept of harmonium rectification. A rectified harmonium is one for which the prior of the harmonium is conjugate to the posterior. Beyond efficient Bayesian inference, a rectified harmonium can also be exactly sampled without simulating long Gibbs chains. Building on this theory, I develop a few classes of harmonium family which contain exactly rectified harmoniums. Since most harmoniums are not rectified, I then develop algorithms for approximately rectifying harmoniums while fitting them to data.

I then consider rectification in the context of deep harmoniums. As I show, the harmonium factorization theorem allows us to understand rectification in deep harmoniums in terms of the rectification of the individual bilayers of the deep harmonium. This allows rectified deep harmoniums to inherit many of the advantages of rectified second-order harmoniums.

In this section I make a certain abuse of notation. Namely, where $\mathcal{M}_{(X_i)_{i=1}^n}$, $\mathcal{H}_{(X_i)_{i=1}^n}$, and $\mathcal{D}_{(X_i)_{i=1}^n}$ are a product family, a harmonium family, and a deep harmonium family respectively, I will use $\boldsymbol{\theta}_{(X_i)_{i=1}^n}$, $\mathbf{s}_{(X_i)_{i=1}^n}$, $\psi_{(X_i)_{i=1}^n}$ to indicate any of their natural parameters, sufficient statistics, log-partition functions, etcetera. Context will separate these different cases, and I will declare them explicitly when necessary.

4.1 Second-Order Harmonium Rectification

In general, the marginal distributions of a harmonium are more complicated – that is, depend on more parameters – than the elements of the component exponential families used to define the harmonium. This complexity is often cited as a reason why harmoniums can model complex distributions over observations. However, the fact that the marginal distribution of the observable variables ought be complex is not in and of itself an argument for why the marginal distribution of the latent variables ought to be complex. In fact, when the marginal distribution of the latent variables is in the same exponential family as the distribution of the latent variables given the observable variables, both sampling and Bayesian inference become much more tractable.

In this section I consider conditions under which the latent distribution of a harmonium is in the same exponential family as the distribution of the latent variables given the observables, and I refer to this form of harmonium as a rectified harmonium. I then present cases in which the harmoniums of a particular class are always rectified. Since most harmoniums cannot be exactly rectified, I then develop a gradi-

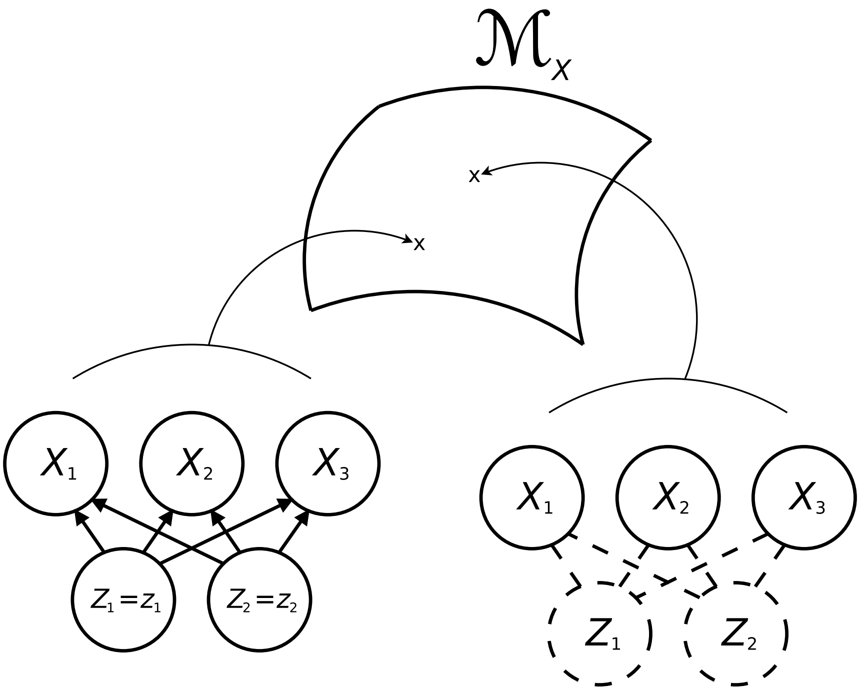

Figure 4.1: A visualization of rectification. Both the conditional distributions $P_{X|Z=\mathbf{z}}$ for any $\mathbf{z} \in \Omega_Z$, and the marginal distribution P_X (represented by graphical model of the harmonium with dashed lines around the variables which have been marginalized out) are in the exponential family $\mathcal{M}_X$.

ent descent approach to fitting a harmonium to data in parallel with training it to be approximately rectified. This allows us to learn harmonium models which are good models of the data, and which enjoy the many advantages of rectification. Although I develop the theory with regards to harmoniums of only two random variables, the harmonium factorization theorem (3.7) can be used to generalize this theory to higher-order harmoniums.

4.1.1 Rectified Second-Order Harmoniums

According to definition 3.1, the conditional distributions of a harmonium (X, Z) defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$ satisfy $P_{X|Z} \in \mathcal{M}_X$ and $P_{Z|X} \in \mathcal{M}_Z$, respectively. In general, however, the marginal distributions P_X and P_Z of a given harmonium are not elements of exponential families. In this section, I analyze harmoniums for which $P_X \in \mathcal{M}_X$.

Definition 4.1 (Rectified Harmonium)**.** The harmonium (X, Z) defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$ is a rectified harmonium if $P_X \in \mathcal{M}_X$[6].

I present an intuitive depiction of this definition in figure 4.1. Rectified harmoniums have a number of useful properties. Firstly, by definition, the prior of a rectified harmonium is conjugate to its posterior. Secondly, rectified harmoniums are also often trivial to sample from. Assuming that we may tractably sample from distributions in the families $\mathcal{M}_X$ and $\mathcal{M}_Z$, we may generate samples from $P_{XZ} \in \mathcal{H}_{XZ}$ by simply drawing a sample from P_X and then a sample from $P_{Z|X}$.

---

[6] The word "rectified" also arises in deep learning when discussing "rectified linear units" (Nair and Hinton, 2010), which are a popular transfer function used for constructing multilayer perceptrons. The name clash is unfortunate, but I think rectification is the best name for what I describe in definition 4.1. In any case, rectification in my sense refers to whole distributions, whereas a "rectified linear unit" is a single neuron, and so in practice these names should not overlap.

The following theorem provides necessary and sufficient conditions for a harmonium to be rectified in terms of the natural parameters of the harmonium. It also provides the reason why I refer to harmoniums which satisfy definition 4.1 as rectified. That is, rectifying a harmonium constrains the (typically nonlinear) log-partition function of the likelihood of a harmonium (3.6) to be affine in the sufficient statistic of the latent variables.

Theorem 4.1 (The Rectification Theorem). *Suppose that $\mathcal{H}_{XZ}$ is a harmonium family defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$, and suppose that the harmonium (X, Z) with distribution $P_{XZ} \in \mathcal{H}_{XZ}$ has parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$. Then (X, Z) is rectified if and only if there exists parameters $\boldsymbol{\rho}_X$ and a constant ρ_0 such that*

$$\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) = \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\rho}_X + \rho_0, \tag{4.1}$$

for any $\mathbf{x} \in \Omega_X$.

Proof. On one hand, recall that the prior distribution P_X of a harmonium is given by equation 3.4, and that if the harmonium in question is rectified, then $P_X \in \mathcal{M}_X$ with some natural parameters $\boldsymbol{\theta}^*_X$. Then

$$\begin{aligned} & p_X(\mathbf{x}) \propto e^{\boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ})} \propto e^{\boldsymbol{\theta}^*_X \cdot \mathbf{s}_X(\mathbf{x})} \\ \Longrightarrow \quad & \boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) = \boldsymbol{\theta}^*_X \cdot \mathbf{s}_X(\mathbf{x}) + \rho_0 \\ \Longrightarrow \quad & \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) = \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\rho}_X + \rho_0. \end{aligned}$$

for some ρ_0, and where $\boldsymbol{\rho}_X = \boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X$.

On the other hand, if we first assume that equation 4.1 holds, then P_X is given by

$$p_X(\mathbf{x}) \propto e^{\boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ})} \propto e^{(\boldsymbol{\theta}_X + \boldsymbol{\rho}_X) \cdot \mathbf{s}_X(\mathbf{x})},$$

which implies that $P_X \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta}_X + \boldsymbol{\rho}_X$. □

Assuming that they exist, I refer to the parameters $\boldsymbol{\rho}_X$ and ρ_0 of a rectified harmonium as the rectification parameters of the harmonium, and I refer to equation 4.1 as the rectification equation. The following corollary is contained in the proof of the previous theorem, however it is helpful to state on its own.

Corollary 4.2. *Suppose that $\mathcal{H}_{XZ}$ is a harmonium family defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$, and suppose that the harmonium (X, Z) with distribution $P_{XZ} \in \mathcal{H}_{XZ}$ and parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$ is rectified with rectification parameters $\boldsymbol{\rho}_X$ and ρ_0. Then the parameters $\boldsymbol{\theta}^*_X$ of $P_X \in \mathcal{M}_X$ are given by*

$$\boldsymbol{\theta}^*_X = \boldsymbol{\theta}_X + \boldsymbol{\rho}_X.$$

Proof. This follows from the second part of the proof of theorem 4.1. □

Notice that equation 4.1 does not depend on the parameters $\boldsymbol{\theta}_X$ of the harmonium. Because of this, one may rather think of the likelihood $P_{Z|X}$ as being rectified, such that given any prior $P_X \in \mathcal{M}_X$, the harmonium (X, Z) with density $p_{Z|X} \cdot p_X$ is rectified.

Definition 4.2 (Rectified Likelihood)**.** The likelihood $P_{Z|X}$ with the form of equation 3.6 and parameters $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{XZ}$ is rectified if the parameters satisfy equation 4.1 for some rectification parameters $\boldsymbol{\rho}_X$ and ρ_0.

This formulation of rectification is especially useful in the context of Bayesian inference, where ensuring that the likelihood is rectified is enough to ensure the existence of conjugate priors. When given a rectified likelihood and a conjugate prior, the following corollary will help us not to forget the exact form of the resulting harmonium.

Corollary 4.3. *Let $\mathcal{M}_X$ and $\mathcal{M}_Z$ be exponential families, and suppose that $P_{Z|X} \in \mathcal{M}_Z$ is rectified with parameters $\boldsymbol{\rho}_X$ and ρ_0, and that $P_X \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta}^*_X$. Then the distribution $P_{XZ} \in \mathcal{H}_{XZ}$ with density $p_{Z|X} \cdot p_X$ has parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$, where $\boldsymbol{\theta}_X = \boldsymbol{\theta}^*_X - \boldsymbol{\rho}_X$.*

An alternative to theorem 4.1 for describing rectification for harmoniums involves the log-partition function of P_{XZ}.

Corollary 4.4. *Suppose that $\mathcal{H}_{XZ}$ is a harmonium family defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$. Then the harmonium $P_{XZ} \in \mathcal{H}_{XZ}$ with parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$ is rectified, with latent distribution $P_X \in \mathcal{M}_X$ and parameters $\boldsymbol{\theta}^*_X$, if and only if the log-partition function of P_{XZ} satisfies*

$$\psi_{XZ}(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ}) = \psi_X(\boldsymbol{\theta}^*_X) + \rho_0.$$

Proof. According to theorem 4.1 and corollary 4.2,

$$\begin{aligned}
p_X(\mathbf{x}) &= e^{\boldsymbol{\theta}^*_X \cdot \mathbf{s}_X(\mathbf{x}) - \psi_X(\boldsymbol{\theta}^*_X)} \\
&= e^{\boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\rho}_X \cdot \mathbf{s}_X(\mathbf{x}) + \rho_0 - \psi_{XZ}(\boldsymbol{\theta}_{XZ})} \\
\iff \qquad \psi_{XZ}(\boldsymbol{\theta}) &= (\boldsymbol{\theta}_X + \boldsymbol{\rho}_X - \boldsymbol{\theta}^*_X) \cdot \mathbf{s}_X(\mathbf{x}) + \psi_X(\boldsymbol{\theta}^*_X) + \rho_0 \\
&= \psi_X(\boldsymbol{\theta}^*_X) + \rho_0.
\end{aligned}$$

□

Theorem 4.1 and corollary 4.4 provide different perspectives on the constraints placed on a harmonium when it is rectified. Whereas theorem 4.1 states that the log-partition function of the likelihood is restricted to be an affine function, corollary 4.4 states that the log-partition function of the joint distribution of a harmonium must be no more complex than the log-partition function of the exponential family of the latent variables.

4.1.2 Exact Rectification

Although approximate rectification is all we can aim for when working with most harmonium families, there are at least two cases where the exact rectification parameters of a harmonium are given by closed-form expressions, namely, when the harmonium family in question involves either normal or categorical families (see appendix A). Of course, these are exactly the cases where Bayes' rule can typically be solved analytically; nevertheless, showing how these solutions can be expressed as instances of rectification provides both an introduction to understanding rectification in practice, and a new perspective on these known solutions. This also serves to emphasize the importance of approximate rectification, which allows approximate Bayesian inference to be implemented for cases which do not otherwise not yield to analytic solutions.

Let us refer to the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$ which define the harmonium family $\mathcal{H}_{XZ}$ as the latent and observable families, respectively. The first case I consider is where the latent exponential family of the harmonium in question is the categorical family (see appendix A). Because the categorical distribution contains all probability distributions over a given finite set, it follows immediately that any harmonium is rectified if its latent family is the categorical family. Nevertheless, this example illustrates how to calculate the rectification parameters, which have many practical applications even in this trivial case.

Proposition 4.5. *Let $\mathcal{H}_{XZ}$ be the harmonium family defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$, and suppose that $\mathcal{M}_X$ is the categorical family where $\Omega_X = \{x_i\}_{i=1}^n$. Moreover, suppose that (X, Z) is a harmonium with distribution $P_{XZ} \in \mathcal{H}_{XZ}$ and parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$ Then (X, Z) is rectified with rectification parameters*

$$\begin{aligned}\rho_{X,i} &= \psi_Z(\boldsymbol{\theta}_Z + \boldsymbol{\theta}_{XZ,i}) - \psi_Z(\boldsymbol{\theta}_Z),\\ \rho_0 &= \psi_Z(\boldsymbol{\theta}_Z),\end{aligned}$$

where $\boldsymbol{\theta}_{XZ,i}$ is the ith row of $\boldsymbol{\Theta}_{XZ}$.

Proof. First note that if $\Omega_X = \{x_i\}_{i=1}^n$, then $\mathcal{M}_X$ is $n-1$ dimensional, and the sufficient statistic of $\mathcal{M}_X$ is given by $s_{X,j}(x_i) = 1$ if $i = j$, and 0 otherwise, for every $1 \leq j < n$ (appendix A). In order for P_{XZ} to be rectified, equation 4.1 must be satisfied for every element of Ω_X. For any $x_i \in \Omega_X$ where $i < n$,

$$\begin{aligned}&\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(x_i) \cdot \boldsymbol{\Theta}_{XZ}) = \mathbf{s}_X(x_i) \cdot \boldsymbol{\rho}_X + \rho_0\\ \iff &\psi_Z(\boldsymbol{\theta}_Z + \boldsymbol{\Theta}_{XZ,i}) = \rho_{X,i} + \rho_0\\ \iff &\rho_{X,i} = \psi_Z(\boldsymbol{\theta}_Z + \boldsymbol{\Theta}_{XZ,i}) - \rho_0.\end{aligned}$$

For $x_n \in \Omega_X$,

$$\begin{aligned}&\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(x_n) \cdot \boldsymbol{\Theta}_{XZ}) = \mathbf{s}_X(x_n) \cdot \boldsymbol{\rho}_X + \rho_0\\ \iff &\rho_0 = \psi_Z(\boldsymbol{\theta}_Z).\end{aligned}$$

□

Because the prior of a harmonium (X, Z) with a categorical latent family is always an element of the categorical family, we may express the marginal density over Z as

$$p_Z(\mathbf{z}) = \sum_{i=1}^{d_X} p_X(x_i) \cdot p_{Z|X}(\mathbf{z} \mid x_i) = \sum_{i=1}^{d_X} \eta_{X,i} \cdot p_{Z|X}(\mathbf{z} \mid x_i),$$

where we define $\eta_{X,d_X} = 1 - \sum_{i=1}^{d_Z - 1} \eta_{X,i}$. This is simply the general definition of a mixture model with mixture weights $\eta_{X,i}$ and mixture components $p_{Z|X}(\mathbf{z} \mid x_i)$ (Murphy, 2012). Harmonium models defined by categorical latent families therefore provide a general interface to mixture modelling, and can be trained with the gradient

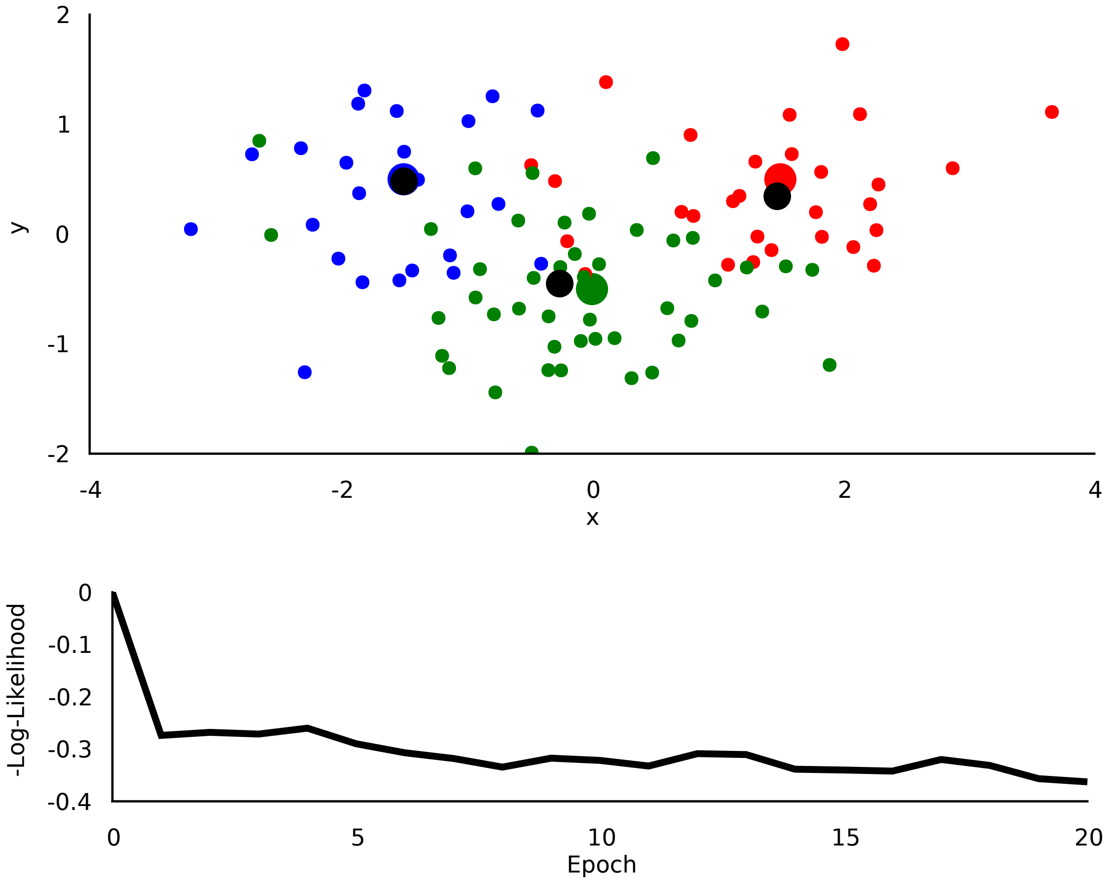

Figure 4.2: A simulation of fitting the harmonium model of the harmonium family $\mathcal{H}_{XZ}$, where $\mathcal{M}_X$ is is the categorical family with three states $\Omega_X = \{0, 1, 2\}$, and $\mathcal{M}_Z$ is the bivariate normal family with known, diagonal covariance matrix, where the variance of the first and second variables is 1 and 0.5, respectively. *Top*: The true distribution is defined by three mixture components with weights 0.25, 0.25, and 0.5, and means $(1.5, 0.5)$, $(-1.5, 0.5)$, and $(0, -0.5)$, depicted by the large red, green, and blue circles, respectively. The small dots represent the data used for training. The black circles represent the means of the learned mixture components, which have weights 0.27, 0.31, and 0.43, and means (1.66, 0.71), (-1.51, 0.57), and (0.15, -0.58), respectively. *Bottom*: The average negative-log likelihood of the model on 1,000 test samples drawn from the true mixture model, over the course of 20 training epochs on the samples depicted in the top plot.

descent procedures developed throughout this dissertation[7]. In figure 4.2 I present an example of training a mixture of normal distributions based on theorem 4.5 by following the gradient given by the derivatives in equations 3.8.

The second case I consider is where the observable family $\mathcal{M}_Z$ of the harmonium family $\mathcal{H}_{XZ}$ in question is a multivariate normal family with known covariance, and the sufficient statistic of the latent family $\mathcal{M}_X$ is given by the all first- and second-order statistics of the latent variables. By this I mean that the sufficient statistic of $\mathcal{M}_X$ satisfies

$$\boldsymbol{\theta}_X \cdot \mathbf{s}_X(\mathbf{x}) = \boldsymbol{\theta} \cdot \mathbf{x} + \mathbf{x} \cdot \boldsymbol{\Theta} \cdot \mathbf{x}, \tag{4.2}$$

for arbitrary parameters $\boldsymbol{\theta}_X$, where $\boldsymbol{\theta}_X = (\boldsymbol{\theta}, \boldsymbol{\Theta})$, and where $\boldsymbol{\Theta}$ is expressed in row-major form. For the following theorem, I also use $\mathrm{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to denote the multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.

---

[7] See Montúfar and Morton (2015b) for an analysis of restricted Boltzmann machines as mixtures of mixture models.

Theorem 4.6. *Let $\mathcal{H}_{XZ}$ be a harmonium family defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$, where $\mathcal{M}_Z$ is the multivariate normal family with fixed covariance Σ, and the sufficient statistic of $\mathcal{M}_X$ satisfies equation 4.2. Suppose that (X, Z) is a harmonium with distribution $P_{XZ} \in \mathcal{H}_{XZ}$ and a likelihood given by $P_{Z|X=\mathbf{x}} = \mathrm{N}(\mathbf{m} + \mathbf{M} \cdot \mathbf{x}, \boldsymbol{\Sigma})$. Then P_{XZ} is rectified, with rectification parameters*

$$\begin{aligned} \rho_0 &= \frac{1}{2}\mathbf{m} \cdot \boldsymbol{\Sigma}^{-1} \cdot \mathbf{m} - \frac{1}{2}\log|\boldsymbol{\Sigma}^{-1}|, \\ \boldsymbol{\rho}_{Z,\boldsymbol{\mu}} &= \mathbf{m} \cdot \boldsymbol{\Sigma}^{-1} \cdot \mathbf{M}, \\ \mathbf{P}_{Z,\boldsymbol{\Sigma}} &= \frac{1}{2}\mathbf{M}^{\top} \cdot \boldsymbol{\Sigma}^{-1} \cdot \mathbf{M}, \end{aligned} \tag{4.3}$$

where $\boldsymbol{\rho}_X = (\boldsymbol{\rho}_{Z,\boldsymbol{\mu}}, \mathbf{P}_{Z,\boldsymbol{\Sigma}})$.

Proof. The natural parameters of a multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ are given by

$$\boldsymbol{\theta}_{\boldsymbol{\mu}} = -2\boldsymbol{\mu} \cdot \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}, \qquad \boldsymbol{\Theta}_{\boldsymbol{\Sigma}} = -\frac{1}{2}\boldsymbol{\Sigma}^{-1}. \tag{4.4}$$

The log-partition function of $\mathcal{M}_Z$ (appendix A) expressed as a function of these parameters is

$$\psi_Z(\boldsymbol{\theta}_{\boldsymbol{\mu}}, \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}) = -\frac{1}{4}\boldsymbol{\theta}_{\boldsymbol{\mu}} \cdot \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}^{-1} \cdot \boldsymbol{\theta}_{\boldsymbol{\mu}} - \frac{1}{2}\log|-2\boldsymbol{\Theta}_{\boldsymbol{\Sigma}}|. \tag{4.5}$$

We may therefore express $\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ})$ for this model in mean and covariance parameters of a multivariate normal by substituting $\boldsymbol{\mu} = \mathbf{m} + \mathbf{M} \cdot \mathbf{x}$ into equation 4.4 and then result into equation 4.5. In this vein, the first term of equation 4.5 may be expressed as

$$\begin{aligned} -\frac{1}{4}&\boldsymbol{\theta}_{\boldsymbol{\mu}} \cdot \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}^{-1} \cdot \boldsymbol{\theta}_{\boldsymbol{\mu}} \\ &= -((\mathbf{m} + \mathbf{M} \cdot \mathbf{x}) \cdot \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}) \cdot \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}^{-1} \cdot ((\mathbf{m} + \mathbf{M} \cdot \mathbf{x}) \cdot \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}) \\ &= -(\mathbf{m} + \mathbf{M} \cdot \mathbf{x}) \cdot ((\mathbf{m} + \mathbf{M} \cdot \mathbf{x}) \cdot \boldsymbol{\Theta}_{\boldsymbol{\Sigma}}) \\ &= \frac{1}{2}(\mathbf{m} + \mathbf{M} \cdot \mathbf{x}) \cdot \boldsymbol{\Sigma}^{-1} \cdot (\mathbf{m} + \mathbf{M} \cdot \mathbf{x}) \\ &= \frac{1}{2}\mathbf{m} \cdot \boldsymbol{\Sigma}^{-1} \cdot \mathbf{m} + \mathbf{m} \cdot \boldsymbol{\Sigma}^{-1} \cdot \mathbf{M} \cdot \mathbf{x} + \frac{1}{2}\mathbf{x} \cdot \mathbf{M}^{\top} \cdot \boldsymbol{\Sigma}^{-1} \cdot \mathbf{M} \cdot \mathbf{x}, \end{aligned} \tag{4.6}$$

which is a second-order polynomial. The second term in equation 4.5 may be expressed as

$$-\frac{1}{2}\log|-2\boldsymbol{\Theta}_{\boldsymbol{\Sigma}}| = -\frac{1}{2}\log|\boldsymbol{\Sigma}^{-1}|. \tag{4.7}$$

By combining equations 4.3, 4.5, 4.6, and 4.7, we may solve the rectification equation (4.1) by

$$\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}) = \boldsymbol{\rho}_{Z,\boldsymbol{\mu}} \cdot \mathbf{x} + \mathbf{x} \cdot \mathbf{P}_{Z,\boldsymbol{\Sigma}} \cdot \mathbf{x} + \rho_0 = \boldsymbol{\rho}_X \cdot \mathbf{s}_X(\mathbf{x}) + \rho_0,$$

where ρ_0, $\boldsymbol{\rho}_{Z,\boldsymbol{\mu}}$, and $\mathbf{P}_{Z,\boldsymbol{\Sigma}}$ are as defined in the theorem statement. □

The form of likelihood function $P_{Z|X}$ in this theorem is the one applied in the context of Kalman filtering, and models observations of the latent state which are corrupted by additive Gaussian noise. The example of multisensory Bayesian inference which I depicted in figure 2.7 is also a simple instance of this scheme. Nevertheless, this theorem may also be applied to latent families $\mathcal{M}_X$ which are not multivariate normal families, as long as the sufficient statistic satisfies equation 4.2. One example of such families is the family of Boltzmann machines or Ising models, which model the second-order statistics between binary random variables.

4.1.3 Approximate Rectification

Since the rectification equation (4.1) can not in most cases be exactly satisfied, let us now endeavour to train a harmonium to approximately do so. Suppose that Q_{XZ} is the generative model defined by the harmonium family $\mathcal{H}_{XZ}$, such that $Q_{XZ} \in \mathcal{H}_{XZ}$ for any parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ}) \in \Theta_{XZ}$. The purpose of rectification is to drive the marginal distribution Q_X of the generative model to be an element of the exponential family $\mathcal{M}_X$, and so we define the objective of rectification as minimizing the rectification error

$$D(Q_X^* \parallel Q_X),$$

where $Q_X^* \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta}_X^* \in \Theta_X$; I refer to the model Q_X^* as the rectifier of the harmonium model Q_{XZ}. Note how, in contrast to minimizing the relative entropy with respect to some target distribution P_Z, this divergence depends not only on the parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$ of Q_{XZ}, but also the parameters $\boldsymbol{\theta}_X^*$ of the rectifier Q_X^*.

Since Q_X^* does not depend on the parameters of Q_{XZ}, computing the derivatives of the rectification error with respect to $\boldsymbol{\theta}_X$, $\boldsymbol{\theta}_Z$, and $\boldsymbol{\Theta}_{XZ}$ results in derivatives analogous to derivatives 3.8. In particular,

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_X} D(Q_X^* \parallel Q_X) &= \mathbb{E}_Q[\mathbf{s}_X(X)] - \mathbb{E}_{Q^*}[\mathbf{s}_X(X)], \\
\partial_{\boldsymbol{\theta}_Z} D(Q_X^* \parallel Q_X) &= \mathbb{E}_Q[\mathbf{s}_Z(Z)] - \mathbb{E}_{Q^*}[\mathbb{E}_Q[\mathbf{s}_Z(Z) \mid X]], \\
\partial_{\boldsymbol{\Theta}_{XZ}} D(Q_X^* \parallel Q_X) &= \\
&\mathbb{E}_Q[\mathbf{s}_X(X) \otimes \mathbf{s}_Z(Z)] - \mathbb{E}_{Q^*}[\mathbb{E}_Q[\mathbf{s}_X(X) \otimes \mathbf{s}_Z(Z) \mid X]].
\end{aligned} \tag{4.8}$$

Since rectification, as described by equation 4.1, does not depend on the parameters $\boldsymbol{\theta}_X$ of the harmonium, one may also choose not to minimize the rectification error with respect to $\boldsymbol{\theta}_X$, and rather rely solely on $\boldsymbol{\theta}_X^*$ to minimize the error with respect to the latent biases.

When computing the derivatives of $D(Q_X^* \parallel Q_X)$ with respect to $\boldsymbol{\theta}_X^*$, we can no longer ignore the entropy of Q_X^*. Nevertheless, as long as the sufficient statistic of $\mathcal{M}_X$ is minimal, we may express the derivatives as a particular kind of covariance. Where we define the covariance function as

$$\mathbf{C}_P(X, Y) \mapsto \mathbb{E}_P[X \otimes Y] - \mathbb{E}_P[X] \otimes \mathbb{E}_P[Y],$$

we may prove the following theorem.

Theorem 4.7. *Let $\mathcal{H}_{XZ}$ be the second-order harmonium family defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$, and suppose that the sufficient statistic $\mathbf{s}_X$ is minimal. Moreover, suppose that*

$Q_{XZ} \in \mathcal{H}_{XZ}$ with parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$, and that $Q^*_X \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta}^*_X$. Then

$$\begin{aligned} &\partial_{\boldsymbol{\theta}^*_X} D(Q^*_X \parallel Q_X) = \\ &\qquad \mathbf{C}_{Q^*}(\mathbf{s}_X(X) \cdot (\boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X) - \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(X) \cdot \boldsymbol{\Theta}_{XZ}), \mathbf{s}_X(X)). \end{aligned} \tag{4.9}$$

Moreover, if $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{XZ}$ satisfy equation 4.1 where $\boldsymbol{\rho}_X = \boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X$, then $\partial_{\boldsymbol{\theta}^*_X} D(Q^*_X \parallel Q_X) = \mathbf{0}$.

Proof. To begin, we expand the relative entropy into the difference of the cross-entropy and entropy, such that

$$\partial_{\boldsymbol{\theta}^*_X} D(Q^*_X \parallel Q_X) = \partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\log q_X(X)] - \partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\log q^*_X(X)]. \tag{4.10}$$

Now, $\mathbb{E}_{Q^*}[\log q^*_X(X)]$ is simply the negative entropy of Q^*_X. According to theorem 2.7, $\mathbb{E}_{Q^*}[\log q^*_X(X)] = \phi_X(\boldsymbol{\eta}^*_X)$, where $\boldsymbol{\eta}^*_X$ are the mean parameters of Q^*_X. According to proposition 2.4, $\boldsymbol{\eta}^*_X = \partial_{\boldsymbol{\theta}^*_X} \psi_X(\boldsymbol{\theta}^*_X) = \boldsymbol{\tau}_X(\boldsymbol{\theta}^*_X)$. Moreover, according to proposition 2.6, since $\mathbf{s}_X$ is minimal, $\partial_{\boldsymbol{\eta}^*_X}(\phi(\boldsymbol{\tau}_X(\boldsymbol{\theta}^*_X))) = \boldsymbol{\tau}^*_X(\boldsymbol{\tau}_X(\boldsymbol{\theta}^*_X))) = \boldsymbol{\theta}^*_X$. Therefore

$$\begin{aligned} \partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\log q^*_X(X)] &= \partial_{\boldsymbol{\theta}^*_X}(\phi_X(\boldsymbol{\tau}_X(\boldsymbol{\theta}^*_X))) \\ &= \boldsymbol{\tau}^*_X(\boldsymbol{\tau}_X(\boldsymbol{\theta}^*_X)) \cdot \partial_{\boldsymbol{\theta}^*_X}(\partial_{\boldsymbol{\theta}^*_X} \psi_X(\boldsymbol{\theta}^*_X)) \\ &= \partial_{\boldsymbol{\theta}^*_X \boldsymbol{\theta}^*_X} \psi_X(\boldsymbol{\theta}^*_X) \cdot \boldsymbol{\theta}^*_X. \end{aligned} \tag{4.11}$$

In order to evaluate the derivative $\partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\log q_X(X)]$, let us define the function $\boldsymbol{\theta}_{Z|X}(\mathbf{x}) \mapsto \boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ}$, and recall that $\boldsymbol{\theta}_{XZ} = (\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$. By substituting the definition of the harmonium prior from equation 3.4 into $\partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\log q_X(X)]$ we may write

$$\begin{aligned} &\partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\log q_X(X)] \\ &\qquad = \partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\boldsymbol{\theta}_X \cdot \mathbf{s}_X(X) + \psi_Z(\boldsymbol{\theta}_{Z|X}(X)) - \psi_{XZ}(\boldsymbol{\theta}_{XZ})] \\ &\qquad = \partial_{\boldsymbol{\theta}^*_X}(\boldsymbol{\theta}_X \cdot \mathbb{E}_{Q^*}[\mathbf{s}_X(X)] + \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_{Z|X}(X))] - \psi_{XZ}(\boldsymbol{\theta}_{XZ})) \\ &\qquad = \partial_{\boldsymbol{\theta}^*_X \boldsymbol{\theta}^*_X} \psi_X(\boldsymbol{\theta}^*_X) \cdot \boldsymbol{\theta}_X + \partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_{Z|X}(X))], \end{aligned} \tag{4.12}$$

where $\partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\mathbf{s}_X(X)] = \partial_{\boldsymbol{\theta}^*_X \boldsymbol{\theta}^*_X} \psi_X(\boldsymbol{\theta}^*_X)$.

If we then consider the derivatives $\partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_{Z|X}(X))]$, we find that

$$\begin{aligned} &\partial_{\boldsymbol{\theta}^*_X} \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_{Z|X}(X))] \\ &\qquad = \partial_{\boldsymbol{\theta}^*_X} \int_{\Omega_X} q^*_X(\mathbf{x}) \psi_Z(\boldsymbol{\theta}_{Z|X}(X)) \mu_X(d\mathbf{x}) \\ &\qquad = \partial_{\boldsymbol{\theta}^*_X} \int_{\Omega_X} e^{\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\theta}^*_X - \psi_X(\boldsymbol{\theta}^*_X)} \psi_Z(\boldsymbol{\theta}_{Z|X}(X)) \mu_X(d\mathbf{x}) \\ &\qquad = \int_{\Omega_X} (\mathbf{s}_X(\mathbf{x}) - \mathbb{E}_{Q^*}[\mathbf{s}(X)]) e^{\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\theta}^*_X - \psi_X(\boldsymbol{\theta}^*_X)} \psi_Z(\boldsymbol{\theta}_{Z|X}(X)) \mu_X(d\mathbf{x}) \\ &\qquad = \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_{Z|X}(X)) \mathbf{s}_X(X)] - \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_{Z|X}(X))] \mathbb{E}_{Q^*}[\mathbf{s}_X(X)]. \end{aligned} \tag{4.13}$$

Finally, by combining equations 4.10, 4.11, 4.12, and 4.13, and expanding the function $\boldsymbol{\theta}_{Z|X}$, we arrive at the derivative

$$\begin{aligned} \partial_{\boldsymbol{\theta}^*_X} D(Q^*_X \parallel Q_X) &= \partial_{\boldsymbol{\theta}^*_X \boldsymbol{\theta}^*_X} \psi_X(\boldsymbol{\theta}^*_X) \cdot (\boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X) \\ &\quad + \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(X) \cdot \boldsymbol{\Theta}_{XZ})] \mathbb{E}_{Q^*}[\mathbf{s}_X(X)] \\ &\quad - \mathbb{E}_{Q^*}[\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(X) \cdot \boldsymbol{\Theta}_{XZ}) \mathbf{s}_X(X)]. \end{aligned} \tag{4.14}$$

Notice how the difference of expectations in the derivative of the rectification error is the covariance of the random variables $\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(X) \cdot \boldsymbol{\Theta}_{XZ})$ and $\mathbf{s}_X(X)$. Moreover, according to proposition 2.4, the Hessian of the log-partition function is simply the covariance of the sufficient statistic $\mathbf{s}_X$ under the distribution $Q^*_X \in \mathcal{M}_X$. We may therefore rewrite equation 4.14 more compactly as

$$\begin{aligned}
&\partial_{\boldsymbol{\theta}^*_X} D(Q^*_X \parallel Q_X) \\
&\quad= \mathbf{C}_{Q^*}(\mathbf{s}_X(X), \mathbf{s}_X(X)) \cdot (\boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X) \\
&\qquad\qquad - \mathbf{C}_{Q^*}(\psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(X) \cdot \boldsymbol{\Theta}_{XZ}), \mathbf{s}_X(X)) \\
&\quad= \mathbf{C}_{Q^*}(\mathbf{s}_X(X) \cdot (\boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X) - \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(X) \cdot \boldsymbol{\Theta}_{XZ}), \mathbf{s}_X(X)).
\end{aligned}$$

Moreover, if the parameters of the models satisfy equation 4.1, where $\boldsymbol{\rho}_X = \boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X$, then

$$\mathbf{s}_X(X) \cdot (\boldsymbol{\theta}^*_X - \boldsymbol{\theta}_X) - \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(X) \cdot \boldsymbol{\Theta}_{XZ}) = \rho_0,$$

which implies that the covariance in equation 4.9 is 0. □

In order to model a rectified harmonium with Q_{XZ}, we must minimize the rectification error while at the same time training Q_{XZ} to model some target distribution over the observable variables. Consider the target distribution P_Z, the harmonium model Q_{XZ}, and the rectifier Q^*_X. In order to both approximate P_Z with Q_Z as well as rectify Q_{XZ}, we evaluate

$$\underset{\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ}, \boldsymbol{\theta}^*_X}{\arg\min} \left\{ \alpha_Z D(P_Z \parallel Q_Z) + \alpha_X D(Q^*_X \parallel Q_X) \right\},$$

for some positive constants α_Z and α_X. In order to minimize this combined error, we may descend the derivatives in equations 3.8, 4.8, and 4.9.

Despite its superficially more complicated form, descending the rectification error gradient with respect to $\boldsymbol{\theta}^*_X$ is relatively straightforward, as long as we can generate samples from the elements of $\mathcal{M}_X$. On the other hand, evaluating the derivatives of $D(P_Z \parallel Q_Z)$ (3.8) and $D(Q^*_X \parallel Q_X)$ (4.8) with respect to the parameters $\boldsymbol{\theta}_X$, $\boldsymbol{\theta}_Z$, and $\boldsymbol{\Theta}_{XZ}$ requires approximating more complicated expectations.

As described in subsection 3.1.2, we may apply contrastive divergence minimization to approximately minimize $D(P_Z \parallel Q_Z)$. In fact, we can also approximate the derivatives of $D(Q^*_X \parallel Q_X)$ with contrastive divergence. As long as we can generate samples from Q^*_X, we can use these samples to initiate the short Gibbs chains required to estimate the approximate contrastive divergence derivatives.

At the same time, rectification can improve the efficiency of the minimization of $D(P_Z \parallel Q_Z)$. If we initialize the parameters such that $\boldsymbol{\Theta}_{XZ} = \mathbf{0}$ and $\boldsymbol{\theta}^*_X = \boldsymbol{\theta}_X$, then $D(Q^*_X \parallel Q_X)$ is 0, and Q_{XZ} models an exactly rectified harmonium. This implies that we can estimate the derivatives of $D(P_Z \parallel Q_Z)$ with exact samples drawn from P_X and then $P_{Z|X}$. As long as the rectification error remains small, then this strategy should allow us to avoid any amount of Gibbs sampling in fitting the model. On the other hand, since $D(Q^*_X \parallel Q_X)$ is small, Q^*_X is a good approximation to Q_X, which implies that the Gibbs sampler for estimating the rectification error derivatives should converge quickly to Q_X given a sample from Q^*_X. Algorithmically this is the same as contrastive divergence minimization, but practically it means that by generating a handful of iterations of the Gibbs sampler, we may tractably approximate the true derivatives of $D(Q^*_X \parallel Q_X)$. In section 7.2 I implement and validate this approach with restricted Boltzmann machines.

4.2 Deep Harmonium Rectification

In section 3.2 I generalized second-order harmoniums to collections of n random variables, where the conditional distribution of each random variable given all the others is always an element of the same exponential family. Although theoretically interesting, the complexity of most computations with higher-order harmoniums grow exponentially with the order. One of the primary motivations for considering deep harmoniums in particular (subsection 3.2.3), and deep generative models in general, is that when carefully designed, they may capture rich hidden structure while reducing the complexity of the incumbent computations to be linear in the number of layers. Unfortunately, in spite of its second-order structure, sampling and training deep harmoniums in general remains intractable.

I address these issues in this section by applying the harmonium factorization theorem to generalize the theory of rectification developed in the previous section for deep harmoniums. In a three-layer deep harmonium (X, Y, Z), for example, there are two factorizations of the deep harmonium to consider, namely, the harmonium $((X, Y), Z)$, and the harmonium $(X, (Y, Z))$. I show that when both of these factorizations are rectified, we can reduce the complexity of sampling the distribution of the deep harmonium to the sum of the complexity of sampling each bilayer. I also show how these two forms of rectification can be understood in terms of the rectification of the lower and upper bilayers of the deep harmonium. Based on this theory, I then develop a novel algorithm for fitting a deep harmonium to data.

4.2.1 Rectified Deep Harmoniums

Let us consider the deep harmonium (X, Y, Z) defined by the exponential families $\mathcal{M}_X$, $\mathcal{M}_Y$, and $\mathcal{M}_Z$ in light of the theory of rectification developed in subsection 4.1.1. According to the harmonium factorization theorem (3.7), $((X, Y), Z)$ is a second order harmonium defined by $\mathcal{M}_X \otimes \mathcal{M}_Y$ and $\mathcal{M}_Z$. If the factorization $((X, Y), Z)$ of (X, Y, Z) is rectified, then $P_{XY} \in \mathcal{M}_X \otimes \mathcal{M}_Y$ and $P_{XY|Z} \in \mathcal{M}_X \otimes \mathcal{M}_Z$, such that P_{XY} is conjugate to $P_{XY|Z}$. Where X and Y are the latent random variables and Z is the observable random variable, this means that Bayesian inference on the latent variables of a rectified deep harmonium (X, Y, Z) is about as straightforward as it is for rectified second-order harmoniums.

Sampling of the distribution P_{XYZ} based on rectification is less straightforward. If the factorization $((X, Y), Z)$ is rectified, we must still somehow generate a sample from P_{XY} before we can sample from $P_{Z|XY}$, which is often intractable. If the factorization $(X, (Y, Z))$ is rectified, then we may generate samples from P_X, but in this case we must generate a sample from $P_{XY|Z}$ in order to generate a sample from P_{XYZ}, which again is often intractable.

However, as I show, if both factorizations $((X, Y), Z))$ and $(X, (Y, Z))$ are rectified, then we may generate a sample from P_{XYZ} by first sampling from P_X, and then generating samples from $P_{Y|X}$ and $P_{Z|Y}$. In general, I define a deep harmonium as rectified if it is rectified at every level of its hierarchy.

Definition 4.3 (Rectified Deep Harmonium)**.** A deep harmonium $(X_i)_{i=1}^n$ is rectified if $((X_i)_{i=1}^k, (X_i)_{i=k+1}^n)$ is rectified for every $0 < k < n$.

As previously described, the following theorem states that we can sample from the distribution $P_{(X_i)_{i=1}^n}$ of a rectified deep harmonium as long we can sample from the component exponential families which define its corresponding deep harmonium family.

Theorem 4.8. *Let $\mathcal{D}_{(X_i)_{i=1}^n}$ be the deep harmonium family defined by the sequence of exponential families $(\mathcal{M}_{X_i})_{i=1}^n$ with minimal sufficient statistics. If $(X_i)_{i=1}^n$ is a rectified deep harmonium with distribution $P_{(X_i)_{i=1}^n} \in \mathcal{D}_{(X_i)_{i=1}^n}$, then $P_{X_1} \in \mathcal{M}_{X_1}$, and $P_{X_k \mid X_{k-1}} \in \mathcal{M}_{X_k}$ for every $1 < k \leq n$.*

Proof. $P_{X_1} \in \mathcal{M}_{X_1}$ because $(X_1, (X_i)_{i=2}^n)$ is a rectified harmonium. Because $(X_i)_{i=1}^n$ is an nth-order harmonium, $P_{X_n \mid (X_i)_{i=1}^{n-1}} \in \mathcal{M}_{X_n}$, and because of the graphical structure of the deep harmonium, $P_{X_n \mid (X_i)_{i=1}^{n-1}} = P_{X_n \mid X_{n-1}}$. Finally, because of the harmonium factorization theorem (3.7), for $1 < k < n$, $P_{(X_i)_{i=1}^k} \in \mathcal{D}_{(X_i)_{i=1}^k}$, which implies that $P_{X_k \mid (X_l)_{l=1}^{k-1}} = P_{X_k \mid X_{k-1}} \in \mathcal{M}_{X_k}$. ☐

Rectified deep harmoniums inherit many of the properties of rectified second-order harmoniums, but in order take advantage of these properties, we must develop an analogue to the rectification theorem (4.1) for deep harmoniums. Ultimately I will show that a deep harmonium is rectified if and only if every bilayer in the deep harmonium is rectified. Proving this claim requires the following lemma.

Lemma 4.9. *Let $\mathcal{D}_{(X_i)_{i=1}^n}$ be a deep harmonium family defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$ with minimal sufficient statistics. Let $(X_i)_{i=1}^n$ be a deep harmonium with distribution $P_{(X_i)_{i=1}^n} \in \mathcal{D}_{(X_i)_{i=1}^n}$ and with parameters given by the biases $\boldsymbol{\theta}_{X_1}, \ldots, \boldsymbol{\theta}_{X_n}$ and interaction matrices $\boldsymbol{\Theta}_{X_1 X_2}, \ldots, \boldsymbol{\Theta}_{X_{n-1} X_n}$. Then $((X_i)_{i=1}^{n-1}, X_n)$ is rectified if and only if*

$$\psi_{X_n}(\boldsymbol{\theta}_{X_n} + \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \boldsymbol{\Theta}_{X_{n-1} X_n}) = \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \boldsymbol{\rho}_{X_{n-1}} + \rho_{n-1}, \tag{4.15}$$

for rectification parameters $\boldsymbol{\rho}_{X_{n-1}}$ and ρ_{n-1}, such that the bias $\boldsymbol{\theta}^*_{X_{n-1}}$ of X_{n-1} in the distribution $P_{(X_i)_{i=1}^{n-1}} \in \mathcal{D}_{(X_i)_{i=1}^{n-1}}$ is $\boldsymbol{\theta}^*_{X_{n-1}} = \boldsymbol{\theta}_{X_{n-1}} + \boldsymbol{\rho}_{X_{n-1}}$.

Proof. Suppose that equation 4.15 holds, and let us denote the sufficient statistic of the deep harmonium family $\mathcal{D}_{(X_i)_{i=1}^{n-1}}$ by $\mathbf{s}_{(X_i)_{i=1}^{n-1}}$. Then

$$\begin{aligned}
p_{(X_i)_{i=1}^{n-1}}(\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}) &= \int_{\Omega_{X_n}} p_{(X_i)_{i=1}^n}(\mathbf{x}_1, \ldots, \mathbf{x}_n) \mu_{X_n}(d\mathbf{x}_n) \\
&\propto e^{\sum_{i=1}^{n-1} \boldsymbol{\theta}_{X_i} \cdot \mathbf{s}_{X_i}(\mathbf{x}_i) + \sum_{i=1}^{n-2} \mathbf{s}_{X_i}(\mathbf{x}_i) \cdot \boldsymbol{\Theta}_{X_i X_{i+1}} \cdot \mathbf{s}_{X_{i+1}}(\mathbf{x}_{i+1})} \\
&\qquad \int_{\Omega_{X_n}} e^{\boldsymbol{\theta}_{X_n} \cdot \mathbf{s}_{X_n}(\mathbf{x}_n) + \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \boldsymbol{\Theta}_{X_{n-1} X_n} \cdot \mathbf{s}_{X_n}(\mathbf{x}_n)} \mu_{X_n}(d\mathbf{x}_n) \\
&= e^{\sum_{i=1}^{n-1} \boldsymbol{\theta}_{X_i} \cdot \mathbf{s}_{X_i}(\mathbf{x}_i) + \sum_{i=1}^{n-2} \mathbf{s}_{X_i}(\mathbf{x}_i) \cdot \boldsymbol{\Theta}_{X_i X_{i+1}} \cdot \mathbf{s}_{X_{i+1}}(\mathbf{x}_{i+1})} \\
&\qquad + e^{\psi_{X_n}(\boldsymbol{\theta}_{X_n} + \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \boldsymbol{\Theta}_{X_{n-1} X_n})} \\
&= e^{\sum_{i=1}^{n-1} \boldsymbol{\theta}_{X_i} \cdot \mathbf{s}_{X_i}(\mathbf{x}_i) + \sum_{i=1}^{n-2} \mathbf{s}_{X_i}(\mathbf{x}_i) \cdot \boldsymbol{\Theta}_{X_i X_{i+1}} \cdot \mathbf{s}_{X_{i+1}}(\mathbf{x}_{i+1}) + \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \boldsymbol{\rho}_{X_{n-1}} + \rho_{n-1}} \\
&\propto e^{\sum_{i=1}^{n-2} \boldsymbol{\theta}_{X_i} \cdot \mathbf{s}_{X_i}(\mathbf{x}_i) + \sum_{i=1}^{n-2} \mathbf{s}_{X_i}(\mathbf{x}_i) \cdot \boldsymbol{\Theta}_{X_i X_{i+1}} \cdot \mathbf{s}_{X_{i+1}}(\mathbf{x}_{i+1}) + \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot (\boldsymbol{\theta}_{X_{n-1}} + \boldsymbol{\rho}_{X_{n-1}})},
\end{aligned}$$

which implies that $P_{(X_i)_{i=1}^{n-1}} \in \mathcal{D}_{(X_i)_{i=1}^{n-1}}$.

Conversely, suppose that $P_{(X_i)_{i=1}^{n-1}} \in \mathcal{D}_{(X_i)_{i=1}^{n-1}}$. Then according to the rectification theorem (4.1),

$$\psi_{X_n}(\boldsymbol{\theta}_{X_n} + \mathbf{s}_{(X_i)_{i=1}^{n-1}}((\mathbf{x}_i)_{i=1}^{n-1}) \cdot \boldsymbol{\Theta}_{(X_i)_{i=1}^{n-1} X_n}) = \mathbf{s}_{(X_i)_{i=1}^{n-1}}((\mathbf{x}_i)_{i=1}^{n-1}) \cdot \boldsymbol{\rho}_{(X_i)_{i=1}^{n-1}} + \rho_{n-1}.$$

Because $(X_i)_{i=1}^n$ is defined by exponential families with minimal sufficient statistics, and X_n only interacts with X_{n-1}, theorem 3.9 implies that all interactions between X_n and X_k for $k < n-1$ are 0, and therefore that $\mathbf{s}_{(X_i)_{i=1}^{n-1}}((\mathbf{x}_i)_{i=1}^{n-1}) \cdot \mathbf{\Theta}_{(X_i)_{i=1}^{n-1}X_n} = \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \mathbf{\Theta}_{X_{n-1}X_n}$.

This implies that $\mathbf{s}_{(X_i)_{i=1}^{n-1}}((\mathbf{x}_i)_{i=1}^{n-1}) \cdot \boldsymbol{\rho}_{(X_i)_{i=1}^{n-1}}$ depends only on $\mathbf{x}_{n-1}$. Let us reexpress the rectification parameters $\boldsymbol{\rho}_{(X_i)_{i=1}^{n-1}}$ as the tuple $\boldsymbol{\rho}_{(X_i)_{i=1}^{n-1}} = (\boldsymbol{\rho}_{(X_i)_{i=1}^{n-2}}, \boldsymbol{\rho}_{X_{n-1}}, \mathbf{P}_{(X_i)_{i=1}^{n-2}X_{n-1}})$, such that

$$\begin{aligned}\mathbf{s}_{(X_i)_{i=1}^{n-1}}((\mathbf{x}_i)_{i=1}^{n-1}) \cdot \boldsymbol{\rho}_{(X_i)_{i=1}^{n-1}} =&\mathbf{s}_{(X_i)_{i=1}^{n-2}}((\mathbf{x}_i)_{i=1}^{n-2}) \cdot \boldsymbol{\rho}_{(X_i)_{i=1}^{n-2}} + \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \boldsymbol{\rho}_{X_{n-1}} \\ &+ \mathbf{s}_{(X_i)_{i=1}^{n-2}}((\mathbf{x}_i)_{i=1}^{n-2}) \cdot \mathbf{P}_{(X_i)_{i=1}^{n-2}X_{n-1}} \cdot \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}).\end{aligned}$$

If the right hand side of this equation depends only on $\mathbf{x}_{n-1}$, then because all sufficient statistics under consideration are minimal, proposition 3.3 implies that $\boldsymbol{\rho}_{(X_i)_{i=1}^{n-2}}$ and $\mathbf{P}_{(X_i)_{i=1}^{n-2}X_{n-1}}$ must be zero. Therefore

$$\begin{aligned}\psi_{X_n}(\boldsymbol{\theta}_{X_n} + \mathbf{s}_{(X_i)_{i=1}^{n-1}}((\mathbf{x}_i)_{i=1}^{n-1}) \cdot \mathbf{\Theta}_{(X_i)_{i=1}^{n-1}X_n}) =&\psi_{X_n}(\boldsymbol{\theta}_{X_n} + \mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \mathbf{\Theta}_{X_{n-1}X_n}) \\ =&\mathbf{s}_{(X_i)_{i=1}^{n-1}}((\mathbf{x}_i)_{i=1}^{n-1}) \cdot \boldsymbol{\rho}_{(X_i)_{i=1}^{n-1}} + \rho_{n-1} \\ =&\mathbf{s}_{X_{n-1}}(\mathbf{x}_{n-1}) \cdot \boldsymbol{\rho}_{X_{n-1}} + \rho_{n-1}.\end{aligned}$$

□

This lemma describes how to rectify the factorization $((X_i)_{i=1}^{n-1}, X_n)$ of the deep harmonium $(X_i)_{i=1}^n$. This lemma can be recursively applied to every bilayer of a deep harmonium by simply again using the fact that subsequences $((X_i)_{i=1}^k)$ of a rectified deep harmonium are themselves deep harmoniums.

Theorem 4.10 (The Deep Rectification Theorem)**.** *Let $\mathcal{D}_{(X_i)_{i=1}^n}$ be a deep harmonium family defined by the exponential families $(\mathcal{M}_{X_i})_{i=1}^n$ with minimal sufficient statistics. Let $(X_i)_{i=1}^n$ be a deep harmonium with distribution $P_{(X_i)_{i=1}^n} \in \mathcal{D}_{(X_i)_{i=1}^n}$ and with parameters given by the biases $\boldsymbol{\theta}_{X_1}, \dots, \boldsymbol{\theta}_{X_n}$ and interaction matrices $\mathbf{\Theta}_{X_1X_2}, \dots, \mathbf{\Theta}_{X_{n-1}X_n}$. Then $(X_i)_{i=1}^n$ is rectified if and only if for some parameters $(\boldsymbol{\rho}_{X_i})_{i=1}^{n-1}$ and $(\rho_i)_{i=1}^{n-1}$,*

$$\psi_{X_k}(\boldsymbol{\theta}_{X_k} + \boldsymbol{\rho}_{X_k} + \mathbf{s}_{X_{k-1}}(\mathbf{x}_{k-1}) \cdot \mathbf{\Theta}_{X_{k-1}X_k}) = \mathbf{s}_{X_{k-1}}(\mathbf{x}_{k-1}) \cdot \boldsymbol{\rho}_{X_{k-1}} + \rho_{k-1}, \tag{4.16}$$

for every $1 < k \leq n$, where $\boldsymbol{\rho}_{X_n} = \mathbf{0}$.

Proof. Lemma 4.9 implies that 4.16 holds for $k = n$ for some rectification parameters $\boldsymbol{\rho}_{X_{n-1}}$ and ρ_{n-1}.

As a consequence, $P_{(X_i)_{i=1}^{n-1}} \in \mathcal{D}_{(X_i)_{i=1}^{n-1}}$ such that $(X_i)_{i=1}^k$ is a deep harmonium, where the bias of X_{n-1} is $\boldsymbol{\theta}_{X_{n-1}} + \boldsymbol{\rho}_{X_{n-1}}$. Lemma 4.9 therefore implies that $P_{(X_i)_{i=1}^{k-1}} \in \mathcal{D}_{(X_i)_{i=1}^{k-1}}$ if and only if

$$\begin{aligned}&\psi_{X_{n-1}}(\boldsymbol{\theta}_{X_{n-1}} + \boldsymbol{\rho}_{X_{n-1}} + \mathbf{s}_{X_{n-2}}(\mathbf{x}_{n-2}) \cdot \mathbf{\Theta}_{X_{n-2}X_{n-1}}) \\ &= \mathbf{s}_{X_{n-2}}(\mathbf{x}_{n-2}) \cdot \boldsymbol{\rho}_{X_{n-2}} + \rho_{n-2},\end{aligned}$$

for some parameters $\boldsymbol{\rho}_{X_{k-2}}$ and ρ_{k-2}. This establishes equation 4.16 for $k = n-1$.

By recursively applying lemma 4.9 in this manner to every bilayer of $(X_i)_{i=1}^n$, we may establish equation 4.16 for every k. □

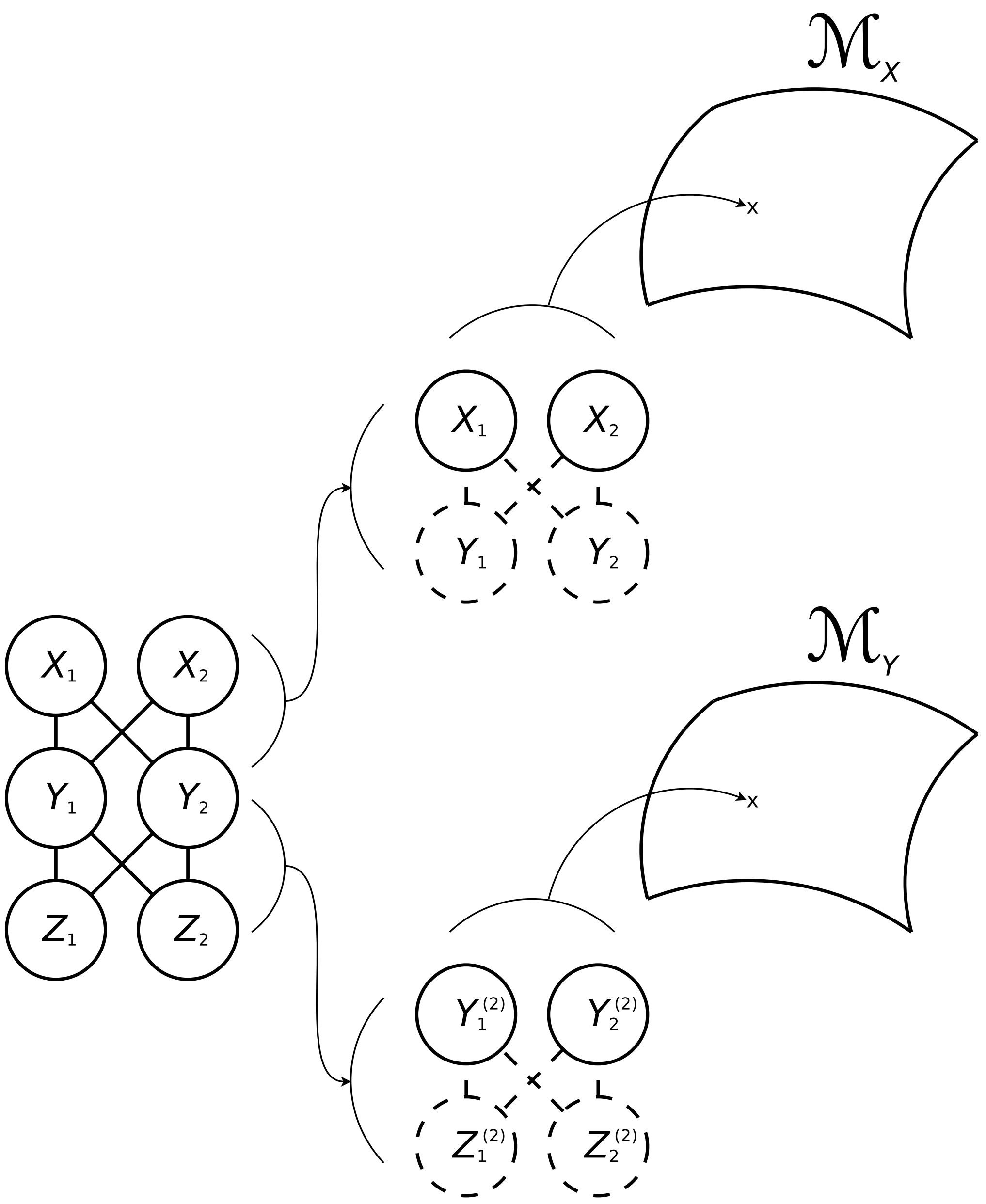

Figure 4.3: The deep harmonium (X, Y, Z) is rectified if the factorizations $((X, Y), Z)$ and $(X, (Y, Z))$ are rectified, and the rectification of $((X, Y), Z)$ and $(X, (Y, Z))$ reduces to the rectification of the bilayers $(X^{(1)}, Y^{(1)})$ and $(Y^{(2)}, Z^{(2)})$. In particular, $((X, Y), Z)$ is rectified if and only if the harmonium bilayer $(Y^{(2)}, Z^{(2)})$ is rectified. If $((X, Y), Z)$ is rectified, then the upper bilayer $(X^{(1)}, Y^{(1)})$ is equivalent to (X, Y), and $((X, Y), Z)$ is rectified if and only if (X, Y) is rectified.

Intuitively, theorem 4.10 reduces the rectification of a deep harmonium to the rectification of its bilayers. In order to formalize this intuition for a rectified deep harmonium $(X_i)_{i=1}^n$ with rectification parameters $(\boldsymbol{\rho}_{X_k})_{i=1}^{n-1}$ and $(\rho_i)_{i=1}^{n-1}$, I will often define the bilayer random variables $(X_{k-1}^{(k-1)}, X_k^{(k-1)})$ as the second-order harmonium with distribution $P_{X_{k-1}^{(k-1)} X_k^{(k-1)}} \in \mathcal{H}_{X_{k-1} X_k}$ and parameters $(\boldsymbol{\theta}_{X_{k-1}}, \boldsymbol{\theta}_{X_k} + \boldsymbol{\rho}_{X_k}, \boldsymbol{\Theta}_{X_{k-1} X_k})$. Based on this definition, $(X_i)_{i=1}^n$ is rectified if and only if $(X_{k-1}^{(k-1)}, X_k^{(k-1)})$ is rectified for every $1 < k \leq n$; I visualize this in figure 4.3. Note that based on this construction, the uppermost bilayer $(X_1^{(1)}, X_2^{(1)})$ is equivalent to (X_1, X_2).

Theorem 4.10 implies that the complexity of rectification only grows linearly with the number of layers in the deep harmonium. Analogous to corollary 4.4, we may also show that as a consequence of this theorem, the log-partition function of a rectified deep harmonium is a necessarily simple structure.

Corollary 4.11. *Let* $\mathcal{D}_{(X_i)_{i=1}^n}$ *be a deep harmonium family defined by the exponential families* $(\mathcal{M}_{X_i})_{i=1}^n$*. Moreover, let* $(X_i)_{i=1}^n$ *be a deep harmonium with distribution* $P_{(X_i)_{i=1}^n} \in \mathcal{D}_{(X_i)_{i=1}^n}$ *and parameters* $\boldsymbol{\theta}_{(X_i)_{i=1}^n}$ *given by the biases* $\boldsymbol{\theta}_{X_1}, \dots, \boldsymbol{\theta}_{X_n}$ *and interaction matrices* $\boldsymbol{\Theta}_{X_1 X_2}, \dots, \boldsymbol{\Theta}_{X_{n-1} X_n}$*, and suppose that* $(X_i)_{i=1}^n$ *is rectified with rectification parameters* $(\boldsymbol{\rho}_{X_i})_{i=1}^{n-1}$ *and* $(\rho_i)_{i=1}^{n-1}$*. Then the log-partition function of the harmonium is given by*

$$\psi_{(X_i)_{i=1}^n}(\boldsymbol{\theta}_{(X_i)_{i=1}^n}) = \psi_{X_1}(\boldsymbol{\theta}_{X_1} + \boldsymbol{\rho}_{X_1}) + \sum_{i=1}^{n-1} \rho_i.$$

Proof. Since $((X_i)_{i=1}^{n-1}, X_n)$ is rectified, corollary 4.4 implies that

$$\psi_{(X_i)_{i=1}^n}(\boldsymbol{\theta}_{(X_i)_{i=1}^n}) = \psi_{(X_i)_{i=1}^{n-1}}(\boldsymbol{\theta}_{(X_i)_{i=1}^{n-1}}) + \rho_{n-1}.$$

Since $(X_i)_{i=1}^{n-1}$ is also rectified, we may again apply corollary 4.4 to conclude that

$$\psi_{(X_i)_{i=1}^n}(\boldsymbol{\theta}_{(X_i)_{i=1}^n}) = \psi_{(X_i)_{i=1}^{n-2}}(\boldsymbol{\theta}^*_{(X_i)_{i=1}^{n-2}}) + \rho_{n-2} + \rho_{n-1},$$

where $\boldsymbol{\theta}^*_{X_{n-2}}$ are the natural parameters of the distribution $P_{(X_i)_{i=1}^{n-2}} \in D_{(X_i)_{i=1}^{n-2}}$, with biases on X_{n-2} equal to $\boldsymbol{\theta}_{X_{n-2}} + \boldsymbol{\rho}_{X_{n-2}}$. By recursively applying corollary 4.4 in this manner, we may derive the equation in the theorem statement. $\square$

4.2.2 Fitting and Rectifying Deep Harmoniums

The approach developed in subsection 4.1.3 for optimizing harmoniums can be extended to deep harmoniums, although it unsurprisingly requires additional considerations. In general, where $\mathcal{D}_{(X_i)_{i=1}^n}$ is the deep harmonium family defined by $(\mathcal{M}_{X_I})_{i=1}^n$, our goal will be to optimize the deep harmonium model $Q_{(X_i)_{i=1}^n} \in \mathcal{D}_{(X_i)_{i=1}^n}$. In order to rectify a deep harmonium, for every $1 < k \leq n$, we will consider the rectifier $Q^*_{X_k} \in \mathcal{M}_{X_k}$, and attempt to rectify the bilayer model $Q_{X_{k-1}^{(k-1)} X_k^{(k-1)}} \in \mathcal{H}_{X_{k-1} X_k}$. For the sake of clarity, but without loss of generality, I will develop the algorithm for the three-layer deep harmonium model Q_{XYZ}. At the end of this subsection I will explain how to generalize this algorithm to the n-layer case.

For the three-layer case, the algorithm I present is a three-stage algorithm based on the pretraining and fine-tuning approach first introduced by Hinton et al. (2006). The first stage of the optimization algorithm is to optimize the parameters $\boldsymbol{\theta}_Y$, $\boldsymbol{\theta}_Z$, and $\boldsymbol{\Theta}_{YZ}$ of the lower bilayer model $Q_{Y^{(2)} Z^{(2)}}$, and the parameters $\boldsymbol{\theta}^*_Y$ of the rectifier Q^*_Y. In this case our objective is to minimize the relative entropy of P_Z with respect to $Q_{Z^{(2)}}$, and we also to minimize the rectification error of Q^*_Y with respect to $Q_{Y^{(2)}}$. Since the lower bilayer is a second-order harmonium, we may achieve these goals in the same manner described in subsection 4.1.3. The objective of the first stage of the algorithm is therefore to evaluate

$$\underset{\boldsymbol{\theta}_Y, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{YZ}, \boldsymbol{\theta}^*_Y}{\arg\min} \left\{ \alpha_Z D(P_Z \parallel Q_{Z^{(2)}}) + \alpha_Y D(Q^*_Y \parallel Q_{Y^{(2)}} \right\}, \tag{4.17}$$

for some positive constants α_Z and α_Y. Once this objective function has been minimized, we move on to stage two.

In stage two we freeze the parameters $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{YZ}$, and we optimize the parameters of the upper bilayer $\boldsymbol{\theta}_X$, $\boldsymbol{\theta}_Y$, and $\boldsymbol{\Theta}_{XY}$. Stage two is based on the following upper-bound on the training objective $D(P_Z \parallel Q_Z)$.

Theorem 4.12. *Let $\mathcal{D}_{XYZ}$ be a deep harmonium family defined by the exponential families $\mathcal{M}_X$, $\mathcal{M}_Y$, and $\mathcal{M}_Z$ and let $Q_{XYZ} \in \mathcal{D}_{XYZ}$ be a deep harmonium model. Finally, let P_Z be a target observable distribution, and Y be the random variable with distribution P_Y given by*

$$p_Y(\mathbf{y}) = \int_{\Omega_Z} p_{Y|Z}(\mathbf{y} \mid \mathbf{z}) P_Z(d\mathbf{z}),$$

for an arbitrary, strictly positive conditional density $p_{Y|Z}$. Then

$$D(P_Z \parallel Q_Z) \leq D(P_Y \parallel Q_Y) + c.$$

for a constant c which is independent of the parameters $\boldsymbol{\theta}_X$, $\boldsymbol{\theta}_Y$, and $\boldsymbol{\Theta}_{XY}$ of the deep harmonium.

Proof. The following derivation is adapted from Salakhutdinov (2015).

Recall from equation 2.4 that $D(P_Z \parallel Q_Z) = H(P_Z, Q_Z) - H(P_Z)$, and from equation 2.3 that

$$H(P_Z, Q_Z) = -\int_{\Omega_Z} \log \frac{dQ_Z}{d\mu_Z} dP_Z = -\mathbb{E}_P[\log q_Z(Z)]. \tag{4.18}$$

If we consider $\log q_Z(Z)$ and apply Jensen's inequality we find that

$$\begin{aligned}
\log q_Z(Z) &= \log \int_{\Omega_Y} q_{YZ}(\mathbf{y}, Z) \mu_Y(d\mathbf{y}) \\
&= \log \int_{\Omega_Y} \frac{q_{YZ}(\mathbf{y}, Z)}{p_{Y|Z}(\mathbf{y} \mid Z)} p_{Y|Z}(\mathbf{y} \mid Z) \mu_Y(d\mathbf{y}) \\
&= \log \mathbb{E}_P\Big[\frac{q_{YZ}(Y, Z)}{p_{Y|Z}(Y \mid Z)} \mid Z\Big] \\
&\geq \mathbb{E}_P\Big[\log \frac{q_{YZ}(Y, Z)}{p_{Y|Z}(Y \mid Z)} \mid Z\Big] \\
&= \mathbb{E}_P[\log q_Y(Y) \mid Z] + \mathbb{E}_P[\log q_{Z|Y}(Z \mid Y) \mid Z] \\
&\qquad - \mathbb{E}_P[\log p_{Y|Z}(Y) \mid Z].
\end{aligned} \tag{4.19}$$

If we insert equation 4.19 back into equation 4.18 we may write

$$\begin{aligned}
H(P_Z, Q_Z) \leq H(P_Y, Q_Y) &- \mathbb{E}_P[\mathbb{E}_P[\log q_{Z|Y}(Z \mid Y) \mid Z]] \\
&+ \mathbb{E}_P[\mathbb{E}_P[\log p_{Y|Z}(Y \mid Z) \mid Z]],
\end{aligned} \tag{4.20}$$

where

$$-\mathbb{E}_P[\mathbb{E}_P[\log q_Y(Y) \mid Z]] = \mathbb{E}_P[\log q_Y(Y)] = H(P_Y, Q_Y). \tag{4.21}$$

Notice that the second term on the right hand side of inequality 4.20 depends only on the parameters $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{YZ}$, and the third term depends on none of the parameters of (X, Y, Z). Therefore, by defining the constant

$$\begin{aligned} c = &-\mathbb{E}_P[\mathbb{E}_P[\log q_{Z|Y}(Z \mid Y) \mid Z]] \\ &+ \mathbb{E}_P[\mathbb{E}_P[\log p_{Y|Z}(Y \mid Z) \mid Z]] - H(P_Z) - H(P_Y), \end{aligned} \tag{4.22}$$

inserting equation 4.22 and equation 4.21 back into equation 4.20, and rearranging terms, we establish the result of this theorem. □

The consequence of this theorem is that by reducing $D(P_Y \parallel Q_Y)$ with respect to the parameters $\boldsymbol{\theta}_X$, $\boldsymbol{\theta}_Y$, and $\boldsymbol{\Theta}_{XY}$, we may lower an upper-bound on $D(P_Z \parallel Q_Z)$ and hopefully, by extension, reduce it. As discussed in Hinton et al. (2006) and Salakhutdinov (2015), and shown in Neal and Hinton (1998), if $P_{Y|Z}$ is the conditional distribution of a harmonium and $p_{Y|Z} \cdot p_Z = p_{YZ} \in \mathcal{M}_{YZ}$, then the inequality is tight. Moreover, if P_Z is not the marginal of a harmonium, we can nevertheless tighten the bound by minimizing the relative entropy of a harmonium with respect to P_Z. Given this, where $Q_{Y^{(2)}Z^{(2)}}$ is the result of evaluating objective 4.17, we define $P_{Y|Z}$ as the conditional distribution of a harmonium with parameters $\boldsymbol{\theta}'_Y = \boldsymbol{\theta}_Y$ and $\boldsymbol{\Theta}'_{YZ} = \boldsymbol{\Theta}_{YZ}$. When we further optimize the parameters of the harmonium in stage two, we leave these parameters fixed.

Now we must figure out how to minimize $D(P_Y \parallel Q_Y)$. In the original pretraining approach presented in Hinton et al. (2006) for training a deep belief network, Q_Y is replaced with the marginal of a harmonium which is defined as the transpose of the lower bilayer, which ensures that this new marginal is equal to the replaced marginal. This is a severe restriction, and is rarely done in practice, and so inequality 4.20 is rather used heuristically to motivate pretraining. The form of pretraining developed for deep Boltzmann machines in (Salakhutdinov and Hinton, 2012) also requires a number of additional approximations. By applying the theory of rectification to pretraining, we may avoid these approximations and optimize Q_Y in a principled and practical manner.

As described in the previous subsection, if the harmonium $(Y^{(2)}, Z^{(2)})$ in stage one is rectified, then the marginal Q_{XY} of the deep harmonium model Q_{XYZ} is equal to the upper bilayer model $Q_{X^{(1)}Y^{(1)}}$. Since the lower bilayer is approximately rectified, we may instead conclude that Q_{XY} is approximately equal to the distribution $Q_{X^{(1)}Y^{(1)}}$. Therefore, we can approximately minimize $D(P_Y \parallel Q_Y)$ with respect to $\boldsymbol{\theta}_X$, $\boldsymbol{\theta}_Y$, and $\boldsymbol{\Theta}_{XY}$ by minimizing $D(P_Y \parallel Q_{Y^{(1)}})$. By theorem 4.12, this approximately reduces an upper bound on $D(P_Z \parallel Q_Z)$.

In addition to training the deep harmonium on data in stage two, we also wish to rectify the upper bilayer of the model. However, because the lower bilayer is approximately rectified, theorem 4.10 allows us to approximately reduce this rectification to the rectification of $(X^{(1)}, Y^{(1)})$. Therefore, where we define $Q^*_X \in \mathcal{M}_X$ as the rectifying distribution of the upper bilayer with parameters $\boldsymbol{\theta}^*_X$, we define the objective of stage two of the deep harmonium optimization algorithm as evaluating

$$\underset{\boldsymbol{\theta}_X, \boldsymbol{\theta}_Y, \boldsymbol{\Theta}_{XY}, \boldsymbol{\theta}^*_X}{\arg\min} \left\{ \alpha_Y D(P_Y \parallel Q_{Y^{(1)}}) + \alpha_X D(Q^*_X \parallel Q_{X^{(1)}}) \right\},$$

for some positive constants α_Y and α_X. Note that the rectification of the factorized harmonium $((X, Y), Z)$ does not depend on these parameters, and so evaluating this objective does not modify the rectification error of the lower bilayer. Since this objective once again has the form of the objective defined in the previous subsection for second-order harmoniums, we may again evaluate it with the techniques developed therein. Once this objective function has been minimized, we move on to stage three, wherein we attempt to train all the parameters of the harmonium model in parallel.

Now, based on equation 2.18, if we are given a sample $\mathbf{z} \in \Omega_Z$ from the target distribution P_Z, then stochastically minimizing $D(P_Z \parallel Q_Z)$ requires that we estimate the stochastic cotangent vector which starts at $\mathbb{E}_Q[\mathbf{s}_{XYZ}(X, Y, Z)]$ and ends at $\mathbb{E}_Q[\mathbf{s}_{XYZ}(X, Y, Z) \mid Z = \mathbf{z}]$. Let us refer to the starting point of this vector as the model point, and the end point of this vector as the posterior point. If Q_{XYZ} approximately models a rectified deep harmonium, then we can generate approximate samples from Q_{XYZ} with a single pass of sampling, as long as the elements of the component exponential families which define $\mathcal{D}_{XYZ}$ can be sampled. By generating such samples, we may estimate the model point of a strongly rectified deep harmonium with minimal computational cost.

Estimating the posterior point without extensive Gibbs sampling is more challenging, yet we can draw inspiration for solving this problem from contrastive divergence minimization. Although contrastive divergence minimization can only be applied to second-order harmoniums, it is nevertheless an algorithm for approximating the cotangent vector defined by the model and posterior points. In the case of second-order harmoniums, however, it is the posterior point that is trivial to calculate, and contrastive divergence minimization is rather designed to minimize the number of Gibbs-sampling steps required to estimate the model point.

Contrastive divergence minimization can be intuitively understood as the following procedure. Given an observation, the posterior is sampled in order to estimate the posterior point. Then, using the posterior sample to initialize the Gibbs sampler, each step of Gibbs sampling, on average, moves away from the posterior point in the direction of the model point. We then define the approximate cotangent vector as starting from the last step of the Gibbs sampler and ending at the estimated posterior point. If we adapt this strategy to our problem, we may begin by generating a sample from the model distribution to estimate the model point. We then initialize the Gibbs sampler with this model sample, and use Gibbs sampling to move from the model point in the direction of the posterior point, and use the end point of the Gibbs sampler to estimate the cotangent vector. I refer to this algorithm as dual contrastive divergence minimization, and I visualize these algorithms in figure 4.4.

As is conventional, let us refer to the contrastive divergence minimization algorithm with n steps of Gibbs sampling as CD-n, and let us similarly refer to the dual contrastive minimization algorithm as DCD-n. We may intuitively understand the efficiency of CD-n for low n by considering figure 4.4. Because each step of Gibbs sampling moves the posterior distribution in the direction of the model distribution, and since it is only the direction that we truly care about, CD-1 is often sufficient to produce good samples, and each additional step only improves the quality of the direction. Moreover, the shortness or bias of the CD-1 cotangent vector is at least partially compensated by the fact that the CD-1 estimation procedure has a smaller variance than the theoretically correct Gibbs sampler. Also note that if the generative model is perfect, such that the model and posterior distributions are the same, then the average cotangent vector produced by CD-1 will be 0.

All of these arguments for the efficiency of CD-n translate, *mutatis mutandis*, into

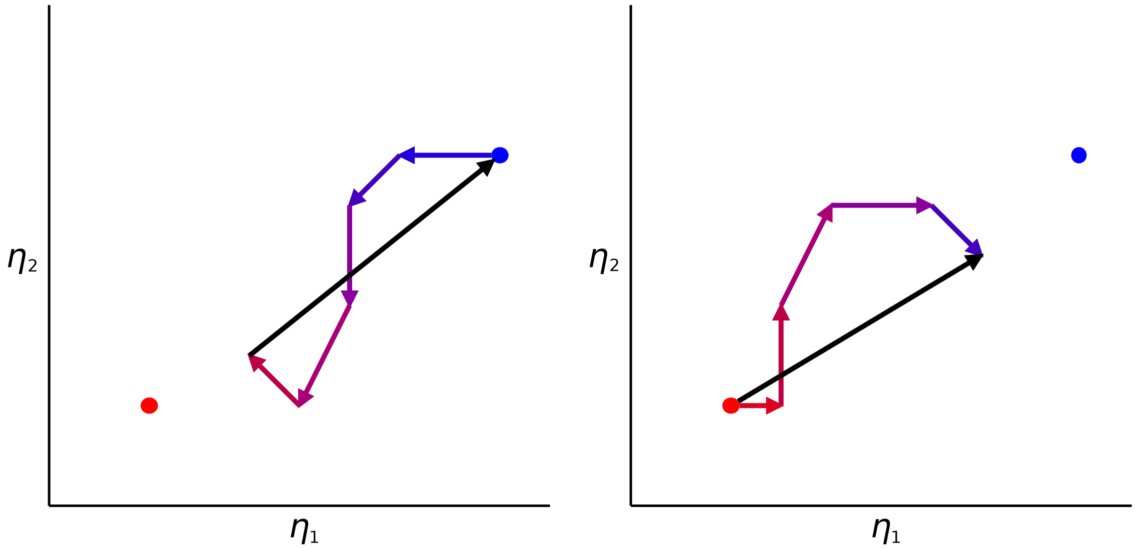

Figure 4.4: An intuitive depiction of estimating effective cotangent vectors with CD-5 and DCD-5 in mean coordinates. The red dot represents the model point and the blue point represents the posterior point. Each coloured vector starts at the estimated average of the sufficient statistic at step k of the corresponding algorithm, and ends at step $k+1$. The black vector represents the resulting cotangent vector. In both cases, using a short Gibbs chain would still result in a cotangent vector which moves the parameters of the harmonium in the correct direction. *Left*: A depiction of CD-5. CD-n starts from the posterior point and moves toward the model point. *Right*: A depiction of DCD-5. DCD-5 stats from the model point and moves toward the posterior point.

arguments for the efficiency of DCD-n. To the best of my understanding, there are two primary reasons why the gradients estimated by CD-n may nevertheless fail to point in the right direction, and why general convergence proofs for CD-n have not yet been found. On one hand, the cotangent vectors are defined by the expected values of the sufficient statistic of the harmonium in question, but the marginal distributions of the Gibbs sampler Markov chain are not elements of the corresponding harmonium family. As such, even though the distribution of the Markov chain may move in the right direction, the sufficient statistic may capture the wrong features of that distribution, such that the resulting cotangent vector will point away from the local optima.

On the other hand, as explained in section 2.2, cotangent vectors must theoretically be corrected by the metric tensor in order to compute the true gradient. In practice however, this is rarely done, because computing the metric is too expensive. Moreover, when the cotangent vectors are unbiased in direction, failing to correct the cotangent vector is not a fundamental problem, since, loosely speaking, the correction made by the metric tensor is never more than 90 degrees, and the local minima of the corrected and uncorrected gradients are the same. However, when estimated cotangent vectors are biased, failing to correct for the curvature of the manifold of harmoniums may also result in a misleading gradient.

All that being said, contrastive divergence minimization is a highly effective algorithm in practice, and by extension, dual contrastive divergence minimization should be as well. Let us now finally formalize DCD-n. To begin, let us define the transition distribution of the Gibbs chain $(X_k, Y_k)_{k \in \mathbb{N}}$ for the posterior $Q_{XY|Z=\mathbf{z}}$ of (X, Y, Z) as

$$p_{X'Y'|X,Y}(\mathbf{x}', \mathbf{y}' \mid \mathbf{x}, \mathbf{y}) = q_{X|Y}(\mathbf{x}' \mid \mathbf{y}') q_{Y|X,Z}(\mathbf{y}' \mid \mathbf{x}, \mathbf{z}),$$

and define the initial distribution of the Gibbs sampler by $P_{X_0 Y_0} = Q_{XY}$. The DCD-n

objective is then to minimize the relative entropy $D(P_{X_n Y_n Z} \parallel Q_{XYZ})$. Putting all of this together, we define the objective of rectifying and fitting a deep harmonium as

$$\underset{\theta_{XYZ}, \theta_X^*, \theta_Y^*}{\arg\min} \left\{ \alpha_Z D(P_{X_n Y_n Z} \parallel Q_{XYZ}) + \alpha_Y D(Q_Y^* \parallel Q_{Y^{(2)}}) + \alpha_X D(Q_X^* \parallel Q_{X^{(1)}}) \right\},$$

for some positive constants α_Z, α_Y, and α_X.

This algorithm may be generalized to an n-layered harmonium model $Q_{(X_i)_{i=1}^n}$ in the following way. Stage 1 remains the same, and is used to train and rectify the lowest bilayer $Q_{X_{n-1}^{(n-1)}, X_n^{(n-1)}}$. When stage 1 is complete, stage two may be iteratively applied from $k = 2$ to $k = n$ in order to train and rectify $Q_{X_{k-1}^{(k-1)} X_k^{(k-1)}})$ for every $1 < k \leq n$. Once the deep harmonium model is approximately rectified and partially trained, we may apply stage three to train the entire model $Q_{(X_i)_{i=1}^n}$ and minimize the rectification errors $D(Q_{X_k}^* \parallel Q_{X_k^{(k)}})$ in parallel, for all $1 < k \leq n$.

Chapter 5

Bayesian Inference with Artificial Neural Networks

In chapters 3 and 4 I described a large class of models which have priors which are conjugate to their posteriors, and thereby support efficient Bayesian inference. Nevertheless, this efficiency has remained relatively abstract, as I have yet to show how to implement Bayesian inference computationally. In this chapter I develop a theory of how to implement Bayesian inference with artificial neural networks.

In the first section of this chapter I consider Bayesian inference on static (i.e. non-dynamic) latent variables. There I introduce the concepts of implementing inference with a function, and encoding beliefs in some computational medium. I begin the second section by describing how to implement Bayesian prediction with abstract functions, and I introduce a class of multilayer perceptron designed for this purpose. By combining these multilayer perceptrons with a model of the emission distribution, I then show how to construct a class of recurrent neural networks, and train them to approximately implement a Bayes filter.

5.1 Implementing Static Inference

In this section I begin to formalize the concept of "implementing Bayesian inference". In the case of Bayesian inference on static latent variables, this reduces to implementing Bayes' rule and supporting conjugate priors. I then develop a simple theory of how to encode belief parameters with a different set of parameters, while still implementing Bayes' rule. This will allow me to unify a number of models from deep learning and computational neuroscience. Moreover, where we interpret these alternative parameters as the vector of firing rates of a population of neurons, this will allow us to describe how to encode beliefs in a neural population.

5.1.1 Implementing Bayes' Rule

In order to implement Bayes' rule in a machine or a mammalian brain, we must be able reduce Bayesian inference to the evaluation of tractable functions. One approach to this is to assume that the prior beliefs P_X and posterior beliefs $P_{X|Z}$ are in some sets of probability distributions M and M' parameterized by Θ and Θ', respectively. In this case we may interpret Bayes' rule as a function which computes the parameters of the posterior given the observation $\mathbf{z} \in \Omega_Z$ and the parameters of the prior $\boldsymbol{\theta} \in \Theta$.

Definition 5.1 (Implementing Bayes' Rule 1)**.** A function $\mathbf{f} \colon \Omega_Z \times \Theta \to \Theta'$ implements Bayes' rule with respect to the likelihood $P_{Z|X}$ if the posterior $P_{X|Z=\mathbf{z}} \in M'$ has parameters $\mathbf{f}(\mathbf{z}, \boldsymbol{\theta}) \in \Theta'$, for any observation $\mathbf{z} \in \Omega_Z$ and prior $P_X \in M$ with parameters $\boldsymbol{\theta} \in \Theta$.

Consider the harmonium family $\mathcal{H}_{XZ}$ defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$. We saw in subsection 3.1.2 that the parameters of the posterior of a harmonium are given by an affine function, and so let us endeavour to use this function to define an implementation of Bayes' rule. In general, the posterior $P_{X|Z}$ of a harmonium distribution in the family $\mathcal{H}_{XZ}$ is always in the exponential family $\mathcal{M}_X$, such that we may define the set of posteriors as $M' = \mathcal{M}_X$.

Let us fix a harmonium likelihood $P_{Z|X}$ (equation 3.6) with parameters $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{XZ}$. By construction, any prior P_X in the set of prior distributions $M = \{P_X \colon P_{Z|X} \cdot P_X \in \mathcal{H}_{XZ}\}$ ensures that the posterior of $P_{Z|X} \cdot P_X$ has the desired affine form. Moreover, because the posterior fixes the parameters $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{XZ}$, $\boldsymbol{\theta}_X$ is the only set of free parameters in the equation for the prior distribution (3.4), such that the parameter space $\Theta = \Theta_X$ parameterizes M. Based on this construction the function

$$\begin{aligned} \mathbf{f} \colon \Omega_Z \times \Theta_X &\to \Theta_X \\ (\mathbf{z}, \boldsymbol{\theta}_X) &\mapsto \boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}) \end{aligned}$$

implements Bayes' rule with respect to the likelihood $P_{Z|X}$. Note however, that although $\Theta = \Theta_X$, $M \neq \mathcal{M}_X$, because the elements of M do not typically have the exponential family form of the elements of $\mathcal{M}_X$.

As discussed in section 2.6, we often wish to evaluate Bayes' rule given a sequence of observations, and conjugate priors can greatly reduce the complexity of these computations. In general, the priors of a harmonium are not conjugate to their posteriors, and cannot support the recursive application of Bayes' rule in relation 2.23. However, if the likelihood $P_{Z|X}$ is rectified, then $M = \mathcal{M}_X$, such that P_X is conjugate to $P_{X|Z}$. In this case we may define an implementation of Bayes' rule as

$$\begin{aligned} \mathbf{f} \colon \Omega_Z \times \Theta_X &\to \Theta_X \\ (\mathbf{z}, \boldsymbol{\theta}^*_X) &\mapsto \boldsymbol{\theta}^*_X - \boldsymbol{\rho}_X + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}), \end{aligned} \tag{5.1}$$

where $\boldsymbol{\rho}_X$ and ρ_0 are the rectification parameters of $P_{Z|X}$, and $\boldsymbol{\theta}^*_X$ are the natural parameters of $P_X \in \mathcal{M}_X$. Based on this implementation of Bayes' rule with respect to a rectified likelihood, we may implement a general solution to recursive relation 2.23.

Proposition 5.1. *Let $\mathcal{H}_{XZ}$ be the harmonium family defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$. Let X be the latent variable with distribution $P_X \in \mathcal{M}_X$, and let $(Z_i)^n_{=1}$ be a sequence of random observations such that*

1. *$(Z_i)^n_{i=1}$ are conditionally independent given X,*
2. *$P_{Z_i|X}$ is a harmonium likelihood with parameters $\boldsymbol{\theta}_{Z_i}$ and $\boldsymbol{\Theta}_{XZ_i}$,*
3. *and $P_{Z_i|X}$ is rectified with rectification parameters $\boldsymbol{\rho}^i_X$ and ρ^i_0.*

Then the function

$$\begin{aligned} \mathbf{f} \colon \Omega_Z \times \cdots \times \Omega_Z \times \Theta_X &\to \Theta_X \\ (\mathbf{z}_1, \dots, \mathbf{z}_n, \boldsymbol{\theta}^*_X) &\mapsto \boldsymbol{\theta}^*_X - \sum_{i=1}^{n} \boldsymbol{\rho}^i_X + \boldsymbol{\Theta}_{XZ_i} \cdot \mathbf{s}_Z(\mathbf{z}_i), \end{aligned}$$

implements Bayes' rule with respect to the likelihood $P_{(Z_i)_{i=1}^n|X}$.

Proof. Let $(\mathbf{f}_i)_{i=1}^n$ be the sequence of functions

$$\begin{aligned} \mathbf{f}_i \colon \Omega_Z \times \Theta_X &\to \Theta_X \\ (\mathbf{z}, \boldsymbol{\theta}^*_X) &\mapsto \boldsymbol{\theta}^*_X - \boldsymbol{\rho}^i_X + \boldsymbol{\Theta}_{XZ_i} \cdot \mathbf{s}_Z(\mathbf{z}), \end{aligned} \tag{5.2}$$

and let $\mathbf{f}^{(n)}$ be the function given recursively by

$$\begin{aligned} \mathbf{f}^{(i)}(\mathbf{z}_1, \dots, \mathbf{z}_i, \boldsymbol{\theta}^*_X) &= \mathbf{f}_i(\mathbf{z}_i, \mathbf{f}^{(i-1)}(\mathbf{z}_1, \dots, \mathbf{z}_{i-1}, \boldsymbol{\theta}^*_X)), \\ \mathbf{f}^{(0)}(\boldsymbol{\theta}^*_X) &= \boldsymbol{\theta}^*_X. \end{aligned} \tag{5.3}$$

By construction, $\mathbf{f}^{(n)} = \mathbf{f}$, and we need only show that it indeed implements recursive Bayesian inference.

For the case $i = 1$, $\mathbf{f}^{(1)}(\mathbf{z}_1, \boldsymbol{\theta}^*_X) = \boldsymbol{\theta}^*_X - \boldsymbol{\rho}^1_X + \boldsymbol{\Theta}_{XZ_1} \cdot \mathbf{s}_Z(\mathbf{z}_1)$, which are the parameters of the posterior $P_{Z_1|X} \in \mathcal{M}_X$.

For $i > 1$, suppose that $\mathbf{f}^{(i-1)}(\mathbf{z}_1, \dots, \mathbf{z}_{i-1}, \boldsymbol{\theta}_X)$ are the parameters of the posterior $P_{X|(Z_j)_{j=1}^{i-1}} \in \mathcal{M}_X$. Because the observations $(Z_j)_{j=1}^{i-1}$ are conditionally independent given X, $P_{XZ_i|(Z_j)_{j=1}^{i-1}} = P_{Z_i|X} \cdot P_{X|(Z_j)_{j=1}^{i-1}}$. Since $P_{Z_i|X}$ is rectified, corollary 4.3 implies that $P_{Z_i|X} \cdot P_{X|(Z_j)_{j=1}^{i-1}} \in \mathcal{H}_{XZ}$ with parameters $(\mathbf{f}^{(i)} - \boldsymbol{\rho}^i_X, \boldsymbol{\theta}_{Z_i}, \boldsymbol{\Theta}_{XZ_i})$. According to equation 3.5, the posterior of this distribution at $\mathbf{z}_i$ has parameters $\mathbf{f}^{(i)} - \boldsymbol{\rho}^i_X + \boldsymbol{\Theta}_{XZ_i} \cdot \mathbf{s}_Z(\mathbf{z}_i)$, which according to equations 5.2 and 5.3, is the value of $\mathbf{f}^{(i)}(\mathbf{z}_1, \dots, \mathbf{z}_i, \boldsymbol{\theta}_X)$. Thus $P_{X|(Z_j)_{j=1}^{i}} \in \mathcal{M}_X$ has parameters $\mathbf{f}^{(i)}(\mathbf{z}_1, \dots, \mathbf{z}_i, \boldsymbol{\theta}_X)$.

Therefore, for any i, $\mathbf{f}^{(i)}$ implements Bayes' rule with respect to $P_{(Z_j)_{j=1}^{i}|X}$, and in particular when $i = n$. □

In the proof of this proposition we saw the functions $(\mathbf{f}_i)_{i=1}^n$ which transform the prior parameters and an observation into the parameters of the sequence of posteriors. Intuitively, each of these functions may correspond to different sensory modalities, which provide independent information about the latent state. These different sources of information are then summed by the function $\mathbf{f}$, which integrates the independent observations into a single set of beliefs, and combines them with the parameters of the prior $\boldsymbol{\theta}_X$.

If $\mathbf{f}_i = \mathbf{f}$ for any i and some function $\mathbf{f}$, then the single function $\mathbf{f}$ may be recursively applied to infer the latent state given an arbitrary number of observations. In this case we may think of $\mathbf{f}$ as specifying a recurrent neural network, which accumulates information about a static latent variable over time. The case where the stimulus is dynamic leads to implementing Bayesian filtering, which I consider later in this chapter.

5.1.2 Encoding Beliefs

In this subsection my goal is show how we can encode the prior and posterior distributions of a harmonium in a pair of additional random variables, and to implement Bayesian inference with respect to these encodings.

Definition 5.2 (Encoding)**.** Given the random variables $X \in \Omega_X$ and $Z \in \Omega_Z$, the distribution over X encoded by $\mathbf{z} \in \Omega_Z$ is $P_{X|Z=\mathbf{z}}$.

The distribution encoded by $\mathbf{z}$ is simply another name for the conditional distribution of X given Z at $\mathbf{z}$. The name however emphasizes that it is the distribution encoded by $\mathbf{z}$ for which we aim to implement Bayesian inference.

Let us consider the harmonium family $\mathcal{H}_{XZ}$ and the harmonium (X, Z) with distribution P_{XZ} and parameters $(\boldsymbol{\theta}_X, \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$, and two additional exponential families $\mathcal{M}_V$ and $\mathcal{M}_W$. Let us refer to $V \in \mathrm{H}_V$ as the random prior encoding and $W \in \mathrm{H}_W$ as the random posterior encoding[8]. When W and V are large random vectors composed of positive values, we may use them to represent the rates of neural populations in a brain.

In order to justify referring to V and W as encodings of the prior and posterior, respectively, I want to ensure that the conditional distribution distributions of the latent variable given these variables satisfy $P_{X|V} = P_X$ and $P_{X|W} = P_{X|Z}$. I will do this with a new definition of implementation analogous to what I provided in the previous subsection.

Definition 5.3 (Implementing Bayes' Rule 2)**.** A function $\mathbf{w}\colon \Omega_Z \times \mathrm{H}_V \to \Theta$ implements Bayes' rule with respect (X, Z, V, W) where $W = \mathbf{w}(Z, V)$ if $P_{X|V} = P_X$ and $P_{X|W} = P_{X|Z}$.

If we assume that $P_{X|V} = P_X$, then we may characterize this definition as requiring that the random variables (X, Z, V, W) are a Bayesian network represented by the graph in figure 5.1, such that their distribution can be factorized as

$$P_{VWXZ} = P_Z \cdot P_V \cdot P_{W|V,Z} \cdot P_{X|W}.$$

This graphical representation implies that W captures all the information available in V and Z about X. Enforcing this representation requires that $P_{ZV} = P_Z \cdot P_V$, and more importantly that $P_{X|ZVW} = P_{X|W}$. Because $P_{X|V} = P_X$ and $\mathbf{w}$ is a function of Z, this implies that $P_{X|Z} = P_{X|ZVW}$, and therefore that $P_{X|W} = P_{X|Z}$.

Let us begin constructing an implementation of Bayes' rule for encodings by defining V as independent of Z, and defining the density encoded by the prior encoding $\mathbf{v}$ as

$$p_{X|V}(\mathbf{x} \mid \mathbf{v}) \propto e^{\mathbf{s}_X(\mathbf{x})\cdot(\boldsymbol{\theta}_X^V + \boldsymbol{\Theta}_{XV}\cdot\mathbf{v}) + \psi_Z(\boldsymbol{\theta}_Z + \mathbf{s}_X(\mathbf{x})\cdot\boldsymbol{\Theta}_{XZ})}, \tag{5.4}$$

where $\boldsymbol{\Theta}_{XV} \in \Theta_X \otimes \Theta_V$ is the prior decoder matrix, and $\boldsymbol{\theta}_X^V \in \Theta_X$ is the decoder bias. Based on equation 3.4, $p_{X|V=\mathbf{v}}$ is equal to the prior P_X of the harmonium P_{XZ} when $\boldsymbol{\theta}_X = \boldsymbol{\theta}_X^V + \boldsymbol{\Theta}_{XV} \cdot \mathbf{v}$. Therefore, we may ensure that $P_{X|V} = P_X$, where

---

[8] Note that, although I use the notation H_V and H_W to define the state spaces of V and W, the distributions of V and W will not be elements of the exponential families $\mathcal{M}_V$ and $\mathcal{M}_W$. This notation is nevertheless convenient when we later build larger neural networks involving these belief encodings, as the state spaces of V and W must overlap with certain exponential family sample spaces.

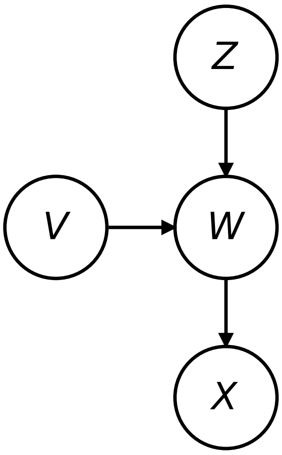

Figure 5.1: Here I depict a directed graphical representation of the distribution P_{VWXZ}. This graph captures all the features required to describe the notion of prior and posterior encodings, except for the fact that $P_{X|V} = P_X$.

$P_X \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta}_X$, by defining P_V as any distribution over the pre-image $\{\mathbf{v} \colon \boldsymbol{\theta}_X = \boldsymbol{\theta}_X^V + \boldsymbol{\Theta}_{XV} \cdot \mathbf{v}\}$.

Let us now define W by $W = \mathbf{w}(Z, V)$, where

$$\begin{aligned} \mathbf{w} \colon \Omega_Z \times \mathrm{H}_V &\to \mathrm{H}_W \\ (\mathbf{z}, \mathbf{v}) &\mapsto \mathbf{A} \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{B} \cdot \mathbf{v} + \mathbf{w}_0, \end{aligned} \tag{5.5}$$

where $\mathbf{w}_0 \in \mathrm{H}_W$, and where $\mathbf{A} \in \mathrm{H}_W \otimes \Theta_Z$ and $\mathbf{B} \in \mathrm{H}_W \otimes \Theta_V$ are matrices which we refer to as the observation and prior recoder matrices, respectively. Based on this construction, we may state the following proposition about the conditions under which $\mathbf{w}$ implements Bayesian inference.

Proposition 5.2. *Let $\mathcal{H}_{XZ}$ be a harmonium family, and let the harmonium distribution P_{XZ} be an element of $\mathcal{H}_{XZ}$. Suppose that $P_{X|V}$ is given by equation 5.4, that $P_{X|V} = P_X$, and that $W = \mathbf{w}(Z, V)$ is given by equation 5.5. Then $\mathbf{w}$ implements Bayesian inference with respect to (X, Z, V, W) if and only if*

$$\begin{aligned} \boldsymbol{\Theta}_{XZ} &= \boldsymbol{\Theta}_{XW} \cdot \mathbf{A}, \\ \boldsymbol{\Theta}_{XV} &= \boldsymbol{\Theta}_{XW} \cdot \mathbf{B}, \\ \boldsymbol{\theta}_X^V &= \boldsymbol{\theta}_X^W + \boldsymbol{\Theta}_{XW} \cdot \mathbf{w}_0, \end{aligned} \tag{5.6}$$

for some $\boldsymbol{\Theta}_{XW} \in \Theta_X \otimes \Theta_W$, $\boldsymbol{\theta}_X^W \in \Theta_W$, and $\mathbf{w}_0 \in H_W$, and

$$p_{X|W}(\mathbf{x} \mid \mathbf{w}) \propto e^{\mathbf{s}_X(\mathbf{x}) \cdot (\boldsymbol{\theta}_X^W + \boldsymbol{\Theta}_{XW} \cdot \mathbf{w})}.$$

Proof. Because W is a function of V and Z, $P_{X|ZVW} = P_{X|ZV}$. Therefore equations 5.6 are satisfied if and only if

$$\begin{aligned} p_{X|ZV}(\mathbf{x} \mid \mathbf{z}, \mathbf{v}) &\propto e^{\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{s}_X(\mathbf{x}) \cdot (\boldsymbol{\theta}_X^V + \boldsymbol{\Theta}_{XV} \cdot \mathbf{v})} \\ &= e^{\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\theta}_X^W + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XW} \cdot (\mathbf{A} \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{B} \cdot \mathbf{v} + \mathbf{w}_0)} \\ &= e^{\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\theta}_X^W + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XW} \cdot \mathbf{w}(\mathbf{z}, \mathbf{v})}. \end{aligned}$$

In this equation, because $\mathbf{z}$ and $\mathbf{v}$ interact with $\mathbf{x}$ exclusively through $\mathbf{w}$, the Fisher-Neyman factorization theorem[9] implies that this equation holds if and only if $P_{X|ZV} = P_{X|W}$, and

$$p_{X|W}(\mathbf{x} \mid \mathbf{w}) \propto e^{\mathbf{s}_X(\mathbf{x})\cdot(\boldsymbol{\theta}_X^W + \boldsymbol{\Theta}_{XW}\cdot\mathbf{w})}.$$

□

5.2 Implementing Bayesian Filtering

In this section I show how to implement and learn to implement Bayes filters. I begin by defining the concept of implementation for Bayesian prediction in particular, and Bayesian filtering in general. I then develop a form of recurrent neural network which I call exponential family multilayer perceptrons, or EFMLPs for short. EFMLPs are a reformulation of classic multilayer perceptrons using the language of exponential families, and provide a general class of regression architecture for approximately implementing Bayesian prediction.

I then define two classes of neural network for modelling Bayes filters based on EFMLPs. I show how these neural networks improve on existing models based on the universal representation theory of multilayer perceptrons (Hornik, 1993). Finally, I derive a gradient descent approach to training these neural networks given the observations from a hidden Markov chain.

5.2.1 Implementing Bayesian Prediction

Let us now generalize the theory from the previous section on implementing Bayesian inference given sequences of independent observations, to include Bayesian filtering over a hidden Markov chain $(X_k, Z_k)_{k\in\mathbb{N}}$ (see section 2.6). For simplicity, I will assume that the emission distribution $P_{Z|X}$ of the Markov chain is time-invariant. This will allow us to reduce the implementation of a Bayes filter to a pair of mutually recursive functions.

A Bayes filter is an online combination of Bayes' rule and prediction, and so the theory we have so far developed addresses the first part of implementing a Bayes filter. In order to implement Bayesian prediction, let us assume that the predictions $P_{X_{k+1}|(Z_i)_{i=0}^k}$ and posteriors $P_{X_k|(Z_i)_{i=0}^k}$ for any $k \in \mathbb{N}$ are in some sets M and M' parameterized by Θ and Θ', respectively. In this case we may interpret prediction as a function which computes the parameters $\boldsymbol{\theta}_{k+1} \in \Theta$ of the prediction distribution at time $k+1$ given the parameters $\boldsymbol{\theta}'_k \in \Theta'$ of the posterior at time k.

Definition 5.4 (Implementing Bayesian Prediction 1)**.** The function $\mathbf{g}\colon \Theta' \to \Theta$ implements Bayesian prediction with respect to the transition distribution $P_{X'|X}$ if the prediction $P_{X_{k+1}|(Z_i)_{i=0}^k} \in M$ has parameters $\mathbf{g}(\boldsymbol{\theta}'_k) \in \Theta$, for any $k \in \mathbb{N}$, and any posterior $P_{X_k|(Z_i)_{i=0}^k} \in M'$ with parameters $\boldsymbol{\theta}'_k \in \Theta'$.

In the context of a time-invariant emission distribution $P_{X|Z}$, I define an implementation of a Bayes filter as a pair of functions which implement Bayes' rule and implement Bayesian prediction.

---

[9] See the Wikipedia article on sufficient statistics at `https://en.wikipedia.org/w/index.php?title=Sufficient_statistic&oldid=795500815`.

Definition 5.5 (Implementing a Bayes Filter 1)**.** The pair of functions $\mathbf{f} \colon \Omega_Z \times \Theta \to \Theta'$ and $\mathbf{g} \colon \Theta' \to \Theta$ implement a Bayes filter with respect to the Markov chain $(X_k, Z_k)_{k \in \mathbb{N}}$ if $\mathbf{f}$ implements Bayes' rule with respect to the emission distribution $P_{Z|X}$, and $\mathbf{g}$ implements Bayesian prediction with respect to the transition distribution $P_{X'|X}$.

Because the implementations of Bayes' rule and Bayesian prediction map into each others domains, we can use the implementation of a Bayes filter to compute the posteriors at any time k.

Proposition 5.3. *If the pair of functions* $(\mathbf{f}, \mathbf{g})$ *implement a Bayes filter with respect to the hidden Markov chain* $(X_k, Z_k)_{k \in \mathbb{N}}$*, then for any* $k > 0$*, the posterior* $P_{X_k | (Z_i)_{i=0}^k = (\mathbf{z}_i)_{i=0}^k} \in M'$ *has parameters* $\mathbf{f}(\mathbf{z}_k, \mathbf{g}(\boldsymbol{\theta}'_{k-1})) \in \Theta'$*, for any observation* $\mathbf{z}_k \in \Omega_Z$ *and any previous posterior* $P_{X_{k-1} | (Z_i)_{i=0}^{k-1} = (\mathbf{z}_i)_{i=0}^{k-1}} \in M'$ *with parameters* $\boldsymbol{\theta}'_k \in \Theta'$.

Proof. This follows from a similar recursive argument to the one used in proposition 5.1, where the components function under considerations are given by

$$\mathbf{f}^{(k+1)}(\mathbf{z}_1, \dots, \mathbf{z}_{k+1}) = \mathbf{f}(\mathbf{z}_{k+1}, \mathbf{g}(\mathbf{f}^{(k)}(\mathbf{z}_1, \dots, \mathbf{z}_k))).$$

In this case however, we do not need to make use of rectified likelihoods. ☐

In contrast with the case of static latent variables, this proposition describes how to process sequences of observations without depending on conjugate priors or rectified likelihoods. Nevertheless, there remain several advantages to conjugate priors in the context of Bayesian filtering.

Let $\mathcal{H}_{XZ}$ be the harmonium family defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$. First of all, if the observations come from multiple independent senses, and therefore the implementation of Bayes' rule must integrate multiple, conditionally independent observations at every timestep, then rectified emission distributions and proposition 5.1 provide a practical means of doing so. Secondly, if the emission distribution is rectified, then when we later attempt to train neural networks to implement a Bayes filter, we may still take advantage of the algorithms developed in chapter 4 to efficiently estimate the learning gradients. Finally, in most cases where solutions of the prediction and update equations afford closed-form expressions, the predictions of a Bayes filter are indeed conjugate to the posteriors.

The simplest example of this is when $\mathcal{M}_X$ is the categorical distribution with states $\Omega_X = \{x_i\}_{i=1}^n$. In this case the prediction equation may be evaluated brute-force by computing

$$p_{X_k | (Z_i)_{i=0}^{k-1}}(x_k \mid (\mathbf{z}_i)_{i=0}^{k-1}) = \sum_{i=1}^{n} p_{X'|X}(x_k \mid x_i) p_{X_{k-1} | (Z_i)_{i=0}^{k-1}}(x_i \mid (\mathbf{z}_i)_{i=0}^{k-1}).$$

This equation corresponds more or less to computing the mean parameters of a categorical distribution, and by using the forward and backward mappings, we may construct a corresponding function $\mathbf{g} \colon \Theta_X \to \Theta_X$ which implements this equation in the natural parameter space of $\mathcal{M}_X$. Moreover, since any emission distribution $P_{Z|X}$ is rectified if $\mathcal{M}_X$ is the categorical distribution (theorem 4.5), we may combine this

$\mathbf{g}$ with the function $\mathbf{f}$ defined in equation 5.1 to construct a general implementation $(\mathbf{f}, \mathbf{g})$ of a Bayes filter over a latent variable with finite states.

Let us now consider the case where $\mathcal{M}_X$ and $\mathcal{M}_Z$ are multivariate normal families. Theorem 4.6 implies that if the emission distribution is given by $P_{Z|X=\mathbf{x}} = \mathrm{N}(\mathbf{m}+\mathbf{M}\cdot\mathbf{x}, \boldsymbol{\Sigma})$, then $P_{Z|X}$ is rectified, and evaluating Bayes' rule is trivial. Similarly, suppose that the transition distribution of $(X_k)_{k\in\mathbb{N}}$ is given by $P_{X'|X=\mathbf{x}} = \mathrm{N}(\mathbf{N}\cdot\mathbf{x}, \mathbf{T})$, and that the posterior $P_{X_k|(Z_i)^k_{i=0}} \in \mathcal{M}_X$ with parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$. Then the prediction distribution $P_{X_{k+1}|(Z_i)^k_{i=0}}$ has parameters $\boldsymbol{\mu}^0_{k+1}$ and $\boldsymbol{\Sigma}^0_{k+1}$ given by

$$\begin{aligned}\boldsymbol{\mu}^0_{k+1} &= \mathbf{N}\cdot\boldsymbol{\mu}_k,\\ \boldsymbol{\Sigma}^0_{k+1} &= \mathbf{N}\cdot\boldsymbol{\Sigma}_k\cdot\mathbf{N}^\top + \mathbf{T},\end{aligned}$$

(Thrun et al., 2005; Särkkä, 2013).

By again relying on the isomorphisms between the different parameterizations of $\mathcal{M}_X$, we may use these equations to define the implementations $\mathbf{f}$ and $\mathbf{g}$, and construct the implementation $(\mathbf{f}, \mathbf{g})$ of a Bayes filter. Moreover, we need not limit ourselves to multivariate normal observations, and we may combine the implementation $\mathbf{g}$ of Bayesian prediction with the implementation $\mathbf{f}$ of Bayes' rule of any rectified emission distribution. The pair $(\mathbf{f}, \mathbf{g})$ is then an implementation of a Bayes filter on a hidden Markov chain with observations with a form of our choosing.

Moreover, this technique for implementing Bayes filters is not limited to optimal solutions. The most well-known extension of Kalman filtering to nonlinear dynamical systems is the extended Kalman filter (Thrun et al., 2005; Särkkä, 2013). Although the extended Kalman filter does not exactly solve the prediction equation, it does compute good multivariate normal approximations to the true solutions.

The extended Kalman is applied to hidden Markov models with transition distributions of the form $P_{X'|X=\mathbf{x}} = \mathrm{N}(\mathbf{h}(\mathbf{x}), \mathbf{T})$. Where $\hat{P}_{X_k|(Z_i)^k_{i=0}}$ is the approximate posterior at time k with parameters $\boldsymbol{\mu}^0_{k+1}$ and $\boldsymbol{\Sigma}^0_{k+1}$, the approximate, multivariate-normal prediction $\hat{P}_{X_{k+1}|(Z_i)^k_{i=0}}$ of the extended Kalman filter has parameters

$$\begin{aligned}\boldsymbol{\mu}^0_{k+1} &= \mathbf{h}(\boldsymbol{\mu}_k),\\ \boldsymbol{\Sigma}^0_{k+1} &= \partial_{\boldsymbol{\mu}_k}\mathbf{h}(\boldsymbol{\mu}_k)\cdot\boldsymbol{\Sigma}_k\cdot\partial_{\boldsymbol{\mu}_k}\mathbf{h}(\boldsymbol{\mu}_k)^\top + \mathbf{T}.\end{aligned} \tag{5.7}$$

Observe that these equations reduce to the Kalman filter prediction equations when $\mathbf{h}$ is a linear function.

There is another approximate solution to the prediction equation that I make use of in this dissertation. The von Mises family is an exponential family with a state space equal to $[-\pi, \pi]$, and is naturally suited to describing probability distributions over rotational data. The von Mises family can be parameterized by the mean μ and so-called concentration parameter κ. When κ is small, the von Mises distribution $\mathrm{vM}(\mu, \kappa)$ is approximately equal to the normal distribution $\mathrm{N}(\mu, \kappa^{-1})$.

Suppose we are given the transition distribution defined by $P_{X'|X=\mathbf{x}} = \mathrm{vM}(h(\mu), \lambda)$. If we are given the approximate, von Mises posterior $\hat{P}_{X_k|(Z_i)^k_{i=0}}$ with parameters (μ_k, κ_k), and assume that λ and κ_k are small, then we may compute approximate predictions $\hat{P}_{X_{k+1}|(Z_i)^0_{i=1}k}$ by evaluating the parameters

$$\begin{aligned}\mu^0_{k+1} &= h(\mu_k),\\ \frac{1}{\kappa^0_{k+1}} &= \frac{(h'(\mu_k))^2}{\kappa_k} + \lambda^{-1},\end{aligned}$$

based on equations 5.7. I will apply this technique for implementing an approximate von Mises Bayes filter later in this dissertation.

5.2.2 Exponential Family Multilayer Perceptrons

A discriminative model is a statistical model which is a function of an additional variable, and multilayer perceptrons are one of the most well-known classes of discriminative model. In this subsection I develop a general form of multilayer perceptron based on the theory of exponential families, which I refer to as exponential family multilayer perceptrons (EFMLPs). An exponential family multilayer perceptron (EFMLP) is designed to process the natural parameters of belief distributions, and as we will see in the next section, EFMLPs are especially well suited to implementing the predictions of a Bayes filter.

Suppose we are given a sequence of n exponential families $(\mathcal{M}_{Y_i})_{i=1}^{n}$, with dual coordinate systems H_{Y_i} and Θ_{Y_i}. Where $\mathcal{M}_{Y_{i+1}Y_i}$ are the harmonium families defined by these exponential families $\mathcal{M}_{Y_{i+1}}$ and $\mathcal{M}_{Y_i}$, I recursively define an n-layered EFMLP as the function

$$\begin{aligned} \boldsymbol{\theta}_{Y_n|Y_1} \colon \mathrm{H}_{Y_1} &\to \Theta_{Y_n} \\ \boldsymbol{\eta}_{Y_1} &\mapsto \boldsymbol{\theta}_{Y_n} + \boldsymbol{\Theta}_{Y_nY_{n-1}} \cdot \boldsymbol{\eta}_{Y_{n-1}|Y_1}(\boldsymbol{\eta}_{Y_1}), \end{aligned} \tag{5.8}$$

where $\boldsymbol{\eta}_{Y_n|Y_1}$ is defined as

$$\begin{aligned} \boldsymbol{\eta}_{Y_n|Y_1} \colon \mathrm{H}_{Y_1} &\to \mathrm{H}_{Y_n} \\ \boldsymbol{\tau}_{Y_1} &\mapsto \boldsymbol{\eta}_{Y_n}(\boldsymbol{\theta}_{Y_n|Y_1}(\boldsymbol{\eta}_{Y_1})) \end{aligned}$$

for $n > 1$, and where $\boldsymbol{\eta}_{Y_1|Y_1}(\boldsymbol{\eta}_{Y_1}) = \boldsymbol{\eta}_{Y_1}$ for $n = 1$. Taken together, I denote the parameters of an EFMLP by $\boldsymbol{\chi} = (\boldsymbol{\Theta}_{Y_2Y_1}, \boldsymbol{\theta}_{Y_2}, \ldots, \boldsymbol{\Theta}_{Y_nY_{n-1}}, \boldsymbol{\theta}_{Y_n})$ and the parameter space by $\mathcal{X}$.

Where we denote Y_1 by W and Y_n by X, I define the EFMLP discriminative model $Q_{X|W}$ of X given W as

$$q_{X|W}(\mathbf{x} \mid \mathbf{w}) = e^{\mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\theta}_{X|W}(\mathbf{w}) - \psi_X(\boldsymbol{\theta}_{X|W}(\mathbf{w}))}. \tag{5.9}$$

which is the exponential family distribution $Q_{X|W=\mathbf{w}} \in \mathcal{M}_X$ with parameters $\boldsymbol{\theta}_{X|W}(\mathbf{w})$[10]. I depict an example three-layer EFMLP in figure 5.2.

Now suppose we wish to train the EFMLP discriminative model $Q_{X|W}$ with parameters $\boldsymbol{\chi}$. Let $X \in \Omega_X$ and $W \in \mathrm{H}_W$ be random variables, and let us define the conditional relative entropy as

$$D(P_{X|W} \parallel Q_{X|W}) = \int_{\mathrm{H}_W} \int_{\Omega_X} \log \frac{p_{X|W}(\mathbf{x} \mid \mathbf{w})}{q_{X|W}(\mathbf{x} \mid \mathbf{w})} P_{XW}(d\mathbf{x}, d\mathbf{w}).$$

---

[10] Note that although a generative model can be thought of as a parameterized joint distribution, a discriminative model is not in general a parameterized conditional distribution. A conditional distribution is derived from a joint distribution, whereas a discriminative model is simply a function which takes an input and returns a distribution. The input of a discriminative model is simply another set of parameters of the model, and may not have any well define probabilistic relationship with the random variable of the output distribution.

$$\begin{tikzcd}
 & \Theta_Y \arrow[d, "\boldsymbol{\tau}_Y"'] & \Theta_X \\
\mathrm{H}_W \arrow[ur, "\boldsymbol{\theta}_{Y|W}"] & \mathrm{H}_Y \arrow[ur, "\boldsymbol{\theta}_{X|Y}"] &
\end{tikzcd}$$

Figure 5.2: A diagram of the composition of a generic three-layer perceptron where the domain of $\boldsymbol{\theta}_{X|W}$ is H_W and the codomain is H_X. The functions $\boldsymbol{\theta}_{Y|W}$ and $\boldsymbol{\theta}_{X|Y}$ represent the affine steps of the evaluation of the multilayer perceptron. If we define the hidden layer in terms of the product family $\mathcal{M}_Y$ of independent Bernoulli distributions, then the coordinate transform $\boldsymbol{\tau}_Y$ defines the transfer function of the multilayer perceptron as the logistic function.

Then based on equation 2.12, it follows that

$$\begin{aligned}
&\partial_{\chi_i} D(P_{X|W} \parallel Q_{X|W}) = \\
&\qquad \mathbb{E}_P[(\mathbb{E}_Q[\mathbf{s}_X(X) \mid W] - \mathbb{E}_P[\mathbf{s}_X(X) \mid W]) \cdot \partial_{\chi_i} \boldsymbol{\theta}_{X|W}(\mathbf{w})].
\end{aligned} \tag{5.10}$$

The derivative in equation 5.10 is the expected value of the dot product of a pair of random vectors on $(\Omega, \mathcal{F}, P)$. Let us denote the first random vector by

$$R_X = \mathbb{E}_Q[\mathbf{s}_X(X) \mid W] - \mathbb{E}_P[\mathbf{s}_X(X) \mid W],$$

and refer to it as the residual of the EFMLP. The second vector in this dot product is the derivative of the EFMLP $\boldsymbol{\theta}_{X|W}$ at $\mathbf{w}$ with respect to the parameter χ_i. The derivatives of $\boldsymbol{\theta}_{X|W}(\mathbf{w})$ with respect to the final-layer parameters $\boldsymbol{\theta}_{Y_n}$ and $\boldsymbol{\Theta}_{Y_n Y_{n-1}}$ are given by

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_{Y_n}} D(P_{X|W} \parallel Q_{X|W}) &= \mathbb{E}_P[R_X], \\
\partial_{\boldsymbol{\Theta}_{Y_n Y_{n-1}}} D(P_{X|W} \parallel Q_{X|W}) &= \mathbb{E}_P[R_X \otimes \boldsymbol{\eta}_{Y_{n-1}|Y_1}(W)].
\end{aligned} \tag{5.11}$$

To evaluate the derivatives with respect to the remaining parameters of the network, we must apply the chain rule to $\boldsymbol{\eta}_{Y_{n-1}|Y_1}(W)$. Observe that the derivatives of $\boldsymbol{\eta}_{Y_{n-1}|Y_1}(W)$ at $\boldsymbol{\theta}_{Y_{n-1}|Y_1}(W)$ are equal to the Hessian $\partial_{\boldsymbol{\theta}_{Y_{n-1}|Y_1} \boldsymbol{\theta}_{Y_{n-1}|Y_1}} \psi_{Y_{n-1}}(\boldsymbol{\theta}_{Y_{n-1}|Y_1}(W))$, and that the derivatives of $\boldsymbol{\theta}_{Y_{n-1}|Y_1}(W)$ with respect to the parameters $\boldsymbol{\theta}_{Y_{n-1}}$ and $\boldsymbol{\Theta}_{Y_{n-1}Y_{n-2}}$ have the same form as equations 5.11 with a different random residual. Therefore, by defining the recursive residuals as the random variables

$$R_{Y_i} = R_{Y_{i+1}} \cdot \boldsymbol{\Theta}_{Y_{i+1}Y_i} \cdot \partial_{\boldsymbol{\theta}_{Y_i|Y_1} \boldsymbol{\theta}_{Y_i|Y_1}} \psi_{Y_i}(\boldsymbol{\theta}_{Y_i|Y_1}(W)),$$

for $R_{Y_n} = R_X$, we may express the derivatives in equation 5.10 with respect to all the component parameters of $\boldsymbol{\chi}$ as

$$\begin{aligned}
\partial_{\boldsymbol{\theta}_{Y_i}} D(P_{X|W} \parallel Q_{X|W}) &= \mathbb{E}_P[R_{Y_i}], \\
\partial_{\boldsymbol{\Theta}_{Y_i Y_{i-1}}} D(P_{X|W} \parallel Q_{X|W}) &= \mathbb{E}_P[R_{Y_i} \otimes \boldsymbol{\eta}_{Y_{i-1}|Y_1}(W)].
\end{aligned}$$

Backpropagation is the algorithm of recursively evaluating these expectations based on realizations of W (Rumelhart et al., 1986), and this recursion ends at the input

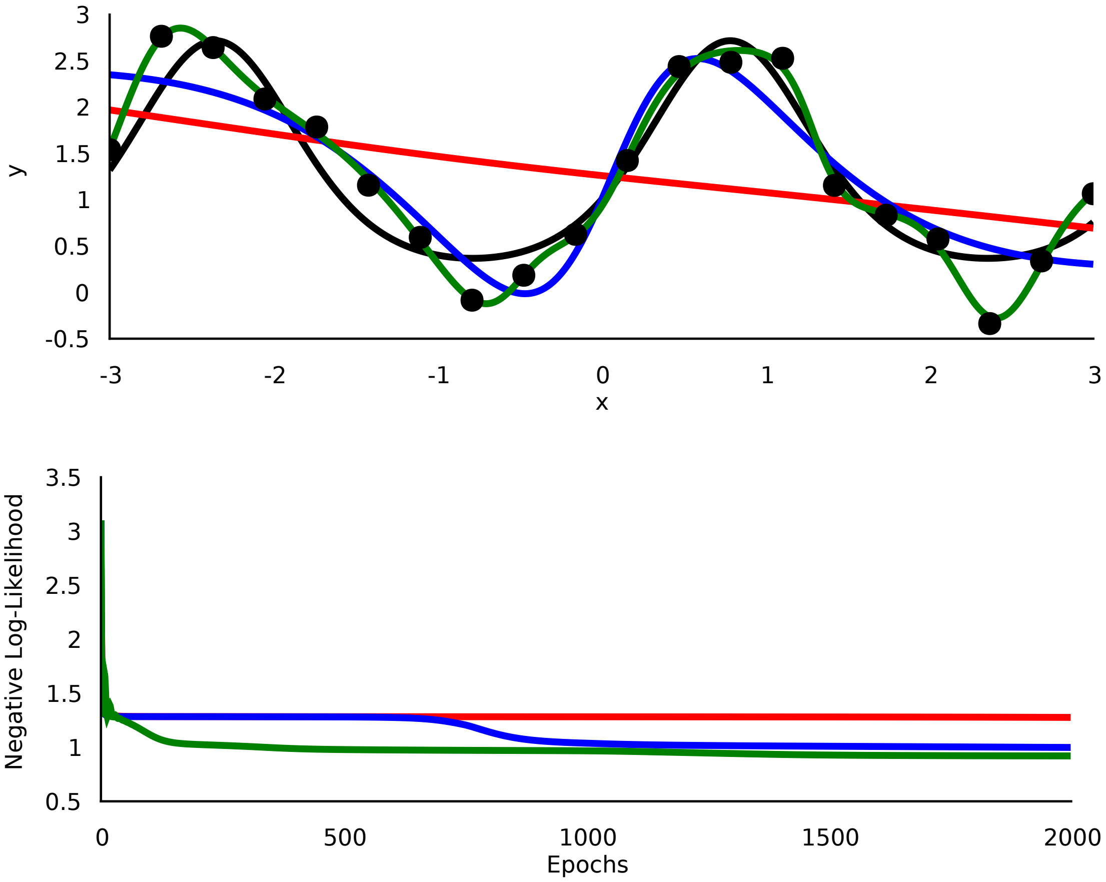

Figure 5.3: An application of backpropagation to training EFMLPs where $\mathcal{M}_W = \mathcal{M}_X$ is the family of normal distributions with known variance and $\mathcal{M}_Y$ is the product family of 20 Bernoulli families. *Top*: A depiction of the learned and target functions. In particular, I depict the mean of the target function (black line), the noisy samples used to train the network (black dots), as well as the mean of the EFMLP trained with vanilla gradient descent (red line), classic momentum (blue line), and the Adam algorithm (green line). *Bottom*: The negative log-likelihood of the data for the three learned EFMLPs (red, blue, green, as before).

W, which does not depend on the parameters $\boldsymbol{\chi}$. I present an example of learning an EFMLP with backpropagation in figure 5.3.

My primary use for EFMLPs in this dissertation is in modelling harmoniums for Bayesian inference. That is, where $\mathcal{H}_{XZ}$ is the harmonium family defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$, I consider EFMLPs for which $\mathcal{M}_{Y_n} = \mathcal{H}_{XZ}$. As defined by equation 5.9, training this EFMLP discriminative model for all but the simplest harmonium families is intractable, since training the final layer would involve optimizing a third-order tensor. However, we may construct a powerful, yet tractable class of EFMLP harmonium model by considering discriminative harmonium models with densities of the form

$$q_{XZ|W}(\mathbf{x}, \mathbf{z} \mid \mathbf{w}) \propto e^{\boldsymbol{\theta}_{X|W}(\mathbf{w}) \cdot \mathbf{s}_X(\mathbf{x}) + \boldsymbol{\theta}_Z \cdot \mathbf{s}_Z(\mathbf{z}) + \mathbf{s}_X(\mathbf{x}) \cdot \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(\mathbf{z})}, \tag{5.12}$$

which is the density of a harmonium with distribution $Q_{XZ|W=\mathbf{w}} \in \mathcal{H}_{XZ}$ and parameters $(\boldsymbol{\theta}_{X|W}(\mathbf{w}), \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$, where $\boldsymbol{\theta}_{X|W}$ is an EFMLP with domain H_W and codomain Θ_X.

As long as the residual R_X can be effectively sampled, then training a harmonium discriminative model with the form of equation 5.12 is no more complicated than the one presented in equation 5.9. Moreover, the remaining, non-conditional parameters $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{XZ}$ of the conditional harmonium may be trained by the methods discussed and developed in chapters 3 and 4. Especially in the case of rectified harmoniums, the resulting model also affords an intuitive interpretation. That is, we may think of the EFMLP $\boldsymbol{\theta}_{X|W}(\mathbf{w})$ as computing the parameters the prior $Q_{X|W=\mathbf{w}}$. Given this prior and the likelihood $Q_{Z|X}$ which is independent of the parameters $\boldsymbol{\theta}_{X|W}(\mathbf{w})$, we may then compute the posterior of the harmonium model $Q_{XZ|W=\mathbf{w}}$ given the observation $\mathbf{z} \in \Omega_Z$ by evaluating $Q_{X|Z=\mathbf{z},W=\mathbf{w}}$.

5.2.3 Modelling a Bayes Filter

In this subsection I develop theory for modelling Bayes filters. This work generalizes and unifies the work on modelling time series in Sutskever et al. (2009) and Boulanger-Lewandowski et al. (2012), and the work on implementing Bayesian filtering in neural circuits in Beck et al. (2011). As I show, Sutskever et al. (2009) present a model for approximate filtering which exactly implements Bayes' rule, but which has no guarantees on its ability to approximate the predictions; Boulanger-Lewandowski et al. (2012) present a model which can approximate the prediction equation in a principled manner, but fails to implement Bayes' rule in general; and Beck et al. (2011) develop a theoretical neural circuit which can exactly implement a Kalman filter, but training the neural circuit was not considered.

I present a pair of model Bayes filters which extend the machine learning approaches of Sutskever et al. (2009) and Boulanger-Lewandowski et al. (2012), and which exactly implements Bayes' rule and can approximate the solution of the prediction equation with the marginal distribution of a harmonium arbitrarily well. The model in Beck et al. (2011) can be interpreted as a form of the second model introduced here, and I discuss the proposed neural circuit directly in subsection 6.2. The general approach I take is to rely on the theory of harmoniums to compute exact posteriors, and to use the EFMLPs introduced in the previous subsection to compute approximate predictions.

In all cases we start with a hidden Markov chain $(X_k, Z_k)_{k\in\mathbb{N}}$, defined by the initial distribution P_{X_0}, the transition distribution $P_{X'|X}$, and we assume that the emission distribution $P_{Z|X}$ is time-invariant. At a any time k, our goal is to model the conditional distribution $P_{X_kZ_k|(Z_i)_{i=0}^{k-1}}$ of the hidden Markov chain with a discriminative harmonium model $Q_{X_kZ_k|(Z_i)_{i=0}^{k-1}}$, which is an element of the harmonium family $\mathcal{H}_{XZ}$ at every k. Now, if $Q_{X_kZ_k|(Z_i)_{i=0}^{k-1}}$ depends directly on the entire history of random observations $(Z_i)_{i=0}^{k-1}$, then the complexity of $Q_{X_k|(Z_i)_{i=0}^{k-1}}$ grows linearly with time, and a distinct model must be learned for every k.

In order to avoid the complexity issues of maintaining the complete history of observations and learning a sequence of models, let us consider an EFMLP which computes the parameters of the discriminative harmonium model $Q_{X_kZ_k|(Z_i)_{i=0}^{k-1}}$ as a function of the parameters of the previous model posterior $Q_{X_{k-1}|(Z_i)_{i=0}^{k-1}}$. Observe that we may tractably compute the model posterior $Q_{X_k|(Z_i)_{i=0}^{k}}$ as a function of the previous model posterior by evaluating the EFMLP, and computing the posterior distribution of $Q_{X_kZ_k|(Z_i)_{i=0}^{k-1}}$. By repeating this procedure, we define a recurrent neural network based on the EFMLP which models a Bayes filter at every time k.

Based on the theory which I have so far presented in the dissertation, there are two forms of recurrent neural network which we might define for computing the parame-

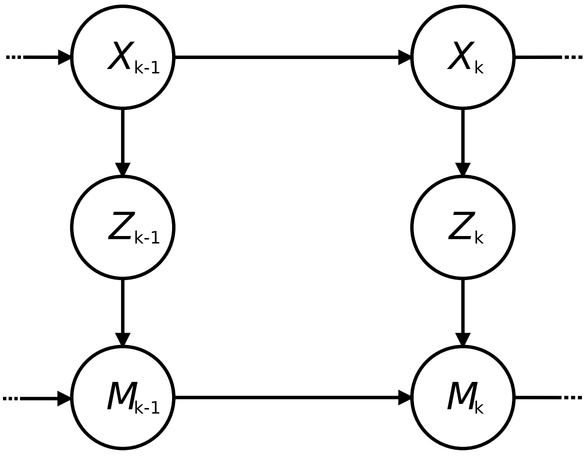

Figure 5.4: Here I depict the directed graphical representation of an EFMLP Bayes filter. The random variables in the graph are latent states X_k, the observations Z_k, and the mean parameters M_k of the dynamic model posterior.

ters of the discriminative harmonium model. In both cases the EFMLP parameterizes a discriminative harmonium model with the form given in relation 5.12, such that the observable biases $\boldsymbol{\theta}_Z$ and interactions parameters $\boldsymbol{\Theta}_{XZ}$ of the harmonium are time-invariant, and the two models differ only in how they compute the dynamic latent biases.

I refer to the first kind of recurrent neural network as the EFMLP Bayes filter. In an EFMLP Bayes filter, the latent parameters of the discriminative harmonium model are computed by the EFMLP $\boldsymbol{\theta}_{X|M} \colon \mathrm{H}_X \to \Theta_X$, which takes the mean parameters of the previous model posterior and returns the latent biases of the discriminative harmonium model. More formally, let us recursively define the random posterior mean parameters by

$$M_k = \boldsymbol{\tau}_X(\boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(Z_k) + \boldsymbol{\theta}_{X|M}(M_{k-1})), \tag{5.13}$$

where $M_0 = \boldsymbol{\eta}_{X_0}$ are the mean parameters of the model prior Q_{X_0}. The discriminative harmonium model $Q_{X_k Z_k|(Z_i)_{i=0}^{k-1}}$ of the EFMLP Bayes filter at time k is then $Q_{XZ|M=M_{k-1}}$, where $Q_{XZ|M=M_{k-1}} \in \mathcal{H}_{XZ}$ with parameters $(\boldsymbol{\theta}_{X|M}(M_{k-1}), \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$. I depict the graphical representation of an EFMLP Bayes filter in figure 5.4.

The RTRBM model of Sutskever et al. (2009) is a form of EFMLP Bayes filter where $\boldsymbol{\theta}_{X|M}$ a 2-layer EFMLP, such that $\boldsymbol{\theta}_{X|M}$ is an affine function[11]. Established results on universal approximation (Hornik, 1993) imply that an EFMLP architecture requires at least one hidden layer of an arbitrary size with a non-polynomial transfer function in order to model arbitrary latent biases. Although the RTRBM does apply the logistic transfer function when computing the mean parameters of the previous posterior, it does not first apply a parameterized linear transformation to the inputs of these logistic units, and the size of the logistic layer is fixed by the dimension of the parameter space of the posterior. Therefore, the RTRBM cannot model arbitrary latent biases as a function of the parameters of the previous posterior.

---

[11] Both Sutskever et al. (2009) and Boulanger-Lewandowski et al. (2012) also consider observable biases which depend on the previous posterior. However, since observable biases do not affect posterior distributions in harmonium generative models, this has no affect on the analysis of this section, and I elide this detail for the sake of simplicity.

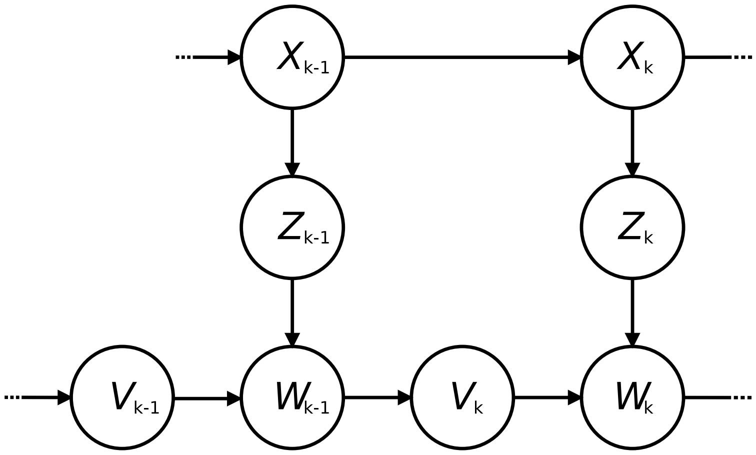

Figure 5.5: Here I depict the directed graphical representation of a neural Bayes filter. The chain is composed of latent states X_k, observations Z_k, the prediction encoding V_k, and the posterior encoding W_k. The arrow from W_{k-1} to V_k represents the computation of the neural network $\boldsymbol{\eta}_{V|W}$, such that $V_k = \boldsymbol{\eta}_{V|W}(W_{k-1})$.

The second model I consider is based on the theory of belief encodings developed in subsection 5.1.2. Consider two exponential families $\mathcal{M}_V$ and $\mathcal{M}_W$. Let us define two dynamic encodings $(V_k)_{k\in\mathbb{N}}$ and $(W_k)_{k\in\mathbb{N}}$, with state spaces H_V and H_W, and refer to them as the prediction and posterior encodings, respectively. We define the conditional distribution of the posterior encodings $P_{W_k|Z_k,V_k}$ by

$$W_k = \mathbf{A} \cdot \mathbf{s}_Z(Z_k) + \mathbf{B} \cdot V_k + \mathbf{w}_0,$$

where $\mathbf{A}$ and $\mathbf{B}$ are the observation and prediction recoders, and $\mathbf{w}_0$ is the encoding bias.

Now suppose that $\boldsymbol{\theta}_{X|W} \colon \mathrm{H}_W \to \Theta_X$ is an n-layered EFMLP defined by the sequence of exponential families $(\mathcal{M}_{Y_i})_{i=1}^n$, where $\mathcal{M}_{Y_1} = \mathcal{M}_W$, $\mathcal{M}_{Y_n} = \mathcal{M}_X$, and $\mathcal{M}_{Y_{n-1}} = \mathcal{M}_V$. Since $\boldsymbol{\theta}_{X|W}(\mathbf{w}) = \boldsymbol{\theta}_X + \boldsymbol{\Theta}_{XV} \cdot \boldsymbol{\eta}_{V|W}(\mathbf{w})$ by mapping 5.8, if we define $V_k = \boldsymbol{\eta}_{V|W}(W_{k-1})$, then V_k encodes predictions over X_k as a function of the encoding of the previous posterior, where $\boldsymbol{\Theta}_{XV}$ is the decoder matrix, and $\boldsymbol{\theta}_X$ is the decoder bias.

By combining the conditional distributions $P_{X'|X}$, $P_{Z|X}$, $P_{W_k|Z_k,V_k}$, and $P_{V_k|W_{k-1}}$, we may construct the Markov chain $(X_k, Z_k, V_k, W_k)_{k\in\mathbb{N}}$. We define the discriminative harmonium model $Q_{X_k Z_k|(Z_i)_{i=0}^{k-1}}$ at time k as the discriminative harmonium model $Q_{ZX|W=W_{k-1}}$ defined by the EFMLP $\boldsymbol{\theta}_{X|W}$. This model is also a form of recurrent neural network, but it does not implement Bayes' rule in general, unlike the EFMLP Bayes filter. Therefore, we place an additional restriction on this recurrent neural network, in order to ensure it does not lose information during the update step.

Definition 5.6 (Neural Bayes Filter)**.** The Markov chain $(X_k, Z_k, V_k, W_k)_{k\in\mathbb{N}}$ is a neural Bayes filter if at every time k, $\mathbf{w}$ implements Bayes' rule with respect to (X_k, Z_k, V_k, W_k).

I depict the graphical representation of a neural Bayes filter in figure 5.5.

The RNN-RBM model presented in Boulanger-Lewandowski et al. (2012) is a form of neural Bayes filter where $\boldsymbol{\eta}_{V|W}(W) = \boldsymbol{\tau}_V(W)$, such that $\boldsymbol{\theta}_V = \mathbf{0}$ and $\boldsymbol{\Theta}_{VW} = \mathbf{I}$. In the RNN-RBM, the parameters $\mathbf{A}$, $\mathbf{B}$, $\mathbf{w}_0$, $\boldsymbol{\theta}_X$, $\boldsymbol{\Theta}_{XV}$, $\boldsymbol{\Theta}_{XZ}$, and $\boldsymbol{\theta}_Z$ are treated as independent, trainable parameters. Since the dimension of the state-space of W_k

is not fixed by the complexity of the posterior, and is defined as an arbitrary affine transformation of the sufficient statistics of the observation $\mathbf{s}_Z(Z_k)$ and the encoded prediction V_k, an RNN-RBM can in theory approximate any latent bias arbitrary well.

However, according to proposition 5.2, W_k encodes the posterior over X_k given the predictions encoded by V_k if and only if V_k and W_k satisfy equations 5.6. If the parameters of the RNN-RBM do not satisfy these equations, then W_k encodes suboptimal posteriors, which implies that the RNN-RBM loses information about the latent state. This loss of information can only be mitigated by ensuring that equations 5.6 are satisfied, which prevents the EFMLP of the RNN-RBM from being a universal approximator.

In this subsection I have introduced two recurrent neural networks – the EFMLP Bayes filter, and the neural Bayes filter – both of which compute exact solutions to the update equation (2.24) by computing the posterior of a harmonium model, and can approximate the solution of the prediction equation (2.25) with the marginal of a harmonium model with arbitrary latent biases. Due to the universal approximation theory of multilayer perceptrons (Hornik, 1993), the recurrent neural networks I have introduced in this subsection can model Bayes filters arbitrarily well, when the predictions are given by the marginal of a harmonium distribution. By contrast, the RTRBM and RNN-RBM lack the requisite complexity to model these Bayes filters in general.

5.2.4 Training a Model Bayes Filter

In this section I show how to train EFMLP Bayes filters and neural Bayes filters. Let us once again consider the hidden Markov chain $(X_k, Z_k)_{k \in \mathbb{N}}$, defined by the initial distribution P_{X_0}, the transition distribution $P_{X'|X}$, and the time-invariant emission distribution $P_{Z|X}$. Training the neural Bayes filter requires additional considerations which do not arise when training EFMLP Bayes filters, and so we begin with the latter.

Let us consider the harmonium family $\mathcal{H}_{XZ}$ defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$, and the n-layer EFMLP $\boldsymbol{\theta}_{X|M}$ defined by the exponential families $(\mathcal{M}_{Y_i})_{i=1}^n$ where $\mathcal{M}_{Y_1} = \mathcal{M}_{Y_n} = \mathcal{M}_X$, with parameters $\boldsymbol{\chi}$. Let $Q_{XZ|M=M_{k-1}} \in \mathcal{H}_{XZ}$ be the discriminative harmonium model of the EFMLP Bayes filter at time k with parameters $(\boldsymbol{\theta}_{X|M}(M_{k-1}), \boldsymbol{\theta}_Z, \boldsymbol{\Theta}_{XZ})$. Since we do not observe the latent states of the hidden Markov chain directly, we train the EFMLP Bayes filter by minimizing the conditional relative entropy

$$D(P_{Z_k|M_{k-1}} \parallel Q_{Z|M=M_{k-1}}) \tag{5.14}$$

for every k.

The derivatives of entropy 5.14 with respect to the parameters $\boldsymbol{\theta}_Z$ and $\boldsymbol{\Theta}_{XZ}$ are simply the derivatives of the harmonium parameters given by equations 3.8. Based on equations 2.17 and 5.10, the derivatives of entropy 5.14 with respect to χ_i is

$$\partial_{\chi_i} D(P_{Z_k|M_{k-1}} \parallel Q_{Z|M=M_{k-1}}) = \mathbb{E}_P[R_{Z_k} \cdot \partial_{\chi_i} \boldsymbol{\theta}_{X|M=M_{k-1}}(M_{k-1})], \tag{5.15}$$

where

$$R_{Z_k} = \mathbf{r}_{Z_k}(M_{k-1})$$

are the residuals of the neural network at time k, and where

$$\begin{aligned}\mathbf{r}_{Z_k}(\mathbf{m}_{k-1}) = \mathbb{E}_Q[&\mathbf{s}_{XZ}(X, Z) \mid M = \mathbf{m}_{k-1}] \\ &- \mathbb{E}_P[\mathbb{E}_Q[\mathbf{s}_{XZ}(X, Z) \mid Z = Z_k, M = \mathbf{m}_{k-1}]].\end{aligned}$$

Note that $\mathbf{r}_{Z_k}$ at $\mathbf{m}_{k-1}$ is again simply the relative entropy derivatives of a harmonium (3.8).

Since we have principled methods for estimating the residuals, all we must do to estimate the expectation in equation 5.15 is to compute $\partial_{\chi_i}\boldsymbol{\theta}_{X|M}(M_{k-1})$. Although backpropagation allows us to partially evaluate these derivatives, M_{k-1} is defined recursively by the EFMLP, and so backpropagation does not simply end when M_{k-1} is reached. This leads to the algorithm known as backpropagation-through-time (Williams and Zipser, 1989; Werbos, 1990; Sutskever et al., 2009; Makin et al., 2016) for computing the complete derivatives of $\partial_{\chi_i}\boldsymbol{\theta}_{X|M}(M_{k-1})$.

Let us rewrite equation 5.13 as

$$M_{k-1} = \boldsymbol{\tau}_X(T_{k-1} + \boldsymbol{\Theta}_{XY_{n-1}} \cdot \boldsymbol{\eta}_{Y_{n-1}|M}(M_{k-2})),$$

where $T_{k-1} = \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_Z(Z_{k-1}) + \boldsymbol{\theta}_X$. Then we may express $\boldsymbol{\theta}_{X|M}(M_{k-1})$ as

$$\boldsymbol{\theta}_{X|M}(M_{k-1}) = \boldsymbol{\theta}_{X|M}(\boldsymbol{\tau}_X(T_{k-1} + \boldsymbol{\Theta}_{XY_{n-1}} \cdot \boldsymbol{\eta}_{Y_{n-1}|M}(M_{k-2}))).$$

If we compare this equation to the definition of an EFMLP in mapping 5.8, we see that $\boldsymbol{\theta}_{X|M}$ at M_{k-1} is equivalently a $(2n+1)$-layered EFMLP at M_{k-2} with random bias T_{k-1} in layer $n+1$. By repeatedly expanding M_{k-i} for all i, we may express backpropagation-through-time on $\boldsymbol{\theta}_{X|M}(M_{k-1})$ as simply backpropagation on the $(kn+k-1)$-layered EFMLP with random biases $(T_i)_{i=1}^{k-1}$, where $\partial_{\chi_i}T_i = \partial_{\chi_i}\boldsymbol{\theta}_X$. Since the parameters of each subnetwork in the $(kn+k-1)$-layered EFMLP are shared, we then sum the derivatives of χ_i computed at each subnetwork to compute the complete derivative of χ_i.

Although well-defined, backpropagation-through-time can be problematic in practice (Bengio et al., 1994; Pascanu et al., 2013). Moreover, because the posterior $P_{X_k|(Z_i)_{i=0}^k}$ captures all the information about the latent state available in the observations, if the conditional relative entropy in expression 5.14 is 0, then $(M_k)_{k\in\mathbb{N}}$ is a Markov chain. This suggests that as entropy 5.14 is minimized, the long-range dependencies in backpropagation-through-time can be ignored when modelling Bayes filters. When training model Bayes filters in chapter 7, I therefore consider a one-step approximation to the true derivative of $\boldsymbol{\theta}_{X|M}(M_{k-1})$ with respect to χ_i, and assume that (M_{k-1}) is independent of the parameters $\boldsymbol{\chi}$. As we will see, this works well in practice.

Let us now consider how to train a neural Bayes filter. Expression 5.14, equation 5.15, and the subsequent derivation of backpropagation-through-time continue to apply to neural Bayes filters. The additional consideration we must address when training neural Bayes filters is how we train the additional parameters of the belief encoding (see equations 5.6) which arise when computing the derivatives of W_{k-1}, while ensuring that equations 5.6 remain satisfied. In theory, we would need apply some form of projected gradient descent to ensure that equations 5.6 are satisfied, but this is highly nontrivial in the context of the non-convex optimization of minimizing the marginal relative entropy.

There are two advantages of neural Bayes filters over EFMLP Bayes filters. The first is in using the encodings $(V_k)_{k \in \mathbb{N}}$ and $(W_k)_{k \in \mathbb{N}}$ to model real neural populations, which involves matching the encoding and decoding parameters to real data, and not necessarily optimizing them with respect to entropy 5.14. The second is in defining the encoding and decoding transformations to confer additional useful properties on the respective populations, such as defining the distributions encoded by $(V_k)_{k \in \mathbb{N}}$ and $(W_k)_{k \in \mathbb{N}}$ to be independent of the mean firing rate of the populations, as done in Beck et al. (2011). As such, in practice, the parameters of the affine belief encoding of a neural Bayes filter should be trained with an independent objective or set by hand, which is the strategy I take in chapter 7.

Chapter 6

Bayesian Inference with Biological Neural Networks

In this chapter I have two principle goals. Firstly, I show that a large class of well-known models of the brain from the computational neuroscience literature are special cases of the theory I have developed in this dissertation. Secondly, I wish to highlight some of the properties of these models which demonstrate them to be especially good at implementing Bayesian inference, and thereby to show that the biological neural networks of the brain are the right kind of neural networks for solving the problems of embodied cognition.

I begin this chapter by reviewing linear-nonlinear (LN) models of single neuron activity. I then introduce the theory of neural coding known as probabilistic population codes (PPCs), which defines neural coding in terms of Bayesian inference. An important class of PPC is the family of linear probabilistic population codes (LPPCs), which are a kind of PPC that can tractably implement Bayes' rule.

I then develop the connection between LPPCs and harmoniums, as well as their connection with the theory of rectification. As I show, the structure of neural noise as modelled by LN models results in simple conditions for rectification, which demonstrates that biological neurons are especially well-suited to implementing Bayesian inference.

As we move into the field of theoretical neuroscience, it will be helpful to modify our vocabulary so that our discussions are consistent with the terminology used therein. In particular, I will tend to refer to latent variables X as stimuli, and observable variables Z as neurons or populations of neurons. In the latter case, Z represents the spike count or firing rate of the neuron(s) in question.

6.1 Linear-Nonlinear Neurons

A linear-nonlinear (LN) model is a simple model of how a neuron responds to a stimulus (Gerstner and Kistler, 2002; Simoncelli et al., 2004), yet one which can nevertheless provides a good approximation to the statistics of neural activity (Plesser and Gerstner, 2000; Paninski, 2004; Ostojic and Brunel, 2011). LN models are typically applied to modelling the discrete-time dynamic activity of a neuron. Although the brain exists in continuous time, typical mammalian neurons have a maximum firing rate between 500 and 1000 Hertz (Sterling and Laughlin, 2015), so that the brain can essentially be modelled as a discrete-time stochastic process where each time k covers an interval on the order of of $h = 1000^{-1}$ seconds. Moreover, as we will see, at this timescale LN neurons are approximately equal to a continuous-time, inhomogeneous Poisson process (see Jost, 2014, chapter 3).

Suppose that $(X_k, Z_k)_{k\in\mathbb{N}}$ is a hidden Markov chain, and let us refer to $(X_k)_{k\in\mathbb{N}}$ as the random dynamic stimulus, and $(Z_k)_{k\in\mathbb{N}}$ as the LN neuron. An LN neuron is a binary, dynamic random variable defined by the emission distribution

$$P(Z_k = 1 \mid \mathbf{x}_k) = hf(\mathbf{a}\cdot\mathbf{x}_k),$$

where $\mathbf{a}$ is a so-called linear filter, h is the time-step, and $f\colon \Omega_X \to \mathbb{R}^+$ is known as the transfer function. The value $f(\mathbf{a}\cdot\mathbf{x}_k)$ at time k is the firing rate of $(Z_k)_{k\in\mathbb{N}}$ at time k, and the LN neuron is "spiking" when $Z_k = 1$, and the neuron is "quiescent" when $Z_k = 0$. The number of spikes in the interval of $s = nh$ seconds is $Z^n = \sum_{i=0}^{n-1} Z_i$. When the LN neuron has a constant firing rate c over this interval, then Z^n is a binomial random variable defined by a spiking probability hc over n trials.

The Binomial distribution converges to a Poisson distribution as the probability of spikes per time-step decreases and the number of trials increases, and the ratio between them is held constant. Formally, if N is a Poisson random variable with rate λ, B is a Binomial random variable defined by n trials and a spiking probability $p = \lambda/n$, then $\lim_{n\to\infty} P_B = P_N$ (Jost, 2014), where

$$P(N = k) = \frac{e^{-\lambda}\lambda^k}{k!}.$$

I depict the relationship between Bernoulli, binomial, and Poisson distribution in figure 6.1.

Consider an LN neuron $(Z_k)_{k\in\mathbb{N}}$ with a firing rate $f(\mathbf{a}\cdot X_k)$ at time k. Neurons usually fire well below their maximum rate; in one study, the average firing rate of neurons in the hippocampal and entorhinal cortex was found to be less than 1Hz, with a maximum of about 10Hz (Mizuseki and Buzsáki, 2013). If we assume that the firing rate of $(Z_k)_{k\in\mathbb{N}}$ at time k is at most 10Hz, and that $h = 0.001$, then the probability that a neuron fires at time k is 0.01. On the other hand, the chance that a Poisson random variable with rate $\lambda = 0.01$ is equal to 0 or 1 is more than 99.995%. Therefore, if we approximate a hippocampal/entorhinal LN neuron with an inhomogeneous Poisson process with instantaneous firing rate $f(\mathbf{a}\cdot X_t)$, where $(X_t)_{t=1}^{\mathbb{R}^+}$ is a continuous-time dynamic stimulus, then we would only expect the continuous-time model to deviate from the LN neuron by only about one spike every 20 seconds.

The approximate equivalence between these discrete- and continuous-time models is both intuitively and technically advantageous. On one hand, the brain is indeed a continuous-time system, and it is convenient to describe its continuous-time state as the collection of instantaneous firing rates of all its constituent neurons. On the other hand, real-world stimuli are also continuous-time phenomena, and a continuous-time model of the coupled stimulus-brain system allows us to avoid unnecessarily discretizing the dynamics of the stimulus. Moreover, being able to treat neurons as either Bernoulli or Poisson random variables means that we may apply and combine the theory of both types of random variable when analyzing LN neurons.

6.2 Linear Probabilistic Population Codes

In this section we move beyond modelling the response statistics of single neurons, to modelling Bayesian inference in populations of neurons. There are many established

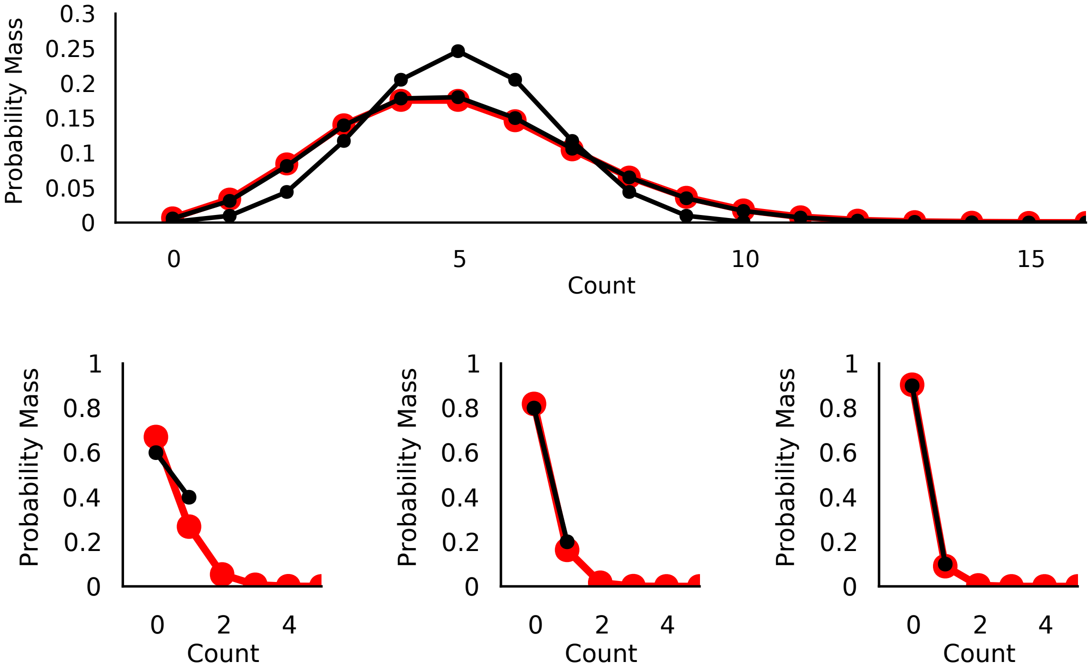

Figure 6.1: A depiction of the convergence of Bernoulli and Binomial distributions (black) to Poisson distributions (red). *Top*: A Poisson distribution with rate $\lambda = 5$, and two Bernoulli distributions with $n = 10$ and $n = 100$ trials, and probabilities of success $p = 0.5$ and 0.05. *Bottom*: Three plots comparing three pairs of Bernoulli and Poisson distributions, with parameters $\lambda = p = 0.4$, $\lambda = p = 0.2$, and $\lambda = p = 0.1$, respectively. As depicted in the third plot, the probability that a Poisson random variable is greater than 1 when $\lambda = 0.1$ is less than 0.005.

frameworks for understanding how populations of neurons store and process information about stimuli (see Pouget et al., 2013), but it is the theory of probabilistic population codes which I consider in this dissertation. An example of a probabilistic population code is given by collections of independent LN neurons paired with their stimuli. However, the definition of an LPPC is more general than what is described by a product of LN neurons alone.

Let us consider a random stimulus X and a population of d_Z neurons $Z = (Z_i)_{i=1}^{d_Z}$. A code is a definition of how to encode and decode information about one thing into and from some alternative representation. A probabilistic population code (PPC) is a stochastic code defined as a pair of random variables (X, Z), where encoding information about the stimulus $\mathbf{x}$ in Z is defined as sampling the likelihood $P_{Z|X=\mathbf{x}}$, and decoding the information in the sampled vector $\mathbf{z} \in \Omega_Z$ is defined as evaluating the posterior $P_{X|Z=\mathbf{z}}$ (Zemel et al., 1998).

A linear probabilistic population code (LPPC) is a PPC for which the likelihood $P_{Z|X} \in \mathcal{M}_Z$, where $\mathcal{M}_Z$ is an exponential family with a sufficient statistic equal to the identity function. LPPCs – in particular those where $\mathcal{M}_Z$ is the product family of independent Poisson distributions – have been shown to effectively model the statistics of neural responses (Jazayeri and Movshon, 2006; Graf et al., 2011). The mean of the ith LPPC neuron conditioned on $\mathbf{x}$ is defined as $\gamma f_i(\mathbf{x})$, where $f_i \colon \Omega_X \to \mathbb{R}^+$ is the so-called tuning curve of the neuron, and γ is the gain. Because $\mathbf{s}_Z$ is the identity function, the mean parameters of the likelihood $P_{Z|X=\mathbf{x}}$ are simply $\mathbb{E}[\mathbf{s}_Z(Z) \mid X = \mathbf{x}] = \mathbb{E}[Z \mid X = \mathbf{x}] = \gamma \mathbf{f}(\mathbf{x})$. We may thus write the likelihood of an

LPPC in exponential family form as

$$p_{Z|X}(\mathbf{z} \mid \mathbf{x}) = e^{\boldsymbol{\tau}_Z^*(\gamma \mathbf{f}(\mathbf{x}))\cdot\mathbf{z} - \psi_Z(\boldsymbol{\tau}_Z^*(\gamma \mathbf{f}(\mathbf{x})))}. \tag{6.1}$$

Based on this form, an LPPC is called a *linear* PPC because the log-likelihood is linear in the population response $\mathbf{z}$[12].

Let us consider a few examples of LPPCs, as defined by their tuning curves. When $\mathcal{M}_Z$ is the product family of independent Poisson distributions, the two most widely applied tuning curves in the context of LPPCs are the Gaussian and von Mises tuning curves. The 1-dimensional Gaussian tuning curve is

$$\begin{aligned} f_i \colon \mathbb{R} &\to \mathbb{R}^+ \\ x &\mapsto e^{\frac{-(x-x_i^0)^2}{2\sigma_i^2}}, \end{aligned} \tag{6.2}$$

with preferred stimuli x_i^0, and tuning widths σ_i^2. The von Mises tuning curve is

$$\begin{aligned} f_i \colon [-\pi, \pi] &\to \mathbb{R}^+ \\ x &\mapsto e^{\kappa_i \cos(x - x_0^i)}, \end{aligned} \tag{6.3}$$

with preferred orientations x_i^0, and concentrations κ_i.

On the other hand, if $\mathcal{M}_Z$ is the product family of Bernoulli distributions, then a standard tuning curve for d_X-dimensional stimuli is the logistic function combined with a linear transformation

$$\begin{aligned} f_i \colon \Omega_X &\to [0, 1] \\ \mathbf{x} &\mapsto (1 + e^{\mathbf{a}_i \cdot \mathbf{x}})^{-1} \end{aligned}$$

where $\mathbf{a}_i$ is the linear filter of the ith neuron. This defines each neuron in the LPPC as an LN neuron at some time k with a logistic transfer function.

Another assumption which is often made in the context of LPPCs with Poisson neurons is that the sum of the tuning curves is independent of the stimulus (Ma et al., 2006; Beck et al., 2011), such that

$$\sum_{i=1}^{d_Z} f_i(\mathbf{x}) = \lambda, \tag{6.4}$$

for some constant λ. Because the gain is proportional to the sum of the tuning curves, this constraint allows the gain to model so-called nuisance parameters such as contrast, which influence the response of the neurons but do not contain information about the stimuli. Another important consequence of LPPCs which satisfy equation 6.4 is that they can trivially implement Bayesian inference with conjugate priors. In the next section I show how this result is a special case of rectification, but for now let us develop some intuition about this phenomenon through simulation.

---

[12] In Ma et al. (2006); Beck et al. (2008, 2011) LPPCs are defined in a slightly more general way, as they assume that the stimulus dependent parameters of $P_{Z|X}$ can be modulated by the covariance matrix of Z given X. However, because an LPPC has an identity sufficient statistic, the covariance of Z given X can only vary as a function of the base measure μ_Z. This way of adding covariance to the neural population is difficult to generalize beyond the case of one-dimensional stimuli, and so I do not include it in my presentation.

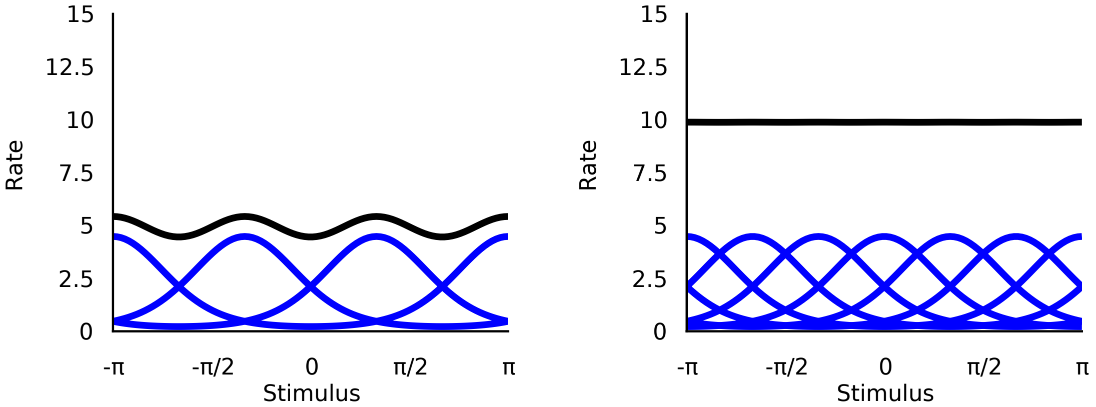

Figure 6.2: The sum (black line) of the tuning curves (blue lines) of two von Mises LPPCs. In both cases $\kappa_i = 1.5$ and $\gamma = 1$. *Left*: A von Mises LPPC with 3 neurons. The sum of the tuning curves varies noticeably over the stimulus space. *Right*: A von Mises LPPC with 6 neurons. Doubling the number of neurons results in a sum of tuning curves which is nearly indistinguishable from a straight line.

I depict the sum of the tuning curves of two von Mises LPPCs in figure 6.2. Notice how the sum of the tuning curves quickly converges to a straight line as the number of neurons in the population is increased, such that equation 6.4 can be approximately satisfied with ease in the one-dimensional case. These plots provide an important intuition about Bayesian inference in LPPCs. If we consider the left plot in figure 6.4, a null response $\mathbf{z} = \mathbf{0}$ suggests that the stimulus is in between the preferred stimuli of the model. However, when the sum of the tuning curves is constant, a null response provides no information about the stimulus, and the posterior over the stimulus given the null response is equal to the prior. The result of this is a high degree of regularity in the beliefs computed by the LPPC, and ultimately the existence of conjugate priors.

I depict an example of Bayesian inference with a von Mises LPPC in figure 6.3. This figure illustrates Bayesian inference in LPPCs as originally formulated in Ma et al. (2006) where it was presented as a form of “cue integration”. That is, where both Z_1 and Z_2 are defined as responses to a stimulus X with a likelihood given by equation 6.1, the authors showed that the response Z_3 defined as a certain linear combination of Z_1 and Z_2 retains all the information in Z_1 and Z_2 about X under the assumption of a flat prior. This linear circuit for implementing Bayesian inference over encoded beliefs is ultimately a special case of the belief encodings I developed in subsection 5.1.2.

Bayesian inference in LPPCs was also extended and applied to implementing Bayes filters in neural circuits (Beck and Pouget, 2007; Beck et al., 2011); these models are ultimately special cases of the neural Bayes filters introduced in subsection 5.2.3 with emission distributions defined by LPPC likelihoods. In the case of linear dynamical systems and Gaussian tuning curves addressed in Beck et al. (2011), the optimal 2-layer EFMLP has the form

$$\frac{\boldsymbol{\theta}_{V|W}(\mathbf{w})}{h} = \boldsymbol{\Theta}^{(2)} \cdot \mathbf{w} + \mathbf{w} \cdot \boldsymbol{\Theta}^{(3)} \cdot \mathbf{w} + 1(w^0 - \frac{1 \cdot \mathbf{w}}{d_W}),$$

under the assumption that the emission distribution satisfies equation 6.4, and where h is the time-step in the time-discretized system. Intuitively, $\boldsymbol{\Theta}^{(2)}$ drives the rate of

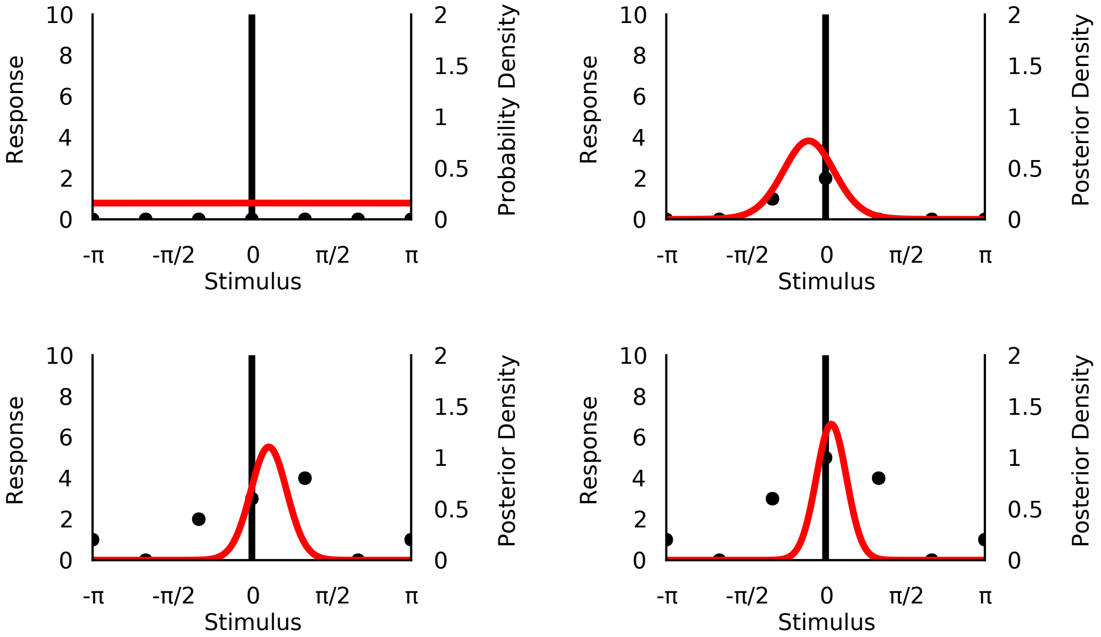

Figure 6.3: A realization of Bayesian inference with the second von Mises LPPC from figure 6.2. In each plot I show the true stimulus (black line), the population response (black dots), where each black dot shows the number of spikes produced by the neuron with the given preferred stimulus, and the belief density (red line). *Top Left*: In the absence of spikes, the prior is flat. *Top Right*: The posterior given the flat prior and a response $\mathbf{z}_1$ of the LPPC with gain $\gamma = 0.5$. *Bottom Left*: The posterior given the flat prior and a response $\mathbf{z}_2$ of the LPPC with gain $\gamma = 1$. *Bottom Right*: The posterior given the flat prior with the response $\mathbf{z}_1 + \mathbf{z}_2$, which is a more accurate posterior than in the other two cases.

the population in proportion to the linear dynamics, $\mathbf{\Theta}^{(3)}$ quadratically drives the rate of the population in proportion to the noise in the dynamics, and w^0 is a parameter which encourages the component-wise average of the rates of the computed by the network to remain near w^0.

Moreover, Beck et al. (2011) show how to take the limit of this equation as $h \to 0$ by approximating the Poisson-distributed responses with Bernoulli distributions as described in the previous section; they also show how to resample the encoding random variable W with minimal loss of information, such that W may be interpret as a population of spiking neurons. The result is a continuous-time, spiking, recurrent neural network which implements a Bayes filter to an arbitrary degree of precision. In future work I hope to combine this technique with the methods that I present in my dissertation, towards developing a continuous-time, spiking, recurrent neural network which can learn to implement a Bayes filter without knowledge of either the transition or emission distributions.

6.3 LPPCs and Harmoniums

Both harmoniums and LPPCs are pairs of random variables (X, Z), where the likelihood $P_{X|Z}$ is an element of some exponential family $\mathcal{M}_Z$. Moreover, when implementing Bayesian inference based on equation 6.4, the posterior $P_{X|Z}$ of an LPPC is typically an element of some exponential family $\mathcal{M}_X$. As such, although it may seem as though equation 6.1 is more general than the affine form of a harmonium likelihood (3.6), theorem 3.1 implies that in practice, an LPPC is typically a form of harmonium.

Proposition 6.1. *Suppose that there exists an LPPC* (X, Z) *with likelihood* $P_{Z|X} \in \mathcal{M}_Z$ *defined by the tuning curves* $\mathbf{f}$ *and gain* γ *which satisfies* $P_{X|Z} \in \mathcal{M}_X$*. Then*

$$\boldsymbol{\tau}^*(\gamma \mathbf{f}(\mathbf{x})) = \boldsymbol{\theta}_Z + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_X(\mathbf{x}), \tag{6.5}$$

for some parameters $\boldsymbol{\theta}_Z$ *and* $\boldsymbol{\Theta}_{XZ}$*.*

Proof. If P_{XZ} has an LPPC likelihood $P_{X|Z}$ and a posterior $P_{Z|X}$, then this follows from theorem 3.1 and by combining equations 3.6 and 6.1 for the likelihood of (X, Z). □

Note that this proposition does not depend on the form of $P_{X|Z}$, but rather only its existence for some (X, Z), and thus applies in general to tuning curves $\mathbf{f}$ and gain γ which satisfy the conditions.

For example, let us suppose that $\mathcal{M}_Z$ is the product family of independent Poisson distributions, such that $\tau^*_{Z,i}(\boldsymbol{\theta}_Z) = \log(\theta_{Z,i})$ (appendix A), and let us again consider the Gaussian and von Mises tuning curves. In the case of Gaussian tuning curves, we can satisfy equation 6.5 by setting the elements of the interaction matrix $\boldsymbol{\Theta}_{XZ}$ equal to

$$\theta_{XZ,1,i} = \frac{x_i^0}{\sigma_i^2}, \quad \theta_{XZ,2,i} = -\frac{1}{2\sigma_i^2}, \tag{6.6}$$

and setting the elements of the bias $\boldsymbol{\theta}_Z$ equal to

$$\theta_{Z,i} = \log \gamma - \frac{(x_i^0)^2}{2\sigma_i^2}, \tag{6.7}$$

and by letting $\mathbf{s}(x) = (x, x^2)$, which is the sufficient statistic of the family of normal distributions. On the other hand, we can satisfy equation 6.5 for von Mises tuning curves by setting the elements of the interaction matrix $\boldsymbol{\Theta}_{XZ}$ equal to

$$\theta_{XZ,1,i} = \kappa \cos(x_i^0), \quad \theta_{XZ,2,i} = \kappa \sin(x_i^0),$$

and setting the elements of the bias $\boldsymbol{\theta}_Z$ equal to

$$\theta_{Z,i} = \log \gamma,$$

and by letting $\mathbf{s}(x) = (\cos(x), \sin(x))$, which is the sufficient statistic of the family of von Mises distributions.

By definition, LPPCs with exponential family posteriors are always a form of harmonium, and in the previous subsection we found that LPPCs with Poisson neurons afford conjugate priors if the sum of the tuning curves is independent of the stimulus. By definition, a rectified harmonium has conjugate priors, and therefore equation 6.4 must somehow be related to the rectification equation (4.1). In this next proposition I show that we may in fact generalize the approach to conjugate priors provided by equation 6.4 with the theory of rectification.

Proposition 6.2. *Let $\mathcal{H}_{XZ}$ be a harmonium family defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$, where $\mathcal{M}_Z$ is the product family of independent Poisson distributions. Let (X, Z) be a harmonium with distribution $P_{XZ} \in \mathcal{H}_{XZ}$. If*

$$\sum_{i=1}^{d_Z} \mathbb{E}_P[Z_i \mid X = \mathbf{x}] = \boldsymbol{\rho}_X \cdot \mathbf{s}_X(\mathbf{x}) + \rho_0, \tag{6.8}$$

for some parameters $\boldsymbol{\rho}_X$ and ρ_0, then (X, Z) is rectified with rectification parameters $\boldsymbol{\rho}_X$ and ρ_0.

Proof. Note that the log-partition function of a single Poisson exponential family $\mathcal{M}_{Z_i}$ is given by $\psi_{Z_i}(\theta_{Z_i}) = e^{\theta_{Z_i}} = \tau_{Z_i}(\theta_{Z_i})$. Therefore, by equation 3.7, $P_{Z|X=\mathbf{x}} \in \mathcal{M}_Z$ for any $\mathbf{x} \in \Omega_X$ satisfies

$$\begin{aligned} \psi_Z(\boldsymbol{\theta}_Z + \boldsymbol{\Theta}_{XZ} \cdot \mathbf{s}_X(\mathbf{x})) &= \sum_{i=1}^{d_Z} \tau_{Z,i}(\boldsymbol{\theta}_Z + \boldsymbol{\Theta}_{XZ} \cdot \boldsymbol{\theta}_Z) \\ &= \sum_{i=1}^{d_Z} \mathbb{E}_P[Z_i \mid X = \mathbf{x}] \\ &= \boldsymbol{\rho}_X \cdot \mathbf{s}_X(\mathbf{x}) + \rho_0, \end{aligned}$$

which implies by theorem 4.1 that (X, Z) is rectified with rectification parameters $\boldsymbol{\rho}_X$ and ρ_0. □

It is also worth noting that at low rates, this proposition can be approximately applied to LN neurons, as discussed in subsection 6.1.

6.4 Homogeneous LPPCs

Let us analyze the general conditions under which LPPCs can satisfy the rectification equation. Consider the Gaussian (6.2) and von Mises (6.3) tuning curves. In both cases, if we assume that the set of tuning curves $\{f_i\}_{i=1}^{d_Z}$ have preferred stimuli x_i^0 and shared shape parameters σ^2 or κ, then we may express the ith tuning curve as $f_i(x) = f(x - x_i^0)$, where f is a tuning curve with preferred stimulus 0 and shape parameter σ^2 or κ. Let us refer to LPPCs with this structure as homogeneous.

Definition 6.1 (Homogeneous LPPC)**.** The LPPC (X, Z) defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$ is a homogeneous LPPC with tuning function $f \colon \Omega_X \to \mathbb{R}^+$ and gain function $\gamma \colon \Omega_X \to \mathbb{R}^+$ if:

1. $\mathcal{M}_Z$ is the product family of independent Poisson distributions.
2. For every i, $\mathbb{E}_P[Z_i \mid X = \mathbf{x}] = \gamma(\mathbf{x}_i^0) f(\mathbf{x} - \mathbf{x}_i^0)$.

It turns out that homogeneous LPPCs which cover the latent space Ω_X with a sufficient number of tuning curves can always be rectified by modulating the gain function. Proving this requires rudimentary harmonic analysis (see Deitmar and Echterhoff (2014) for a thorough treatment of the topic).

Generally speaking, suppose that Ω is a locally compact, abelian group with group action $+$ and inverse $-$. Then there exists a Haar measure ν on the Borel σ-algebra of Ω, with which we define the convolution of g and h as

$$(g * h)(x) = \int_{\Omega} g(x)h(x-y)\nu(dy).$$

In this case there also exists an invertible function F known as the Fourier transform for which

$$F(g * h) = F(g)F(h). \tag{6.9}$$

Proposition 6.3. *Let (X, Z) be a homogeneous LPPC defined by $\mathcal{M}_X$ and $\mathcal{M}_Z$, with a tuning function f and gain function γ. Moreover, suppose that Ω_X is a locally compact abelian group with action $+$ and inverse $-$, and that*

$$\sum_{i=1}^{d_Z} \gamma(\mathbf{x}_i^0) f(\mathbf{x} - \mathbf{x}_i^0) = \int_{\Omega_X} \gamma(\mathbf{x}^0) f(\mathbf{x} - \mathbf{x}^0)\nu(\mathbf{x}^0) = (\gamma * f)(\mathbf{x}). \tag{6.10}$$

Then there exists a gain function γ given by

$$\gamma = F^{-1}\Big(\frac{F(\mathbf{s}_X \cdot \boldsymbol{\rho}_X + \rho_0)}{F(f)}\Big),$$

such that (X, Z) is rectified with rectification parameters $\boldsymbol{\rho}_X$ and ρ_0.

Proof. By combining equation 6.10 with proposition 6.2 and the definition of a homogeneous tuning curve, we find that

$$(\gamma * f)(\mathbf{x}) = \boldsymbol{\rho}_X \cdot \mathbf{s}_X(\mathbf{x}) + \rho_0.$$

Let us define the function $g(x) = \boldsymbol{\rho}_X \cdot \mathbf{s}_X(x) + \rho_0$. Then equation 6.9 implies that

$$\begin{aligned} &\gamma * f = g \\ \Longrightarrow\ &F(\gamma)F(f) = F(g) \\ \Longrightarrow\ &F(\gamma) = \frac{F(g)}{F(f)} \\ \Longrightarrow\ &\gamma = F^{-1}\Big(\frac{F(g)}{F(f)}\Big). \end{aligned}$$

□

In order to apply this proposition we must verify whether equation 6.10 is satisfied. In the case where Ω_X is a discrete space, this is relatively straightforward, as the Haar measure ν is simply the counting measure. In this case equation 6.10 is satisfied if there exists a tuning curve with a preferred stimulus at every point in the space. Of course, this may require a prohibitively large number of neurons, and this strategy will not work at all when Ω_X is uncountable. Nevertheless, as we will see and as already suggested in figure 6.2, a sum of a small number of neural tuning curves quickly approximates the conditions for rectification.

In order to analyze rectification in a LPPC where $\Omega_X \subset \mathbb{R}$, let us us assume that $(X, Z_k)_{k \in \mathbb{N}}$ is a sequence of LPPCs, where (X, Z_k) has $d_{Z_k} = k$ neurons and preferred

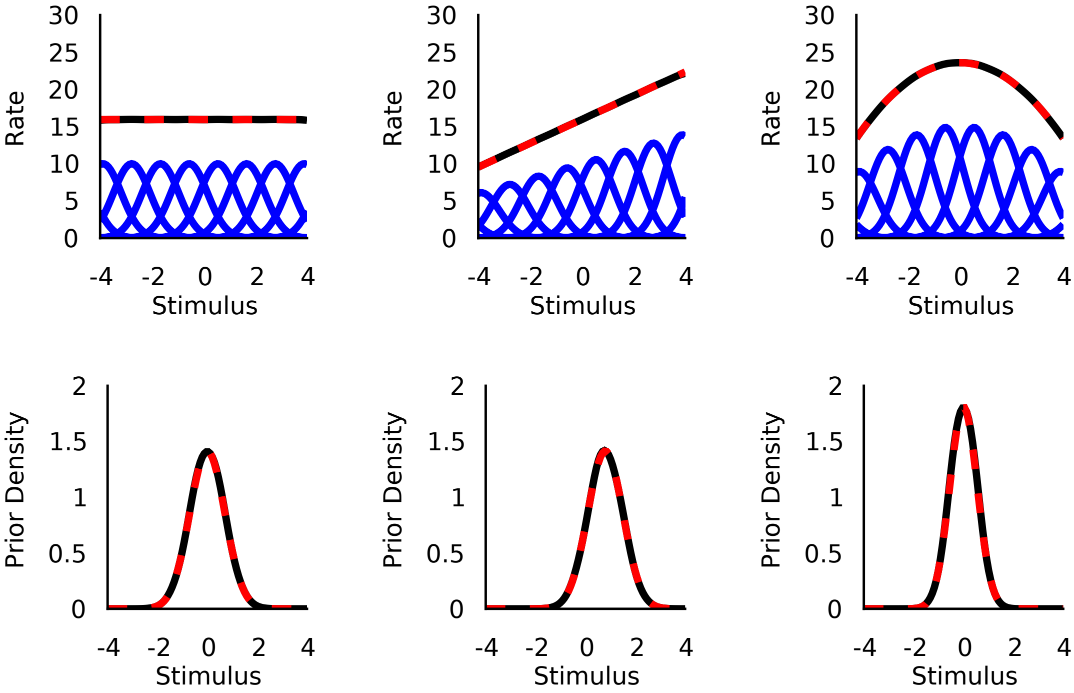

Figure 6.4: In this figure I present three pairs of plots which visualize rectification in homogeneous LPPCs with Gaussian tuning curves, where $\sigma^2 = 0.5$. In the top plot of each example I present the tuning curves of the LPPC (blue lines), the sum of the tuning curves as a function of the stimulus (black line), and the right hand side of equation 6.8 as a function of the stimulus (dashed red line), based on rectification parameters $\boldsymbol{\rho}_X$ and ρ_0 which approximately solve equation 6.8. In the bottom plot I depict the true prior of the model computed with numerical integration (black line), and the approximate prior with parameters $\boldsymbol{\theta}^*_X = \boldsymbol{\theta}_X + \boldsymbol{\rho}_X$ (dashed red line). In all cases, the $\boldsymbol{\theta}_X$ are the natural parameters of a normal distribution with mean 0 and variance 1. *Left*: A rectified LPPC with a constant sum of tuning curves such that $\boldsymbol{\rho}_X = 0$. The parameters of the prior are simply $\boldsymbol{\theta}_X$. *Middle*: A rectified LPPC where $\rho_{X,1} = 1.6$, and $\rho_{X,2} = 0$. The mean of the prior with parameters $\boldsymbol{\theta}^*_X$ is shifted to the right. *Right*: A rectified LPPC where $\rho_{X,1} = 0$, and $\rho_{X,2} = -0.635$. The mean of the prior is unchanged, but the precision is higher.

stimuli $(x^0_{k,i})^{d_Z}_{i=1}$. Let us also assume that (X, Z_k) is homogeneous with tuning function f and gain function γ scaled by $\frac{a-b}{k}$, and with preferred stimuli distributed uniformly over the interval $[a, b]$. If γ and f are integrable, then in the limit of infinite neurons, the left hand side of equation 6.8 is given by the integral

$$\lim_{k\to\infty} \sum_{i=1}^{k} \gamma(x^0_{k,i}) f(x - x^0_{k,i}) \frac{a-b}{k} = \int_{[a,b]} \gamma(x^0) f(x - x^0) \lambda(dx^0),$$

where λ is the Lebesgue measure. Since the Lebesgue measure is the Haar measure on $\mathbb{R}$, this construction nearly results in a convolution. If γ and f are periodic on $[a, b]$, such as is the case when $\mathcal{M}_X$ is the von Mises family, and $a = -\pi$ and $b = \pi$, then this equation is indeed a convolution. When $\Omega_X = \mathbb{R}$, then we may take the limit of this equation as $a \to -\infty$ and $b \to \infty$.

For example, in the case of an infinite-neuron LPPC with Gaussian tuning curves and tuning width σ^2, the solution for γ is

$$\gamma(x) = \frac{\rho_{X,1} x + \rho_{X,2}(x^2 - \sigma^2) + \rho_0}{\sqrt{2\pi\sigma^2}}.$$

In figure 6.4 I present three examples of approximately rectified, homogeneous LPPCs with Gaussian tuning curves based on the gain function defined in the above equation. As stated in corollary 4.2, if the likelihood of the LPPC (X, Z) satisfies equation 6.8, then the prior P_X of (X, Z) has parameters $\boldsymbol{\theta}_X^* = \boldsymbol{\theta}_X + \boldsymbol{\rho}_X$. In each example in figure 6.4, the parameters $\boldsymbol{\theta}_X$ are the same, and the prior density p_X depicted in the lower plots is changed by modulating the gain γ in equation 6.1 as a function of the preferred stimuli of the neurons.

There is another advantage of gain-modulated homogeneous LPPCs which is worth highlighting. Note that in the equation for the interactions 6.6 and observable biases 6.7 of a harmonium based on 1-dimensional Gaussian tuning curves, the gain only appears in the observable biases $\boldsymbol{\theta}_Z$. Since the posterior of a harmonium $P_{X|Z}$ is independent of the observable biases (3.5), this means that we may freely alter the gain of the neurons, without changing the beliefs encoded by a particular population response. This suggests that homogeneous LPPCs are particularly flexible in their ability to satisfy the rectification equation (4.1).

Chapter 7

Simulations

In this dissertation I have presented numerous plots which demonstrate theory as I have developed it. However, I have for the most part not presented plots which validate the gradient descent algorithms that I have developed, as these algorithms take hours to execute, and require more careful analysis in order to evaluate their progress and success.

In this chapter I present the most sophisticated simulations that I have produced based on the theory developed in this dissertation. I begin by introducing the libraries that I have developed for the Haskell programming language, with which I have implement the complex numerical simulations that I present. I then continue by presenting my simulations of training and rectifying a restricted Boltzmann machine on the MNIST dataset, followed by my simulations of training neural Bayes filters, with which I model particular neural circuits from the mammalian brain.

7.1 Geometric Optimization Libraries

Although I have not prominently featured my programming work in this dissertation, programming is how I have spent the bulk of my majority during my doctoral work. The fruits of this labour are a collection of libraries for the Haskell programming language, which I call the Geometric Optimization Libraries, or simply Goal for short. In my experience with implementing numerical algorithms, the majority of my time as a programmer is spent debugging. This is, again in my opinion, the least enjoyable part of programming, and I have developed Goal in order minimize this tedium.

At the lowest level, programs are engaged with manipulating indistinguishable numbers in memory. In the language of computer science, a type system is a way of assigning extra information to these numbers in order to ensure that they are not manipulated and combined inappropriately. In this vein, Goal provides a type system for numerical optimization. In particular, Goal provides a type system which treats numbers as points on a manifold, and provides functions for finding particular points on these manifolds.

I begin this section with a short introduction to type systems and Haskell. I then introduce Goal, and its core types such as manifolds, points, and tangent spaces, and the application of these types to describing exponential family manifolds. For the sake of simplicity, the definitions I provide are not always the same as they are in the Goal libraries, and this introduction is not a complete survey of all the code I have implemented. The purpose of this introduction is rather to provide a clear and concise overview of the novelty and practical value of Goal.

7.1.1 Type Systems and Haskell

Code is typically compiled before it is executed, and so there is a difference between errors which arise while compiling code, and errors which arise while executing the program which it describes. On one hand, the most straightforward kind of compile-time error is a parsing error, where the compiler simply cannot understand what the programmer has written. On the other hand, a program can be compiled, but then still result in errors when executed. For example, a compiler can compile a program which adds two numbers from specified locations in memory, but this programming will crash if these locations do not exist.

In general, it is better to recognize errors at compile-time rather then run-time. Compile-time errors are revealed more or less instantly, and do not depend on the inputs to the program. Conversely, programs may need to be run for extended periods of time before run-time errors emerge, and they may only emerge under very specific conditions. However, compile-time errors can only be found through some kind of analysis on the part of the compiler, and so defining and catching compile-time errors requires a system for performing this analysis.

If we think of code as essentially a collection of expressions, then a type system is a system for assigning types to these expressions. The primary purpose of a type system is to help identity and reduce errors. Although type systems can be applied to manage run-time errors, type systems are fundamental to defining and identifying nontrivial compile-time errors. When every expression in the code of a given language must have an unambiguous type before it can be compiled, then the language is known as strongly-typed.

An important element of a strongly-typed language is that the expressions can be analyzed in isolation – if one expression depends implicitly on another expression, then it may be difficult to assign that expression a coherent type. Programming languages which forbid implicit interactions between expressions (also known as side-effects) are known as functional programming languages. Arguably the most well-known, strongly-typed, functional programming language is Haskell, and all the simulations I have run and plots I have generated in this dissertation were written in the Haskell programming language.

In order to illustrate the Haskell type system, let us write a program for managing the types of animals that live in a house. The fundamental building blocks of the Haskell type system are algebraic data types (ADTs). We can define an ADT to describe the world of possible pets that live in a house:

```haskell
data Pet = Dog | Hamster | Goldfish
```

In the language of Haskell, *Pet* is a type constructor, and *Dog*, *Hamster*, and *Goldfish* are value constructors, all of which in this case have zero arguments.

Let us create two pets:

```haskell
fluffy :: Pet
fluffy = Dog
```

```
goldy :: Pet
goldy = Goldfish
```

Fluffy is a *Dog*, and Goldy is a *Goldfish*, and both are of the *Pet* type. Let us suppose that Fluffly and Goldy belong to Susie. We can define a list of Susie's pets with Haskell lists:

```
susiesPets :: [Pet]
susiesPets = [fluffy, goldy]
```

The type of *susiesPets* is a list of pets. The square brackets in the type declaration of *susiesPets* is the type constructor of a list, and it takes a single argument. As such, a list is a polymorphic type, because we could make a list of any given type. For example, the *length* function counts the number of elements in a list, and it has the following type:

```
length :: [a] -> Int
```

This is to say that the length function can calculate the length of any list, regardless of the type of element it contains.

Let us create another kind of animal:

```
data Pest = Mouse | Cockroach | Spider
```

If we wish to describe the list of pests in Susie's house, we can again use the list structure:

```
susiesPests :: [Pest]
susiesPests = [Mouse, Mouse, Mouse, Spider]
```

We do not care so much about pests, and so we do not give them names. Unfortunately, Susie has three mice and a spider in her house. Thankfully, she has no cockroaches. We can calculate the number of mice in someones house with a few functions. First, let us define a function which checks whether or not a pest is a mouse:

```
isMouse :: Pest -> Bool
isMouse Mouse = True
isMouse _ = False
```

This function is *True* if the input is a *Mouse*, and *False* otherwise, as indicated by the underscore. We will then use the *filter* function, which has the following type:

```
filter :: (a -> Bool) -> [a] -> [a]
```

The *filter* function takes as input a function which either accepts or rejects a value of some type, a list of that type, and returns a list of accepted values. We can then count mice with the following function:

```
countMice :: [Pest] -> Int
countMice ps = length (filter isMouse ps)
```

That is, we count the number of elements of the given list which are mice.

It is important to note that the list type in Haskell is homogeneous. Trying to create a list out of a dog and a mouse would result in a compile-time error, as dogs and mice do not have the same type. Nevertheless, we sometimes want to group together different types. The algebraic structure of Haskell types allows us to do this easily:

```
data HouseAnimal = HousePet Pet | HousePest Pest
```

A house animal is either a house pet or a house pest, and we may construct a *HouseAnimal* with either of the single-argument value constructors *HousePet* or *HousePest*. We may then construct a list of house animals as follows:

```
someAnimals :: [HouseAnimal]
someAnimals = [HousePet Dog, HousePest Rat, HousePest Mouse]
```

If we want to create a single function which converts from the base type to the house type, we cannot do this directly, because the function must have a single type. We can get around this by using Haskell typeclasses. For example, we may define a *ToHouseAnimal* typeclass:

```
class ToHouseAnimal a where
    toHouseAnimal :: a -> HouseAnimal
```

We may then instantiate this class for Pets and Pests as follows:

```
instance ToHouseAnimal Pet where
    toHouseAnimal p = HousePet p

instance ToHouseAnimal Pest where
    toHouseAnimal p = HousePest p
```

The *map* function applies a function to every element of a list, and the ++ operator concatenates two lists. With these various definitions, we may list all of the animals that live in Susie's house:

```
susiesAnimals :: [HouseAnimal]
susiesAnimals = map toHouseAnimal susiesPets
    ++ map toHouseAnimal susiesPests
```

Although sound, this use of typeclasses is not very practical. We could model the animals that live in a house by better organizing the ADTs which define them, while avoiding the more complicated syntax of typeclasses. In the following subsections, as I introduce my Goal library, we will see more practical applications of these various parts of the Haskell type system.

7.1.2 Manifolds

Goal is a Haskell library for numerical optimization. The core datatypes, typeclasses, and functions of Goal are modelled after concepts from differential geometry. As I hope to demonstrate, drawing on concepts from differential geometry provides an interface for numerical optimization algorithms which is effective, intuitive, and concise. In particular, Goal is effective because it provides good definitions for the compile-time evaluation of numerical programs; Goal is intuitive because the various definitions which it employs ensure that the purpose of complex numerical algorithms remain transparent; and Goal is concise because the rich structure associated with every numerical object allows us to implement complex algorithms with a few high-level functions.

To begin, the most fundamental typeclass in Goal is *Manifold*:

```
class Manifold m where
    dimension :: m -> Int
```

Any type which is a *Manifold* must have a function which defines its associated dimension. Let us define a few manifolds. The simplest manifold is Euclidean space:

```
data Euclidean = Euclidean Int

instance Manifold Euclidean where
    dimension (Euclidean n) = n
```

The value constructor *Euclidean* is defined simply by an integer, which specifies its dimension. If we take the tensor product of two manifolds, we can define a third manifold with a dimension equal to the product of the dimensions of the input manifolds:

```
data Tensor m n = Tensor m n

instance (Manifold m, Manifold n) => Manifold (Tensor n m) where
    dimension (Tensor n m) = dimension m * dimension n
```

Optimization is concerned with finding the best points on a manifold. As such, we must define a type for a point on a manifold. Now a manifold itself is simply a set, and can in principle be made of just about anything. As such, it is not easy to define a general type for a point on a manifold. What we can easily do, however, is define a point on a manifold in a particular coordinate system, or chart:

```
data c :#: m = Point m [Double]
```

Although I have called the value constructor of a point *Point*, the type constructor of a point is rather the operator $:\#:$. As we will later see, defining the type constructor of a point as an operator leads to much more readable types for the resulting points. I have chosen this symbol because the datatype operators in Haskell must be surrounded by colons, and the # symbol looks rather like the grid of a coordinate system.

The $:\#:$ type constructor is a function of two types, c and m. m indicates the manifold on which the point lies, and c indicates the chart in which the point is expressed. The value constructor *Point* is also a function of two arguments, but in this case it is the coordinates of the point as given by a list of doubles (real values), and the manifold m on which the point lies.

Notice that the chart c in the type constructor : # : does not appear as an argument of the value constructor *Point*. This means that c is a so-called phantom type. A phantom type is a type which only lives at the type level. In practice what this means is that the only way for the chart to influence computations is through typeclasses. The most fundamental typeclass in Goal which depends on the chart of a point is the *Transition* typeclass, which re-expresses a point in a different chart.

For example, let us consider two charts on Euclidean space:

```
data Cartesian

data Polar
```

Observe that because charts are phantom types, they do not need value constructors. The *Transition* typeclass is then defined as:

```
class Transition c d m where
    transition :: c :#: m -> d :#: m
```

The *transition* function takes a point in c-coordinates, and returns the same point in d-coordinates. In the case of the 2-dimensional Euclidean space for example, we may instantiate the coordinate transforms between the *Cartesian* and *Polar* charts as follows:

```
instance Transition Cartesian Polar Euclidean where
    transition (Point (Euclidean 2)) [x,y]) =
        Point (Euclidean 2) [sqrt (x^2 + y^2), atan2 y x]

instance Transition Polar Cartesian Euclidean where
    transition (Point (Euclidean 2)) [r,a]) =
        Point (Euclidean 2) [r * cos a, r * sin a]
```

Note that we do not need to specify the charts of the input and output of the *transition* function, as they are inferred from the definition of the function in the *Transition* class. I depict an example of transforming from *Cartesian* into *Polar* coordinates in figure 7.1.

7.1.3 Differential Geometry

We have now amassed a number of definitions for working numerically with manifolds. However, we are still not in a position to perform numerical optimization, as this requires that we define types and functions for working with differentiable and Riemannian manifolds. Ultimately we want to define gradient descent of a function on a manifold, which requires definitions for tangent spaces and metric tensors. To begin, let us define tangent spaces:

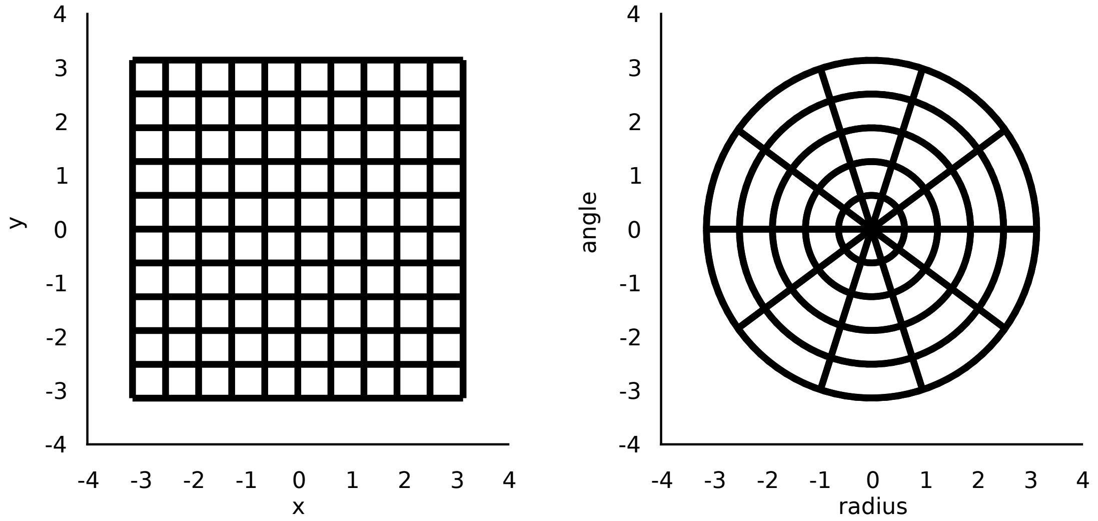

Figure 7.1: Here I depict an application of the *transition* function from *Cartesian* to *Polar* coordinates. *Left*: A regular grid of points in *Cartesian* coordinates. *Right*: I map the grid of points from the left plot into *Polar* coordinates with the *transition* function, and plot the angle and radius of the resulting points.

```
data Tangent c m = Tangent (c :#: m)

instance Manifold m => Manifold (Tangent c m) where
    dimension (Tangent (Point m _)) = dimension m
```

A tangent space is a kind of manifold with a more complicated definition than what we have seen so far. Mathematically, a tangent space is a vector space defined at a single point. In Goal, however, we do not work with points directly, but rather the coordinates of points in particular charts. In Goal I therefore define the tangent space at a point $c : \# : m$ as the tangent space of the manifold m embedded locally in the chart c. The value constructor *Tangent* thus takes a single point as an argument, and the type constructor *Tangent* takes two arguments, which indicate the chart and manifold of the underlying point. Finally, the *dimension* of a given tangent space is the dimension of the underlying manifold, in accordance with the mathematical definition.

In order to define gradient descent on a manifold, we also need to define cotangent spaces. In order to express cotangent spaces in Goal I apply another practical trick. Rather than defining a cotangent space as a distinct kind of manifold, I define a cotangent space as an element of a tangent space in a different coordinate system. This works in practice because tangent and cotangent spaces are isomorphic. I therefore define two charts for tangent spaces:

```
data Directional

data Differential
```

A point in a tangent space in directional coordinates represents a tangent vector, and a point in differential coordinates represents a cotangent vector. The last step toward implementing gradient descent on manifolds is defining the metric tensor. In order to express the metric tensor in a particular chart, we define the *Function* chart:

```
data Function c d
```

The type constructor *Function* takes two arguments, which correspond to the chart of the domain and codomain of the function question. Given this definition, we may then define a manifold as Riemannian with the following typeclass:

```
class Manifold m => Riemannian c m where
    metric :: c :#: m -> Function Directional Differential
                            :#: Tensor (Tangent c m) (Tangent c m)
    flat :: Directional :#: Tangent c m
               -> Differential :#: Tangent c m
    sharp :: Differential :#: Tangent c m
               -> Directional :#: Tangent c m
```

This is the most verbose typeclass definition that I present, but it is not especially complicated. Firstly, a *Riemannian* manifold is defined by the function *metric* which takes a point, and returns a matrix in the tensor product space of the tangent and cotangent space. As such, it is a bilinear form on two tangent vectors, which matches the definition of the metric tensor. Notice that we have two additional functions, *flat*, and *sharp*, in this class. These functions are simply the applications of the metric tensor and its transpose to a tangent and cotangent vector, respectively.

We are now ready to implement gradient descent on a manifold. Firstly, we define a single step of gradient descent as the function:

```
gradientStep
    :: Manifold m
    => Double
    -> Directional :#: Tangent c m -> c :#: m
gradientStep eps (Point (Tangent (Point m xs)) fs') =
     Point m (zipWith (+) xs (map (*negate eps) fs'))
```

The most complicated part of this function is simply deconstructing the given tangent vector. We break the tangent vector down into the underlying manifold *m*, the coordinates of the current point *xs*, and the coordinates of the tangent vector *fs'*. We then multiply the tangent vector by the negative of the scalar *eps*, which is the step size of the gradient descent, and we add this to the coordinates *xs*, and return a new point on the manifold *m* with these coordinates.

Finally, we may implement gradient descent on a manifold with the following function:

```
gradientDescent :: (Riemannian c m, Manifold m)
    => Double
    -> (c :#: m -> Differentials :#: Tangent c m)
    -> (c :#: m)
    -> [c :#: m]
gradientDescent eps df p0 =
    iterate (gradientStep (-eps) . sharp . df) p0
```

The three arguments to *gradientDescent* are the step size *eps*, the differential map of the function we are trying to minimize *df*, and the initial point of the gradient descent *p0*. The *iterate* function returns a list which contains repeated applications of the given function on the initial point. In this case, the initial point is *p0*, and the given function is the composition of applying the differential map, applying the *transition* function to compute the tangent vector, and taking a step along the vector. I use this gradient descent function in computing figure 2.2, in combination with the definitions I provide in the next subsection.

7.1.4 Exponential Families

Now that we have some general tools for working numerically with points on manifolds, let us implement exponential family manifolds using this tool set. The first step is to define some charts:

```
data Standard

data Natural

data Mean
```

The *Natural* and *Mean* coordinate systems indicate the natural and mean coordinate systems developed for exponential family manifolds in section 2.3. The *Standard* chart is used to represent some standard set of parameters for a family, for example the mean and variance of the normal family.

The next step is to define the class of exponential family manifolds:

```
class Manifold m => ExponentialFamily m where
    sufficientStatistic :: m -> Double -> Mean :#: m
    baseMeasure :: m -> Double -> Double
    logPartitionFuntion :: Natural :#: m -> Double
    negativeEntropy :: Mean :#: m -> Double
```

The *sufficientStatistic* and *baseMeasure* define an exponential family. We must also include the *logPartitionFunction* and *negativeEntropy* functions in this definition, as there is no automatic way to compute the integrals which define them.

Based on the *ExponentialFamily* class, we may define the probability density of a point on an exponential family manifold in general as:

```
density :: ExponentialFamily m
    => Natural :#: m -> Double -> Double
density p@(Point m _) x =
    exp (p <.> sufficientStatistic m x - logPartitionFunction p)
```

Here the $<.>$ function is the dot product between the two points in natural and mean coordinates, respectively.

In contrast to the case of integrals, we may use automatic differentiation libraries to implement the *transition* function between the *Mean* and *Natural* coordinates by computing the derivatives of the *logPartitionFunction* and *negativeEntropy*. The *transition* functions from *Natural* to *Mean* coordinates and *Mean* to *Natural* coordinates implement the forward and backward mappings, and so merely by defining the *logPartitionFunction* and *negativeEntropy* functions, we may automatically implement a variety of statistical algorithms. For example, given a list of data, we may compute the maximum likelihood estimator of a normal distribution in *Natural* coordinates as follows:

```
normalMLE :: ExponentialFamily m
    => [Double] -> Natural :#: Normal
normalMLE xs = transition $ sufficientStatisticN Normal xs
```

Here, the *sufficientStatisticN* function is the average of the *sufficientStatistic* over the given data *xs*. By combining the exponential family class with automatic differentiation, we may also automatically instantiate the *Riemannian* class for exponential families, as well as easily implement the exponential family multiplayer perceptrons defined in section 5.2.2.

At this point Goal is composed of roughly ten thousand lines of code, and what I have described here is only an overview of some of the core functionality of Goal. Beyond this I have implemented types and functions for working with manifolds of neural networks and mechanical systems, for simulating mechanical systems and stochastic systems, as well as additional optimization techniques, and for implementing filtering and control theory algorithms. All of these libraries, as well as the scripts for generating the figures in this dissertation, are available online at my repository `https://hub.darcs.net/alex404/goal`.

7.2 Restricted Boltzmann Machines

A restricted Boltzmann machine (RBM) is a harmonium model defined by a pair of product families $\mathcal{M}_X$ and $\mathcal{M}_Z$ of independent Bernoulli distributions. In this section I train a pair of RBMs on the MNIST dataset of handwritten digits. I train the first RBM with the rectification-based algorithm which I introduced in subsection 4.1.3, and I compare the result with the second RBM which I train with contrastive divergence minimization (Hinton, 2002). This experiment demonstrates that models which are not naturally rectified can be approximately rectified and still explain data.

The MNIST dataset is a collection of 8-bit greyscale images with 28×28 pixels, such that each element of the MNIST set is a 784-dimensional vector of integers between 0 and 255. For the purposes of this experiment, I transform the MNIST dataset into a target distribution P_Z of vectors of 0s and 1s. Since I ultimately use stochastic gradient descent (see section 2.2) to train the RBMs, I define P_Z implicitly with the following sampling procedure: I select a random element of the MNIST training set with a uniform probability; I normalize each pixel to be between 0 and 1 such that each value can be interpreted as a probability of the pixel being 0 or 1; I then resample these pixels to be either 0 or 1 based on the assigned probability.

In this experiment I train a pair of RBMs to model this P_Z. In both cases the RBMs are defined by the exponential families $\mathcal{M}_X$ and $\mathcal{M}_Z$, where $\mathcal{M}_X$ is the product family of 64 independent Bernoulli distributions, and $\mathcal{M}_Z$ is the product family of 784 independent Bernoulli distributions. The first model Q^R_{XZ} is an RBM trained with rectification-based algorithms, and the second model Q^{CD}_{XZ} is trained with contrastive

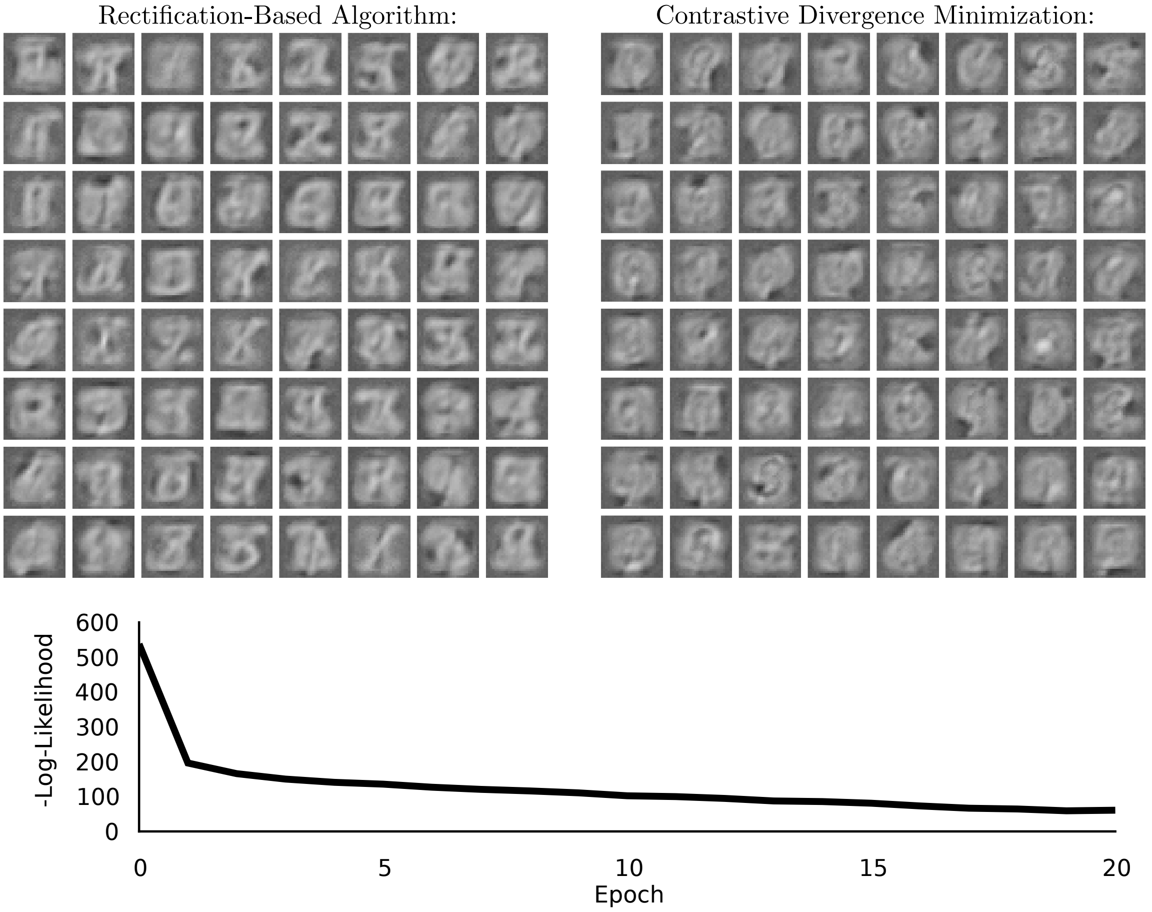

Figure 7.2: In this figure I compare the results of training Q^R_{XZ} and Q^{CD}_{XZ} on MNIST. *Top*: The receptive fields of each of the 64 neurons in the hidden layers of the trained RBMs. *Bottom*: The approximate negative log-likelihood of Q^R_X on the validation date of MNIST.

divergence minimization. In the second case, contrastive divergence minimization is a widely-applied algorithm for approximating the derivatives of $D(P_Z \parallel Q^{CD}_Z)$ with respect to the parameters of the RBM.

In the case of Q^R_{XZ}, I minimize the training objective $D(P_Z \parallel Q^R_Z)$ in parallel with the rectification error $D(P^*_X \parallel Q^R_X)$, where P^*_X is the rectifier described in section 4.1.3. Under the assumption that the rectification error is small, I approximate the expectations in the derivatives of $D(P_Z \parallel Q^R_Z)$ (see equations 4.8) with approximate samples from $Q_{XZ}R$ generated by sampling Q^R_X followed by $Q^R_{Z|X}$.

I implement gradient descent on these approximate gradients with the Adam algorithm (Kingma and Ba, 2014). For this I use a learning rate of $\epsilon = 0.001$, and otherwise I use the standard parameters listed in the paper. I trained each model for 20 epochs, where each epoch is composed of 1000 gradient steps, and each gradient step is the average of the approximate gradients of 10 samples from P_Z. I plot the results of this experiment in figure 7.2.

In the upper plots of this figure I display the receptive fields of the hidden neurons in both RBMs. The receptive field of neuron i is the vector of probabilities $Q_{Z|X=\boldsymbol{\delta}_i}$, where $\boldsymbol{\delta}_i$ is a vector where the ith element is 1, and all other elements are 0. The receptive field visualizes the image to which the given neuron most strongly responds. There are many similarities between these two collections of receptive fields: the trained neurons respond weakly to pixels on the edge of the image, and many neurons have patches of low-response embedded in a high-response area. On the other hand, the receptive fields of Q^R_{XZ} reveal that the hidden neurons of Q^R_{XZ} respond most strongly to images which look roughly like digits. By contrast, the representation learned by Q^{CD}_{XZ} appears much more distributed in nature.

In the lower plot I depict the approximate negative log-likelihood of Q^R_Z given 10,000 samples from the MNIST validation dataset after each training epoch. In general, there is no closed-form expression for the negative log-likelihood of a harmonium model. However, when the harmonium model is approximately rectified, then we may use corollary 4.4 to compute the approximate negative-log likelihood. As can be seen, the rectification-based algorithm minimizes this approximate negative log-likelihood, and thereby appears to effectively train Q^R_{XZ} to represent the MNIST dataset.

7.3 Biological Neural Circuits

In this section I run three simulations in which I model how a biological neural circuit learns to implement a Bayes filter and thereby compute beliefs about a dynamic stimulus. The three stimuli are colour sequences which I model as a finite-state Markov chain; the position of a mouse on a track which I model as a 1-dimensional linear dynamical system; and the angle and angular velocity of a human arm, which I model as a pendulum.

I model these neural circuits with neural Bayes filters $(X_k, Z_k, V_k, W_k)_{k \in \mathbb{N}}$, with emission distributions given by the likelihood of a homogeneous LPPC. For the purposes of this section, I refer to $(Z_k)_{k\in\mathbb{N}}$ as the observation population, $(V_k)_{k\in\mathbb{N}}$ as the prediction population, and $(W_k)_{k\in\mathbb{N}}$ as the posterior population.

A neural Bayes filter is defined by the parameters of the emission distribution $\mathbf{\Theta}_Z$ and $\boldsymbol{\theta}_Z$, and the recoder $\mathbf{A}$; the parameters of the prediction population $\boldsymbol{\theta}^V_X$ and $\mathbf{\Theta}_V$, and recoder $\mathbf{B}$; the parameters of the posterior population $\boldsymbol{\theta}^W_X$ and $\mathbf{\Theta}_W$; and the initial prior encoding $\mathbf{v}_0$. For the purposes of the experiments in this section, I set these parameters by hand. The remaining parameters are the parameters $\boldsymbol{\chi}$ of the EFMLP $\mathbf{v}$, which I optimize with the method described in subsection 5.2.4, except for the weights in the output layer $\mathbf{\Theta}_W$.

In order to test what kind of population codes are likely used by the brain, I train two candidate neural circuits which differ in the parameters of the prediction and posterior populations. Moreover, when training the EFMLP in each circuit, I apply and compare contrastive divergence minimization with a rectification-based algorithm, leading to a total of four sub-experiments in each experiment. In each experiment I also validate the learned filter against the corresponding optimal, or mostly optimal, filter. The result of this section are adapted from my article Sokoloski (2017).

7.3.1 Training and Validation Procedure

In each simulation I define the sum of the tuning curves of the homogeneous LPPC as constant, so that $\boldsymbol{\rho}_X = \mathbf{0}$. I define the parameters of the prediction and posterior populations by $\boldsymbol{\theta}^V_X = \boldsymbol{\theta}^W_X = \mathbf{0}$, and $\mathbf{\Theta}_V = \mathbf{\Theta}_W$. This implies that the recoder $\mathbf{B} = \mathbf{I}$. In this case we may intuitively think of the two neural populations as a single population for encoding approximate beliefs. Finally, I set the initial rates of

Parameter	Experiment 1	Experiment 2	Experiment 3
$d_Z = d_V = d_W$	10	10	20
d_Y	100	200	500
n_t	$10,000$	$10,000$	$20,000$

Table 7.1: In this table I show the simulation parameters which change in each experiment. These are the sizes of the observation, prediction, and posterior populations d_N, d_Y, d_Z, respectively; the number of hidden neurons in the EFMLP d_Y; and the number of steps in the training simulation n_t. In all experiments, $\mathbf{\Theta}_Z = \mathbf{\Theta}_Y$ and $\mathbf{y}_0 = \mathbf{0}$; in the naive circuit $\mathbf{\Theta}_Z = \mathbf{\Theta}_N$, and in the orthogonal circuit $\mathbf{\Theta}_Z$ is constructed from orthogonal rows which are also orthogonal to 1. Where i_e is the epoch, the initial learning rate of the Adam algorithm is $\epsilon = 0.00005 \cdot 1.25^{-(i_e-1)}$, and contrastive divergence is run with i_e contrastive divergence steps.

the prediction population to $\mathbf{y}_0 = \mathbf{0}$, such that the prediction population initially encodes a flat prior over the stimulus.

I define the parameters of the posterior population $\mathbf{\Theta}_W$ in one of two ways. The first is by setting $\mathbf{\Theta}_W = \mathbf{\Theta}_Z$, which I refer to as the naive code. In the case $\mathbf{A} = \mathbf{I}$. I refer to the second code as the orthogonal code, based on the code presented in the supplementary material of Beck et al. (2011). In this case we construct $\mathbf{\Theta}_W$ from a set of mutually orthogonal rows, which are also orthogonal to the vector of ones, such that $\mathbf{\Theta}_{Z,i} \cdot \mathbf{\Theta}_{Z,j} = 0$ for $i \neq j$, and $\mathbf{\Theta}_{Z,i} \cdot 1 = 0$. As a result of this, $\mathbf{\Theta}_W \cdot 1 = \mathbf{0}$, which implies that for any rates of the posterior population $\mathbf{w}$, the rates $\mathbf{w} + c1$ encode the same beliefs for any scalar c. The details of constructing $\mathbf{\Theta}_W$, as well as a corresponding recoder $\mathbf{A}$ which satisfies equations 5.6, can be found in the supplementary material of Beck et al. (2011).

For the naive code, because $\mathbf{\Theta}_Z = \mathbf{\Theta}_V = \mathbf{\Theta}_W$, all three neural populations in the circuit have the same number of neurons. In the case of the orthogonal code, because $\mathbf{\Theta}_V = \mathbf{\Theta}_W$, the prediction and posterior populations have the same number of neurons; this number however need not equal the number of neurons in the observation population. Although we could set the number of neurons in the prediction and posterior populations to be different from the number in the observation population, I define $\mathbf{\Theta}_W$ to have the same number of columns as $\mathbf{\Theta}_Z$ in order to ensure that differences in circuit performance are not simply due to differences in the number of parameters. This implies that in all the cases we consider, $d_Z = d_V = d_W$, which is to say that the observation, prediction, and posterior populations always have the same number of neurons. For the number of neurons in each experiment, and a summary of all simulation parameters, see table 7.1.

In all experiments I define the EFMLP $: \mathrm{H}_W \to \mathrm{H}_V$ by the three exponential families $\mathcal{M}_W$, $\mathcal{M}_Y$, and $\mathcal{M}_V$. I define $\mathcal{M}_Y$ to be the product family of d_Y independent Bernoulli distributions, so that the transfer function in the hidden layer of the EFMLP is the logistic function. I define $\mathcal{M}_W$ and $\mathcal{M}_V$ to be product families of independent Poisson distributions, so that the EFMLP has an exponential transfer function in the output layer. I use the exponential function on the output layer to ensure that the rates computed by the EFMLP are always positive. This is especially important in the case of the naive code, as negative rates cannot be reliably decoded by $\mathbf{\Theta}_Z$.

The training objective for $\mathbf{v}$ is to minimize the relative entropy in expression 5.14. In order to approximate the derivatives of this expression, I apply either contrastive divergence minimization, or I use the rectification of the emission distribution to estimate the expectations in the derivatives with approximate samples (see equation 5.15). Given the proposed neural circuits and gradient approximation schemes, I run four parallel simulations in every experiment, by applying either contrastive divergence minimization (CD) or the rectification-based algorithm (R) to approximating the stochastic gradient (2.5), in order to train the neural circuits based on either the naive (NV) or orthogonal (OT) population codes. I denote these four simulations and corresponding circuits by NV-R, NV-CD, OT-R, and OT-CD.

In each experiment I train each neural circuit over the course of twenty epochs, where each epoch is composed of a training simulation of n_t steps. During the training simulation, where i_e is the number of the current epoch, I reset the rates of the prediction population V_k to $\mathbf{0}$ every $(i_e - 1)^2$ number of steps. I do this because newly initialized EFMLPs $\mathbf{v}$ tend to be unstable, in that recursively evaluating W_k for large k tends to result in rates which diverge and fail to encode accurate beliefs. By first training $\boldsymbol{\theta}_{W|V}$ on shorter, stable paths, we may avoid this problem and better maximize the likelihood of the parameters $\boldsymbol{\chi}$.

I use the Adam algorithm to update the parameters $\boldsymbol{\chi}$ (Kingma and Ba, 2014). At every epoch I define the initial learning rate to be $\epsilon = \frac{0.00005}{1.25^{i_e - 1}}$, and otherwise I use the parameters listed in the paper. Finally, when applying contrastive divergence minimization, I also set the number of contrastive divergence steps equal to the epoch number i_e.

After each training epoch I validate the trained neural circuits on a simulation of $n_v = 200,000$ steps. I compute the sequence of rates $\mathbf{w}_0, \dots, \mathbf{w}_{n_v}$ of the posterior population as a function of the sequence of validation responses $\mathbf{z}_0, \dots, \mathbf{z}_{n_v}$ without resetting the rates of the prediction population. Where ε is a function which measures error given a stimulus and the natural parameters of a belief distribution, I then compute the average of the error measure $E_W = \sum_{i=0}^{n_v} \frac{\varepsilon(\mathbf{x}_i, \boldsymbol{\Theta}_W \cdot \mathbf{w}_i)}{n_v}$. In the first and second experiments I define ε as the negative log-likelihood, and in the third experiment, since the negative log-likelihood does not have a closed-form expression, I define ε as the mean-squared error.

By computing the average error $E_{Opt} = \sum_{i=0}^{n_v} \frac{\varepsilon(\mathbf{x}_i, \boldsymbol{\theta}_k)}{n_v}$ on the belief parameters $\boldsymbol{\theta}_k$ of the closed-form filters described in section 5.2.1, I compute a lower-bound on the error of the trained circuits. Conversely, since any useful filter must provide more information about the stimulus then the instantaneous responses, I compute $E_Z = \sum_{i=0}^{n_v} \frac{\varepsilon(\mathbf{x}_i, \boldsymbol{\Theta}_Z \cdot \mathbf{z}_i)}{n_v}$ to provide a performance upper-bound. Finally, by computing the ratio $r = \frac{E_W - E_Z}{E_{Opt} - E_Z}$, I express the performance of the neural circuit in question as a percentage of the distance achieved from the upper- to the lower-bound.

Finally, in the last two experiments I also estimate the tuning curves of the hidden layer of the EFMLP with respect to the stimuli. I estimate these tuning curves by simulating the trained neural circuits for n_v steps, and then sorting these steps into bins, where each bin contains the activity of the hidden layer of the EFMLP when the stimulus is near a particular stimulus value. I then average the rate of each hidden neuron in each bin, in order to estimate the mean activity of the hidden neuron given the stimulus that corresponds to the bin.

7.3.2 Colour Sequence Learning

In this simulated experiment I imagine that subjects are shown sequences of colours drawn from red, green, and blue. The colours are described by a Markov chain, such that each colour has a certain probability of appearing based on the previously seen colour. Subjects must learn to predict the sequence as well as possible. I assume that the stimuli change quickly, so that subjects do not always perceive the stimulus before it transitions to the next stimulus value. I consider how this problem might be solved in the ventral stream, and model colour-sensitive neurons in the visual area V4 with the observation population, and sequence-learning neurons in the inferior temporal cortex with the prediction and posterior populations (Roe et al., 2012).

For simplicity, let us denote the three colour values by r, g, and b. The transition probabilities of the Markov chain are

$$\begin{aligned} &p_{X'|X}(r \mid r) = p_{X'|X}(b \mid b) = 0.8, && p_{X'|X}(g \mid g) = 0.5, \\ &p_{X'|X}(r \mid g) = p_{X'|X}(b \mid g) = 0.25, && p_{X'|X}(g \mid r) = p_{X'|X}(g \mid b) = 0.15, \\ &p_{X'|X}(b \mid r) = p_{X'|X}(r \mid b) = 0.05. \end{aligned}$$

Intuitively, blue tends to stay blue and red tends to stay red, whereas green is a relatively transitory state. Moreover, red and blue tend to first transition through green before reaching blue and red, respectively.

I assume that the observation population has $d_N = 10$ neurons, and that the gain $\gamma = 1$. I define the tuning curve of neuron i given blue as $f_i(b) = e^{0.4(i-1)-5}$, given red as $f_i(r) = f_{10-i}(b)$ and given green as $f_i(g) = \frac{1}{n}\sum_{i=1}^{10} f_i(b)$. This construction ensures that equation 6.4 is satisfied exactly. Intuitively, the low-index neurons of the observation population respond to red, the high-index neurons respond to blue, and the observation population responds with a uniform pattern of activity to green, which provides little information about the true colour. Finally, I set the number of hidden neurons in the EFMLP to $d_Y = 100$.

I depict the results of the simulations of the NV-R, NV-CD, OT-R, and OT-CD circuits in figure 7.3. As displayed in the left panel, the circuit which best approximates the true beliefs of the Bayes filter is the orthogonal circuit trained with the rectification-based approximation of the stochastic cross-entropy gradient (solid red), which achieves $r = 95.4\%$ of the performance of the Bayes filter. In this experiment, because the emission distribution is exactly rectified, it is unsurprising that the rectification-based gradient produces the best results, as it is in fact equal to the true stochastic gradient. It is surprising, however, that the choice of population code has such a dramatic effect on the learning. Where the orthogonal circuits more or less completely recover the true beliefs, the naive circuits cannot even surpass the baseline provided by the responses.

In the right panels of figure 7.3 I display 30 steps of a simulation from this system. In the top right panel I use the opacities of coloured squares to show the dynamic beliefs of both the optimal and OT-R filter, and one can see how the beliefs of the two filters are nearly identical. In the bottom right panel I show the corresponding population responses. In total, the observation population spikes 25 times over the course of the simulation. Both filters initially recognize that the stimulus is blue. However, in the middle of the simulation when the stimulus changes to red and back to blue again, the filters cannot recognize this, as no spikes reveal this transition.

7.3.3 Self-Localization

In this simulation I model self-localization in a mouse confined to a one-dimensional track, which explores the local track while avoiding straying too far from its home

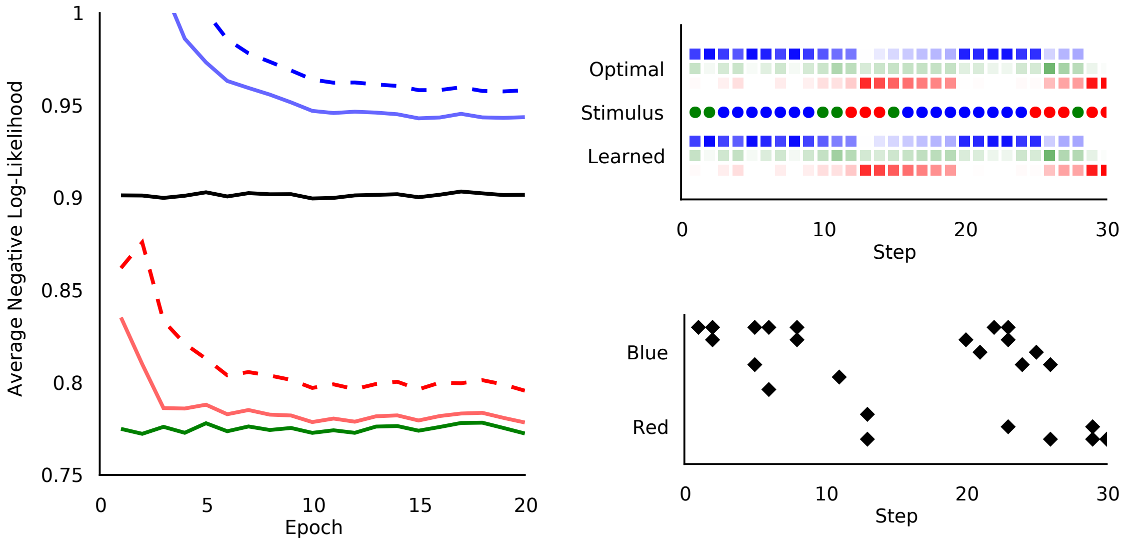

Figure 7.3: Here I depict the training of the proposed neural circuits, and a simulation with the OT-R circuit. *Left*: The average negative log-likelihood of the approximate beliefs given the stimuli over each epoch. I display the descent of the NV-R circuit (blue), the NV-CD circuit (dashed blue), the OT-R circuit (red) and the OT-CD circuit (dashed red). I also depict the baseline (black) provided by the population responses and the optimum (green) computed by the discrete Bayes filter. *Top Right*: A simulation from the Markov chain (coloured circles), as well as the learned an optimal filters. The beliefs of the optimal filter (top) and the learned filter (bottom) are indicated by the opacity of a colour, which corresponds to the inferred probability of the stimulus value. *Bottom Right*: The responses of the observation population over the 30 steps of the simulation. Spikes (black diamonds) from a particular neuron are arranged along the x-axis in accordance with the neuron index.

position. I model the dynamics of the position of the mouse with a stochastic, one-dimensional, linear dynamical system. By applying a neural Bayes filter, I model how the mouse learns to track its position in a novel environment with place cells in the hippocampus, given noisy position estimates provided by visual cues (McNaughton et al., 2006). I model place cells with the posterior population, and cue-sensitive cells with the observation population.

Since the position of the mouse is a continuous-time variable, we may describe it with the linear stochastic differential equation

$$dX_t = aX_t dt + bdW_t,$$

where W_t is a Wiener process. Where h is the time-step, this implies that the transition distribution $P_{X'|X}$ of the time-discretized system at x is a normal distribution with mean $x + hax$ and variance hb^2. In this case I set $a = -1$, $b = 1$, and $h = 0.02$. I then define the observation population to have $d_Z = 10$ neurons with the 1-d Gaussian tuning curves defined in equation 6.2, with gain $\gamma = h \cdot 100 = 2$, preferred stimuli x_i^0 distributed uniformly over the interval $[-7, 7]$, and variance $\sigma^2 = 2$. Finally, I set the number of hidden neurons of the EFMLP to $d_Y = 200$.

I depict the results of the four simulations in figure 7.4. As depicted in the top right panel, the orthogonal circuit trained with the rectification-based algorithm (red) best approximates the optimal filter, achieving $r = 96.0\%$ of the performance of the optimal filter, which is slightly better than the OT-CD circuit. In the left panel I display a 2 second simulation from the system. The black dots indicate the mean of

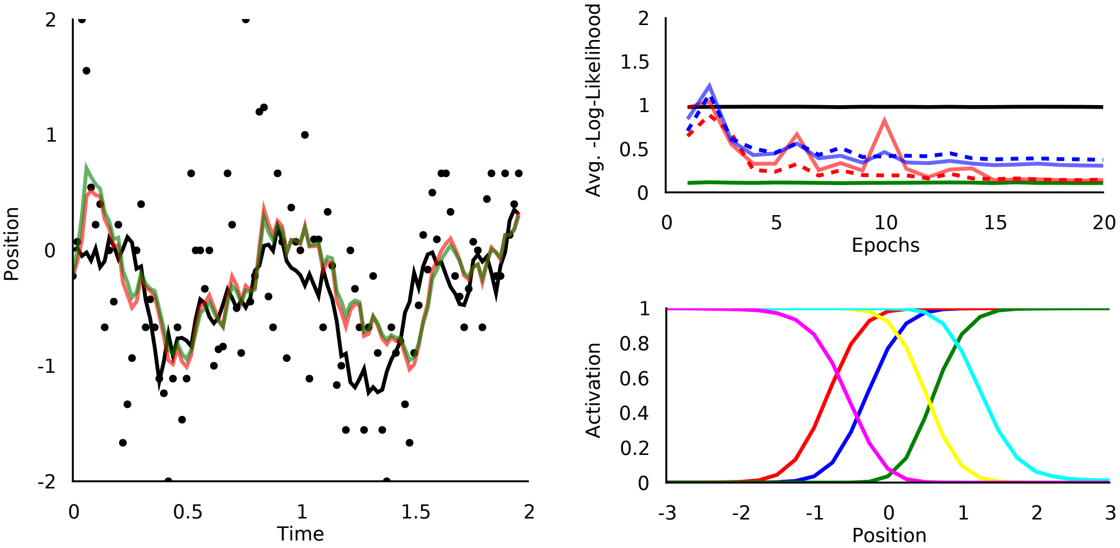

Figure 7.4: Here I depict the training of the proposed neural circuits, and simulations with the OT-R circuit. *Top Right*: The descent of the negative log-likelihood using the same colour scheme as in figure 7.3. *Left*: A simulation from the dynamical system, where I depict the stimulus (black line), and the dynamic mean of the response posteriors (black dots), the optimal belief distribution (green line), and the learned belief distribution (red line). *Bottom Right*: Six tuning curves from the hidden layer of the EFMLP.

the posterior $P_{X|Z=\mathbf{z}_k}$ of each response $\mathbf{z}_k$ under the assumption of a flat prior. The mean of the optimal beliefs (green line) given these responses is very close to the true stimulus, and the mean of the learned beliefs (red line) is nearly identical to the optimum. In the bottom right panel I display six approximate tuning curves from the hidden layer of the trained EFMLP $\mathbf{v}$, which has learned sigmoid tuning curves over the stimuli.

7.3.4 Proprioception

In this final simulated experiment I model how a human optimizes its forward model of the swing of its arm. I focus on the role of the cerebellum in proprioception, and assume that Purkinje cells in the cerebellum receive information about the angle and angular velocity of the shoulder from proprioceptors, and use this information to drive a forward model of arm position (Kawato et al., 2003; Franklin and Wolpert, 2011). I model the neural populations in the cerebellum with the prediction and posterior populations, and the proprioceptors with the observation population.

For simplicity, I assume that the arm may be described by a single rigid body at a joint, and that the subject uses random motions of the arm in order to explore its dynamics. I therefore model the arm as a stochastic pendulum, which is a two-dimensional stochastic process over the angular position q, and the angular velocity $\dot{q}$. I define the discrete-time transition dynamics $P_{X'|X}$ of the stochastic pendulum at $\mathbf{x} = (q, \dot{q})$ as a multivariate normal distribution with mean $\mathbf{x} + h\mathbf{a}(\mathbf{x})$ and covariance matrix $h^2\mathbf{\Sigma}$, where h is the time-step. The function $\mathbf{a}$ is known as the drift, and is given by

$$a_1(q, \dot{q}) = \dot{q},$$
$$a_2(q, \dot{q}) = -g \sin(q) - c\dot{q},$$

where $g = 9.81$ is the gravitational constant and $c = 0.1$ is the coefficient of friction. I define the covariance matrix of the process by

$$\mathbf{\Sigma}_{(1,1)}(q, \dot{q}) = \mathbf{\Sigma}_{(1,2)}(q, \dot{q}) = \mathbf{\Sigma}_{(2,1)}(q, \dot{q}) = 0,$$
$$\mathbf{\Sigma}_{(2,2)}(q, \dot{q}) = \sigma_{\dot{q}}^2,$$

where $\sigma_{\dot{q}}^2 = 1$ is the variance of the noise process. By restricting the noise to the velocity, we may interpret the noise to be the result of the subject applying random forces to its arm. Finally, I define $h = 0.02$.

I define the gain of the emission distribution $P_{Z|X}$ as $\gamma = h \cdot 100 = 2$, and I define the tuning curves of $P_{Z|X}$ with two independent sets of tuning curves over the angle and angular velocity, such that half the neurons in the observation population respond to angle, and the other half to angular velocity. Since the angle is periodic, I define the tuning curves over the angle as a set of von Mises tuning curves (6.3) with 10 preferred stimuli q_i^0 distributed uniformly over the period $[-\pi, \pi]$, and concentration $\kappa = 1/2$. The tuning curves over the angular velocity are again 1-dimensional Gaussian tuning curves as defined in equation 6.2 with 10 preferred stimuli distributed uniformly over the interval $[-12, 12]$, and covariance $\sigma^2 = 4$. The sufficient statistic of the exponential family $\mathcal{M}_X$ determined by these tuning curves is $\mathbf{s}_X(q, \dot{q}) = (\cos q, \sin q, \dot{q}, \dot{q}^2)$. In total, the neural populations have $d_Z = d_W = d_V = 20$ neurons, and I set the number of neurons in the hidden layer of the EFMLP $\mathbf{v}$ to $d_Y = 500$.

I depict the results of the four simulations in figure 7.5. For the purposes of validation, I apply an approximate EKF based on the von Mises approximation developed in section 5.2.1. As displayed in the top left panel, the circuit which best approximates the optimal beliefs, by an extremely slim margin over the OT-R circuit, is the orthogonal circuit trained with the contrastive divergence minimization (dashed red), which achieves $r = 89.7\%$ of the performance of the approximate EKF.

In the lower two panels I depict a 4 second simulation from the system. The black dots, green line, and red line depict the mean of the response posteriors, the approximate EKF, and the learned beliefs, as in the previous section. The blue line depicts the mean of the posteriors of an approximate Kalman filter with linear dynamics given by the small-angle approximation, and which updates its beliefs with the same strategy as the approximate EKF. As can be seen, a straightforward linear model is not sufficient for tracking the nonlinear stimulus.

In the upper right two panels I depict two tuning curves from the hidden layer of the trained EFMLP I plot the two-dimensional tuning curves by plotting the stimulus angle on the x-axis, and indicating the angular velocity with the colour of the line, where black corresponds to -6, and red to 6. As can be seen in these plots, the tuning curve over the angle is a von Mises tuning curve, and the angular velocity is a monotonic function, and the two components interact multiplicatively.

7.3.5 Analysis of Results

One of the most striking results of these simulations was that the performance of the naive circuit was significantly worse than the orthogonal circuit. In the self-

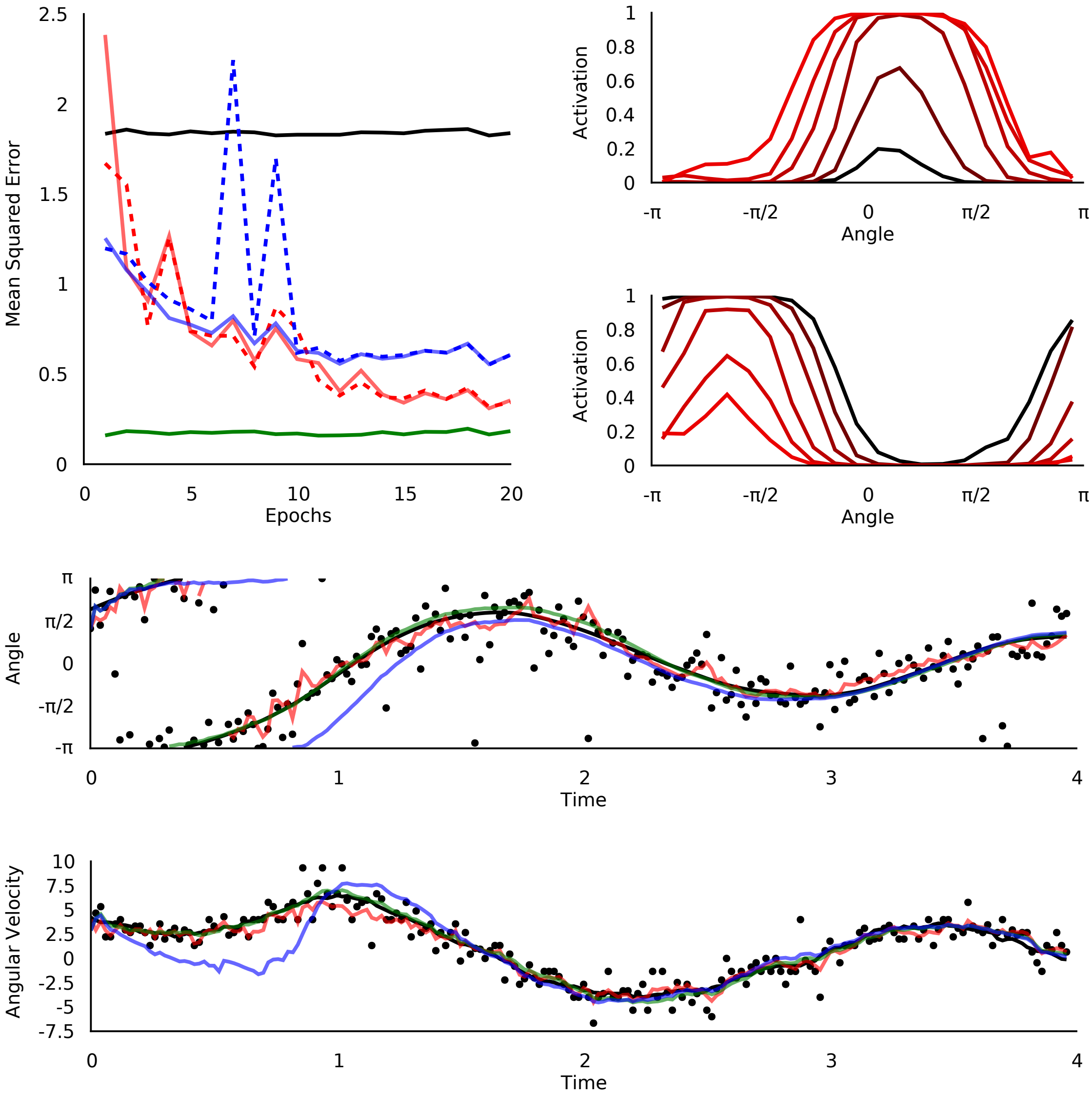

Figure 7.5: Here I depict the training of the proposed neural circuits, and simulations with the OT-CD circuit. *Top Right*: The descent of the mean squared error of the approximate beliefs using the same colour scheme in figure 7.3, where green indicates the approximate EKF beliefs. *Top Right*: Two tuning curves from the hidden layer of the multilayer perceptron. The stimulus angle is plotted on the x-axis, and the stimulus angular velocity is indicated by the colours from red to black, where black indicates an angular velocity of -6, and red indicates a velocity of 6. *Bottom*: Simulation of the angle and angular velocity from the dynamical system. I depict the stimulus (black line), and the dynamic mean of the response posteriors (black dots), the approximate EKF belief density (green line), the approximate KF belief density (blue line), and the learned belief density (red line).

localization and proprioception experiments, the naive circuit performs reasonably well, though not nearly as well as the orthogonal circuit. In the sequence learning experiment, however, the naive circuit fails to even achieve the upper-bound on the error provided by the instantaneous information in the responses. This is surprising, as the theory of universal approximation ensures that there exists parameters for the naive circuits for which their performance should be comparable to the orthogonal circuits. Nevertheless, it appears that the learning itself is heavily impacted by the choice of population code, and this finding is in line with computational (Boulanger-Lewandowski et al., 2012) and experimental (Chang and Snyder, 2010) evidence which suggests that diverse population codes improves performance in neural circuits.

I also found that the hidden layer of the prediction network learns tuning curves over stimuli. In the self-localization experiment (7.3.3), training the network resulted in sigmoid tuning curves over the unobserved stimuli. Although sigmoid tuning curves are often found in the brain (Pouget and Sejnowski, 1995; Pouget et al., 2000), sigmoid tuning curves for self-location have not been found in the limbic system. Although it could be the case that there exists an as of yet undiscovered neural population in the limbic system with tuning curves which match those of the learned hidden layer, I rather suspect that the proposed model neural circuits fail to capture essential features of the self-localization circuitry.

In simulation 7.3.4, training the neural circuit resulted in von Mises tuning curves over the angle, and sigmoid tuning curves over the angular velocity, which interact via multiplication. When the tuning curve over one stimulus interacts multiplicatively with the tuning curve over another stimulus, it is known as a gain-field or gain modulation (Salinas and Thier, 2000), and gain-fields have been found in many areas of the brain (Salinas and Thier, 2000; Hwang et al., 2003; Paninski et al., 2004). In particular, Herzfeld et al. (2015) demonstrated that eye position and velocity is encoded by Purkinje cells in the cerebellum with the same gain-field structure as in the arm-localization circuit. Although the stimuli in this experiment and our simulated experiment are different, both respective neural circuits must ultimately predict the motion of parts of the body, and they do so in similar manners.

It is well-known that given data which match population activity for encoding stimuli, gain-fields can arise spontaneously in the hidden layer of multilayer perceptrons (Zipser and Andersen, 1988). At the same time, Gaussian/sigmoid gain-fields have been used to model the neural computation of coordinate transformations in the posterior parietal cortex (Pouget and Sejnowski, 1995), and were found to be especially apt for computing addition over the encoded variables. In the proprioception experiment, although the neural circuit is not performing a coordinate transformation per se, it is learning to add velocity to position at every time, and therefore the emergence of this particular gain-field fits well into the existing theory. What is novel in my work however, is that the neural network is not trained solve a standard regression problem as in Zipser and Andersen (1988), but is trained rather as part of a more complex neural circuit for implementing a Bayes filter. As I have demonstrated, gain-fields continue to emerge in this context.

Chapter 8

Conclusion

In this dissertation I have developed a general theory of how embodied agents can implement Bayesian inference with neural networks. I began the development of this theory with an analysis of a large class of log-linear graphical model called nth-order harmoniums. I then developed the theory of rectified harmoniums, which are a constrained form of harmonium which support conjugate priors and exact sampling. In order to implement Bayesian filtering, I combined rectified harmoniums with a recurrent neural network for computing approximate predictions.

I then applied this theory within the context of computational neuroscience. In particular, I developed the connections between harmoniums and linear-nonlinear neurons and linear probabilistic populations codes, and showed how my theory can model phenomena such as proprioception and gain-modulated tuning curves in the brain. Moreover, I developed an innovative library for the programming language Haskell within which I implemented the large range of simulations which support my dissertation.

In concluding this dissertation I present an argument for the fundamental importance of neural computation to embodied cognition. This argument serves as a review of a number of the topics I covered in my dissertation, but also serves to emphasize how the theories I have developed address fundamental issues in embodied perception. I will then present an outlook of how these results can be extended in the future.

8.1 Brains, a Priori

What is cognition, and what distinguishes it from life or matter? The theory of optimization has been applied in physics, biology, and cognitive science to model, respectively, the dynamics of particles, the evolution of populations, and learning and inference in cognitive agents. Beyond its efficiency as a compact explanation of diverse phenomena, the way in which optimization is applied in these three fields can also help us distinguish their respective subjects.

In physics, Lagrangian mechanics describes the dynamics of particles as the solution of an optimization problem. Nevertheless, mechanical systems always realize the optimum of the Lagrangian optimization problem, whereas living things and cognitive agents are never so perfect. Mechanical systems are always optimized, whereas life and cognition are always adapting to the environment.

So what then distinguishes cognition from life? Let us formalize optimization as the process of maximizing some value as a function of some variables. In the case of life, we would formalize natural selection as the process of maximizing the fitness of a genetic population as a function of all the variables which describe the environmental niche of the population in question.

At an abstract level, we may think of cognition as the maximization of reward. In a particular context, this reward will correspond to the goal of some task, such as find food and water or hide from predators. However, whereas natural selection is a physical process realized directly by the cycle of death and reproduction, a cognitive agent does not know the values of the variables which define its rewards. Cognitive agents must first infer the state of the world before they can realize how to effectively interact with it.

A cognitive agent gathers information about the world through its senses, and the agent must optimally combine the information that each of the senses provide. Moreover, the agent must account for the dynamics of the environment, and keep collected information up-to-date as the environment changes. Bayesian inference is the most general theory of probabilistic inference which accounts for and unifies all these forms of inference.

Two of the most well-known theoretical arguments for the generality of Bayesian inference are Cox's theorem and the class of Dutch book arguments (Jaynes, 2003; Talbott, 2015). On one hand, Cox's theorem demonstrates that the consistent extension of propositional logic on binary truth values to continuous probabilities is given by Bayesian inference. On the other hand, Dutch book arguments demonstrate that failing to follow the principles of Bayesian inference can lead gamblers to make wagers which they are guaranteed to lose.

Nevertheless, Bayesian inference only describes optimal inference given a probabilistic description of all the relevant physical and cognitive variables. Before an agent can infer the unknown environment in a particular context, the agent must learn a model of the environment in general. Therefore, part of the optimization problem that a cognitive agent must solve is to optimize the parameters of its model of the world. One of the most principled ways of defining this optimization problem is as minimizing the relative entropy of the world distribution with respect to the internal model of the agent (Shore and Johnson, 1980).

A fundamental constraint on a cognitive agent is that it must dynamically solve the problems of learning and inference within a finite computational medium. This entails that both the complexity of the beliefs inferred by the agent and the complexity of the generative model used to compute these beliefs must be bounded. To address this, a cognitive agent may combine exponential family models with conjugate priors, as I studied in chapter 4 under the rubric of rectified harmoniums.

On one hand, it is possible to bound the complexity of a generative model by defining a statistic that summarizes the information gathered about the model parameters. It turns out that under mild conditions, the most general class of models with summary statistics is the class of exponential families (Koopman, 1936). On the other hand, the complexity of the probability distributions – or beliefs – that the agent infers may also be bounded by assuming that the agent represents its beliefs parametrically. In this case the agent may employ prior beliefs which are conjugate to posterior beliefs, which can also ensure that the complexity of posterior beliefs remains bounded (see section 2.6).

In order to further reduce the complexity of learning and inference, a cognitive agent may also employ models with a particular representational topology. That is, an agent may assume that some physical or cognitive variables only interact through intermediate representations, as represented by a graphical model. The theory of graphical models provides a manner of formalizing this concept of intermediate representation, which can greatly simplify computations amongst the constituent variables.

By analyzing several ways of bounding complexity, we have already arrived at a description of the internal model of a cognitive agent with a high degree of structure – a structure which is exemplified in my theory of rectified deep harmoniums (section 4.2). Nevertheless, we have yet to address how an agent learns to update its beliefs to account for a changing environment. As I showed in section 5.2, implementing optimal updates can be reduced to solving a regression problem – compute the parameters which best predict future observations as a function of current beliefs. Moreover, multilayer perceptrons are an excellent candidate for implementing these predictions, as they approximate functions arbitrarily well (Hornik, 1993).

The theory of universal representation of multilayer perceptrons depends on an arbitrarily large intermediate layer of simple pattern detectors (Hornik, 1993). These pattern detectors are sometimes called "neurons", and pattern detectors which emulate biological neurons have proven to be highly efficient computationally. In the context of regression, multilayer perceptrons are often composed of pattern detectors which emulate the average firing rate of biological neurons. Moreover, as I showed in section 6.3, large populations of Poisson neurons can be rectified in a straightforward manner, and thereby implement Bayesian inference and support efficient learning algorithms.

Finally, as discussed in section 6.1, spiking neurons support a continuous-time semantics, which is an important feature when describing embodied systems. In general, continuous-time stochastic processes can be described as jump processes, or diffusions, or a combination of the two. In the context of computation, jump processes implement digital computation, whereas diffusions implement analogue computation. Both digital and analogue computation are used in animal brains, but spiking neurons are the most energy efficient for transmitting information over the long distances of large, mammalian brains (Sterling and Laughlin, 2015).

In spite of the title, the arguments I have made in this section have not been strictly a priori. Rather, they have been a collection of arguments based on reason, fact, and speculation, and whatever else might help make the case for the importance of neural computation. Nevertheless, when all assembled, I believe these arguments indeed make the case that the ubiquity of neural computation is not an accident of evolution. Rather, neural computation is the result of the requirements imposed by learning to implement Bayesian inference in embodied, dynamic, cognitive systems.

8.2 Outlook

In my dissertation I raised a number of questions which I have yet to answer, and there are a number of ways I hope to extend this work in the future. In particular, I derived a number of results concerning the efficiency of rectified deep harmoniums in subsection 4.2.1. In spite of the potential indicated by these models, I have yet to validate the algorithms I presented in subsection 4.2.2 for fitting and rectifying deep harmonium models. The reason for this is simply that validating models of this complexity can take months, if not years of work. Nevertheless, in section 7.2 I provided an initial validation of an algorithm for fitting and rectifying a two-layer harmonium, which is a key component of the algorithm I developed for deep harmoniums.

Neural adaptation is the change over time in the statistics of how sensory neurons respond to stimuli (Kohn, 2007; Wark et al., 2007). Neural adaptation is a diverse phenomenon, covering effects in many parts of the brain (Kohn, 2007; Symonds et al., 2017), on different timescales (Wark et al., 2009; Zavitz et al., 2016), and at the single neuron and population level (Benucci et al., 2013; Solomon and Kohn, 2014). One application of my theory of rectification is in modelling how a neural population could adapt to stimuli without losing its ability to implement Bayesian inference. To the best of my knowledge, there are no models which can explain this combination of adaptation and inference, and I hope to investigate this proposal further.

I also hope to extended my theory of learning to implement Bayesian filtering to include continuous-time models. Much recent work has been done on how to efficiently solve Bayes filters for continuous-time processes (Susemihl et al., 2013; Harel et al., 2015; Cseke et al., 2016), but this work does not address training a recurrent neural network to approximately implement continuous-time filtering. Such a neural network would compute differential predictions which change the beliefs of an agent smoothly over time. These networks have the intuitive feature that when the neural network evaluates to 0, the beliefs of the agent do not change. However, training a continuous-time recurrent neural network requires addressing additional technical challenges which do not arise in the discrete-time case.

Applying the theory of rectification to linear probabilistic population codes led naturally to defining homogeneous LPPCs. In the context of large populations of neurons, the activity of a homogeneous LPPC can be described by a convolution of a shift-invariant tuning curve and a gain function. Moreover, any rectification parameters can be implemented by a homogeneous LPPC with a particular tuning curve with an appropriate choice of gain function. In deep learning, neural networks based on convolutions have been widely applied in image classification tasks (LeCun and Bengio, 1995; Krizhevsky et al., 2012). My theory of rectification suggests that deep, generative, convolutional models may support especially efficient learning and inference algorithms. In future work I hope to combine rectification with convolutional neural networks, towards developing a new class of highly-efficient models of natural images.

Appendix A

List of Exponential Families

In this appendix I list a number of exponential families that arise in this dissertation. For a definition of the various notation and terms, see either the Notation or Glossary sections at the beginning of the dissertation, or consult section 2.3. The definition of a given exponential family can very depending on where constants and coefficients are included in the base measure and sufficient statistic. My primary references for these definitions are Nielsen and Garcia (2009) and the Wikipedia entry on exponential families at `https://en.wikipedia.org/w/index.php?title=Exponential_family&oldid=780623893`.

Categorical Family

The family of all distributions over a finite set of elements.

$$\begin{aligned}
\Omega &= \{x_i\}_{i=1}^{n+1} \\
\mathbf{s}(x) &= (s_1(x), \ldots, s_n(x)) \\
s_i(x) &= \begin{cases} 1, & \text{if } x = x_i \\ 0, & \text{otherwise} \end{cases} \\
\mu(\{x\}) &= 1 \\
\psi(\boldsymbol{\theta}) &= \log(1 + \sum_{i=1}^{n} e^{\theta_i}) \\
\phi(\boldsymbol{\eta}) &= \sum_{i=1}^{n} \eta_i \log(\eta_i) + (1 - \sum_{i=1}^{n} \eta_i) \log(1 - \sum_{i=1}^{n} \eta_i).
\end{aligned}$$

Bernoulli Family

The family of all distributions over 0 and 1.

$$\begin{aligned}
\Omega &= \{0, 1\} \\
s(x) &= x \\
\mu(\{x\}) &= 1
\end{aligned}$$

$$\begin{aligned}\psi(\theta) &= \log(1 + e^{\theta}) \\ \phi(\eta) &= \eta \log(\eta) + (1-\eta)\log(1-\eta).\end{aligned}$$

Poisson Family

A family of single-parameter distributions over the natural numbers.

$$\begin{aligned}\Omega &= \mathbb{N} \\ s(x) &= x \\ \mu(\{x\}) &= \frac{1}{x!} \\ \psi(\theta) &= e^{\theta} \\ \phi(\eta) &= \eta \log(\eta) - \eta.\end{aligned}$$

Von Mises Family

A family of unimodal distributions over rotations. The log-partition function and negative entropy of the von Mises family do not have closed-form expressions, however there are algorithms for sampling from a von Mises distribution, and so many operations with von Mises distributions can be approximated.

$$\begin{aligned}\Omega &= [-\pi, \pi] \\ \mathbf{s}(x) &= (\cos x, \sin x) \\ \mu([0,1]) &= 2\pi.\end{aligned}$$

Normal Family

The family of normal distributions over the real line.

$$\begin{aligned}\Omega &= \mathbb{R} \\ \mathbf{s}(x) &= (x, x^2) \\ \mu([0,1]) &= \frac{1}{\sqrt{2\pi}} \\ \psi(\theta_1, \theta_2) &= -\frac{\theta_1^2}{4\theta_2} - \frac{1}{2}\log(-2\theta_2) \\ \phi(\eta_1, \eta_2) &= -\frac{1}{2}\log(\eta_2 - \eta_1^2) - \frac{1}{2}.\end{aligned}$$

Multivariate Normal Family

The family of multivariate normal distributions over Euclidean space.

$$
\begin{aligned}
\Omega &= \mathbb{R}^n \\
\mathbf{s}(\mathbf{x}) &= (\mathbf{x}, \mathbf{x} \otimes \mathbf{x}) \\
\mu([0,1]^n) &= (2\pi)^{-\frac{n}{2}} \\
\psi(\boldsymbol{\theta}, \boldsymbol{\Theta}) &= -\frac{1}{4}\boldsymbol{\theta} \cdot \boldsymbol{\Theta}^{-1} \cdot \boldsymbol{\theta} - \frac{1}{2}\log(-2|\boldsymbol{\Theta}|) \\
\phi(\boldsymbol{\eta}, \mathbf{H}) &= -\frac{1}{2}(1 + \boldsymbol{\eta} \cdot \mathbf{H} \cdot \boldsymbol{\eta} + \log|\mathbf{H}|).
\end{aligned}
$$

Bibliography

Ackley, D. H., G. E. Hinton, and T. J. Sejnowski (1985). A learning algorithm for Boltzmann machines. *Cognitive science 9*(1), 147–169.

Altun, Y., A. J. Smola, and T. Hofmann (2004). Exponential families for conditional random fields. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 2–9. AUAI Press.

Amari, S.-i. and H. Nagaoka (2007). *Methods of Information Geometry*, Volume 191. American Mathematical Soc.

Arnold, B. C., E. Castillo, and J. M. Sarabia (2001). Conditionally specified distributions: An introduction (with comments and a rejoinder by the authors). *Statistical Science 16*(3), 249–274.

Arnold, B. C. and S. J. Press (1989). Compatible conditional distributions. *Journal of the American Statistical Association 84*(405), 152–156.

Athreya, K. B. and S. N. Lahiri (2006). *Measure Theory and Probability Theory (Springer Texts in Statistics).* Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Beck, J., P. Latham, and A. Pouget (2011). Marginalization in Neural Circuits with Divisive Normalization. *The Journal of Neuroscience 31*(43), 15310–15319.

Beck, J., W. J. Ma, R. Kiani, T. Hanks, A. Churchland, J. Roitman, M. Shadlen, P. Latham, and A. Pouget (2008, December). Probabilistic Population Codes for Bayesian Decision Making. *Neuron 60*(6), 1142–1152.

Beck, J. M. and A. Pouget (2007). Exact inferences in a neural implementation of a hidden Markov model. *Neural computation 19*(5), 1344–1361.

Bengio, Y., P. Simard, and P. Frasconi (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks 5*(2), 157–166.

Benucci, A., A. B. Saleem, and M. Carandini (2013, June). Adaptation maintains population homeostasis in primary visual cortex. *Nature Neuroscience 16*(6), 724–729.

Bernigau, H. (2015, April). Causal Models over Infinite Graphs and their Application to the Sensorimotor Loop.

Besag, J. (1974). Spatial Interaction and the Statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society. Series B (Methodological) 36*(2), 192–236.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pp. 177–186. Springer.

Boulanger-Lewandowski, N., Y. Bengio, and P. Vincent (2012). Modeling Temporal Dependencies in High-dimensional Sequences: Application to Polyphonic Music Generation and Transcription. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, USA, pp. 1881–1888. Omnipress.

Boyd, S. P. and L. Vandenberghe (2004). *Convex Optimization.* Cambridge, UK; New York: Cambridge University Press.

Casella, G. and E. I. George (1992, August). Explaining the Gibbs Sampler. *The American Statistician 46*(3), 167.

Chang, S. W. C. and L. H. Snyder (2010, April). Idiosyncratic and systematic aspects of spatial representations in the macaque parietal cortex. *Proceedings of the National Academy of Sciences 107*(17), 7951–7956.

Coen-Cagli, R., A. Kohn, and O. Schwartz (2015, October). Flexible gating of contextual influences in natural vision. *Nature Neuroscience 18*(11), 1648–1655.

Cseke, B., D. Schnoerr, M. Opper, and G. Sanguinetti (2016). Expectation propagation for continuous time stochastic processes. *Journal of Physics A: Mathematical and Theoretical 49*(49), 494002.

Deitmar, A. and S. Echterhoff (2014). *Principles of Harmonic Analysis.* Universitext. Cham: Springer International Publishing.

Doya, K. (2007). *Bayesian Brain: Probabilistic Approaches to Neural Coding.* MIT press.

Ernst, M. O. and M. S. Banks (2002, January). Humans integrate visual and haptic information in a statistically optimal fashion. *Nature 415*(6870), 429–433.

Fischer, B. J. and J. L. Peña (2011, August). Owl's behavior and neural representation predicted by Bayesian inference. *Nature Neuroscience 14*(8), 1061–1066.

Franklin, D. W. and D. M. Wolpert (2011). Computational mechanisms of sensorimotor control. *Neuron 72*(3), 425–442.

Friston, K. (2003, November). Learning and inference in the brain. *Neural Networks 16*(9), 1325–1352.

Friston, K. (2010, January). The free-energy principle: A unified brain theory? *Nature Reviews Neuroscience 11*(2), 127–138.

Friston, K. J., J. Daunizeau, J. Kilner, and S. J. Kiebel (2010, February). Action and behavior: A free-energy formulation. *Biological Cybernetics 102*(3), 227–260.

Funamizu, A., B. Kuhn, and K. Doya (2016, December). Neural substrate of dynamic Bayesian inference in the cerebral cortex. *Nature Neuroscience 19*(12), 1682–1689.

Ganguli, D. and E. P. Simoncelli (2014, October). Efficient Sensory Encoding and Bayesian Inference with Heterogeneous Neural Populations. *Neural Computation 26*(10), 2103–2134.

Geman, S. and D. Geman (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* (6), 721–741.

Gerstner, W. and W. M. Kistler (2002, August). *Spiking Neuron Models: Single Neurons, Populations, Plasticity.* Cambridge University Press.

Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. *Nature 521*(7553), 452–459.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680.

Graf, A. B. A., A. Kohn, M. Jazayeri, and J. A. Movshon (2011, February). Decoding the activity of neuronal populations in macaque primary visual cortex. *Nature Neuroscience 14*(2), 239–245.

Harel, Y., R. Meir, and M. Opper (2015). A Tractable Approximation to Optimal Point Process Filtering: Application to Neural Encoding. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.), *Advances in Neural Information Processing Systems 28*, pp. 1603–1611. Curran Associates, Inc.

Herzfeld, D. J., Y. Kojima, R. Soetedjo, and R. Shadmehr (2015). Encoding of action by the Purkinje cells of the cerebellum. *Nature 526*(7573), 439–442.

Hinton, G. E. (1989). Connectionist learning procedures. *Artificial intelligence 40*(1), 185–234.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural computation 14*(8), 1771–1800.

Hinton, G. E., P. Dayan, B. J. Frey, and R. M. Neal (1995). The" wake-sleep" algorithm for unsupervised neural networks. *SCIENCE-NEW YORK THEN WASHINGTON-*, 1158–1158.

Hinton, G. E., S. Osindero, and Y. W. Teh (2006). A fast learning algorithm for deep belief nets. *Neural computation 18*(7), 1527–1554.

Hornik, K. (1993, January). Some new results on neural network approximation. *Neural Networks 6*(8), 1069–1072.

Hwang, E. J., O. Donchin, M. A. Smith, and R. Shadmehr (2003, November). A Gain-Field Encoding of Limb Position and Velocity in the Internal Model of Arm Dynamics. *PLOS Biol 1*(2), e25.

Jaynes, E. T. (2003). *Probability Theory: The Logic of Science.* Cambridge university press.

Jazayeri, M. and J. A. Movshon (2006, May). Optimal representation of sensory information by neural populations. *Nature Neuroscience; New York 9*(5), 690–6.

Jost, J. (2014). *Mathematical Methods in Biology and Neurobiology.* Universitext. London: Springer London.

Kappen, H. J., V. Gómez, and M. Opper (2012, February). Optimal control as a graphical model inference problem. *Machine Learning 87*(2), 159–182.

Kawato, M., T. Kuroda, H. Imamizu, E. Nakano, S. Miyauchi, and T. Yoshioka (2003). Internal forward models in the cerebellum: fMRI study on grip force and load force coupling. *Progress in brain research 142*, 171–188.

Kersten, D., P. Mamassian, and A. Yuille (2004, February). Object Perception as Bayesian Inference. *Annual Review of Psychology 55*(1), 271–304.

Kingma, D. and J. Ba (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Knill, D. C. and A. Pouget (2004). The Bayesian brain: The role of uncertainty in neural coding and computation. *TRENDS in Neurosciences 27*(12), 712–719.

Knill, D. C. and W. Richards (1996). *Perception as Bayesian Inference.* Cambridge University Press.

Kohn, A. (2007, March). Visual Adaptation: Physiology, Mechanisms, and Functional Benefits. *Journal of Neurophysiology 97*(5), 3155–3164.

Kollar, D. and N. Friedman (2009). *Probabilistic Graphical Models: Principles and Techniques.* The MIT Press.

Koopman, B. O. (1936). On distributions admitting a sufficient statistic. *Transactions of the American Mathematical society 39*(3), 399–409.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc.

Kullback, S. and R. A. Leibler (1951). On information and sufficiency. *The annals of mathematical statistics 22*(1), 79–86.

Le Roux, N. and Y. Bengio (2008). Representational power of restricted Boltzmann machines and deep belief networks. *Neural computation 20*(6), 1631–1649.

LeCun, Y. and Y. Bengio (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks 3361.*

LeCun, Y., Y. Bengio, and G. Hinton (2015). Deep learning. *Nature 521*(7553), 436–444.

Ma, W. J., J. Beck, P. Latham, and A. Pouget (2006, October). Bayesian inference with probabilistic population codes. *Nature Neuroscience 9*(11), 1432–1438.

Makin, J. G., B. K. Dichter, and P. N. Sabes (2015, November). Learning to Estimate Dynamical State with Probabilistic Population Codes. *PLoS Comput Biol 11*(11), e1004554.

Makin, J. G., B. K. Dichter, and P. N. Sabes (2016, May). Recurrent Exponential-Family Harmoniums without Backprop-Through-Time. *arXiv:1605.05799 [cs, stat]*.

Matsubara, T., V. Gómez, and H. J. Kappen (2014, June). Latent Kullback Leibler Control for Continuous-State Systems using Probabilistic Graphical Models. *arXiv:1406.0993 [cs]*.

McCullagh, P. (2002). What Is a Statistical Model? *The Annals of Statistics 30*(5), 1225–1267.

McGeer, T. (1990). Passive dynamic walking. *I. J. Robotic Res. 9*(2), 62–82.

McNaughton, B. L., F. P. Battaglia, O. Jensen, E. I. Moser, and M.-B. Moser (2006, August). Path integration and the neural basis of the 'cognitive map'. *Nature Reviews Neuroscience 7*(8), 663–678.

Meyn, S. P. and R. L. Tweedie (2009). *Markov Chains and Stochastic Stability.* Cambridge university press.

Mizuseki, K. and G. Buzsáki (2013, September). Preconfigured, Skewed Distribution of Firing Rates in the Hippocampus and Entorhinal Cortex. *Cell Reports 4*(5), 1010–1021.

Montúfar, G. and N. Ay (2011, February). Refinements of Universal Approximation Results for Deep Belief Networks and Restricted Boltzmann Machines. *Neural Computation 23*(5), 1306–1319.

Montúfar, G., K. Ghazi-Zahedi, and N. Ay (2015, September). A Theory of Cheap Control in Embodied Systems. *PLOS Computational Biology 11*(9), e1004427.

Montúfar, G. and J. Morton (2015a, November). Dimension of Marginals of Kronecker Product Models. *arXiv:1511.03570 [cs, math, stat]*.

Montúfar, G. and J. Morton (2015b, January). When Does a Mixture of Products Contain a Product of Mixtures? *SIAM Journal on Discrete Mathematics 29*(1), 321–347.

Montúfar, G. and J. Rauh (2016, September). Hierarchical models as marginals of hierarchical models. *International Journal of Approximate Reasoning*.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective.* MIT press.

Nair, V. and G. E. Hinton (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp. 807–814.

Neal, R. M. (2012). *Bayesian Learning for Neural Networks*, Volume 118. Springer Science & Business Media.

Neal, R. M. and G. E. Hinton (1998). A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pp. 355–368. Springer.

Nielsen, F. and V. Garcia (2009, November). Statistical exponential families: A digest with flash cards. *arXiv:0911.4863 [cs]*.

Ostojic, S. and N. Brunel (2011, January). From Spiking Neuron Models to Linear-Nonlinear Models. *PLoS Computational Biology 7*(1), e1001056.

Paninski, L. (2004, November). Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems 15*(4), 243–262.

Paninski, L., S. Shoham, M. R. Fellows, N. G. Hatsopoulos, and J. P. Donoghue (2004). Superlinear population encoding of dynamic hand trajectory in primary motor cortex. *The Journal of neuroscience 24*(39), 8551–8561.

Pascanu, R., T. Mikolov, and Y. Bengio (2013). On the difficulty of training recurrent neural networks. *ICML (3) 28*, 1310–1318.

Pfeifer, R. and G. Gómez (2009). Morphological computation–connecting brain, body, and environment. *Creating Brain-Like Intelligence*, 66–83.

Plesser, H. E. and W. Gerstner (2000, February). Noise in Integrate-and-Fire Neurons: From Stochastic Input to Escape Rates. *Neural Computation 12*(2), 367–384.

Pouget, A., J. Beck, W. J. Ma, and P. Latham (2013). Probabilistic brains: Knowns and unknowns. *Nature Neuroscience 16*(9), 1170–1178.

Pouget, A., P. Dayan, and R. Zemel (2000). Information processing with population codes. *Nature Reviews Neuroscience 1*(2), 125–132.

Pouget, A. and T. J. Sejnowski (1995). Spatial Representations in the Parietal Cortex May Use Basis Functions. In G. Tesauro, D. S. Touretzky, and T. K. Leen (Eds.), *Advances in Neural Information Processing Systems 7*, pp. 157–164. MIT Press.

Radford, A., L. Metz, and S. Chintala (2015, November). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv:1511.06434 [cs]*.

Rawlik, K., M. Toussaint, and S. Vijayakumar (2012). Path Integral Control by Reproducing Kernel Hilbert Space Embedding. *Arxiv preprint arXiv:1208.2523*.

Robert, C. P. and G. Casella (2004). *Monte Carlo Statistical Methods*. Springer Texts in Statistics. New York, NY: Springer New York.

Roe, A. W., L. Chelazzi, C. E. Connor, B. R. Conway, I. Fujita, J. L. Gallant, H. Lu, and W. Vanduffel (2012, April). Toward a Unified Theory of Visual Area V4. *Neuron 74*(1), 12–29.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986, October). Learning representations by back-propagating errors. *Nature 323*(6088), 533–536.

Salakhutdinov, R. (2015, April). Learning Deep Generative Models. *Annual Review of Statistics and Its Application 2*(1), 361–385.

Salakhutdinov, R. and G. Hinton (2009). Deep boltzmann machines. In *Artificial Intelligence and Statistics*, pp. 448–455.

Salakhutdinov, R. and G. Hinton (2012, April). An Efficient Learning Procedure for Deep Boltzmann Machines. *Neural Computation 24*(8), 1967–2006.

Salinas, E. and P. Thier (2000). Gain modulation: A major computational principle of the central nervous system. *Neuron 27*(1), 15–21.

Särkkä, S. (2013). *Bayesian Filtering and Smoothing.* Number 3. Cambridge University Press.

Shore, J. and R. W. Johnson (1980). Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. *Information Theory, IEEE Transactions on 26*(1), 26–37.

Simoncelli, E. P., L. Paninski, J. Pillow, and O. Schwartz (2004). Characterization of neural responses with stochastic stimuli. *The cognitive neurosciences 3*, 327–338.

Smolensky, P. (1986, February). Information Processing in Dynamical Systems: Foundations of Harmony Theory. Technical report.

Snyder, D. L. (1972). Filtering and detection for doubly stochastic Poisson processes. *Information Theory, IEEE Transactions on 18*(1), 91–102.

Sokoloski, S. (2017, June). Implementing a Bayes Filter in a Neural Circuit: The Case of Unknown Stimulus Dynamics. *Neural Computation 29*(9), 2450–2490.

Solomon, S. G. and A. Kohn (2014, October). Moving Sensory Adaptation beyond Suppressive Effects in Single Neurons. *Current Biology 24*(20), R1012–R1022.

Sterling, P. and S. Laughlin (2015). *Principles of Neural Design.* MIT Press.

Susemihl, A., R. Meir, and M. Opper (2013, March). Dynamic state estimation based on Poisson spike trains—towards a theory of optimal encoding. *Journal of Statistical Mechanics: Theory and Experiment 2013*(03), P03009.

Susemihl, A., R. Meir, and M. Opper (2014, June). Optimal Population Codes for Control and Estimation. *arXiv:1406.7179 [cs, math, q-bio, stat].*

Sutskever, I., G. E. Hinton, and G. W. Taylor (2009). The Recurrent Temporal Restricted Boltzmann Machine. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou (Eds.), *Advances in Neural Information Processing Systems 21*, pp. 1601–1608. Curran Associates, Inc.

Symonds, R. M., W. W. Lee, A. Kohn, O. Schwartz, S. Witkowski, and E. S. Sussman (2017, January). Distinguishing Neural Adaptation and Predictive Coding Hypotheses in Auditory Change Detection. *Brain Topography 30*(1), 136–148.

Talbott, W. (2015). Bayesian Epistemology. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Summer 2015 ed.).

Tansey, W., O. H. M. Padilla, A. S. Suggala, and P. Ravikumar (2015, June). Vector-Space Markov Random Fields via Exponential Families. In *PMLR*, pp. 684–692.

Theodorou, E. A. and E. Todorov (2012). *Relative Entropy and Free Energy Dualities: Connections to Path Integral and KL Control*. Submitted.

Thrun, S., W. Burgard, and D. Fox (2005). *Probabilistic Robotics*. MIT press.

Todorov, E. (2010). Policy gradients in linearly-solvable mdps. In *Advances in Neural Information Processing Systems*, pp. 2298–2306.

Toussaint, M. (2009). Probabilistic inference as a model of planned behavior. *Künstliche Intelligenz 3*(9), 23–29.

Toussaint, M. and C. Goerick (2010). A bayesian view on motor control and planning. In *From Motor Learning to Interaction Learning in Robots*, pp. 227–252. Springer.

Wainwright, M. J. and M. I. Jordan (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning 1*(1-2), 1–305.

Wark, B., A. Fairhall, and F. Rieke (2009, March). Timescales of Inference in Visual Adaptation. *Neuron 61*(5), 750–761.

Wark, B., B. N. Lundstrom, and A. Fairhall (2007, August). Sensory adaptation. *Current Opinion in Neurobiology 17*(4), 423–429.

Wei, X.-X. and A. A. Stocker (2015, September). A Bayesian observer model constrained by efficient coding can explain 'anti-Bayesian' percepts. *Nature Neuroscience 18*(10), 1509–1517.

Welling, M., M. Rosen-zvi, and G. E. Hinton (2005). Exponential Family Harmoniums with an Application to Information Retrieval. In L. K. Saul, Y. Weiss, and L. Bottou (Eds.), *Advances in Neural Information Processing Systems 17*, pp. 1481–1488. MIT Press.

Werbos, P. J. (1990). Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE 78*(10), 1550–1560.

Williams, R. J. and D. Zipser (1989, June). A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation 1*(2), 270–280.

Yang, E., P. Ravikumar, G. I. Allen, and Z. Liu (2013). Conditional Random Fields via Univariate Exponential Families. In *Advances in Neural Information Processing Systems*, pp. 683–691.

Yang, E., P. Ravikumar, G. I. Allen, and Z. Liu (2015). Graphical models via univariate exponential family distributions. *Journal of Machine Learning Research 16*(1), 3813–3847.

Yuille, A. and D. Kersten (2006, July). Vision as Bayesian inference: Analysis by synthesis? *Trends in Cognitive Sciences 10*(7), 301–308.

Zahedi, K. and N. Ay (2013). Quantifying Morphological Computation. *Entropy 15*(5), 1887–1915.

Zavitz, E., H.-H. Yu, E. G. Rowe, M. G. P. Rosa, and N. S. C. Price (2016, April). Rapid Adaptation Induces Persistent Biases in Population Codes for Visual Motion. *Journal of Neuroscience 36*(16), 4579–4590.

Zemel, R. S., P. Dayan, and A. Pouget (1998). Probabilistic interpretation of population codes. *Neural computation 10*(2), 403–430.

Zipser, D. and R. A. Andersen (1988, February). A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons. *Nature 331*(6158), 679–684.

Selbstständigkeitserklärung

Hiermit erkläre ich, die vorliegende Dissertation selbständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialien oder erbrachten Dienstleistungen als solche gekennzeichnet.

New York City, June 25, 2019

....
(Sacha Sokoloski)