Swansea University
Prifysgol Abertawe

THEORETICAL PHYSICS MSC BY RESEARCH

A sufficient condition for the improvement of Restricted Boltzmann Machines

Author name:
Mark Thomas

First supervisor:
Prof. Gert AARTS
Second supervisor:
Prof. Biagio LUCINI

A report submitted in partial fulfillment of the requirements for the degree of Theoretical Physics MSc by Research

in the

Particle Physics and Cosmology Theory Group
School of Biosciences, Geography and Physics

December 11, 2023
Word count: 39,694

Declarations

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed.....[redacted].........

Date......18/12/23......

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed.....[redacted].........................

Date......18/12/23......

I hereby give consent for my thesis, if accepted, to be available for electronic sharing

Signed.....[redacted]

Date......18/12/23......

The University's ethical procedures have been followed and, where appropriate, that ethical approval has been granted.

Signed...[redacted]

Date......18/12/23......

Acknowledgements

I am immensely grateful to my parents for their unwavering support, love, and encouragement throughout my academic journey. Their belief in me and constant encouragement have been instrumental in my accomplishments. I want to express my heartfelt gratitude to my love, Lauren, for her patience, understanding, and support. Undertaking this research alongside a full-time job has been challenging, but her unwavering support has been my anchor.

A special thank you goes to my good friends, Kieran Gay, Kelly Richards, Tim Klee, Suhail Mall, Richard Thompson, Ben Romero-Wilcock, Robert King, Daniel Torren Peraire, Khodr Badih, James Mackie, Matthew Zhang, and Endri Sefgjinaj for providing much-needed downtime and moments of relaxation during the research process. You have all provided valuable encouragement and support through my academic journey.

I am deeply indebted to my two supervisors, Gert Aarts and Biagio Lucini, for their expertise, guidance, and mentorship. Their invaluable feedback and constructive criticism have significantly shaped this thesis and my growth as a researcher.

To all those who have played a part, big or small, in supporting me during this journey, I am genuinely grateful. Your belief in me has been a driving force, and I am honored to have such a fantastic network of support.

Thank you.
Mark Thomas

SWANSEA UNIVERSITY

Abstract

Faculty of Science and Engineering
School of Biosciences, Geography and Physics

Theoretical Physics MSc by Research

A sufficient condition for the improvement of Restricted Boltzmann Machines

Mark Thomas
([redacted])

This thesis explores Restricted Boltzmann Machines (RBMs) and their training, focusing on the minimization of the Kullback-Leibler (KL) divergence. Neural networks and the importance of the KL divergence are introduced and motivated. Examples of KL divergence calculations are demonstrated for various model and target distributions. A demonstration of the non-universality of the ability to improve models by introducing a new parameter without re-training the existing ones is made. The Ising model is explored as an example of available training data, and the work of *G. Cossu et al., 'Machine learning determination of dynamical parameters: The Ising model case,' Phys. Rev. B, 100, 064304 (2019)* in training a set of RBMs on the one-dimensional Ising model is successfully reproduced. Connections between the mathematics of RBMs and lattice Quantum Field Theory (QFT) are explored, and insights from QFT are utilized to inform the design choices of RBMs to consider. Leveraging these insights, a linearisation procedure is employed to produce a sufficient condition for the possibility of improvement of an RBM with bilinear inter-layer mixing and a Gaussian hidden layer through the introduction of new parameters, without the need to re-train already-existing parameters. This condition is tested and potential issues with the linearisation procedure performed are highlighted.

Contents

Chapter 1

Introduction and prerequisites

"There is now a broad consensus that AI research is progressing steadily, and that its impact on society is likely to increase."

-Stephen Hawking

This chapter opens the thesis by introducing the reader to the context of the research, as well as the essential mathematical tools required to understand its significance. This begins in section 1.1, where light introductions to the broad ideas of machine learning and neural networks are made in order to lay down a fundamental understanding of the topics. This will set the stage for a subsequent discussion in section 1.2 regarding *Restricted Boltzmann Machines*, which are a type of neural network, and are the focus of this research.

Section 1.3 follows this by motivating and portraying the significance of the use of a "metric" known as the Kullback-Leibler divergence in benchmarking the performance of Restricted Boltzmann Machines. This measure serves as the primary tool used to compare different Restricted Boltzmann Machines in this research, and so to gain crucial insight into the working of the Kullback-Leibler divergence, section 1.4 builds on the definitions by comparing the performance of pairs of models by use of this tool.

Once the foundations of the Restricted Boltzmann Machine and the Kullback-Leibler divergence have been laid down, section 1.6 highlights particular design choices in both constructing and studying Restricted Boltzmann Machines motivated by Quantum Field Theory and Statistical Mechanics. Finally, once familiarity with the objects of study and the reasons for certain choices has been developed, section 1.7 summarises these discussions and sets out the structure of the rest of the thesis, given the discussed context.

1.1 Introduction to Machine Learning and Neural Networks

This section provides a comprehensive overview of the necessary concepts of Machine Learning and Neural Networks. Subsection 1.1.1 provides an introduction to the notion of Machine Learning and showcases examples of successful applications of Machine Learning in academic and industrial contexts. The importance of learning from data and making predictions without the use of explicit domain-specific algorithms is discussed. This is followed by subsection

1.1.2, which discusses learning as a process of optimizing performance toward a specific task. Gradient descent is introduced as one of the most popular machine learning algorithms, as well as a small number of other popular examples. Subsection 1.1.3 follows on from this by detailing how a gradient descent algorithm can be implemented on *neural nets*. The context for choosing structures like neural nets to make predictions from previously seen data is provided, the idea of an activation function is discussed, and popular examples of activation functions are highlighted. Finally, subsection 1.1.4 introduces the concept of a *deep* neural network and discusses both their strengths and drawbacks, as well as why they will not be used in the context of this research.

1.1.1 Introduction to Machine Learning

Machine Learning (ML) is a field of study focusing on developing models and algorithms capable of learning from data to make decisions or predictions without explicit programming. It is a sub-field of Artificial Intelligence (AI), a tremendously popular and important topic that has grown explosively in recent years due to increased public awareness and available computational power. Machine Learning sets itself apart from traditional programming approaches by enabling computers to automatically learn and improve from experience and new data, rather than relying on explicit instructions or rules provided by human programmers and engineers. Machine Learning algorithms are typically segregated into three main categories: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning is the process of training a model on labeled data. The model learns to associate patterns from input data with the corresponding labels, allowing it to make predictions regarding the correct labeling of new, previously unseen data. Familiar examples of successful supervised learning applications include spam email detection [1], medical diagnoses [2], and credit scoring [3]. All of these look at human-labeled examples of relevant data and use detectable patterns to judge new data.

On the other hand, unsupervised learning deals with unlabeled data. The goal in these contexts is to identify meaningful patterns in data that do not contain predefined labels. Unsupervised learning is usually used in contexts such as anomaly detection, data clustering, or dimensionality reduction. Some familiar examples include the detection of credit card theft [4], identifying customer shopping behaviour [5], and topic identification in speech [6]. The recent release of OpenAi's ChatGPT, an extremely successful chatbot that took less than two months to reach 100 million users [7], used a mixture of both human-supervised and unsupervised learning in its development.

Reinforcement learning is a learning method where an agent actively interacts with an environment. It then learns by taking actions and receiving stimuli following those actions that contribute either positively or negatively toward some reward signal. Through trial and error, the agent learns to maximise the correct actions by adjusting its behaviour to maximize rewards, achieving increased performance over time. Two successful examples of applications of reinforcement learning include Boston Dyanamics' general-purpose agile robot 'Spot' [8] and AlphaGo, the first computer program to successfully beat a human player at the board game Go [9].

Machine Learning has been applied extensively across both academic and industrial domains. In academia, it has significantly contributed to fields such as physics, natural language processing, and Genomics. It has played a transformative role in data analysis, modelling, and simulations. Within experimental physics, machine learning has been applied to large data sets produced by particle accelerators, such as the LHC. Examples of these uses include separating signal and background noise [10], classifying particle interactions [11], and improving the performance of real-time filtering [12]. Another example is in astronomical data, where meaningful patterns have been extracted from the vast Sloan Digital Sky Survey database [13]. In theoretical physics, machine learning methods have been used in cases such as condensed matter physics, where techniques were used to predict properties of new materials, such as their electrical conductivity and magnetic behaviour [14]. There is a plethora of examples of these machine learning algorithms being used to simulate quantum and complex systems. It has also been instrumental in computational physics, where researchers have improved the accuracy of and decreased the time required to perform various calculations by combining machine learning with traditional algorithms [15].

By training an algorithm to learn from data and make predictions without explicit programming, those using the trained algorithm can extract valuable knowledge and patterns from vast amounts of data that may not be apparent via manual analysis. By leveraging ML techniques, hidden relationships can be uncovered, and insights that will significantly enhance decision-making processes can be made. Machine Learning techniques scale well for automation and can reliably handle complexity, saving time and resources compared to manual analysis. By harnessing the combined power of data and computational algorithms, Machine Learning is applied to extract valuable insights, make accurate and helpful predictions, and automate complex processes. Its list of productive applications continues to expand, and its impact on individuals, academic institutions, and private ventures is becoming increasingly significant.

1.1.2 Objective Functions and Optimization

The optimisation of some objective function, also known as a loss function or cost function, plays a crucial role in machine learning, providing a means of quantifying the performance of models. An objective function is a measure or score representing how well a model performs a specific task. Optimisation algorithms aim to find the optimal set of parameters that either minimise or maximise the objective function, where one of these extremes corresponds to optimal model performance. Optimising a particular objective function is a guiding principle for model training in machine learning, as models moving towards optimal on relevant objective functions are considered improving in their performance.

The objective function used in any particular application depends on the specific task at hand. One example of an objective function is binary cross-entropy (to be defined below), which measures the dissimilarity between predicted and true labels on data stemming from a classification task. By minimising this objective function, the model maximises its classification performance. Another example of a commonly used objective function is the mean squared error (MSE), usually used in regression tasks. The MSE indicates the average squared distance between correct values and those predicted by a model. By implementing a learning process that minimises this objective function, the model will minimise its MSE, corresponding to a better regression performance. Another example of an objective function is the within-cluster sum of squares function used in data clustering. This objective function indicates the compactness of data clusters by measuring the average squared distance of data points within that cluster to the cluster centroid. Minimising this objective function leads to cohesive clusters that are well-separated.

Gradient descent, Markov chain Monte Carlo, and simulated annealing are just a few of many popular optimisation algorithms commonly employed to find the optimal parameter set to maximise or minimise some objective function. These algorithms all update a model's parameters iteratively to converge toward an optimal solution gradually. Gradient descent is known as a first-order optimisation algorithm because it considers the first derivative of the objective function. More advanced optimisation algorithms can provide faster convergence to an optimal solution by considering higher-order derivatives; however, they require additional computational resources.

1.1.3 Neural Networks

Neural networks are a powerful class of machine learning models that have gained significant attention in both the academic and public eye. Before diving into neural networks, it is important to step back and acknowledge that the term "machine learning" encompasses a broader range of algorithms and techniques than just neural networks. To name just a few of a diverse set of tools, other examples of machine learning implementations include random forests, support vector machines, and k-nearest neighbours. Each separate implementation type has unique strengths and weaknesses, meaning that the suitability of each depends on the context in which they are being applied. It is important to note that neural networks are only sometimes the optimal choice for a particular machine learning problem. Other machine learning implementations might be more applicable in various circumstances, depending on the availability of labelled data, computational resources, and implementation time.

Neural networks consistently stand out in their ability to solve complex problems with non-linear relationships between input and output, such as in the domains of natural language processing [16], speech recognition [17], and electrical energy demand forecasting [18]. They typically have appealing generalisation properties, where they can make accurate predictions regarding unseen data. They are inspired by the structure and function of the brain, composed of connected nodes called neurons. These neurons are organised into layers, usually referred to as hidden and visible layers (which will be defined in 1.2.2). Each layer processes data by transforming what is input into an output dependent on the parameters of the neural net and the input data. The output of each neural net layer is then passed on as an input to the next until a final layer is reached, at which point the output is interpreted. This flow of information sequentially through the neural network allows it to learn complex relationships within the data by structuring how it organises and processes data hierarchically; this is represented in figure 1.1.

Neural networks learn data through the process of training. This is where the internal parameters, known as weights and biases, are iteratively adjusted to minimise a particular

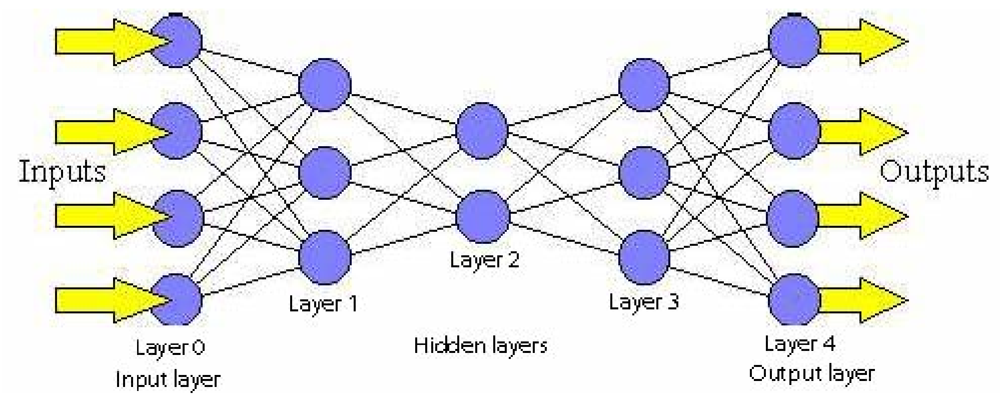


FIGURE 1.1: *Illustration of a typical neural network, taken from [19]. Information flows through the network by being passed via each layer of neurons sequentially.*

objective function. Gradient descent is a popular choice of a numerical algorithm to minimise an objective function for neural networks; the gradient of the objective function with respect to each of the weights and biases is calculated, followed by an updating of these parameters in the direction of steepest descent. Training a neural network involves processes known as forward propagation and back propagation. Forward propagation is where data is passed through each network layer to generate an output, and back propagation is where the error between the predicted output and the actual target output is calculated and used to update the network's parameters in the opposite direction of the gradient. Once this is done, the gradients of the objective function with respect to each of the internal parameters are computed by propagating the error produced at the output back up each layer from the output to the input. The calculated gradients are then used to update the weights in such a way as to minimise the objective function for the seen data. This process is repeated with new data and proceeds until the network reaches a satisfactory level of performance or stagnates in its ability to increase performance.

Key to the operation of a neural network is the activation function, which is the strength by which a node in a neural net passes on a signal it receives. The activation function is a mathematical function inspired as an abstraction of the *action potential* of biological neurons [20]. In that context, the activation function $f(\mathbf{x})$ for a neuron N represents the binary output of either firing, typically where it is taken that $f(\mathbf{x}) = 1$, or not firing, where $f(\mathbf{x}) = 0$. Here, $\mathbf{x}$ is a vector of values representing states of other neurons that have an input into the neuron N. The activation function $f(\mathbf{x})$ is typically taken to perform a weighted sum over the entries of $\mathbf{x}$. For example, the Heaviside activation function representing a neuron with n inputs is

$$f(\mathbf{x}) = \begin{cases} 0 & \text{if } a + \mathbf{x}\cdot\mathbf{b} < 0 \\ 1 & \text{if } a + \mathbf{x}\cdot\mathbf{b} \geq 0 \end{cases}, \quad \text{where } a \in \mathbb{R},\ \mathbf{x} \in \{0,\,1\}^n,\ \mathbf{b} \in \mathbb{R}^n, \tag{1.1}$$

where a represents the neuron *weight* and the entries of $\mathbf{b}$ represent the *biases* of the neuron. A neuron with such an activation function will fire when $\mathbf{x}$ takes on specific configurations. Note here that the weight(s) and biases of a neuron heavily influence whether a neuron will fire for a given input; consequently, the output of a neural network can be modified by tuning the weights and biases in the network. Activation functions introduce non-linearity between features in the network, enabling it to learn the complex relationships between input and output. Without the use of an activation function, a neural network would simply apply a linear transformation to input data in order to produce an output, which would severely restrict their capability to model phenomena.

Many choices of activation functions are used within the science of neural networks, each with its own characteristics and suitability for various tasks. A popular choice of activation is the sigmoid function

$$f(\mathbf{x}) = \sigma(a + \mathbf{x} \cdot \mathbf{b}) = \frac{1}{1 + e^{-(a+\mathbf{x}\cdot\mathbf{b})}}, \tag{1.2}$$

which maps any input $\mathbf{x} \in \mathbb{R}^n$ to an output $y \in [0, 1]$. It is often used in classification problems, where the output of a node corresponds to the probability that the input data corresponds to a certain data class. There are benefits and drawbacks to choosing the sigmoid function as an activation function for nodes in a neural network. To mention a few, the function output is smooth and bounded between 0 and 1, which is helpful for binary classification. It is a differentiable function and lends itself well to interpretation. The boundedness can, however, act as a hindrance on some occasions. Its vanishing gradients near extremes can also slow down the learning process. The sigmoid function mimics the function of neurons in a biological brain; neurons either activate ($\sigma = 1$) or do not ($\sigma = 0$) and do so with a trained probability. The probability of each neuron in a brain passing on a signal then depends on the weights and biases of that neuron. A second popular activation function is the rectified linear unit

$$\mathrm{ReLU}(x) = \max(0, x). \tag{1.3}$$

This is a piecewise linear function making it simple to calculate, but also introduces the necessary non-linearity to capture complex patterns in data whilst simultaneously addressing an issue faced by *deep* neural networks known as the vanishing gradient problem, at least for when $x > 0$.

1.1.4 Deep Neural Networks

Deep neural networks (DNNs) are complex, heavily layered versions of regular neural networks. In practice, neural networks typically only have one or two hidden layers, whereas neural networks with more layers than this are considered deep neural networks. For example, ChatGPT utilises 96 hidden layers (which OpenAi refers to as attention layers [21]). Increased depth in a neural network allows the network to learn more intricate and abstract features in the data; more complex relationships and patterns that may not be discernible with a small number of layers may become interpretable, enhancing performance. More shallow neural networks may need foresight and help to capture and represent intricate relationships within data.

Having a DNN brings many strengths that have contributed to their success in various fields. One of these is their increased ability to handle data where more of the relationships between features are high-dimensional. This makes them particularly effective in tasks such as image and speech recognition. The deep architecture allows the network to extract multiple relevant features at various levels of abstraction, enabling it to have a more comprehensive and nuanced understanding of data. Particularly within the domain of image recognition, there have been remarkable breakthroughs, including examples of where human-level performance has been surpassed in tasks such as object detection and image classification. Other applications, such as autonomous driving, drug research, and particle physics analysis, require the utility offered by deep neural networks' ability to spot patterns in high-dimensionality data.

Despite these strengths, deep neural networks also suffer from certain drawbacks and challenges. A primary challenge is the increased computational cost required to train deeper neural networks and, along with that, the time taken to perform training. Due to their additional complexity, deep neural networks require much larger amounts of training data in order to generalise well and avoid overfitting to any particular data [22]. Should the interpretation of the way the neural network processes information be required, deep neural networks lead to increased difficulty in this interpretation relative to more shallow networks due to the abstract ways the intermediate layers transform data [23].

Despite the fact that deep neural networks have garnered significant attention and success, the study of neural networks that are not deep remains an important and valuable area of research and is the focus of this thesis. Simpler neural network architectures can often provide effective solutions with fewer computational requirements. Furthermore, understanding the underlying mechanisms and principles is crucial for the advancement of the field and of the development of novel algorithms and techniques. The exploration of non-deep neutral networks permits researchers to more thoroughly investigate questions and choices, such as decisions behind the choice of network architecture, activation functions, and learning techniques.

1.2 Restricted Boltzmann Machines

This section introduces Restricted Boltzmann Machines (RBMs) as a class of neural network. Subsection 1.2.1 introduces RBMs, summarises their strengths and weaknesses, and discusses some examples of the successful use of RBMs, as well as discussing the motivation for the thesis. Subsection 1.2.2 discusses the architectural choices that classify a neural network as an RBM. Subsection 1.2.3 further discusses how tunable parameters are set up in an RBM, and the analogues of the framing of the RBM parameters to ideas in statistical physics are presented. Finally, subsection 1.2.4 briefly introduces the training algorithms used by RBMs.

1.2.1 Introduction to Restricted Boltzmann Machines

Restricted Boltzmann Machines (RBMs) are a type of neural network introduced by Paul Smolensky in the 1980s as a statistical physics model [24]. They are employed to tackle unsupervised learning tasks, where they discover patterns and interdependencies within unlabelled data. They have demonstrated their ability in data dimensionality reduction, where they have been used to extract lower dimensionality representations of high dimensional data. They are particularly effective at handling missing data, allowing for robust physical modelling in contexts where data is incomplete. They have been used in broad contexts, such as in image recognition, natural language processing, and recommendation systems. In a notable study, researchers used a Deep Restricted Boltzmann Machine with six layers to classify images from the CIFAR-10 dataset containing 60,000 color images belonging to 10 different classes [25]. The network achieved an accuracy of 87%, a significant improvement on other previously used neural networks. Another example of RBMs used is in recognising handwritten digits from the MNIST dataset, where an accuracy of 99.7% has been achieved. RBMs have been used to generate song lyrics by training them on popular songs, demonstrating their potential for creative application. Netflix uses trained RBMs to predict what users will rate certain movies, aiding them in personalised ranking on user homepages [26].

RBMs are particularly powerful in their ability to extract features from noisy data with minimal explicit feature engineering, lending to their ease of use and immediate application. They are particularly efficient to train, as there is a fast training algorithm that can be employed on them known as the Contrastive Divergence (CD) algorithm, making RBMs suitable for scenarios with heavy amounts of data. The training of an RBM explicitly handles the parameters of hidden units, removing the need for slow backpropagation. This simplifies and accelerates the training process while simultaneously reducing the probability of the neural net getting stuck in local performance optima. The additional transparency offered by explicitly tuning the hidden units allows for a better understanding of why a trained RBM treats data in a certain way, aiding in better analysis and interpretation of model outputs relative to other neural networks. Given their focus on being applied in unsupervised learning algorithms, various training algorithms other than CD can be applied to RBMs to increase training speed further; Persistent Contrastive Divergence, Hybrid Monte Carlo, and Parallel Tempering are a few examples that have been successfully used on RBMs for this reason. They are very flexible in their allowed input data type; they can handle binary, categorical, and continuous data sets without issue.

The work presented by Bachtis, Aarts, and Lucini in *Quantum field-theoretic machine learning (2021)* [27] opens up a unique and exciting avenue for research at the intersection of quantum field theory and machine learning, particularly with regards to applications of Restricted Boltzmann Machines. In the paper, the authors demonstrate that a ϕ^4 scalar field theory with inhomogeneous coupling constants satisfies the Hammersley-Clifford theorem, allowing the field theory to be recast as a machine learning algorithm in a mathematically rigorous way. Thus, scalar ϕ^4 theory, known for its applications in quantum field theory, offers a unique foundation for constructing machine learning models and provides a fresh perspective on the relationship between the two domains. The result prompts the exploration of how principles from quantum field theory can be leveraged to enhance machine learning methodologies. Furthermore, the paper extends beyond theoretical derivations. It presents a set of concrete applications of Restricted Boltzmann Machines derived from ϕ^4 theories, showcasing the versatility of conventional machine learning frameworks inspired by quantum field theory. Additionally, the paper emphasizes the results of the paper *Refinements of Universal Approximation Results for Deep Belief Networks and Restricted Boltzmann Machines (2010)* [28], which demonstrated (with conditions) the ability for RBMs to approximate distributions arbitrarily well; they are so-called *universal approximators*. Overall, Bachtis, Aarts, and Lucini emphasize that RBMs constructed from (particular) scalar field theories can be framed as machine learning algorithms that can be trained to approximate a target distribution arbitrarily well; this is taken as a primary motivation for the study of QFT-inspried RBMs moving forward.

1.2.2 Restricted Boltzmann Machine architecture

The architecture of RBMs is key to their function as unsupervised learning algorithms. They consist of two layers of nodes, known as the visible and hidden layers, and form an undirected bipartite graph; there are no visible-to-visible or hidden-to-hidden connections, and all connections are visible-to-hidden. This is represented in figure 1.2. The visible nodes represent observed variables or features of the input data, while the hidden nodes represent the hidden features and are what allow the neural network to capture the underlying patterns. The allowed values of the nodes are typically chosen given the context of the problem. For example, given a problem with binary input data, the nodes would typically be chosen to take on binary values, but with continuous input data, the nodes would be chosen to take on continuous values. RBMs are characterised by a weight matrix that captures the connections

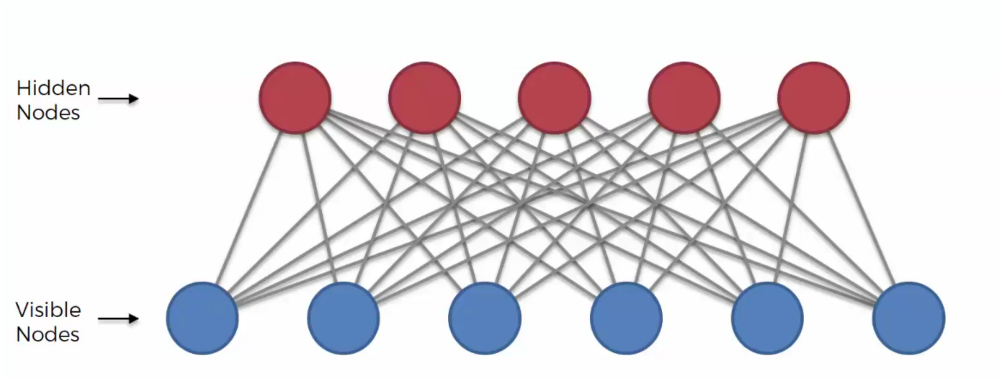


FIGURE 1.2: *Illustration of RBM architecture, taken from [29], where nodes are separated into two layers with no intra-layer connections, but with inter-layer connections between each node in the visible and hidden layers.*

between nodes in the visible and hidden layers. These weights determine the strength with which each node influences the activation probability of the nodes it is connected to. These weights are adjustable parameters tuned during the learning process in order to optimise the RBM performance. All nodes, both visible and hidden, also have a set of (usually many) parameters that shape the probability distribution for the values they take. These allow for the RBM to model the tendency of each node to be activated to a certain value irrespective of input.

1.2.3 Energy functions and node activation

The activation function of each node in an RBM is defined via an energy function, which specifies a probability for each possible configuration of all nodes in the neural network. The activation probability of individual nodes in the neural network is then derived from this energy function. The energy function is dependent on every parameter of the neural network in some way. Training of the neural network parameters translates to modifying the energy function to determine the joint configuration probabilities of the neural network.

For example, for an RBM determined by a set of weights and biases denoted collectively as θ, if the set of visible nodes take on a vector of values $\mathbf{v}$, and the hidden nodes take on the vector of values $\mathbf{h}$, then the joint probability distribution determining the probability of that exact node configuration is given by

$$p\left(\mathbf{v},\, \mathbf{h};\, \theta\right) = \frac{e^{-E\left(\mathbf{v},\, \mathbf{h};\, \theta\right)}}{Z\left(\theta\right)}, \tag{1.4}$$

where here, $E\left(\mathbf{v},\, \mathbf{h};\, \theta\right)$ is the energy associated with the configuration, and $Z\left(\theta\right)$ is the partition function that normalises the distribution. In training the RBM, given the form of $E\left(\mathbf{v},\, \mathbf{h};\, \theta\right)$, the parameters θ are tuned such to cause configurations of the neural network to appear with a frequency in proportion to the appearance of the data represented by that configuration from the system being modelled.

The form of (1.4) mimics the Boltzmann distribution of statistical mechanics, where the probability of the i^{th} configuration of a physical system is given by

$$p_i = \frac{e^{-\frac{\varepsilon_i}{k_B T}}}{Z}, \tag{1.5}$$

with ε_i denoting the energy of the configuration, T the system temperature, and k_B Boltzmann's constant. In statistical physics, the probability of a configuration is then determined by how the ε_i depends on the configuration of a system, as well as the parameters of those systems (for example, the mass of gas particles). From this perspective, the goal of training an RBM then is to adjust the weights and biases determining the energy function to minimise the energy of neural net configurations representing frequently observed data whilst maximising the energy of configurations representing infrequently observed data.

A particular choice frequently made regarding the energy function is to have the weights parameterising the connection between the visible and hidden layer to be bi-linear in the node values. For a system with m visible nodes and n hidden nodes, such a choice constrains the energy function to take the form

$$E\left(\mathbf{v},\, \mathbf{h};\, \theta\right) = \sum_{j=1}^{m} U_j(v_j) + \sum_{i=1}^{n} \tilde{U}_i(h_i) + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i \omega_{ij} v_j. \tag{1.6}$$

Here, v_j is the value taken by visible node j, and h_i is the value taken by hidden node i. $U_j(v_j)$ represents the energy functions dependence on the biases of visible node j, and similarly for the hidden layer's $\tilde{U}_i(h_i)$. The mixing parameter ω_{ij} fully characterises the level of influence hidden node i and visible node j have on each other.

The choice of making the mixing between the visible and hidden layer bi-linear is primarily driven by the mathematical simplicity and computational efficiency it affords. While more complex terms could capture higher-order interactions between the two layers, the bi-linear term balances modelling capability with training practicality. An element-wise product between pairs of visible and hidden nodes does not require the definition of complex operations or the significant use of computational resources.

The mathematical simplicity of this choice also lends to more straightforward interpretations of how the hidden and visible layers will interact for given choices of U_j and $\tilde{U}_i$. This facilitates the theoretical study of the behaviour of RBMs and is of foundational importance to the research presented in this thesis. As demonstrated in 2.1.2, the bi-linear mixing term leads to a well-behaved and relatively simple optimisation process. It allows for the simple calculation of gradients leading to efficient and compact algorithms that drive the learning. Despite being simple, it introduces the necessary interactions between the hidden and visible layer, without which interconnection between features in data would be impossible, meaning the RBM would not be able to learn complex dependencies between variables in presented data.

Although they will not be addressed in this thesis, RBMs with a bi-linear mixing term can still learn high-dimensionality relationships by combining or stacking multiple RBMs to form Deep RBMs (DRBMs) or by incorporating additional fully-connected hidden layers to form Deep Belief Networks (DBNs) [30]. These more complex architectures can capture higher-level patterns in data. In DRBMs, many RBMs are usually intentionally trained on data that spans specific features, followed by stacking these RBMs to ensure that the DRBM transforms input data while respecting each feature learned by the individual RBMs.

Under such a choice for the mixing term, as demonstrated in [31], the bi-linear weights ensure that for binary nodes, the activation function of each node follows a form of the sigmoid function (1.2) or at least the form of its cousin, the softmax function. This carries benefits and drawbacks as discussed in 1.1.3. Considering the framing of energy to label probabilities, it is interesting to note here that the sigmoid function is functionally similar to the Fermi-Dirac distribution

$$f_{\mathrm{FD}}\left(\varepsilon\right) = \frac{1}{1 + e^{\frac{\varepsilon}{k_B T}}} \tag{1.7}$$

which describes the probability of an electron's occupation of a particular state.

1.2.4 RBM learning algorithms

The usual training algorithm RBMs use is the Contrastive Divergence (CD) algorithm. This algorithm begins by initialising the visible nodes of the RBM with some given data. Following this, it updates the configuration of the hidden nodes according to their probability distributions/activation probabilities. Once the hidden layer has been configured, the visible nodes are re-configured probabilistically via their own activation probabilities/probability distributions, given the configuration of the hidden nodes. The difference between the data configuring the initial visible node distribution and the reconstructed visible node distribution is then used to update the parameters of the RBM. In this process, an approximation to the gradient of the log-likelihood function of the RBM is made by performing a process known as Gibbs sampling. Gibbs sampling is discussed further in subsection 2.1.4.

This training procedure is performed over multiple batches of data, improving the performance of the RBM until it reaches a state where it can satisfactorily reconstruct observed data. This iterative process trains the RBM on underlying patterns in the provided data. Once the RBM has been trained, it can be used generatively to produce new samples of data that well represent the original dataset.

1.3 The Kullback-Leibler divergence

The Kullback-Leibler (KL) divergence is a measure that plays a crucial role in the training of RBMs; it serves as the primary objective function guiding the learning process. The KL divergence is a non-negative real number representing the statistical divergence between a model distribution and target data distribution. By minimising the KL divergence of an RBMs configuration probability distribution with respect to a target data distribution, the discrepancies between the two distributions are minimised. Minimising the KL divergence, therefore, corresponds to maximising an RBM's accuracy in representing data. The KL divergence is also a valuable tool for comparing the performance of various RBMs with each other, with better models having lower KL divergences with respect to data. As discussed earlier, RBMs employ the CD algorithm to minimse an objective function. In particular, the CD algorithm approximates the gradient of the KL divergence to find the direction of the steepest descent with respect to model parameter variations; this is used to update parameters accordingly.

One of the main motivations for choosing the KL divergence as the primary objective function stems from its information-theoretic roots. Cross entropy is introduced and explored as a concept in subsection 1.3.1. Subsection 1.3.2 applies the idea of cross-entropy to physics's more familiar idea of entropy to understand entropy from the perspective of cross-entropy. Once an intuition for cross-entropy and entropy has been made from an information-theoretic standpoint, subsection 1.3.3 defines the KL divergence in terms of the cross-entropy of the model and target distributions. To gain further familiarity, this section is followed by section 1.4, where a series of examples of using the KL divergence as a measure of similarity between model and target probability distributions are made.

1.3.1 Cross entropy

To understand the Kullback-Leibler divergence, which is the objective function to be minimised, *cross entropy* must be introduced. The cross entropy H of a distribution Q relative to a second distribution P is given by

$$H\left(P,\, Q\right) = -\mathbb{E}_P\left[\log\left(Q\right)\right], \tag{1.8}$$

where $\mathbb{E}_P$ denotes the expectation value over the distribution P. The cross entropy can be understood as a meaningful quantity by considering the Kraft-McMillan theorem, which states that any method for coding the correct selection of a variable x_i from a collection of possible variables $\{x_1,\, \ldots,\, x_n\}$ can be understood as synonymous with defining a probability distribution

$$Q\left(x_i\right) = \left(\frac{1}{2}\right)^{\ell_i}, \tag{1.9}$$

where ℓ_i is the length of the code for x_i in bits [32]. This interpretation comes from the rearrangement

$$\ell_i = -\log_2\left(Q\left(x_i\right)\right), \tag{1.10}$$

from which the expected message length per datum x for an assumed coding scheme represented by a probability distribution P can be calculated. This is done by integrating over all possible data, weighted by the assumed (but not necessarily correct) probability distribution P associated with the identification of the data. This leaves that

$$\mathbb{E}_P\left[\ell_i\right] = -\mathbb{E}_P\left[\log_2\left(Q\left(x_i\right)\right)\right], \tag{1.11}$$

which is the exactly the cross-entropy identified earlier; $H\left(P,\, Q\right)$ can be interpreted via the equation

$$H\left(P,\, Q\right) = \mathbb{E}_P\left[\ell_i\right]. \tag{1.12}$$

That is, $H\left(P,\, Q\right)$ is the expected number of bits needed to be drawn from a data-set represented by a distribution Q, but that is assumed to be P in order to interpret a datum.

1.3.2 Entropy in Physics

In physics, the concept of entropy is familiar. In particular, physicists referring to entropy are usually referencing *Gibbs entropy.* The Gibbs entropy, S_{Gibbs} of a distribution Q is defined as the sum over the set of possible configurations

$$S_{\text{Gibbs}}\left(Q\right) = -k_B \sum_{i=1}^{n} q_i \log\left(q_i\right), \tag{1.13}$$

where q_i is the probability of the i^{th} system configuration. Up to a factor, this is exactly the cross-entropy of a distribution P with itself. It is constructive to interpret this through the lens of cross-entropy. Consider the cross-entropy of a probability distribution Q with itself

$$H\left(Q,\, Q\right) = \mathbb{E}_Q\left[\log\left(Q\right)\right]. \tag{1.14}$$

Writing the Gibbs entropy (1.13) of a distribution in terms of this yields

$$S_{\text{Gibbs}}\left(Q\right) = -k_B H\left(Q,\, Q\right). \tag{1.15}$$

The “disorder” of a system, which is usually how entropy is interpreted, can now be seen as a measure of the amount of data required to make an interpretation of a physical observation of a system, given that the most efficient interpretation of the data possible is being used ($P = Q$). The larger the entropy of a physical system, the more data is required to fully describe that system.

1.3.3 Defining the KL divergence

The Kullback-Liebler divergence of a distribution Q relative to a second distribution P is defined as

$$D_{KL}\left(P\,||\,Q\right) = H\left(P,\, Q\right) - H\left(P,\, P\right). \tag{1.16}$$

That is, it is the excess message length over the minimum possible message length required to convey the average datum of information from a distribution P when using a coding scheme Q to interpret the data. Q is usually labelled as the “model” distribution which is being used to approximate the “target” P. By using the inequality

$$\ln\left(x\right) \leq x - 1, \tag{1.17}$$

with equality only when $x = 1$, it is simple to prove that

$$D_{KL}\left(P\,||\,Q\right) \geq 0, \tag{1.18}$$

with equality only when Q is equal to P over the entire configuration space of possible events. This is demonstrated in Appendix A.1. It is intuitive to see that, in some sense, the better a model distribution Q describes a target distribution P, the smaller the Kullback-Leibler divergence of that model relative to the target. The simplicity behind this led to the KL divergence becoming a popular choice of objective function to minimise in the training of an RBM and is the choice considered for the work in this thesis.

Finally, for convenience, notice that the definition of the cross entropy leaves a more tidy expression of the Kullback-Leibler divergence:

$$\begin{aligned} D_{KL}\left(P||\,Q\right) &= -\mathbb{E}_P\left[\log(Q)\right] - -\mathbb{E}_P\left[\log\left(P\right)\right] \\ &= \mathbb{E}_P\left[\log\left(P/Q\right)\right]. \end{aligned} \tag{1.19}$$

Using this notation, it is important not to lose track of the fact that the expectation value is taken over the probability space of P and not Q. In the context of simulating a model of the results of a lab, P should be considered the actual data from observations, whereas Q is the probability distribution as represented by the model.

1.4 Examples of calculating the Kullback-Leibler divergence

Now that the Kullback-Leibler divergence has been introduced as the objective function to be minimised when using a model probability distribution Q to represent a target probability distribution P, it is constructive to consider a few simple examples demonstrating how this measure can rank models according to their performance. In this section, three examples of target distributions are considered. These are

$$P_1(x) = \frac{1}{Z_1}\exp\left(-\beta|x|\right), \quad \text{with } Z_1 = \frac{2}{\beta},\ x \in \mathbb{R}, \tag{1.20}$$

$$P_6(x) = \frac{1}{Z_6}\exp\left(-\alpha x^6\right), \quad \text{with } Z_6 = 2\alpha^{-\frac{1}{6}}\Gamma\left(\frac{7}{6}\right),\ x \in \mathbb{R}, \tag{1.21}$$

and finally, with $P(x)$ unspecified, up to knowledge that $x \in \mathbb{R}$. Competing in the representation of these target distributions are two normalised model distributions Q_2 and Q_4, which are given by

$$Q_2(x) = \frac{1}{Z_2}\exp\left(-m^2x^2\right),\ \text{with } Z_2 = 2(m^2)^{-\frac{1}{2}}\Gamma\left(\frac{3}{2}\right), \tag{1.22}$$

and

$$Q_4(x) = \frac{1}{Z_4}\exp\left(-\lambda x^4\right),\ \text{and } Z_4 = 2\lambda^{-1/4}\,\Gamma\left(\frac{5}{4}\right), \tag{1.23}$$

where m^2 and λ are both positive parameters determining the models.

Given these definitions of model and target distributions, the Kullback-Leibler divergence of the models with respect to each target can be calculated in order to understand how well they model each distribution. The performance of the models in representing each of the targets is explored in the following three subsections.

1.4.1 Example calculation 1: $P = P_6$

It is simple to show that

$$D_{\mathrm{KL}}\left(P_6 \,||\, Q_2\right) = \log\left(\frac{\Gamma\left(\frac{3}{2}\right)}{\Gamma\left(\frac{7}{6}\right)}\right) + \frac{1}{6}\left(\log\left(\alpha\right) - 1\right) + \frac{m^2}{\alpha^{\frac{1}{3}}}\frac{\Gamma\left(\frac{3}{2}\right)}{3\Gamma\left(\frac{7}{6}\right)} - \frac{1}{2}\log\left(m^2\right), \tag{1.24}$$

and that similarly

$$D_{\mathrm{KL}}\left(P_6 \,||\, Q_4\right) = \log\left(\frac{\Gamma\left(\frac{5}{4}\right)}{\Gamma\left(\frac{7}{6}\right)}\right) + \frac{1}{6}\left(\log\left(\alpha\right) - 1\right) + \frac{\lambda}{\alpha^{\frac{2}{3}}}\frac{\Gamma\left(\frac{5}{6}\right)}{6\Gamma\left(\frac{7}{6}\right)} - \frac{1}{4}\log\left(\lambda\right). \tag{1.25}$$

The models Q_2 and Q_4 can be optimised by minimisation of their KL divergences by varying the model parameters, m^2 and λ respectively. Differentiation with respect to m^2 reveals a single stationary point in $D_{\mathrm{KL}}\left(P_6 \,||\, Q_2\right)$ when

$$m^2 = m_*^2, \tag{1.26}$$

where

$$m_*^2 = \frac{3\Gamma\left(\frac{7}{6}\right)}{2\Gamma\left(\frac{3}{2}\right)}\alpha^{\frac{1}{3}}. \tag{1.27}$$

Denoting Q_2^* as $Q(x)$ when m^2 is chosen as m_*^2, this then gives the minimum possible Kullback-Leibler divergence of Q_2 relative to P_6 as

$$D_{KL}\left(P_6 \,||\, Q_2^*\right) = \frac{1}{2}\log\left(\frac{2}{3}\left(\frac{\Gamma\left(\frac{3}{2}\right)}{\Gamma\left(\frac{7}{6}\right)}\right)^3\right) + \frac{1}{3} \approx 0.06, \tag{1.28}$$

where the natural log is used. A similar calculation for $D_{\mathrm{KL}}\left(P \,||\, Q_4\right)$ reveals a stationary point when $\lambda = \lambda_*$, where

$$\lambda_* = \frac{3\Gamma\left(\frac{7}{6}\right)}{2\Gamma\left(\frac{5}{6}\right)}\alpha^{\frac{2}{3}}, \tag{1.29}$$

leaving a minimum Kullback-Leibler divergence of

$$D_{\mathrm{KL}}\left(P_6 \,||\, Q_4^*\right) = \frac{1}{4}\log\left(\frac{2}{3}\frac{\Gamma\left(\frac{5}{6}\right)\left(\Gamma\left(\frac{5}{4}\right)\right)^4}{\left(\Gamma\left(\frac{7}{6}\right)\right)^5}\right) + \frac{1}{12} \approx 0.01. \tag{1.30}$$

Clearly the best approximation of the quartic model to the target is much better on average over the domain of the problem than the Gaussian.

1.4.2 Example calculation 2: $P = P_1$

Having seen that the quartic model can better represent P_6 than a Gaussian model, consider instead the target distribution P_1. As above, a simple calculation yields that

$$D_{\mathrm{KL}}\left(P_1\,||\,Q_2\right) = \log\left(\Gamma\left(\frac{3}{2}\right)\right) + \left(\log\left(\beta\right) - 1\right) + \frac{2m^2}{\beta^2} - \frac{1}{2}\log\left(m^2\right), \tag{1.31}$$

and that

$$D_{\mathrm{KL}}\left(P_1\,||\,Q_4\right) = \log\left(\Gamma\left(\frac{5}{4}\right)\right) + \left(\log\left(\beta\right) - 1\right) + \frac{24\lambda}{\beta^4} - \frac{1}{4}\log\left(\lambda\right). \tag{1.32}$$

This time, differentiation of the above with respect to m^2 reveal different optimal parameter choices. Now

$$m_*^2 = \frac{\beta^2}{4} \tag{1.33}$$

and

$$\lambda_* = \frac{\beta^4}{96}. \tag{1.34}$$

Subsitution of these choices for m^2 and λ into their respective distributions gives the minimised Kullback-Leibler divergences as

$$D_{KL}\left(P_1\,||\,Q_2^*\right) = \log\left(2\Gamma\left(\frac{3}{2}\right)\right) - \frac{1}{2} \approx 0.072, \tag{1.35}$$

and

$$D_{KL}\left(P_1\,||\,Q_4^*\right) = \log\left(2\sqrt[4]{6}\,\Gamma\left(\frac{5}{4}\right)\right) - \frac{3}{4} \approx 0.293. \tag{1.36}$$

By these results, the optimal performance of Q_2 will always have a lower Kullback-Leibler divergence than Q_4 when modelling P_1, meaning from this perspective, the Gaussian model better represents Q_1 than the quartic model.

1.4.3 Example calculation 3: $P(x)$ unspecified

Now that the cases where the models are chosen specifically to highlight the behaviour of the models have been seen, consider instead the case where P is an unspecified distribution, that is only known to have a single real argument. In this case now, it can be calculated that

$$D_{KL}\left(P\,||Q_2\right) = \mathbb{E}_P\left[\log\left(P\right)\right] + m^2\mathbb{E}_P\left[x^2\right] + \log\left(2\left(m^2\right)^{-\frac{1}{2}}\Gamma\left(\frac{3}{2}\right)\right), \tag{1.37}$$

and that

$$D_{KL}\left(P\,||Q_4\right) = \mathbb{E}_P\left[\log\left(P\right)\right] + \lambda\mathbb{E}_P\left[x^4\right] + \log\left(2\lambda^{-\frac{1}{4}}\Gamma\left(\frac{5}{4}\right)\right). \tag{1.38}$$

The optimal parameter choices in this case can be shown to be

$$m_*^2 = \frac{1}{2\mathbb{E}_P\left[x^2\right]}, \tag{1.39}$$

and

$$\lambda_* = \frac{1}{4\mathbb{E}_P\left[x^4\right]}. \tag{1.40}$$

This gives optimal Kullback-Leibler divergence values of

$$D_{\mathrm{KL}}\left(P\,||Q_2^*\right) = \mathbb{E}_P\left[\log\left(P\left(x\right)\right)\right] + \frac{1}{2}\log\left(\mathbb{E}_P\left[x^2\right]\right) + \frac{1}{2} + \log\left(2^{\frac{3}{2}}\Gamma\left(\frac{3}{2}\right)\right), \tag{1.41}$$

and

$$D_{\mathrm{KL}}\left(P\,||Q_4^*\right) = \mathbb{E}_P\left[\log\left(P\left(x\right)\right)\right] + \frac{1}{4}\log\left(\mathbb{E}_P\left[x^4\right]\right) + \frac{1}{4} + \log\left(2^{\frac{3}{2}}\Gamma\left(\frac{5}{4}\right)\right). \tag{1.42}$$

Now in order to compare the optimal performance of Q_2 and Q_4 in their representation of P, it is requried to take the difference of the Kullback-Leibler divergences in order to cancel out the contributions towards the KL divergences from the cross entropy of P with itself, which is impossible to compute in this context, but also irrelevant. Now, if it is found that

$$D_{\mathrm{KL}}\left(P\,||Q_4^*\right) - D_{\mathrm{KL}}\left(P\,||Q_2^*\right) < 0, \tag{1.43}$$

then Q_4, when optimised, would be considered as the better at representing P with this metric, and vice versa. Taking this difference yields

$$D_{\mathrm{KL}}\left(P\,||Q_4^*\right) - D_{\mathrm{KL}}\left(P\,||Q_2^*\right) = \log\left(\frac{\left(\mathbb{E}_P\left[x^4\right]\right)^{\frac{1}{4}}}{\left(\mathbb{E}_P\left[x^2\right]\right)^{\frac{1}{2}}}\right) + \log\left(\frac{\Gamma\left(\frac{3}{2}\right)}{\Gamma\left(\frac{5}{4}\right)}\right) - \frac{1}{4}, \tag{1.44}$$

meaning that Q_4 can outperform Q_2 only when

$$R_{P,4,2} \equiv \frac{\mathbb{E}_P\left[x^4\right]}{\left(\mathbb{E}_P\left[x^2\right]\right)^2} < e\left(\frac{\Gamma\left(\frac{5}{4}\right)}{\Gamma\left(\frac{3}{2}\right)}\right)^4 \approx 2.97. \tag{1.45}$$

To verify this result, consider the second and fourth moments of P_1 and P_6 as given earlier. These can be calculated as

$$\mathbb{E}_{P_1}\left[x^2\right] = \frac{2}{\beta^2}, \quad \mathbb{E}_{P_1}\left[x^4\right] = \frac{24}{\beta^4}, \tag{1.46}$$

and

$$\mathbb{E}_{P_6}\left[x^2\right] = \frac{\Gamma\left(\frac{3}{2}\right)}{3\alpha^{\frac{1}{3}}\Gamma\left(\frac{7}{6}\right)}, \quad \mathbb{E}_{P_6}\left[x^4\right] = \frac{\Gamma\left(\frac{5}{6}\right)}{6\alpha^{\frac{2}{3}}\Gamma\left(\frac{7}{6}\right)}. \tag{1.47}$$

These firstly give that

$$R_{P_6,4,2} = 2 < 2.97, \tag{1.48}$$

which is in agreement with why Q_4 had a lower minimum Kullback-Leibler divergence relative to P_6 than Q_1 did. Similarly, these calculations give that

$$R_{P_1,4,2} = 6 > 2.97, \tag{1.49}$$

which correctly estimates that Q_4 has a higher minimum Kullback-Leibler divergence to P_1 than Q_2 does.

It is also worth noting at this point that if the target is a Gaussian, that is $P = P_2$, then

$$R_{P_2,4,2} = 3 > 2.97, \tag{1.50}$$

meaning that the Quartic model cannot outperform the Gaussian model in the case of $P = P_2$.

The validation of these calculations demonstrates the power available to investigate the representation power of various models even when the distribution they are targeting is unknown. Of particular key importance to note is the optimal parameter choices (1.39) and (1.40). A simple calculation shows that

$$\mathbb{E}_{Q_2}\left[x^2\right] = \frac{1}{2m^2} \tag{1.51}$$

and that

$$\mathbb{E}_{Q_4}\left[x^4\right] = \frac{1}{4\lambda}. \tag{1.52}$$

This is insightful, as it means that the optimal choice of m^2 and λ are such that

$$\mathbb{E}_{Q_2^*}\left[x^2\right] = \mathbb{E}_P\left[x^2\right], \tag{1.53}$$

and

$$\mathbb{E}_{Q_4^*}\left[x^4\right] = \mathbb{E}_P\left[x^4\right]. \tag{1.54}$$

The action of minimising the Kullback-Leibler divergence of a model distribution characterised by $Q \propto \exp\left(-c\,x^\gamma\right)$ resulted in the matching of the x^γ moment of the model distribution with that of the target in both cases. This gives a second hint as to what the minimisation of the Kullback-Leibler divergence is actually doing and why a smaller Kullback-Leibler divergence between a model and a target makes them more similar; the action of minimising the KL divergence is parallel to the minimisation of the differences in the moments of the model and target distributions for basic models.

1.5 Example: Upgrading a Gaussian model

In the context of training RBMs, there will arise scenarios where a given model has reached a maximum in its capacity to represent target data. Once this occurs, it would be natural to ask whether it is beneficial to extend the model to gain further representation power at the expense of additional computational requirements. The answer to this is not obvious; a second model with additional complexity would be able to perform as well as the original model. However, it is unclear as to when it could perform better. In this section, the consideration of whether introducing a quartic parameter to a Gaussian model of a one-dimensional probability distribution would necessarily improve the performance of the original Gaussian model.

Consider the case of a single variable probability distribution $P(x)$, where $x \in \mathbb{R}$, and P has a finite fourth moment. Let

$$Q_2(x) = \frac{e^{-m_2^2 x^2}}{Z_2}; \quad Z_2 = \sqrt{\frac{\pi}{m_2^2}}, \tag{1.55}$$

and then minimise $D_{\mathrm{KL}}\left(P||\, Q_2\right)$ such that $m_2^2 = m_*^2$, identically to as in (1.39). Now consider a second model

$$Q_{42}(x) = \frac{e^{-\lambda x^4 - m_4^2 x^2}}{Z_{42}}, \tag{1.56}$$

which has both a Gaussian and quartic component in its construction. Now, the partition function is more complicated, but it works out as

$$\begin{aligned} Z_{42} &= \int_{-\infty}^{\infty} \exp\left(-\lambda x^4 - m_4^2 x^2\right) \mathrm{d}x \\ &= \frac{1}{2}\sqrt{\frac{m_4^2}{\lambda}} \exp\left(\frac{\left(m_4^2\right)^2}{8\lambda}\right) K_{\frac{1}{4}}\left(\frac{\left(m_4^2\right)^2}{8\lambda}\right), \end{aligned} \tag{1.57}$$

where $K_\nu\left(x\right)$ is the modified Bessel function of the second kind. In terms of the convenient variable

$$u = \frac{\left(m_4^2\right)^2}{8\lambda}, \tag{1.58}$$

the differences in the Kullback-Leibler divergences of the two models with respect to the target distribution, $\Delta D_{\mathrm{KL}}\left(P||\, Q_{42},\, Q_2^*\right) := D_{\mathrm{KL}}\left(P||\, Q_{42}\right) - D_{\mathrm{KL}}\left(P||\, Q_2^*\right)$, can be expressed as

$$\begin{aligned} \Delta D_{\mathrm{KL}}\left(P||\, Q_{42},\, Q_2^*\right) = \frac{1}{2}\log\left(\frac{2}{\pi}\right) + \log\left(u^{\frac{1}{2}} e^{u} K_{\frac{1}{4}}\left(u\right)\right) + \frac{1}{2}\log\left(\frac{m_*^2}{m_4^2}\right) \\ + \lambda \mathbb{E}_P\left[x^4\right] + \left(m_4^2 - m_*^2\right)\mathbb{E}_P\left[x^2\right]. \end{aligned} \tag{1.59}$$

Considering now only the case where the new parameter λ is small, the limit of $\lambda \to 0$ with m_4^2 kept finite is explored. This is equivalent to $u \to \infty$ for non-zero m_4^2. By using that

$$\lim_{u\to\infty} K_{\frac{1}{4}}\left(u\right) = e^{-u}\left(\sqrt{\frac{\pi}{2u}} - \frac{3}{32}\sqrt{\frac{\pi}{2u^3}} + \mathcal{O}\left(u^{-\frac{5}{2}}\right)\right), \tag{1.60}$$

and that for small x,

$$\log\left(1+x\right) = x + \mathcal{O}\left(x^2\right), \tag{1.61}$$

the limiting behaviour of the difference between the Kullback-Leibler divergences of the two models can be written up to quadratic order in u^{-1} as as

$$\begin{aligned} \lim_{u\to\infty} \Delta D_{\mathrm{KL}}\left(P||\, Q_{42},\, Q_2^*\right) = -\frac{3}{32u} + \frac{\left(m_4^2\right)^2}{8u}\mathbb{E}_P\left[x^4\right] \\ + \left(m_4^2 - m_*^2\right)\mathbb{E}_P\left[x^2\right] + \frac{1}{2}\log\left(\frac{m_*^2}{m_4^2}\right). \end{aligned} \tag{1.62}$$

The new model is an improvement when $\Delta D_{\mathrm{KL}}\left(P||\,Q_{42},\,Q_2^*\right) < 0$. This inequality can be expressed in terms of λ and m_4^2 as

$$\lambda\left(\mathbb{E}_P\left[x^4\right]-\left(\frac{m_*^2}{m_4^2}\right)^2\mathbb{E}_{Q_2^*}\left[x^4\right]\right)+\left(m_*^2-m_4^2\right)\mathbb{E}_P\left[x^2\right]+\frac{1}{2}\log\left(\frac{m_*^2}{m_4^2}\right)<0, \tag{1.63}$$

correct up to order $\lambda^2/(m_4^2)^2$. That is, if the choice of λ and m_4^2 together satisfy the above inequality (with $\lambda \ll 1$), then the new model is an improvement on the initial. This inequality is satisfied for various choices of λ and m_4^2, but in a way that is entirely dependent on the target distribution P. As discussed in 1.4.3, the Kullback-Leibler divergence is sensitive to the moments of a distribution, and as such, the indicator for the different cases determining the choices of λ and m_4^2 to improve the model performance are dependent on the fourth moment of P. How to choose λ and m_4^2 is discussed in subsections 1.5.1, 1.5.2 and 1.5.3, followed by a discussion in 1.5.4. Figure 1.3 illustrates the behaviour of distributions when

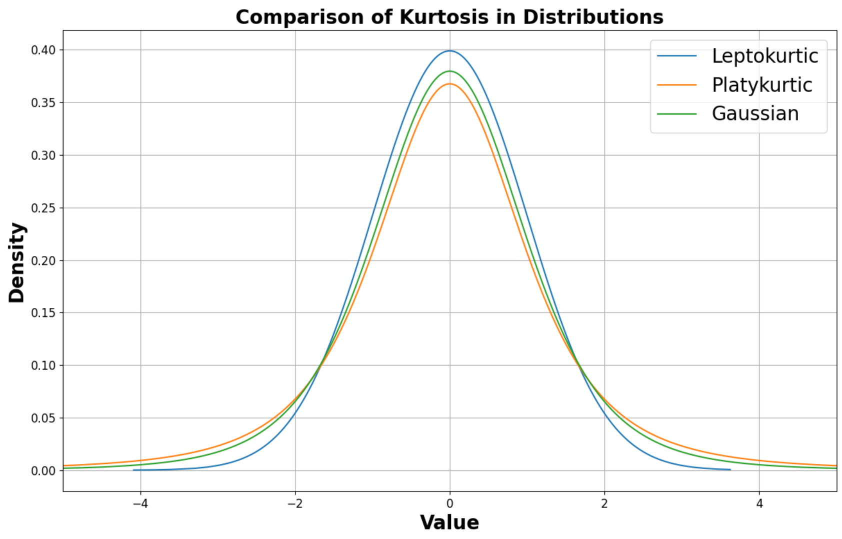


FIGURE 1.3: *Plot illustrating the behavior of leptokurtic and platykurtic distributions in relation to a Gaussian.*

characterised by their standardised fourth moment relative to a Gaussian. The leptokurtic distribution exhibits a higher standardized fourth moment than the Gaussian, resulting in a heavier tail and a sharper peak. On the other hand, the platykurtic distribution has a lower standardized fourth moment compared to the Gaussian, leading to a lighter tail and a flatter peak. The shape of the distribution is emphasized rather than its width.

1.5.1 Case 1: P is Platykurtic

If the target distribution is Platykurtic, that is, where its fourth moment is smaller than for a Gaussian distribution with the same standard deviation as P, then

$$\mathbb{E}_P\left[x^4\right] < \mathbb{E}_{Q_2^*}\left[x^4\right]. \tag{1.64}$$

Under such a condition, the choice $m_4^2 = m_*^2$ clearly allows for a small but positive λ for which the inequality (1.63) is satisfied, meaning in such a situation, the introduction of a new small quartic parameter λ will immediately result in an improvement of the model.

1.5.2 Case 2: P is Leptokurtic

If the target distribution is Leptokurtic, where its fourth moment is larger than that of Gaussian distribution of the same standard deviation as P, then $\mathbb{E}_P\left[x^4\right] > \mathbb{E}_{Q_2^*}\left[x^4\right]$. Under such a situation, in order to improve the model with a small parameter λ, it is necessary to choose

$$m_4^2 \gg m_*^2. \tag{1.65}$$

Under this choice, the change in Kullback-Leibler divergence for small λ approximates as

$$\Delta D_{\mathrm{KL}}\left(P||\, Q_{42},\, Q_2^*\right) = \mathbb{E}_P\left[x^2\right]\left(\lambda\frac{\mathbb{E}_P\left[x^4\right]}{\mathbb{E}_P\left[x^2\right]} - m_4^2\right) + \mathcal{O}\left(\left(\frac{\lambda}{(m_4^2)^2}\right)^2\right), \tag{1.66}$$

where the contribution to the log has been discarded for simplicity as it is always negative in this regime. The indicator then becomes negative for choices of λ where

$$\lambda < m_4^2\frac{\mathbb{E}_P\left[x^4\right]}{\mathbb{E}_P\left[x^2\right]}. \tag{1.67}$$

1.5.3 Case 3: P is Mesokurtic

In the case where $\mathbb{E}_P\left[x^4\right] = \mathbb{E}_{Q_4^*}\left[x^4\right]$, then a choice of m_4^2 of the form

$$m_4^2 = m_*^2 + \delta m^2; \quad \text{with } \delta m^2 \ll m_*^2 \tag{1.68}$$

allows for a small λ for which $\Delta D_{\mathrm{KL}}\left(P||\, Q_{42},\, Q_2^*\right) < 0$. Once again, the contribution from the log is discarded due to the fact that it is always negative in this regime. Under such a choice, by using that

$$\left(m_4^2\right)^{-2} \approx \frac{1}{\left(m_*^2\right)^2}\left(1 - 2\frac{\delta m^2}{m_*^2}\right), \tag{1.69}$$

the differences between the Kullback-Leibler divergences of the two models becomes

$$\Delta D_{\mathrm{KL}}\left(P||\, Q_{42},\, Q_2^*\right) \approx \delta m^2\left(2\lambda\frac{\mathbb{E}_{Q_2^*}\left[x^2\right]}{m_*^2} - \mathbb{E}_P\left[x^2\right]\right), \tag{1.70}$$

which is negative for

$$\lambda < m_*^2\frac{\mathbb{E}_P\left[x^2\right]}{2\mathbb{E}_{Q_2^*}\left[x^2\right]}. \tag{1.71}$$

1.5.4 Discussion on the upgrading of the Gaussian model

The results of the above demonstrate three key points:

1. In this example, it was always possible to improve the model via the introduction of a small parameter λ.

2. However, it was not always possible to improve the model with a small λ without changing the mass parameter m^2 by a significant amount.

3. The measurement of moments of quantities over the target distribution is key in the determination of whether a model can improve its performance without changing already-tuned parameters.

These are important conclusions to take from the above example. While it has not been proven whether or not the introduction of a new parameter can always improve a model, it has been shown that the introduction of a new parameter will occasionally necessitate the changing of the older, already tuned parameters and that understanding when changes to the older parameters do not have to be made can require the calculation of moments over the target distribution.

This focuses the question of interest to cases of understanding when it is possible to improve a neural net via the introduction of new parameters *without* significantly changing the older parameters first. This is an important distinction, as in the context of a many-variable model distribution, understanding the change in the Kullback-Leibler divergence upon the introduction of a new parameter can become particularly complicated if the pre-existing parameters are not fixed.

By restricting the analysis to cases where the older parameters are left unchanged, isolating specific effects of the new parameter will be simpler, allowing for a clearer understanding of its influence on the model performance. Introducing the new parameter this way also eliminates the risk of degrading the existing performance already achieved in the training of the network, as well as also saving on computational resources by leveraging pre-existing knowledge by avoiding the need for re-training any of the models. Also, focusing specifically on the scenarios where the older parameters are kept constant initially allows for a more direct comparative analysis between the models, allowing for potential further insight.

1.6 Motivations from Physics

Now that the KL divergence has been explored, it is instructive to re-visit some ideas from physics in order to motivate how to understand and approach the process of KL divergence minimisations. Subsection 1.6.1 begins by further concreting the relationship between the KL divergence and thermodynamic potentials, whilst also discussing a caveat showing where care must be taken in constructing anaologies between the two. Subsection 1.6.2 then brings in ideas from Statistical Mechanics to explain the idea of the partition function as a generating functional, which can be used to describe the thermodynamic characteristics of a physical theory fully. Subsection 1.6.3 goes on to use the idea of the partition function as a generating functional to demonstrate the importance and convenience of having a quadratic term in the Lagrangian of a quantum field theory, and consequently, in the energy function of a Restricted Boltzmann Machine. Finally, subsection 1.6.4 introduces the concept of lattice field theory, demonstrates the motivations for lattice field theory in theoretical physics contexts, and then discusses why that framework is more relevant in the study of RBMs.

1.6.1 A nod to thermodynamics

As seen in 1.4 and 1.5, calculating the difference in the Kullback-Leibler divergences of two models relative to some target probability distribution requires the calculation of the log of the partition function of the models, as well as moments of the target. To phrase this more cleanly, given a model Q_i for which

$$Q_i\left(\mathbf{v}\right) = \frac{\exp\left(-E_i\right)}{Z_i}, \tag{1.72}$$

the Kullback-Leibler divergence of that model relative to a target P can be expressed as

$$D_{\mathrm{KL}}\left(P||\,Q_i\right) = \mathbb{E}_P\left[\log\left(P\right)\right] + \mathbb{E}_P\left[E_i\right] + \log\left(Z_i\right). \tag{1.73}$$

At this point, consider phrasing the Kullback-Leibler divergence from the point of view of physics. In thermodynamics, the expected energy of a system is known as its *internal* energy, U. This is the mean energy a thermodynamic system will take, given a distribution of energy levels filled by a canonical ensemble of particles populating a system.

This might give the impression that a simple correspondence between thermodynamic and RBM variables could be made, although care must be taken. The correspondence

$$U_i \equiv \mathbb{E}_P\left[E_i\right], \text{ and } F_i \equiv -\log\left(Z_i\right), \tag{1.74}$$

would be incorrect to make, as the quantity E_i defines the distribution Q_i rather than P. Therefore, it would be erroneous to draw a connection between these quantities that describe the RBM and the familiar physical quantities. Consequently, such a connection between the KL divergence and physical quantities would only make sense if the expectation had been taken over Q_i rather than P. Such a scenario does arrive if one calculates the KL divergence $D_{\mathrm{KL}}\left(Q_i||\,P\right)$ rather than $D_{\mathrm{KL}}\left(P||\,Q_i\right)$ via

$$D_{\mathrm{KL}}\left(Q_i||\,P\right) = -\mathbb{E}_{Q_i}\left[\log\left(E_i\right)\right] - \log\left(Z_i\right) - \frac{1}{Z_i Z_P}\int e^{-(E_i+E_P)}\,\mathrm{d}\mathbf{v}, \tag{1.75}$$

where P has been written as taking the form

$$P\left(\mathbf{v}\right) = \frac{e^{-E_P(\mathbf{v})}}{Z_P}. \tag{1.76}$$

However, such route is not of interest due to the preferred calculation of $D_{\mathrm{KL}}\left(Q_i||\,P\right)$ over $D_{\mathrm{KL}}\left(P||\,Q_i\right)$ following the conceptual building discussed in 1.3.1, as well as the fact that the third term in equation (1.75) depends on both the (known) model and the unknown target distribution. This term is problematic, unlike in equation (1.73), where the model and target behaviour separated in a way that could be dealt with when the difference in KL divergences of two models was taken.

Finally, in the hypothetical scenario where the model is chosen and tuned correctly, that is, where $Q_i = P$ everywhere, the energy E_i becomes the energy defining the target distribution E_P, and the partition function Z_i becomes equal to Z_P. In this scenario only, then, equation

(1.73) becomes

$$D_{\mathrm{KL}}\left(P||\, Q_i = P\right) = \mathbb{E}_P\left[\log\left(P\right)\right] + \mathbb{E}_P\left[E_P\right] + \log\left(Z_P\right). \tag{1.77}$$

By then making the correspondences

$$U_P \equiv \mathbb{E}_P\left[E_P\right], \text{ and } F_P \equiv -\log\left(Z_P\right), \tag{1.78}$$

the above becomes

$$D_{\mathrm{KL}}\left(P||\, Q_i = P\right) = \mathbb{E}_P\left[\log\left(P\right)\right] + U_P - F_P. \tag{1.79}$$

Furthermore, by defining the Gibbs entropy, S_P, of P via

$$TS_P = -\mathbb{E}_P\left[\log\left(P\right)\right], \tag{1.80}$$

the KL divergence in this scenario can be written fully in terms of thermodynamic variables

$$D_{\mathrm{KL}}\left(P||\, Q_i = P\right) = -TS_P + U_P - F_P. \tag{1.81}$$

This, however, evaluates to zero (as should be expected). This can either be seen through the relationships between the thermodynamic variables, or simply by the fact that the left hand side is $D_{\mathrm{KL}}\left(P||\, P\right)$, which must be zero. As such, despite the consistency in analogues that can be drawn, the connection to thermodynamics here is not of any particular benefit in this context.

1.6.2 Statistical Field Theory

Statistical Physics reveals that the partition function from physical probability distributions of the form

$$p = \frac{e^{-\beta\varepsilon}}{Z} \tag{1.82}$$

does not only act as constants of normalisation. It turns out that all macroscopic information about physical systems is contained within the partition function, allowing it to act as a *generating function*.

The consideration of a lattice of fields ϕ_k with some Hamiltonian $\hat{H}_0$ has a partition function

$$Z = \mathrm{Tr}\left[e^{-\beta\hat{H}_0}\right], \tag{1.83}$$

and operators $\hat{O}$ defined on that lattice have thermal expectation values given by

$$\langle\hat{O}\rangle = \frac{\mathrm{Tr}\left[\hat{O}\, e^{-\beta\hat{H}_0}\right]}{Z}. \tag{1.84}$$

Whether or not the Hamiltonian has a term linear in each ϕ_k, a *source term* can be introduced, which perturbs the Hamiltonian to give

$$\hat{H} = \hat{H}_0 - \frac{1}{\beta}\sum_k J_k\phi_k. \tag{1.85}$$

The purpose of this is that the partition function for $\hat{H}$ becomes a function of the couplings J_k to become

$$Z(J) = \mathrm{Tr}\left[e^{-\beta \hat{H}_0 + \sum_k J_k \hat{\phi}_k}\right]. \tag{1.86}$$

This is useful because combinations of the thermally averaged field values $\langle \phi_i \rangle$ are of interest in statistical physics and can now be derived from $Z(J)$. Consider the derivative of $Z(J)$ with respect to J_i,

$$\frac{\partial Z(J)}{\partial J_i} = \mathrm{Tr}\left[\hat{\phi}_i \, e^{-\beta \hat{H}_0 + \sum_k J_k \hat{\phi}_k}\right]. \tag{1.87}$$

By then evaluating this at $J = 0$ and dividing by $Z(J = 0)$, it follows that

$$\frac{1}{Z(J=0)} \frac{\partial Z(J)}{\partial J_i}\bigg|_{J=0} = \frac{\mathrm{Tr}\left[\hat{\phi}_i \, e^{-\beta \hat{H}_0}\right]}{Z(0)}, \tag{1.88}$$

which according to (1.84) is exactly the formula for the expected field value $\langle \hat{\phi}_i \rangle$. As such, the thermally averaged field values can be found from the partition function via

$$\langle \hat{\phi}_i \rangle = \frac{1}{Z(J=0)} \frac{\partial Z(J)}{\partial J_i}\bigg|_{J=0}. \tag{1.89}$$

This is not all; the same procedure can be applied to find the correlation functions between fields (also known as the Green's functions) on the lattice. In general [33], it is found that

$$G^{(n)}_{i_1 \ldots i_n} \equiv \langle \hat{\phi}_{i_1} \ldots \hat{\phi}_{i_n} \rangle = \frac{1}{Z(J=0)} \frac{\partial^n Z(J)}{\partial J_{i_1} \ldots J_{i_n}}\bigg|_{J=0}. \tag{1.90}$$

The lesson here to be taken for application on Restricted Boltzmann Machines is that the intentional introduction of terms into the partition function via the Hamiltonian, followed by calculating derivatives with respect to these terms and setting the coupling to zero, can be a quick and straightforward way of calculating quantities of interest with regards to the systems in consideration. This critical technique will be used in chapter 3 to motivate choices of structure for the RBMs being studied, and the usefulness of such design implementations is exploited in chapter 4.

Before moving on to the key inspiration from quantum field theory used to progress calculations on the RBMs, it is important to understand how the above notion of the generating functional is imported into field-theoretic circumstances. There, each lattice site i is upgraded to position coordinate x. Correspondingly, the source terms J_i are upgraded to functions of position $J(x)$, the generating function $Z(J)$ is instead written $Z[J(x)]$, and the Green's functions $G^{(n)}_{i_1 \ldots i_n}$ are written $G^{(n)}(x_1, \ldots, x_n)$. For completeness, the Green's functions of field theory can be written

$$G(x_1, \ldots, x_n) = \frac{1}{i^n} \frac{1}{Z[J=0]} \frac{\delta^n Z[J]}{\delta J(x_1) \ldots \delta J(x_n)}\bigg|_{J=0}, \tag{1.91}$$

where i is a conventional factor of the imaginary unit, and each $\frac{\delta}{\delta J(x_i)}$ represents a functional derivative with respect to each source term $J(x_i)$ [33].

1.6.3 Quantum Field Theory

In the early 20th century, physics faced the problem of reconciling the principles of special relativity with the results of quantum mechanics. One of the main issues at the time was how the two treated time and space. On the one hand, special relativity portrayed the idea that time and space sat on an equal footing, whereas quantum mechanics treated the two manifestly differently. This different treatment between time and space is clear in the one-dimensional Schrödinger equation for a free particle

$$i\hbar \frac{\partial \Psi}{\partial t}(x,\, t) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\Psi(x,\, t)\,, \tag{1.92}$$

which has a single time derivative but a second spatial derivative. Special relativity has revolutionised physicists' understanding of time, space, and high-speed behaviour, and quantum mechanics opened up the world of probabilistic particle behaviour and the wave-particle duality of matter. The need for a theory consistent at their intersection led to the development of Quantum Field Theory.

Rather than treating particles as fundamental entities, QFT introduces the notion of a field, quantities that permeate spacetime. By considering particles as specific stable field configurations, QFT allowed a treatment that encapsulated both particle and wave-like properties. Simultaneously, by constraining the fields to obey select symmetry properties of special relativity, the particles emerging from QFT automatically obeyed the laws of special relativity. Work stemming from QFT has found remarkable success in explaining and predicting a vast swathe of physical phenomena, including the behaviour governing the interactions of elementary particles, the properties of condensed matter systems, and the existence of previously undetected particles. By fusing the principles of special relativity and quantum mechanics, QFT has paved the way for profound advancements in our understanding of the fundamental workings of the universe.

The fields in QFT are assigned continuous labels at each spacetime point x^μ. Dependent on the physics being studied, these fields can be scalar, vector, or spinor valued, and the dynamics captured from the corresponding field are modified accordingly; particularly, the intrinsic spin of excitations of the various fields will differ. Mathematically, the scalar (spin-0) fields ϕ are governed by the Klein-Gordon equation

$$\left[\partial_\mu \partial^\mu + m^2\right]\phi\left(x^\mu\right) = 0, \tag{1.93}$$

Maxwell's equations govern the components of the vector fields A^μ

$$\partial^\mu\left(\partial_\mu A_\nu\left(x^\mu\right) - \partial_\nu A_\mu\left(x^\mu\right)\right) = 0, \tag{1.94}$$

written here without sources, and the Dirac equation

$$\left[i\not{\partial} - m\right]\psi\left(x^\mu\right) = 0 \tag{1.95}$$

governs the spinor fields ψ. Notice how all are written in a Lorentz-covariant manner, allowing the symmetries of special relativity to be respected. The fields of QFT are quantised either via canonical quantisation or via the path integral approach. Field quantisation is the process by which the QFT fields are treated as operators that create and annihilate particles; these particles are then interpreted as the fundamental constituents of matter, carrying various physical properties such as charge, mass, and spin.

Both the canonical quantisation procedure and the path integral approach to QFT require the definition of a Lagrangian density to describe the theory. For scalar fields, the Klein-Gordon Lagrangian is

$$\mathcal{L}_{\mathrm{KG}} = \frac{1}{2}\partial^\mu\phi\partial_\mu\phi - \frac{1}{2}m^2\phi^2, \tag{1.96}$$

the Lagrangian for vector fields is

$$\mathcal{L}_{\mathrm{EM}} = -\frac{1}{4}F^{\alpha\beta}F_{\alpha\beta} - A_\alpha J^\alpha, \tag{1.97}$$

and the Dirac Lagrangian for spinor fields is

$$\mathcal{L}_{\mathrm{D}} = \bar{\psi}\left(i\partial\!\!\!/ - m\right)\psi. \tag{1.98}$$

The action of each field configuration is found by integrating the corresponding Lagrangian density over spacetime

$$S = \int_{\mathcal{M}} \mathrm{d}^4x\,\mathcal{L}, \tag{1.99}$$

where $\mathcal{L} = \mathcal{L}\left(\phi,\, \partial_\mu\phi\right)$ for typical scalar field theories. By varying the action with respect to the fields and implicating that physical field configurations are those for which the action remains stationary, the equations of motion of the fields are redetermined.

The symmetries of theories written down manifest themselves in the partition function. Simple field theories, such as scalar quartic field theory,

$$\mathcal{L} = \frac{1}{2}\left(\partial_\mu\phi\partial^\mu\phi - m^2\phi^2\right) - \frac{1}{4!}\lambda\phi^4, \tag{1.100}$$

exhibit a symmetry that can be spontaneously broken due to their potentially non-zero vacuum expectation value. In the case where ϕ is real, the symmetry here is Z_2, in which the Lagrangian is unchanged under $\phi \to -\phi$. The Higgs field carries a similar symmetry to this. In the absence of gauge fields, the Higgs Lagrangian is

$$\mathcal{L}_{\mathrm{Higgs}} = \frac{1}{2}\left(\partial_\mu\phi^*\partial^\mu\phi + \mu^2\phi^*\phi\right) - \frac{1}{4!}\lambda\left(\phi^*\phi\right)^2. \tag{1.101}$$

This Lagrangian carries with it a global $U(1)$ symmetry, meaning it is unchanged under the transformation $\phi \to e^{i\theta}\phi$, where θ is a real constant. For $\mu^2 > 0$ the symmetry is broken. As a result of this symmetry breaking, by Goldstone's theorem, a massless Goldstone boson appears in the theory. Excitations along the vacuum manifold correspond to these Goldstone modes and excitations in the radial direction correspond to the Higgs. Once the theory is coupled to electromagnetism via the introduction of gauge fields, the symmetry becomes local (in that the Lagrangian is unchanged even if $\theta = \theta(x)$). When the local $U(1)$ symmetry is broken, the gauge fields *"eat"* the massless Goldstone bosons causing them to become massive, and simultaneously remove these (unwanted) massless bosons from the theory. It is thought that through this process, where the non-zero vacuum expectation of the Higgs field causes certain particles to gain mass, a significant portion of the mass of many fundamental particles is generated [34].

Particularly in the path integral formulation of QFT, the partition function plays a central role in characterising the statistical behaviour of quantum fields and in the extraction of crucial physical information, such as particle number densities, correlation functions, and scattering amplitudes. As discussed briefly in 1.6.2, it provides a systematic framework for calculating expectations of field operators by acting as a generating functional. The partition function in quantum field theory can be expressed in terms of an exponential of the action

$$\mathcal{Z}\left[J\right] = \int \mathcal{D}\left[\phi\left(\mathbf{x}\right)\right]\, e^{iS\left[\phi(\mathbf{x}),\, J\right]}, \tag{1.102}$$

where $\mathcal{D}\left[\phi\left(\mathbf{x}\right)\right]$ is the path integral for ϕ. Later, it will be seen that the partition function of restricted Boltzmann machines with linear visible-hidden layer couplings can be written as

$$Z\left(\theta\right) = \int \mathcal{D}\mathbf{v}\, \mathcal{D}\mathbf{h}\, e^{-U_{v,\,h}(\mathbf{v},\,\mathbf{h}\ \theta)}. \tag{1.103}$$

This will be discussed and demonstrated more formally in 2.1.3, although it is worth being aware of *Wick rotations*, a mathematical tool used used to move calculations in QFT from Minkowski space to Euclidean space which cause the partition function (1.102) to become

$$Z\left[J\right] = \int \mathcal{D}\left[\phi\left(x\right)\right] e^{-S_E\left[\phi(x)\right]}, \tag{1.104}$$

where S_E is the *Euclidean action* [35]. For now, however, it is enough to notice the functional similarity between the two equations.

Notably, in a path integral, a sum over all possible field configurations is taken, taking into account the contributions to the amplitude e^{iS} of each possible configuration in order to determine the dynamics of the system. Expectations of operators can be computed this way. The formulation somewhat seamlessly handles the classical and quantum fields; it combines the principles of classical mechanics with the probabilistic nature of quantum mechanics. By summing over all possible field configurations, quantum fluctuations are incorporated, and their effects contribute to the overall dynamics of the system.

On top of this, the path integral formulation offers a more natural framework for incorporating external and internal symmetries into quantum field theories. By manifesting symmetries as invariances under certain transformations under the path integral, more deep understandings of symmetries underlying physical theories can be made.

Exact calculations of the partition function in QFT are often challenging due to the intricate nature of field interactions. To overcome various challenges, many techniques, such as regularisation, and various approximation techniques, have been developed and deployed in the context of QFT to gain insights into the nature of quantum fields. A popular approximation technique is perturbation theory [36], in which the partition function is expanded as a series in terms of a small coupling constant parameter. This is particularly effective in contexts where the interactions between fields are weak, making it legitimate to consider the perturbations as small.

The form of the KL divergence in (1.73) demonstrates that in contexts where the form of the target distribution P is unknown, the partition function must be considered in order to make progress. With this as motivation to attempt a calculation of the partition function, perhaps one of the most useful insights from Quantum Field Theory is in the calculation of the partition function of a quartic scalar field theory via a perturbative expansion about the partition function of a free scalar field theory. Consider the partition function $\mathcal{Z}\left[J\right]$ of a quartic field theory

$$\mathcal{Z}\left[J\right] = \int \mathcal{D}\phi \exp\left(\frac{i}{2}\int \mathrm{d}^4x \left\{\left(\partial_\mu\phi\right)^2 - \frac{\lambda}{4!}\phi^4 - m^2\phi^2\right\} + i\int \mathrm{d}^4x J\phi\right), \tag{1.105}$$

and the partition function of a free field theory

$$\mathcal{Z}_0\left[J\right] = \int \mathcal{D}\phi \exp\left(\frac{i}{2}\int \mathrm{d}^4x \left\{\left(\partial_\mu\phi\right)^2 - m^2\phi^2\right\} + i\int \mathrm{d}^4x J\phi\right), \tag{1.106}$$

both with the additional linear coupling $J\phi$. The free field theory partition function is simple to calculate by use of the propagator [33]. The calculation yields

$$\mathcal{Z}_0\left[J\right] = \exp\left(-\frac{1}{2}\int \mathrm{d}^4x\, \mathrm{d}^4y\, J(x)\Delta\left(x-y\right)J(y)\right), \tag{1.107}$$

where $\Delta\left(x,\, y\right)$ is the free scalar propagator

$$\Delta\left(x,\, y\right) = \int \frac{\mathrm{d}^4p}{\left(2\pi\right)^4}\frac{i\, e^{-ip\cdot(x-y)}}{p^2 - m^2 + i\epsilon}. \tag{1.108}$$

This can also be written as $\Delta\left(x-y\right)$. From this, $\mathcal{Z}_0\left[J\right]$ can now be used as a generating functional, as it can be used to calculate other quantities, such as the partition function (1.105). Appendix A.2 shows that the quartic partition function can be written in terms of the free theory partition function via

$$\mathcal{Z}\left[J\right] = \left[\exp\left(-\frac{i\lambda}{4!}\int \mathrm{d}^4z\, \frac{\delta^4}{\delta J(z)^4}\right)\right]\mathcal{Z}_0\left[J\right], \tag{1.109}$$

where $\frac{\delta}{\delta J(z)}$ represents a functional variation with respect to the coupling $J(z)$. Now, since λ is dimensionless, it can be taken as 'small' and treated as such. By expanding the exponential to linear order in λ, it can be seen that

$$\mathcal{Z}\left[J\right] = \left(1 - \frac{i\lambda}{4!}\int d^4z\, \frac{\delta^4}{\delta J(z)^4}\right)\mathcal{Z}_0\left[J\right] + \mathcal{O}\left(\lambda^2\right), \tag{1.110}$$

which necessitates the calculation of $\frac{\delta^4\mathcal{Z}_0[J]}{\delta J(z)^4}$. To do this, consider the successive results of varying $J(z)$ up until the fourth order. By use of the product rule, application of a single variation on $\mathcal{Z}_0\left[J\right]$ gives that

$$\frac{\delta \mathcal{Z}_0\left[J\right]}{\delta J(z)} = \left[-\int \mathrm{d}^4y\, \Delta\left(z-y\right) J(y)\right] \mathcal{Z}_0\left[J\right]. \tag{1.111}$$

A second time gives that

$$\frac{\delta^2 \mathcal{Z}_0\left[J\right]}{\delta J(z)^2} = \left[-\Delta\left(z-z\right) + \left\{\int \mathrm{d}^4y \Delta\left(z-y\right) J(y)\right\}^2\right] \mathcal{Z}_0\left[J\right]. \tag{1.112}$$

A third time gives that

$$\frac{\delta^3 \mathcal{Z}_0\left[J\right]}{\delta J(z)^3} = \left[3\Delta\left(z-z\right)\left\{\int \mathrm{d}^4y \Delta\left(z-y\right) J(y)\right\} - \left\{\int \mathrm{d}^4y\, \Delta\left(z-y\right) J(y)\right\}^3\right], \tag{1.113}$$

and finally,

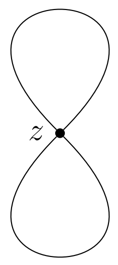
$$\begin{aligned}\frac{\delta^4 \mathcal{Z}_0\left[J\right]}{\delta J(z)^4} = \Bigg[&3\Delta\left(z-z\right)^2 - 6\Delta\left(z-z\right)\left\{\int \mathrm{d}^4y \Delta\left(z-y\right) J(y)\right\}^2 \\ &+ \left\{\int \mathrm{d}^4y \Delta\left(z-y\right) J(y)\right\}^4\Bigg] \mathcal{Z}_0\left[J\right].\end{aligned} \tag{1.114}$$

By integrating this over $\int d^4z$, this gives the final result for the quartic partition function up to linear order in λ as

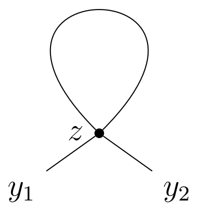
$$\begin{aligned}\mathcal{Z}[J] = \Bigg[1 - i\lambda\Bigg[&\frac{1}{8}\int \mathrm{d}^4z \Delta(z-z)^2 \\ &- \frac{1}{4}\left\{\int \mathrm{d}^4z\, \mathrm{d}^4y_1\, \mathrm{d}^4y_2 \Delta(z-z)\Delta(z-y_1)J(y_1)\Delta(z-y_2)J(y_2)\right\} \\ &+ \frac{1}{4!}\Bigg\{\int \mathrm{d}^4z\, \mathrm{d}^4y_1\, \mathrm{d}^4y_2\, \mathrm{d}^4y_3\, \mathrm{d}^4y_4 \Delta(z-y_1)J(y_1)\Delta(z-y_2)J(y_2) \\ &\qquad \Delta(z-y_3)J(y_3)\Delta(z-y_4)J(y_4)\Bigg\}\Bigg]\Bigg]\mathcal{Z}_0[J].\end{aligned} \tag{1.115}$$

This then allows for the simple computation of $\mathcal{Z}\left[J\right]$ up to first order in λ. In the context of QFT, such a calculation can be represented pictorially in terms of Feynman diagrams. These are shown in figure 1.4.

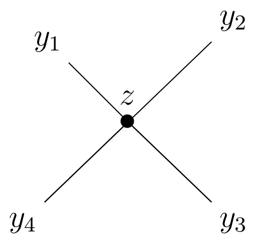
This result from Quantum Field Theory, where the partition function of an interacting scalar field theory was approximated by introducing a generating functional to be applied to the calculable partition function of the free scalar field theory, provides key insight into how the partition function for the Restricted Boltzmann Machines can be treated in order to make progress on understanding their behaviour. In particular, the quadratic dependence of the



$$\sim i\lambda \times \tfrac{1}{8} \int \mathrm{d}^4 z \, \Delta\left(z-z\right)^2$$



$$\sim +i\lambda \times \tfrac{1}{4} \int \mathrm{d}^4 z \, \mathrm{d}^4 y_1 \, \mathrm{d}^4 y_2 \, \Delta\left(z-z\right) J\left(y_1\right) \Delta\left(z-y_2\right) J\left(y_2\right)$$



$$\begin{aligned} &\sim -i\lambda \times \frac{1}{4!} \int \mathrm{d}^4 z \, \mathrm{d}^4 y_1 \, \mathrm{d}^4 y_2 \, \mathrm{d}^4 y_3 \mathrm{d}^4 y_4 \\ &\Delta\left(z-y_1\right) J\left(y_1\right) \Delta\left(z-y_2\right) J\left(y_2\right) \Delta\left(z-y_3\right) \\ &\times J\left(y_3\right) \Delta\left(z-y_4\right) J\left(y_4\right) \end{aligned}$$

FIGURE 1.4: Feynman diagrams corresponding to the first order corrections in λ to the interacting partition function $\mathcal{Z}\left[J\right]$.

free Lagrangian on the field values ϕ is what makes the free partition function simple to calculate, as the calculation for equation (1.107) utilises the Gaussian integral, which is further elaborated on in the appendix A.3. This technique motivates the intentional introduction of a quadratic dependence on the visible node values for the marginalised probability distribution for visible nodes in the RBMs, as the partition functions of such RBMs will be able to be expressed in terms of a related distribution that contains only this quadratic dependence between the nodes.

1.6.4 Field theory on the lattice

There are a few issues with the calculation above that should not be brushed over. The first of these is the propagator $\Delta\left(z-z\right)$ first seen in equation (1.112). The issue here is that it is not well defined. Consider its evaluation

$$\begin{aligned} \Delta\left(z-z\right) &= \int \frac{\mathrm{d}^4 p}{\left(2\pi\right)^4} \frac{i\, e^{-ip\cdot(z-z)}}{p^2-m^2+i\epsilon} \\ &= \int \frac{\mathrm{d}^4 p}{\left(2\pi\right)^4} \frac{i}{p^2-m^2+i\epsilon}. \end{aligned} \tag{1.116}$$

For large p^2, this is approximately

$$\Delta(z-z) \approx \int \frac{\mathrm{d}^4 p}{(2\pi)^4} \frac{i}{p^2}. \tag{1.117}$$

By changing to spherical coordinates with $\rho = |p|$, and using that the 'surface area' of a 3-sphere is $2\pi^2$, this becomes

$$\begin{aligned} \Delta(z-z) &\approx 2\pi^2 i \int^{\infty} \frac{\mathrm{d}\rho\, \rho^3}{(2\pi)^4} \frac{1}{\rho^2} \\ &= \frac{i}{8\pi^2} \int^{\infty} \rho\, \mathrm{d}\rho. \end{aligned} \tag{1.118}$$

This is clearly horribly divergent. However, if the physics is regulated by the replacement of $\int^{\infty} \mathrm{d}\rho \to \int^{\Lambda} \mathrm{d}\rho$, for some finite cutoff to the momentum Λ, then the integral is finite.

A possible way of motivating the introduction of this cutoff is by instead performing the above computations on the lattice [37]. This is where the points of spacetime are treated as a discrete lattice, and the fields are instead defined on the lattice points. This allows for the use of numerical techniques to approximate the partition function, as well as other expectation value calculations [38]. Furthermore, this process of defining QFTs on the lattice provides a well-defined framework for dealing with ultraviolet divergences in calculations appearing in QFT, facilitating calculations [39]. Lattice QFT has also succeeded in studying strong interactions and the phase transitions of quantum fields [40]. More formally, instead of working with points in spacetime $\mathcal{M}$, a discrete Euclidean spacetime Γ with a lattice spacing a is introduced. This is defined by

$$\Gamma = \left\{ x : x = \sum_{\mu=1}^{4} a n_\mu \hat{\mu},\ n_\mu \in \mathbb{Z} \right\}, \tag{1.119}$$

where here $\hat{\mu}$ are four unit basis vectors spanning the Euclidean spacetime. The lattice spacing introduces the ultraviolet cutoff to the theory through

$$a = \frac{1}{\Lambda}. \tag{1.120}$$

For the lattice to be a good approximation to a system, a must be significantly smaller than any other length scale at play in the system. The lattice spacing can be tuned to strike a balance between the computational efficiency of simulations and capturing the physics of relevant scales [41].

To get started in this new version, a discrete version of the action (1.99) is needed. Firstly, for each $\phi(x)$, x must now be contained within the lattice Γ. Kinetic terms are then written in terms of the lattice spacing a via the finite difference scheme

$$\partial_\mu \phi \longrightarrow \frac{\phi(x + a\hat{\mu}) - \phi(x)}{a}. \tag{1.121}$$

Finally, the integral over spacetime is replaced by a sum over the lattice

$$\int_{\mathcal{M}} \mathrm{d}^4 x \longrightarrow a^4 \sum_{x \in \Gamma} . \tag{1.122}$$

Following this procedure, the partition function (1.104) becomes

$$Z = \int \prod_{x \in \Gamma} \mathrm{d}\phi \, e^{-S}, \tag{1.123}$$

where now the action S is

$$S = a^4 \sum_{x \in \Gamma} \mathcal{L} \left(\phi(x), \, \frac{\phi(x + a\hat{\mu}) - \phi(x)}{a} \right) \tag{1.124}$$

A similar process to as in the continuous case can then be followed, but with all integrals over p now cutoff with $|p| \leq \Lambda$ by the lattice, leading to finite integrals.

Due to the discrete nature of the nodes in an RBM, the lattice structure introduced here is a more relevant analogy to draw between the RBM and physics situations. This is explored more deeply in 3.2 and in 3.3, where the connections to QFT are drawn from, but applied to an RBM with an explicitly discrete node structure. From the perspective of lattice field theory, the RBM can be conceptualized as a set of points represented by indices $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$, where the fields h_i and v_j are defined. However, unlike traditional QFT where the energy parameters are determined a priori, in the case of RBMs, these parameters in the energy function $E(\mathbf{v}, \mathbf{h}; \theta)$ are learned through a training process.

1.7 Structure of the thesis

The remainder of this thesis is divided into four further chapters. In Chapter 2, the algorithmic procedures for training RBMs is detailed, and the work of [42] is verified by which mathematical derivations and computational trainings of RBMs are reproduced. In Chapter 3, inspirations from QFT are taken to direct concentration onto a particular class of RBM to be investigated. In Chapter 4, a calculation demonstrating the existence of a sufficient condition for the improvement of a particular class of RBM is made. Chapter 5 then concludes the work.

Chapter 2

Learning Processes and Application of RBMs: Insights and Validation

"The general struggle for existence of animate beings is not a struggle for raw materials – these, for organisms, are air, water and soil, all abundantly available – nor for energy which exists in plenty in any body in the form of heat, but a struggle for [negative] entropy, which becomes available through the transition of energy from the hot Sun to the cold Earth."

-Ludwig Boltzmann

In this chapter, a comprehensive exploration of the learning process of RBMs is presented, followed by an application of these procedures to learning distributions formed in the one-dimensional Ising model. The objective is to better understand the RBM learning process, establish connections with theoretical frameworks, and reproduce and validate the results presented in a reference paper.

Section 2.1 begins by explaining the learning process for RBMs. This includes a demonstration of the link between Kullback-Leibler divergence minimisation and log-likelihood maximisation, the presentation of formulae for the iterative update of RBM parameters, and the introduction of the Contrastive Divergence (CD) algorithm. Section 2.2 then validates the formulae derived in section 2.1.2 by comparing them to a specific case study in a reference paper. It is demonstrated that the formulae there can be considered as a special case of the formulae presented, providing a solid basis for following analyses and acting as an accuracy check of the investigations. Section 2.3 then introduces the Ising model as the system of study in the reference paper. A subset of the results presented in the reference material are then successfully reproduced and validated. This validation showcases the efficacy of RBMs with a linear mixing term and simple biases in capturing essential features in interacting systems. By exploring the learning process of RBMs and validating findings against the established references, this chapter contributes to a comprehensive understanding of the capability of RBMs in their application to complex systems.

2.1 Understanding the RBM learning process

It is essential to understand the learning process now that motivation for studying Restricted Boltzmann Machines has been established and the reasons for using the Kullback-Leibler divergence as an objective function understood. Due to the relative power from the techniques later borrowed from QFT, the RBMs can be treated with significant generality. As such, the learning procedure will be developed without the need for very many constraints. Subsection 2.1.1 begins by elucidating a fundamental principle underlying the RBM learning process; minimisation of the KL divergence is equivalent to maximising the log-likelihood of observed data given a model's parameters. Next, subsection 2.1.2 outlines a general procedure for the computation of the gradient of the RBM log-likelihood when the RBM visible-hidden connections are only bilinear, laying the foundations for subsequent analyses. Section 2.1.3 then delves into calculating the marginal distributions for the hidden and visible layers of such RBMs. Additionally, a demonstration that the activation functions of nodes in these RBMs follow a generalised version of the sigmoid activation function is made. Finally, subsection 2.1.4 introduces the Contrastive Divergence algorithm as a popular method for approximating the RBM log-likelihood gradient, allowing for efficient RBM training.

2.1.1 KL divergence minimisation as likelihood maximisation

Firstly, it is noted that the minimisation of the KL divergence of a model with respect to some target is parallel to the maximisation of the log-likelihood of observing data from the target, given the model's parameters. This can be seen via the following arguments:

1. Given a model distribution $q\left(\mathbf{v}\middle|\,\theta\right)$ with parameters θ, and a set of data values S drawn from a target distribution $p\left(\mathbf{v}\right)$, the likelihood $\mathcal{L}\left(\theta\middle|\,S\right)$ of the parameters θ given the observed data set S is given by

$$\mathcal{L}\left(\theta\middle|\,S\right) = \prod_{\mathbf{v}\in S} q\left(\mathbf{v}|\theta\right), \tag{2.1}$$

as $q\left(\mathbf{v}\middle|\,\theta\right)$ is the model probability for each observed datum $\mathbf{v}$.

2. The function $\log\left(x\right)$ is increasing in x, and so the parameter choice maximising $\mathcal{L}$ will also maximise $\log\left(\mathcal{L}\right)$.

3. By the logarithmic product rule, it then follows that

$$\text{argmax}_\theta \mathcal{L}\left(\theta\middle|\,S\right) = \text{argmax}_\theta \sum_{\mathbf{v}\in S} \log\left(q\left(\mathbf{v}\middle|\,\theta\right)\right). \tag{2.2}$$

4. Since the observed data $\mathbf{v} \in S$ are drawn from P,

$$\text{argmax}_\theta \sum_{\mathbf{v}\in S} \log\left(q\left(\mathbf{v}\middle|\,\theta\right)\right) = \text{argmax}_\theta \mathbb{E}_P\left[\log\left(q\left(\mathbf{v}\middle|\,\theta\right)\right)\right]. \tag{2.3}$$

5. By flipping a sign,

$$\text{argmax}_\theta \mathcal{L}\left(\theta\middle|\,S\right) = \text{argmin}_\theta \mathbb{E}_P\left[-\log\left(q\left(\mathbf{v}\middle|\,\theta\right)\right)\right]. \tag{2.4}$$

6. Since the underlying target distribution $p(\mathbf{v})$ is independent of the model parameters θ,
$$\text{argmax}_{\theta}\mathcal{L}\left(\theta|\, S\right) = \text{argmin}_{\theta}\mathbb{E}_P\left[\log\left(p\left(\mathbf{v}\right)\right) - \log\left(q\left(\mathbf{v}|\,\theta\right)\right)\right]. \tag{2.5}$$

7. Again, by the product rule for logs, this is exactly the Kullback-Leibler divergence as seen in (1.19), meaning
$$\text{argmax}_{\theta}\mathcal{L}\left(\theta|\, S\right) = \text{argmin}_{\theta}D_{\text{KL}}\left(p\left(\mathbf{v}\right)||\, q\left(\mathbf{v}|\,\theta\right)\right). \tag{2.6}$$

During the development of the learning process, it is more mathematically convenient to phrase KL divergence minimisation as likelihood maximisation, and so the learning process will be expressed through this lens.

Given a function of many variables $f(\mathbf{r})$ that undergoes a variation of arguments $\mathbf{r} \to \mathbf{r} + \delta\mathbf{r}$, to first order in the variation, the function itself varies as
$$f\left(\mathbf{r} + \delta\mathbf{r}\right) = f\left(\mathbf{r}\right) + \left(\nabla f\right)\cdot\delta\mathbf{r} + \ldots. \tag{2.7}$$

The change in the function value $(\nabla f)\cdot\delta\mathbf{r}$ is then maximised (positively) when $\delta\mathbf{r}$ is aligned with ∇f. Using this idea for the log-likelihood function, by updating all parameters $\theta_i \to \theta'_i = \theta_i + \delta\theta_i$ such that $\delta\theta$ is aligned with the gradient of $\log\left(\mathcal{L}\left(\theta|\, S\right)\right)$, the log-likelihood of the data batch given the parameters can be increased. By defining a small positive constant of proportionality α, known as the *learning rate*, the updating of the parameters via
$$\theta \to \theta' = \theta + \alpha\,\nabla_{\theta}\log\mathcal{L}\left(\theta|S\right) \tag{2.8}$$
will cause the update of the log-likelihood to be
$$\log\mathcal{L}\left(\theta'|S\right) = \log\mathcal{L}\left(\theta|S\right) + \alpha|\left(\nabla_{\theta}\log\mathcal{L}\right)|^2 + \cdots, \tag{2.9}$$
as seen by substitution of the $\log\mathcal{L}$ and θ' into (2.7). Such a procedure allows for an iterative gradient descent process towards a maximum of log-likelihood and hence a minimum of Kullback-Leibler divergence between the model and the target.

2.1.2 Gradient of the RBM log-likelihood

In order to follow the update procedure (2.8), the gradient of the log-likelihood of the model parameters must be calculated. This is a mathematically straightforward procedure; given a batch of data S, the derivative of the log-likelihood with respect to the i^{th} model parameter θ_i is given by
$$\frac{\partial}{\partial\theta_i}\log\left(\mathcal{L}\left(\theta|\, S\right)\right) = \sum_{\mathbf{v}\in S}\frac{\partial\log\left(q\left(\mathbf{v}|\,\theta\right)\right)}{\partial\theta_i}. \tag{2.10}$$

Now, in the case of a Restriced Botlzmann Machine, the model distribution $q(\mathbf{v}|\,\theta)$ is first found by marginalising over the hidden nodes via
$$q\left(\mathbf{v}|\,\theta\right) = \sum_{\mathbf{h}} q\left(\mathbf{v},\,\mathbf{h}|\,\theta\right), \tag{2.11}$$

where $q\left(\mathbf{v},\, \mathbf{h};\, \theta\right)$ is the usual Boltzmann distribution in terms of the model energy $E\left(\mathbf{v},\, \mathbf{h};\, \theta\right)$,

$$q\left(\mathbf{v},\, \mathbf{h};\, \theta\right) = \frac{1}{Z} e^{-E\left(\mathbf{v},\, \mathbf{h};\, \theta\right)}, \tag{2.12}$$

and the sum $\sum_{\mathbf{h}}$ is over all possible configurations of the hidden layer nodes. Due to this, the gradient of the log-likelihood will depend on some sum over configurations of the hidden layer. By using Bayes' theorem to write the marginalised probability distribution for the hidden layer configurations $q\left(\mathbf{h}|\, \mathbf{v}; \theta\right)$ as

$$q\left(\mathbf{h}|\, \mathbf{v}; \theta\right) = \frac{q\left(\mathbf{v},\, \mathbf{h};\, \theta\right)}{q\left(\mathbf{v}|\, \theta\right)}, \tag{2.13}$$

the gradient of the log-likelihood of the parameters with respect to the i^{th} parameter θ_i given an observed datum $\mathbf{v}$ can be written as

$$\frac{\partial}{\partial \theta_i}\left(\log\left(q\left(\mathbf{v}|\, \theta\right)\right)\right) = -\sum_{\mathbf{h}} q\left(\mathbf{h}|\, \mathbf{v};\, \theta\right) \frac{\partial E\left(\mathbf{v},\, \mathbf{h};\, \theta\right)}{\partial \theta_i} + \sum_{\mathbf{v}',\, \mathbf{h}} q\left(\mathbf{v}',\, \mathbf{h};\, \theta\right) \frac{\partial E\left(\mathbf{v}',\, \mathbf{h};\, \theta\right)}{\partial \theta_i}, \tag{2.14}$$

which is derived fully in appendix A.4. Note here that the double sum in the final term of (2.14) comes from the normalisation factor of the partition function (which introduces $\sum_{\mathbf{v}'}$, the sum over configurations of the visible layer). Following this, the average gradient over a full batch of data S is then given by

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S} \frac{\partial \log \mathcal{L}\left(\theta\, |S\right)}{\partial \theta_i}$$

$$= \frac{1}{|S|}\sum_{\mathbf{v}\in S}\left[-\sum_{\mathbf{h}} q\left(\mathbf{h}|\, \mathbf{v};\, \theta\right) \frac{\partial E\left(\mathbf{v}, \mathbf{h};\, \theta\right)}{\partial \theta_i} + \sum_{\mathbf{v}',\, \mathbf{h}} q\left(\mathbf{v}',\, \mathbf{h}|\theta\right) \frac{\partial E\left(\mathbf{v}',\, \mathbf{h}; \theta\right)}{\partial \theta_i}\right], \tag{2.15}$$

where the sums over $\mathbf{v}'$ and $\mathbf{h}$ are over all possible configurations. From here, the double sum can be split by first rearranging (2.13) for $q\left(\mathbf{v}',\, \mathbf{h};\, \theta\right)$, taking the sum and then decomposing it as

$$\sum_{\mathbf{v}',\, \mathbf{h}} q\left(\mathbf{v}',\, \mathbf{h};\, \theta\right) = \sum_{\mathbf{v}'} q\left(\mathbf{v}'|\, \theta\right) \sum_{\mathbf{h}} q\left(\mathbf{h}|\, \mathbf{v}';\, \theta\right). \tag{2.16}$$

From here, it is convenient to introduce the following two pieces of often-used notation:

$$\mathbb{E}_{\text{data}}\left[F\left(\mathbf{v}\right)\right] = \frac{1}{|S|}\sum_{\mathbf{v}\in S} F\left(\mathbf{v}\right), \tag{2.17}$$

and

$$\mathbb{E}_{\text{model}}\left[F\left(\mathbf{v}\right)\right] = \frac{1}{|S|}\sum_{\mathbf{v}\in V} q\left(\mathbf{v}|\, \theta\right) F\left(\mathbf{v}\right). \tag{2.18}$$

Given the decomposition (2.16) and definitions of $\mathbb{E}_{\text{data}}$ and $\mathbb{E}_{\text{model}}$, the average gradient can be more concisely written as

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S} \frac{\partial \log \mathcal{L}\left(\theta\, |S\right)}{\partial \theta_i}$$

$$= \mathbb{E}_{\text{data}}\left[-\sum_{\mathbf{h}} q\left(\mathbf{h}|\,\mathbf{v};\,\theta\right)\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial \theta_i}\right] - \mathbb{E}_{\text{model}}\left[-\sum_{\mathbf{h}} q\left(\mathbf{h}|\,\mathbf{v};\,\theta\right)\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial \theta_i}\right]. \tag{2.19}$$

This is a useful way to express the gradient of the log-likelihood, as it both highlights the necessity of calculating the quantity

$$\sum_{\mathbf{h}} q\left(\mathbf{h}|\,\mathbf{v};\,\theta\right)\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial \theta_i},$$

and also provides a conceptual framework by which to understand the maximisation of the log-likelihood. Under such a notation, it is understood that by maximising the log-likelihood (and consequently making its gradient zero), the model-averaged value of the above quantity is made to match the data averaged value of the same quantity, where the data is drawn from a representative sample of the target $p\left(\mathbf{v}\right)$.

Since the form of the activation function for the hidden layer $q\left(\mathbf{h}|\,\mathbf{v};\,\theta\right)$ is dependent on the energy function defining the model, progress can only be made from here by specifying $E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)$ to some degree. By restraining the class of RBMs under consideration to only those that have a bi-linear coupling as discussed in 1.2.3, the set of parameters defining the energy of configurations can be broken into three classes as $\theta = \left\{\omega_{ij},\, f^{(v)}_{j\alpha},\, f^{(h)}_{i\alpha}\right\}$, where

1. ω_{ij} are cross terms encoding the interactions between each visible node j and each hidden node i.
2. $f^{(v)}_{j\alpha}$ are a set of parameters characterising the potential U_j of each visible node j.
3. $f^{(h)}_{i\alpha}$ are a set of parameters characterising the potential $\tilde{U}_i$ of each hidden node i.

The label α in $f^{(v)}_{j\alpha}$ and $f^{(h)}_{i\alpha}$ is used to remind of the fact that there may be a number of parameters determining the potential of each node in the RBM. In this notation, the energy of the configurations can be written as

$$E\left(\mathbf{v},\,\mathbf{h};\,\theta\right) = -\left(\sum_{j=1}^{m} U_j\left(v_j,\, f^{(v)}_{j\alpha}\right) + \sum_{i=1}^{n}\tilde{U}_i\left(h_i,\, f^{(h)}_{i\alpha}\right) + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i\omega_{ij}v_j\right). \tag{2.20}$$

The derivatives of the energy with respect to each of these are then

$$\begin{aligned}
\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial \omega_{ij}} &= -h_i v_j, \\
\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial f^{(v)}_{j\alpha}} &= -\frac{\partial U_j\left(v_j,\, f^{(v)}_{j\alpha}\right)}{\partial f^{(v)}_{j\alpha}}, \text{ and} \\
\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial f^{(h)}_{i\alpha}} &= -\frac{\partial \tilde{U}_i\left(h_i,\, f^{(h)}_{i\alpha}\right)}{\partial f^{(h)}_{i\alpha}}.
\end{aligned} \tag{2.21}$$

Then for each parameter θ_i, the relevant expression from here can be substituted back into 2.19 in order to calculate the relevant term composing the gradient. Note from here that the dependence of the node potentials on the parameters $f^{(v)}_{j\alpha}$ and $f^{(h)}_{i\alpha}$ should be understood as implicit. That is, it is taken that

$$U_j\left(v_j\right) \equiv U_j\left(v_j,\, f^{(v)}_{j\alpha}\right), \text{ and } \tilde{U}_i\left(h_i\right) \equiv \tilde{U}_i\left(h_i,\, f^{(h)}_{i\alpha}\right) \tag{2.22}$$

for the remainder of this thesis.

2.1.3 Marginal distributions and the node activation functions

Once the form of the energy (2.20) has been specified, many quantities of use can be calculated. To begin, the partition function is a sum over all possible configurations of both the visible and hidden layer

$$Z\left(\theta\right) = \int_{V\times H} \mathrm{d}\mathbf{v}\,\mathrm{d}\mathbf{h}\,\exp\left(\sum_{j=1}^{m} U_j(v_j) + \sum_{i=1}^{n} \tilde{U}_i(h_i) + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i\omega_{ij}v_j\right), \tag{2.23}$$

where V is the domain of the state of the visible layer, and H is the domain of the state of the hidden layer. This can be expressed instead via a *field integral*; mathematical analogues to Feynman's field path integrals for fields in quantum field theory. By defining the field integral of a function $f\left(\mathbf{x}\right)$ as

$$\int \mathcal{D}\mathbf{x}\, f\left(\mathbf{x}\right) = \prod_{\alpha=1}^{\dim(\mathbf{x})} \int \mathrm{d}x_\alpha\, f_\alpha\left(x_\alpha\right), \tag{2.24}$$

where the integral on the right-hand side is over the domain of the function $f_i\left(x_i\right)$, and where

$$f\left(\mathbf{x}\right) = \prod_{i=\alpha}^{\dim(\mathbf{x})} f_\alpha\left(x_\alpha\right), \tag{2.25}$$

the partition function can be re-expressed (see Appendix A.5) as

$$Z\left(\theta\right) = \int \mathcal{D}\mathbf{v}\,\mathcal{D}\mathbf{h}\, e^{-U_{v,\,h}(\mathbf{v},\,\mathbf{h}\;\theta)}, \tag{2.26}$$

where

$$U_{v,\,h}\left(\mathbf{v},\,\mathbf{h};\,\theta\right) = -\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\frac{U_j(v_j)}{n} + \frac{\tilde{U}_i(h_i)}{m} + h_i\omega_{ij}v_j\right). \tag{2.27}$$

Similarly, by defining the functions

$$U_h\left(\mathbf{v},\,\mathbf{h}\right) = -\sum_{i=1}^{n}\left[\tilde{U}_i(h_i) + h_i\sum_{j=1}^{m}\omega_{ij}v_j\right], \tag{2.28}$$

and

$$U_v\left(\mathbf{v},\,\mathbf{h}\right) = -\sum_{j=1}^{m}\left[U_j(v_j) + v_j\sum_{i=1}^{n}h_i\omega_{ij}\right], \tag{2.29}$$

the marginalised model probability distributions for the visible and hidden nodes can be found as

$$q\left(\mathbf{v}\middle|\,\theta\right) = \frac{1}{Z\left(\theta\right)} \left(\prod_{j=1}^{m} e^{U_j(v_j)}\right) \int \mathcal{D}\mathbf{h}\, e^{-U_h(\mathbf{v},\,\mathbf{h})}, \tag{2.30}$$

and

$$q\left(\mathbf{h}\middle|\,\theta\right) = \frac{1}{Z\left(\theta\right)} \left(\prod_{i=1}^{m} e^{\tilde{U}_i(h_i)}\right) \int \mathcal{D}\mathbf{v}\, e^{-U_v(\mathbf{v},\,\mathbf{h})}. \tag{2.31}$$

These are useful stepping stones in the calculation of the gradient of the log-likelihood (2.19). This is because, from here, it is simple to calculate the probability that a visible or hidden node takes a certain value given the model parameters. As mentioned in 1.1.1, this is the *activation function* of the nodes.

At this stage, it is assumed that all visible and hidden nodes have at least a linear contribution to their energy. That is, each visible node j has a parameter $f^{(v)}_{j\alpha} = b_j$ and each hidden node i has a parameter $f^{(h)}_{i\alpha} = c_i$ such that

$$U_j(v_j) = b_j v_j + U'_j(v_j), \tag{2.32}$$

and

$$\tilde{U}_i(h_i) = c_i h_i + \tilde{U}'_i(h_i), \tag{2.33}$$

where U'_j and $\tilde{U}'_i$ contain all non-linear dependence of each node's potential on v_j and h_i respectively. This is without loss of generality, as the following still holds when $b_j = c_i = 0$. From here, to begin the calculation of the activation functions, it is constructive to consider two quantities. The first of these is $\alpha_\ell\left(\mathbf{h}\right)$, which is the contribution per unit v_ℓ of the ℓ^{th} visible node's linear part of the contribution to its energy, for a given hidden layer configuration;

$$\alpha_\ell\left(\mathbf{h}\right) := -\sum_{i=1}^{n} \omega_{i\ell} h_i - b_\ell. \tag{2.34}$$

The second quantity to consider is β_ℓ, which is the energy of the whole system excluding visible node ℓ;

$$\beta_\ell\left(\mathbf{v}_{-\ell},\,\mathbf{h}\right) := -\sum_{i=1}^{n}\sum_{j\neq\ell}^{m} \omega_{ij} h_i v_j - \sum_{j\neq\ell}^{m}\left(b_j v_j + U'_j(v_j)\right) - \sum_{i=1}^{n}\left(c_i h_i + \tilde{U}'_i(h_i)\right). \tag{2.35}$$

These quantities are useful together as they allow for the writing of the energy of the system as

$$E\left(\mathbf{v},\,\mathbf{h};\,\theta\right) = \beta_\ell\left(\mathbf{v}_{-\ell},\,\mathbf{h}\right) + v_\ell \alpha_\ell\left(\mathbf{h}\right) - U'_\ell\left(v_\ell\right), \tag{2.36}$$

for any choice of visible node ℓ. Given this, consider the probability that visible node ℓ takes on the value v_ℓ, which will be written $q\left(v_\ell\middle|\,\mathbf{h}\right)$. It is useful to begin by writing this as a conditional probability over all other visible nodes, $\mathbf{v}_{-\ell}$:

$$q\left(v_\ell\middle|\,\mathbf{h}\right) = q\left(v_\ell\middle|\mathbf{v}_{-\ell},\,\mathbf{h}\right), \tag{2.37}$$

which, by Bayes' theorem, gives

$$q\left(v_\ell|\,\mathbf{h}\right) = \frac{q\left(v_\ell,\,\mathbf{v}_{-\ell},\,\mathbf{h}\right)}{q\left(\mathbf{v}_{-\ell},\,\mathbf{h}\right)}. \tag{2.38}$$

Summing in the denominator over all possible configurations the ℓ^{th} visible unit given $v_{-\ell}$ and cancelling factors of the partition function gives

$$q\left(v_\ell|\,\mathbf{h}\right) = \frac{\exp\left(-E\left(v_\ell,\,\mathbf{v}_{-\ell},\,\mathbf{h}\right)\right)}{\sum_{a=1}^{K}\exp\left(-E\left(v_\ell,\mathbf{v}_{-\ell},\,\mathbf{h}\right)\right)}, \tag{2.39}$$

where K is the number of allowed configurations of the ℓ^{th} visible node. Breaking these expressions into α_ℓ and β_ℓ now gives

$$q\left(v_\ell|\,\mathbf{h}\right) = \frac{\exp\left(-\beta_\ell\left(\mathbf{v}_{-\ell},\,\mathbf{h}\right) - v_\ell\alpha_\ell\left(\mathbf{h}\right) + U'_\ell\left(v_\ell\right)\right)}{\sum_{a=1}^{K}\exp\left(-\beta_\ell\left(\mathbf{v}_{-\ell},\,\mathbf{h}\right) - v_a\alpha_\ell\left(\mathbf{h}\right) + U'_\ell\left(v_a\right)\right)}. \tag{2.40}$$

Now factorising the content of the numerator out of both the numerator and denominator leaves the result

$$q\left(v_\ell|\,\mathbf{h}\right) = \left[\sum_{a=1}^{K}\exp\left\{-\alpha_\ell\left(\mathbf{h}\right)\left(v_a - v_\ell\right) + \left(U'_\ell(v_a) - U'_\ell(v_\ell)\right)\right\}\right]^{-1} \tag{2.41}$$

$$= \left[\sum_{a=1}^{K}\exp\left(\sum_{i=1}^{n} h_i\omega_{i\ell}\left(v_a - v_\ell\right) + U_\ell\left(v_a\right) - U_\ell\left(v_\ell\right)\right)\right]^{-1}. \tag{2.42}$$

Written this way, the probability distribution for a single visible node's value v_ℓ can be written fully in terms of the state of the hidden layer and the parameters of only the ℓ^{th} visible node. Because the visible and hidden layers are identical up to labelling, the derivation of the activation function for the hidden nodes follows an identical procedure. The results of this procedure are that

$$q\left(h_\ell|\,\mathbf{v}\right) = \left[\sum_{a=1}^{K'}\exp\left\{-\tilde{\alpha}_\ell\left(\mathbf{v}\right)\left(h_a - h_\ell\right) + \left(\tilde{U}'_\ell(h_a) - \tilde{U}'_\ell(h_\ell)\right)\right\}\right]^{-1} \tag{2.43}$$

$$= \left[\sum_{a=1}^{K'}\exp\left(\sum_{j=1}^{m} v_j\omega_{\ell j}\left(h_a - h_\ell\right) + \tilde{U}_\ell\left(h_a\right) - \tilde{U}_\ell\left(h_\ell\right)\right)\right]^{-1}, \tag{2.44}$$

where K' is the number of possible configurations of h_ℓ, and

$$\tilde{\alpha}_\ell\left(\mathbf{v}\right) := -\sum_{j=1}^{m} v_j\omega_{\ell j}h_i - c_\ell. \tag{2.45}$$

This result for the hidden node activation function can then be inserted into the expression for the gradient of the log-likelihood (2.19) fully specifying the quantities that need calculating in order to specify the update procedure for the RBM parameters. Before moving on, it is interesting to note here that activation function (2.41) is actually just a form of the softmax function, which is the multivariate generalisation of the sigmoid function mentioned in 1.1.3.

2.1.4 The Contrastive Divergence algorithm

At this point, it is important to know that a popular algorithm used in the training of Restricted Boltzmann Machines is the Contrastive Divergence (CD) algorithm. The CD algorithm is a Markov Chain Monte Carlo (MCMC) process that approximates the model average term in the equation for the log-likelihood gradient. This procedure is employed because the exact calculation of that term requires a sum over all possible states of the system, making the calculation intractable for most systems. To navigate this issue, the CD algorithm approximates this gradient by contrasting the distributions observed in two phases of the algorithm, known as the positive and negative phases.

The positive phase of the CD algorithm is where the model receives exposure to the observed data. The configurations of the visible nodes are set equal to the data, and correspondingly, each hidden layer node is updated probabilistically using their activation functions in terms of the visible layer configuration. This phase establishes the dependency between the visible and hidden units. During the negative phase, the model generates its own data sample (i.e., visible node configuration) using the activation function in terms of the hidden layer configuration set in the positive phase. This process of taking the visible node configuration, using it to probabilistically generate a hidden layer configuration, followed by probabilistically generating a second visible layer configuration, is known as a Gibbs sampling. This process is repeated k times

$$\mathbf{v} \to \mathbf{h}^{(1)} \to \mathbf{v}^{(1)} \to \mathbf{h}^{(2)} \to \ldots \to \mathbf{h}^{(k)} \to \mathbf{v}^{(k)}, \tag{2.46}$$

allowing the RBM to explore the joint sample space of both the hidden and visible layers in order to form a configuration representative of the model average configuration, from which $E_{\text{model}}[\cdot]$ can be approximated. The number of Gibbs steps k is usually set to a small value (1 or 2) to keep the approximation's computational complexity low. However, it can be increased in order to improve the quality of the approximation for the gradient. The essence of the Gibbs sampling phase is that during this phase, the RBM uses a vector from the data distribution to generate a sample according to the model distribution. That is, $\mathbf{v}^{(k)}$ can be considered as generated by the RBM. This is because each step involves sampling from the conditional probabilities as determined by the RBM to determine the state of the units. The final sample then approximates the RBM's model distribution.

Once the k^{th} step has been performed, the algorithm computes the difference between model averages over the initially observed data and the k^{th} Gibbs step configuration. This procedure approximates the equation for the gradient of the log-likelihood, allowing for a much quicker learning process.

2.2 Verification of formulae

Section 2.1 showed a set of general equations describing the parameter tuning process to minimize the KL divergence between an RBM's model distribution and a target distribution. In the following section, these general equations will be applied to specific cases by selecting appropriate parameter values and forms of the RBM energy function, allowing for the reproduction of the results presented in the referenced paper [42], which is a study of the training of a simple RBMs parameters to learn the behaviour of the Ising model. By specializing the equations to this specific case, it will be demonstrated that the derived equations capture the essence of the results obtained in the paper.

In the paper, all nodes are binary, taking either values 0 or 1. The RBM is also chosen such that the potential energy of all nodes is linear in the node value. As such, the energy of a configuration of the RBM there is given by

$$E\left(\mathbf{v},\,\mathbf{h};\,\theta\right) = -\left(\sum_{j=1}^{m} b_j v_j + \sum_{i=1}^{n} c_i h_i + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i \omega_{ij} v_j\right). \tag{2.47}$$

Since the nodes are binary-valued, the formulae for the activation functions (2.41) and (2.43) simplify massively. Because of the form of (2.47), $U_\ell'\left(v_a\right)$ is zero for all visible nodes, and $\tilde{U}_\ell'\left(h_a\right)$ is zero for all hidden nodes. Following this, the activation probability (the probability that a node takes a value of 1 rather than zero) for the visible nodes can be calculated as

$$\begin{aligned} q\left(v_\ell = 1 \middle|\, \mathbf{h}\right) &= \left[\sum_{a=1}^{2} \exp\left\{-\alpha_\ell\left(\mathbf{h}\right)\left(h_a - 1\right)\right\}\right]^{-1} \\ &= \left[1 + \exp\left(\alpha_\ell\left(\mathbf{h}\right)\right)\right]^{-1} \\ &= \sigma\left(\sum_{i=1}^{n} h_i \omega_{i\ell} + b_\ell\right). \end{aligned} \tag{2.48}$$

A similar calculation for the hidden nodes shows that

$$q\left(h_\ell = 1 \middle|\, \mathbf{v}\right) = \sigma\left(\sum_{j=1}^{m} \omega_{\ell j} v_j + c_\ell\right). \tag{2.49}$$

Both of these are in exact agreement with the formulae for the activation probabilities derived in the referenced paper [42].

Next, the relevant derivatives with respect to each parameter are calculated. As in the general case

$$\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\theta\right)}{\partial \omega_{ij}} = -h_i v_j, \tag{2.50}$$

but now the only parameters determining the potential energy terms of the nodes are b_j and c_i, meaning the only other derivatives needing calculation are

$$\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial b_j} = -v_j, \tag{2.51}$$

and

$$\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial c_i} = -h_i. \tag{2.52}$$

Having calculated these, the derivative of the log-likelihood gradient with respect to each parameter can be calculated. By use of (2.19), it is seen that

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S}\frac{\partial \log \mathcal{L}(\theta\,|S)}{\partial \omega_{ij}} = \mathbb{E}_{\text{data}}\left[\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta)\,h_i v_j\right] - \mathbb{E}_{\text{model}}\left[\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta)\,h_i v_j\right]. \tag{2.53}$$

Similarly,

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S}\frac{\partial \log \mathcal{L}(\theta\,|S)}{\partial b_j} = \mathbb{E}_{\text{data}}\left[\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta)\,v_j\right] - \mathbb{E}_{\text{model}}\left[\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta)\,v_j\right], \tag{2.54}$$

and

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S}\frac{\partial \log \mathcal{L}(\theta\,|S)}{\partial c_i} = \mathbb{E}_{\text{data}}\left[\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta)\,h_i\right] - \mathbb{E}_{\text{model}}\left[\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta)\,h_i\right]. \tag{2.55}$$

This introduces the form of the energy into the algorithm for minimising the model KL divergence. However, the allowed node values must also be specified. Since all nodes take on binary values, the sum $\sum_{\mathbf{h}}$ can be split into the cases where the i^{th} hidden node is 0 or 1. This simplifies the sum via

$$\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta) = 1, \tag{2.56}$$

and

$$\sum_{\mathbf{h}} q(\mathbf{h}|\,\mathbf{v};\,\theta)\;h_i = q(h_i = 1|\,\mathbf{v};\,\theta), \tag{2.57}$$

with $q(h_i = 1|\,\mathbf{v};\,\theta)$ as given in (2.49). Using this trick to simplify the expressions for the derivatives of the log-likelihood with respect to each parameter then gives the final results of

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S}\frac{\partial \log \mathcal{L}(\theta\,|S)}{\partial \omega_{ij}} = \mathbb{E}_{\text{data}}\left[q(h_i = 1|\,\mathbf{v};\,\theta)\,v_j\right] - \mathbb{E}_{\text{model}}\left[q(h_i = 1|\,\mathbf{v};\,\theta)\,h_i v_j\right] \tag{2.58}$$

for the mixing term,

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S}\frac{\partial \log \mathcal{L}(\theta\,|S)}{\partial b_j} = \mathbb{E}_{\text{data}}[v_j] - \mathbb{E}_{\text{model}}[v_j] \tag{2.59}$$

for the visible node bias parameter, and

$$\frac{1}{|S|}\sum_{\mathbf{v}\in S}\frac{\partial \log \mathcal{L}(\theta\,|S)}{\partial c_i} = \mathbb{E}_{\text{data}}\left[q(h_i = 1|\,\mathbf{v};\,\theta)\right] - \mathbb{E}_{\text{model}}\left[q(h_i = 1|\,\mathbf{v};\,\theta)\right]. \tag{2.60}$$

for the hidden node bias parameter. This specification of the form of the RBM energy and of the allowed node configurations correctly reduces the general form of the gradient of the log-likelihood to the formulae derived in the reference paper.

2.3 Reproduction of existing results: Application to the Ising model

The authors in the paper *"Machine learning determination of dynamical parameters: The Ising model case (2019)"* [42] chose to restrict the visible and hidden nodes of the RBM to binary values because they aimed to create a representation that closely aligns with the nature of the data associated with the Ising model. This choice allows the RBM to capture and model the essential characteristics and patterns inherent in the Ising model data, facilitating the learning and inference processes.

Subsection 2.3.1 defines the Ising model and motivates reasons for its study, both in the context of physics and in machine learning. This is followed by subsection 2.3.2 where material from the reference paper that was reproduced is presented.

2.3.1 The Ising model

The Ising model is a concept from statistical physics that was initially a model of magnetic systems but has since been used to study various physical phenomena. The model proposed by Wilhelm Lenz and later worked on by his student Ernst Ising provides a simplified representation of the interactions between the individual spins in a magnetic material. The model has applications in various fields, including computer science, condensed matter physics, and social sciences.

The model is based on the idea that magnetic materials are composed of a lattice of atomic units with a magnetic dipole or spin that can be either aligned or anti-aligned with an applied magnetic field. The model assumes that the only interactions between the spins only occur between nearest neighbours. The lattice can take various shapes and dimensionalities. Each spin is represented by a binary variable s_i, where $+1$ refers to the spin being aligned with the external magnetic field, and -1 refers to the spin being anti-aligned with the external magnetic field.

The interactions between each neighbouring pair of spins are usually characterised by a coupling constant J. Positive values of J incentivise the material to exhibit ferromagnetic behaviour, where neighbouring spins are more likely to be aligned than anti-aligned. On the other hand, negative J values do the opposite and cause neighbouring spins to typically become anti-aligned. The external magnetic field, which is usually represented by a variable h, indirectly influences the system's magnetisation (average spin direction). By varying h, researchers can study phenomena such as phase transitions in interacting materials, which are abrupt changes in the macroscopic behaviour of a physical system resulting from small changes in the microscopic parameters. The Hamiltonian of the system is given by

$$H = -\sum_{\langle ij \rangle} J s_i s_j - h \sum_i s_i, \tag{2.61}$$

where the sum $\langle ij \rangle$ denotes a sum over all nearest neighbours in the lattice. The probability distribution of the spin configurations is then determined by the usual Boltzmann distribution,

$$p_i = \frac{e^{-\frac{\varepsilon_i}{k_B T}}}{Z}, \tag{2.62}$$

where ε_i is the energy eigenvalue of the i^{th} configuration. At lower temperatures, the term ε_i/k_BT becomes more sensitive to energy, making the system tend to favour specific configurations with lower energies. This can lead to the emergence of ordered states and phase transitions. This phenomenon is known as spontaneous symmetry breaking; it occurs when a system's Hamiltonian possesses a symmetry that is not preserved in its ground state. This results in the formation of asymmetric but ordered structures that result from initially symmetric disordered ones. The likelihood of spontaneous symmetry breaking occurring in a system increases as temperature decreases because energy differences between configurations become more significant.

The relative simplicity and mathematical tractability of the Ising model make it a crucial tool in the study of interacting systems. Its study has provided insights into the nature of various complex systems, including their critical phenomena and phase transitions. It also serves as a helpful starting point for more elaborate and representative models that can incorporate additional physical effects.

2.3.2 Reproduced material

The aforementioned paper [42] presents a study focused on applying machine learning techniques to determine dynamical parameters in the context of the Ising model. By leveraging particular machine learning methods, the authors aim to enhance the understanding of using RBMs in complex systems by using the Ising model as a case example.

The paper begins by providing a brief overview of Restricted Boltzmann Machines. It then demonstrates a calculation of the gradient of the log-likelihood for an RBM with an energy distribution of the form mentioned in 2.2. The authors then introduce several monitoring factors for the training procedure, including using the log-likelihood, measuring the constancy of the partition function across training rounds, using a loss function, and a reconstruction error. Their models were trained on data from an available software **Magneto** [43].

In order to allow for an efficient learning process, the partition functions are approximated using a technique known as annealed importance sampling, and the Contrastive Divergence algorithm is used to approximate the differences in the model and data averages for use in equations (2.58), (2.59), and (2.60).

The RBM is trained on both the 1D and 2D Ising model without a magnetisation term h. A site spin of $s = +1$ corresponds to a node configuration of either h_i or v_j equal to 1, and a spin of $s = -1$ corresponds to a node configuration equal to 0. Of interest to this thesis, the learned target distribution is plotted, and the measured log-likelihood for the parameter values is plotted as a function of training epoch.

An original implementation of the above material was reproduced in Python, and is available at [44]. As in the reference paper, an Ising model with couplings $J = 1(= T)$ is simulated, and an RBM is trained on the configurations of this simulation. Firstly, a binary RBM with six hidden nodes was trained on a 1D Ising model lattice with six spins over four epochs with run parameters as documented in table 2.1. There are no phase transitions in the one dimensional Ising model, and so the reference paper did not consider varying the Ising coupling J or the temperature T at this point. The original intention of the authors was to demonstrate that the RBM does successfully learn the distribution presented given enough training, but also checks that the performance reaches a plateau as the data provided saturates the model. The distribution learned in the new python implementation

Epoch	Number of Data batches	Data Batch Size	Learning Rate
1	1000	200	0.01
2	1000	200	0.01
3	1000	200	0.001
4	1000	200	0.0001

TABLE 2.1: Parameters used for the training of the 6 unit 1D Ising model lattice as performed in [42].

for the above parameters is plotted in figure 2.1. Inspection of the figure shows that the later

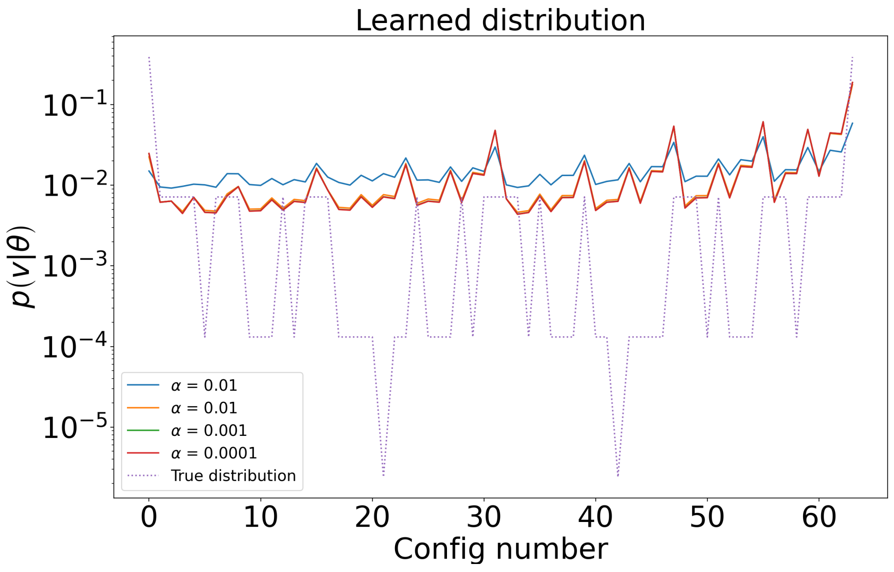


FIGURE 2.1: *Learned distribution for a six-spin Ising model at the ends of each stages of training for an RBM with six visible and six hidden nodes. Four phases of training were underwent sequentially; each phase consisted of 1000 applications of the contrastive divergence algorithm, with each step receiving a batch of data S of size $|S| = 200$. The learning rate in each phase were* $\alpha = 0.1$, 0.1, 0.001 *and* 0.001 *respectively. The final configurations learned at the end of each phase are plotted and labelled in the figure legend.*

stages of the training are, on average, closer to the target (true) distribution, plotted as a dotted line in the figure. The system was trained for $N = 6$ lattice sites, leading to $2^6 = 64$ possible configurations. The configurations were labelled from 0 to 63 via

$$\text{configuration number} = \sum_{i=0}^{N-1} 2^i j_i, \tag{2.63}$$

with $j_i = 0$ when lattice site i is spin down, and $j_i = 1$ when lattice site i is spin up. Such a choice of labelling explains why the true distribution in figure 2.1 is symmetric around lattice sites $31/32$ due to the symmetry of the Hamiltonian in flipping all of the spins.

Shown in figure 2.2 is the log likelihood of the parameters for when the setup described in table 2.1 is run. It can be seen that over the course of the training, the parameter likelihood

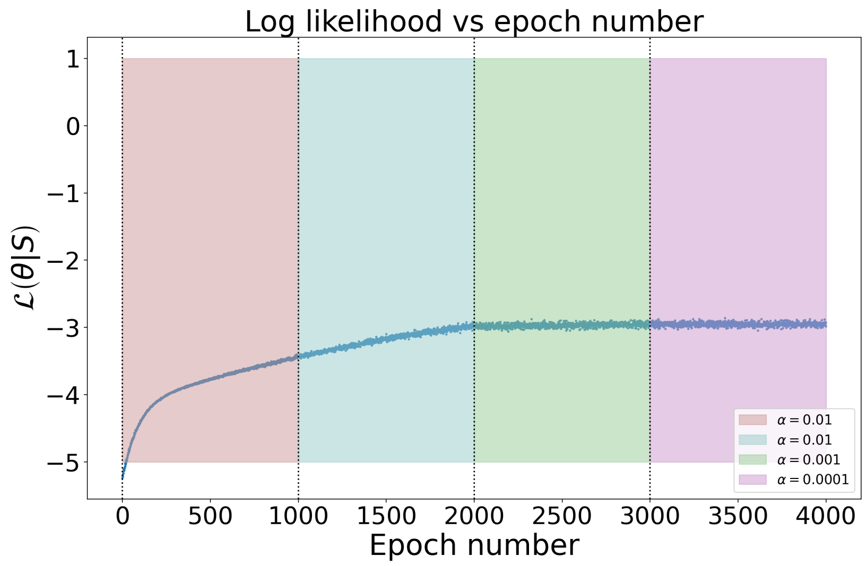


FIGURE 2.2: *Log likekihood of trained parameters θ given the batch S of training data at each epoch for the training procedure detailed in table 2.1. The learning rate in each phase is shaded according to the legend.*

increases and then begins to slow down in its rate of increase after each configuration has likely been seen multiple times. Once the learning rate α decreases from 0.01 to 0.001, the rate of increase of the log-likelihood drops further. The log-likelihoods obtained plateau at around -2.8, similar to those in the reference paper (where the log-likelihood plateaued at around -2.5 for this setup). This provides quantitative evidence of the RBM's learning performance. The increasing log-likelihood followed by a plateau indicates that the model has successfully learned key aspects of the underlying data, leading to a robust and stable learning outcome. The success of correctly training a Restricted Boltzmann Machine (RBM) to learn the Ising model distribution is a significant achievement that reinforces the foundations of this thesis. It demonstrates the capacity of a simple RBM to capture complex behavior in data.

An important note to make here is that despite the fact that the RBM has learned the distribution fairly well, it has had massive exposure to the target distribution. With only 64 available configurations, but $200 \times 1000 \times 4 = 800,000$ data points provided, each state is seen an average of $12,500$ times. In machine learning, overfitting occurs when a model learns training data *too well*. In the reference paper, this was done intentionally to demonstrate the plateau in the log likelihood, however, there are generally concerns with overfitting due to the fact that a model will learn accidental features of the data set provided, capturing noise and fluctuations that are specific to the training set, but that do not generalise to unseen data. Overfitting hinders a model's performance on unseen data as it 'memorizes' the training set rather than the underlying physical patterns. In the case demonstrated above, where there are a small number of configurations that have been learned by an RBM with a large number of training examples, it cannot yet be concluded that the RBM is able to generalise the data it has seen very well without first checking that is able to increase its log-likelihood significantly without getting exposure to multiple rounds of all possible configurations.

To check how the RBM behaves when not overfit, a second training round is demonstrated for a 1D Ising model lattice with 18 sites. The details of the training parameters written in table 2.2, and the RBM used has two hidden nodes. In this training example, there are

Epoch	Number of Data batches	Data Batch Size	Learning Rate
1	500	200	0.01
2	500	200	0.005
3	500	200	0.00025

TABLE 2.2: Parameters used for the training of the 18 unit 1D Ising model lattice.

$2^{18} \approx 216,000$ possible configurations of the Ising model, and 300000 of these are seen by the end of the training. This means that by the end of the training, each configuration has, on average, been observed 1.14 times. The results of this training example are shown in figure 2.3. As in the previous case, the model begins with a steep increase in the log-likelihood, fol-

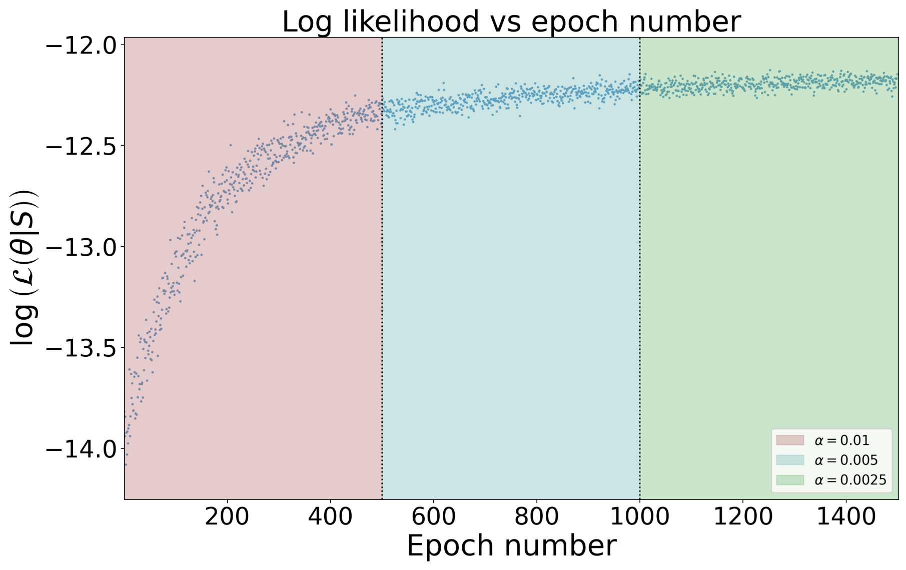


FIGURE 2.3: *Log likelihood of the trained parameters θ given the batch of data S at each epoch for the training run specified in table 2.2.*

lowed by a slowing down in the rate of increase in performance as further training batches are added. As shown in figure 2.4, which is the same data but focused in on the first 500 training batches (where the number of states seen is only approximately 38% the number of possible configurations), the initial learning rate is still steep despite the RBM not having been saturated with data. Over this section of the training the log likelihood steadily increases from

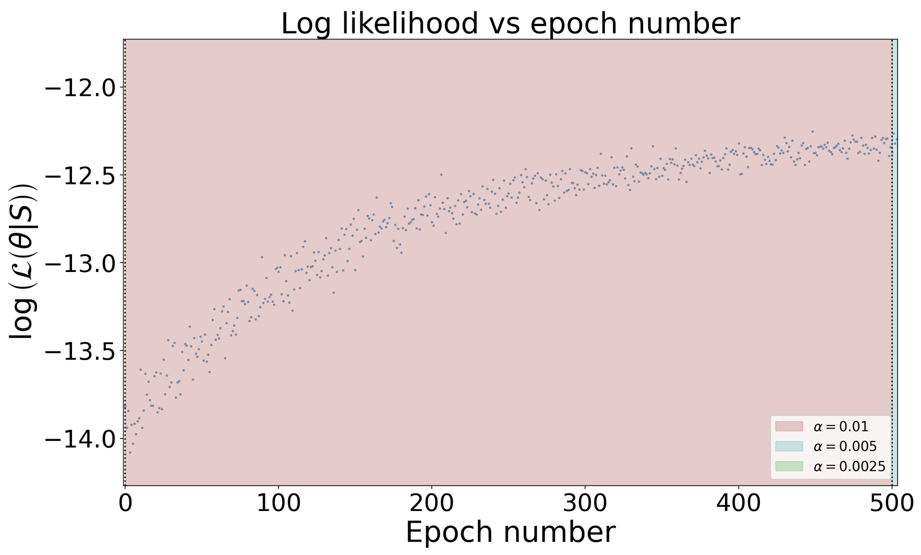


FIGURE 2.4: *Focus on the log likelihood for the first 500 steps in the training. By the end of this stage the RBM had seen* 10^5 *data points, which is approximately* 38% the number of system configurations.

approximately -14.0 to approximately -12.5. This means that the RBM is successfully generalising the information regarding configurations it has seen to configurations that it hasn't, as the log-likelihood is calculated over all data within each batch (and each new batch likely contains new configurations previously unseen). Finally, figure 2.5 demonstrates that much

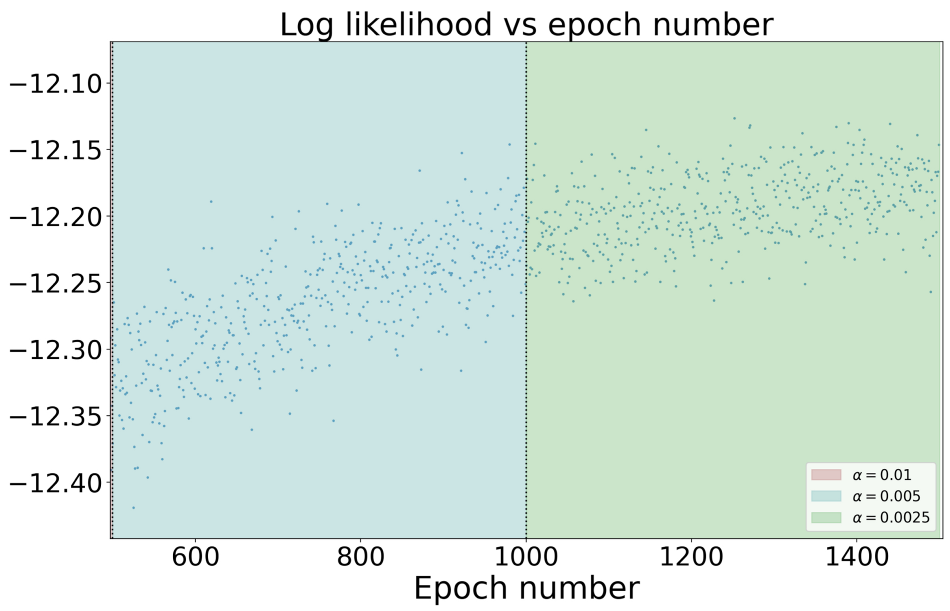


FIGURE 2.5: *Focus on the log likelihood for the first* 500^{th} *to* 1500^{th} *steps in the training. By the end of this stage the RBM had seen* 3×10^5 *data points, which is approximately* 114% the number of system configurations.

later, once the system had seen a number of configurations approaching and exceeding the number of possible system configurations that the log-likelihood does still increase, but more slowly than in the initial phases of training. This is as expected, as the parameters will still be converging towards their optimal state for the data being represented, however now it is important to recognise that in this regime, the model is beginning to simply memorise configurations themselves, rather than correctly generalising the data it has received to unseen configurations.

Overall, these results firmly establish the effectiveness of the chosen approach in learning the target data. It is worth noting that the incorporation of a bilinear coupling between the hidden and visible layers in the RBM architecture stands out as a pivotal design choice. Despite its apparent simplicity, this approach empowers the model to learn complex behaviors in the data. Consequently, RBMs with only a bilinear mixing term can be confidently employed in subsequent analyses, as highly accurate results are yielded by these models while offering computational advantages over more complex counterparts.

Chapter 3

Insights from Quantum Field Theory

“[Quantum mechanics] describes nature as absurd from the point of view of common sense. And yet it fully agrees with experiment. So I hope you can accept nature as She is - absurd.”

-Richard Feynman

This chapter bridges the realms of Restricted Boltzmann Machines and Quantum Field Theory (QFT), connecting principles and insights between the two domains. QFT, as a subject, provides a robust framework from which to understand the nature of the universe by combining special relativity and quantum mechanics. By exploring the connections between specific structures in QFT and RBMs, insightful perspectives on the behaviour of RBMs can be seen. In this chapter, section 3.1 explores the relationship between QFT and the Ising model, framing QFT as a natural generalisation of the Ising model. Some successes of QFT within Theoretical Physics as well as its applications outside, are mentioned, motivating to study RBMs with nodes that take on continuous spins. Section 3.2 demonstrates a simple calculation from QFT of the behaviour of integrating out a field to produce an effective action and highlights the analogues between this process and marginalising over the hidden layer in an RBM. Section 3.3 then uses the results of this discussion to argue for the study of specific types of RBMs and explores the implications of having Gaussian and higher hidden node energies. These choices have significant consequences for the RBMs' representation of target distributions, and the discussions provide valuable insight into the control that can be had over these choices. Finally, section 3.4 demonstrates the calculation of the KL divergence for an RBM under the considerations mentioned and sets out a natural question to be asked, given the insights from QFT.

3.1 Connections between lattice QFT and the Ising model

The Physics of the Ising model can be mapped onto a lattice QFT, allowing for a deeper understanding of the connections between the two frameworks. The lattice variables can be understood as a generalisation of the spin variables of the Ising model. In the Ising model, the system is composed of spins that can take binary values: ± 1, whereas in lattice QFT the lattice variables can take on a continuous label. To establish a connection between the two, consider a 1D lattice of points with both a spin variable s_n and a field variable ϕ_i defined at each lattice site n. The probability distribution for both is then given in terms of exponentials: for the spins, the probability of the i^{th} configuration is given in terms of the Boltzmann distribution

$$p_i = \frac{1}{Z} e^{-\frac{\varepsilon_i}{k_B T}}, \tag{3.1}$$

where ε_i is the energy of the i^{th} configuration, as determined by the Ising model. For the quantum field, the configuration amplitudes are given in terms of

$$\mathcal{A}_i \propto e^{iS_i/\hbar}, \tag{3.2}$$

where S_i is the action of the i^{th} field configuration. To map the lattice QFT to the Ising model, the question becomes how to choose the field Lagrangian of the lattice field theory to make the two probability distributions match in their form. By choosing the lattice field to be kinetic-free and by giving it a mass and quartic interaction term, a Z_2 spontaneous symmetry breaking will occur when the mass term is negative. The action takes the form

$$S_i = \sum_{\text{Sites } j} \frac{1}{2} m^2 \phi_j^2 + \frac{1}{4!} \lambda \phi_j^4. \tag{3.3}$$

When $m^2 < 0$ is chosen, the most likely classical field configurations for each field is

$$\phi_i = \pm \sqrt{\frac{-m^2}{\lambda}}, \tag{3.4}$$

with both being equally likely. By choosing the interaction strength as $\lambda = -m^2 > 0$, followed by taking the limit behaviour of $m^2 \to -\infty$, the model behaves like the zero temperature limit of the non-interacting Ising model, and the two most likely configurations become the only configurations with non-zero probability. Following this, the only configurations of the fields with non-zero probability are $\phi_i = \pm 1$. Such a choice of parameters has forced the fields to have the same space of possible configurations as the spin variables s_i, establishing a link between the two models. The lattice field theory defined above is equivalent to the Ising model with no coupling between neighbours and with no externally applied magnetic field. More careful considerations of the constructions above can cause the appearance of terms corresponding to these. For example, the discretised kinetic term $(\partial_i \phi)^2$ causes the appearance of a nearest-neighbour term via

$$(\partial_i \phi)^2 \to (\phi_{i+1} - \phi_i)^2 = \phi_{i+1}^2 + \phi_i^2 - 2\phi_{i+1}\phi_i, \tag{3.5}$$

where the squared terms are simply absorbed into the mass term.

By connecting the Ising model Hamiltonian and the lattice action, the lattice QFT framework provides a mathematical description of the Ising model in terms of field theory. This is useful, as it allows for the utilization of insights developed in the contexts of lattice QFT and the Ising model to be passed between each other to gain a deeper understanding of both. It also sets up a motivation to use RBMs that well represent the field configurations of QFT, as these can now be seen as a natural generalisation to the study of the Ising model as a toy model for complex interacting systems. This also means that all of the interesting and valuable phenomena exhibited by the Ising model will be contained within lattice QFT, and so the motivations for the study of the Ising model in the first place will hold from here too.

Having seen the above, generalising binary RBMs to RBMs with continuous labels is a natural step that the connections between QFT and the Ising model inspire. By expanding RBMs to handle continuous labels, a more comprehensive range of phenomena with more complex relationships in data can be explored.

Motivated by the success and applicability of RBMs in modelling binary data, the extension to continuous labels opens up further possibilities for modelling continuous variables, such as measurements or sensory inputs. The generalisation allows RBMs to capture the fine-grained nuances and variations present in real-world data. Furthermore, the ability to hold continuous labels will allow the RBMs to hold more realistic representations of the uncertainty and variability present in data from real-world scenarios, allowing the RBMs to represent physical features more faithfully. Given the wealth of effort that has been applied to understanding calculations in QFT, upgrading the RBM labels to continuous will allow for sophisticated techniques developed for QFT that could not be applied to the binary RBMs. This will be demonstrated further in 4.3. By embracing ideas from QFT, an advantage can be taken to aid in the understanding of RBMs with continuous labels, ultimately advancing the ease with which meaningful insights into the workings of RBMs can be made.

3.2 Effective action in QFT

The design choices for the RBMs with continuous labels will draw inspiration from the calculation of a two-field effective action. This calculation will guide the decision-making process and inform the specific parameters chosen for the RBMs considered.

Following the calculations of [45], consider the action for two massive scalar fields coupled by a dimensionless coupling constant λ in a zero-dimensional QFT

$$S\left[\phi,\, \chi\right] = \frac{m^2}{2}\phi^2 + \frac{M^2}{2}\chi^2 + \frac{\lambda}{4}\phi^2\chi^2. \tag{3.6}$$

These fields are defined to exist at a single point in space, and so the path integral over their possible configurations reduces to an integral over the range of the function values. By constraining to the case of real-valued fields (which will be the case for the values represented by RBMs), the partition will be given by

$$\mathcal{Z} = \int_{\mathbb{R}^2} \mathrm{d}\phi\, \mathrm{d}\chi\, e^{-\frac{S(\phi,\, \chi)}{\hbar}}. \tag{3.7}$$

Consider the attempt at calculating this where ϕ is first held constant, and the integral over χ is performed first. Such a process yields an *effective action* $\mathcal{W}\left(\phi\right)$ that can be found via the integral

$$e^{-\frac{\mathcal{W}(\phi)}{\hbar}} = \int_{\mathbb{R}} \mathrm{d}\chi\, e^{-\frac{S[\phi,\, \chi]}{\hbar}}. \tag{3.8}$$

Since $S\left[\phi,\, \chi\right]$ is at most quadratic in χ, this calculation is straightforward. The calculation yields

$$\int_{\mathbb{R}} \mathrm{d}\chi\, e^{-\frac{S[\phi,\, \chi]}{\hbar}} = e^{-\frac{m^2\phi^2}{2\hbar}} \sqrt{\frac{2\pi\hbar}{M^2 + \lambda\phi^2/2}}, \tag{3.9}$$

giving the effective action

$$\mathcal{W}\left(\phi\right) = \frac{1}{2}m^2\phi^2 + \frac{\hbar}{2}\ln\left[1 + \frac{\lambda}{2M^2}\phi^2\right] + \frac{\hbar}{2}\ln\left(\frac{M^2}{2\pi\hbar}\right). \tag{3.10}$$

Dropping the irrelevant constant and expanding the logarithm gives

$$\begin{aligned}\mathcal{W}\left(\phi\right) &= \left(\frac{m^2}{2} + \frac{\hbar\lambda}{4M^2}\right)\phi^2 - \frac{\hbar\lambda^2}{16M^4}\phi^4 + \frac{\hbar\lambda^3}{48M^6}\phi^6 + \dots \\ &= \frac{m_{\text{eff}^2}}{2}\phi^2 + \frac{\lambda_4}{4!}\phi^4 + \frac{\lambda_6}{6!}\phi^6 \dots .\end{aligned} \tag{3.11}$$

In expanding the logarithm, one should be aware that formally $\hbar$ has dimension Js, and hence the expansion is not in $\hbar$, but in terms of a dimensionless ratio. The effect of integrating out the quadratic field χ has the effect of changing how the action sees ϕ. Particularly, the mass term has been shifted by an amount $\frac{\hbar\lambda}{2M^2}$, and an infinite series of new coupling terms have appeared. In the context of QFT, both of these effects are inherently quantum; this can be seen by the factor of $\hbar$ appearing in each of the new contributions to the action. This said, both of these effects will be important in consideration of the type of interactions introduced in the RBM energy, as the integrals done will be similar; the integration of the auxiliary field χ is analogous to integrating over the hidden layer to find the marginalised visible layer probability distribution. The contributions of the hidden layer due to their integration on the "visible layer effective action" will then determine the dependency the corresponding RBM's model distribution will have on each visible node.

A natural generalisation of the action (3.6) that is invariant under parity of the fields would be one that contains all even monomials in ϕ and χ is [45],

$$S'\left(\phi,\, \chi\right) = \sum_{i,\, j} \frac{\lambda_{ij}}{(2i)!\,(2j)!}\phi^{2i}\chi^{2j}. \tag{3.12}$$

Integrating this action over χ would not generate new couplings in ϕ, but instead shift all $\lambda_{i,\,0}$ by some amount. This amount will be non-trivial as the integral over χ would now be very complicated. This demonstrates a lesson - having a coupling that is quadratic or higher in χ generates infinitely many couplings in powers of ϕ. Choosing such a coupling in an RBM would lead to a more difficult time understanding the behaviour of the training of the visible nodes in the RBM. This would also hold true in the training phase; tuning the parameters $\lambda_{0,\,j}$, the coefficients of the biases χ^{2j}, would have non-trivial consequences on the models' dependence on each ϕ^{2i}. This would hold true even when the coefficients $\lambda_{i,\,0}$ are left unchanged. This is not automatically bad for model performance; it does, however, make isolating the behaviour of the RBM at a particular power more difficult, and so is an undesirable behaviour given the context of the research.

Consider instead the hypothetical action that has only a bilinear coupling between ϕ and χ,

$$S''[\phi,\,\chi] = \frac{m^2}{2}\phi^2 + \frac{M^2}{2}\chi^2 + \omega\phi\chi. \tag{3.13}$$

This will act differently to S when the integral over χ is taken as there can be a completion of the square in χ. Given that the contribution to the coupling from ϕ is linear, integration over χ only causes an additional quadratic term in ϕ to appear. In particular, the effective potential here becomes

$$\begin{aligned}\mathcal{W}''(\phi) &= \frac{1}{2}\left(m^2 - \frac{\omega^2}{M^2}\right)\phi^2 - \hbar \ln\left(\sqrt{\frac{2\hbar\pi}{M^2}}\right) \\ &= \frac{1}{2}m_{\text{eff}}^2\phi^2 + \text{const} \\ &\propto S''[\phi,\,\chi=0] + \text{const}.\end{aligned} \tag{3.14}$$

This is an insightful result: an action with a coupling that is at most bi-linear in the fields, and where the auxiliary field has no self-interaction term (it is at most quadratic), has a corresponding effective action that is only perturbed at quadratic order in the remaining field. In the context of RBMs, this means that if the energy of the hidden nodes is chosen to be at most Gaussian, and the coupling term is at most bilinear in the fields, then there will be good interactivity picked up by the mixing term (as demonstrated in the example of the Ising model 2.3.2), but there will only be a shift in terms quadratic in the visible layer; nothing more complicated. This suggests that a good balance can be struck between the complexity the RBMs can represent and the mathematical simplicity of the RBMs by choosing the interaction term to be bilinear and by choosing the hidden layer to be Gaussian. Under such a choice, if higher order dependencies on the visible layer (ϕ here) are set up, their tuning will be fully dependent on their coefficients as they appear when first written down and not in some convoluted way that depends on the coefficients of the hidden node contributions. Additionally, as discussed in 1.6.3, having a Gaussian contribution in the visible layer will also be useful as tricks from perturbation theory to calculate the higher order partition functions in terms of a quadratic one can then be employed.

3.3 RBM with a Gaussian hidden layer and bilinear mixing

Now that the case for a Gaussian hidden layer with a bilinear coupling has been established as an interesting case for study in RBMs with real-valued nodes, consider an RBM characterised by the energy

$$E(\mathbf{v},\,\mathbf{h};\theta) = -\left(\sum_{j=1}^{m} U_j(v_j) + \sum_{i=1}^{n}\tilde{U}_i(h_i) + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i\omega_{ij}v_j\right). \tag{3.15}$$

Now, the hidden layer energies are Gaussian, such that hidden node potential parameters $f_{i_\alpha}^{(h)}$ give that

$$\tilde{U}_i\left(h_i,\,f_{i_\alpha}^{(h)}\right) = -\frac{1}{2}\tilde{m}_i^2 h_i^2 + c_i h_i, \quad \tilde{m}_i^2 > 0, \tag{3.16}$$

such that

$$q\left(\mathbf{v},\, \mathbf{h};\, \theta\right) = \frac{1}{Z\left(\theta\right)} \exp\left(\sum_{j=1}^{m} U_j\left(v_j\right) + \sum_{i=1}^{n}\left(-\frac{1}{2}\tilde{m}_i^2 h_i^2 + c_i h_i\right) + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i \omega_{ij} v_j\right), \tag{3.17}$$

with $\tilde{m_i}^2 > 0$. Here, it is worth emphasising that the hidden layer mass term depends on the location/node as labelled by i which is phenomenologically different to QFT. Substitution of $\tilde{U}_i\left(h_i\right)$ into (2.28) gives

$$U_h\left(\mathbf{v},\, \mathbf{h}\right) = -\sum_{i=1}^{n}\left[-\frac{1}{2}\tilde{m}_i^2 h_i^2 + h_i\left(c_i + \sum_{j=1}^{m}\omega_{ij} v_j\right)\right], \tag{3.18}$$

which can then be inserted into (2.30) to find the marginalised probability distribution for the visible nodes. This results in

$$\begin{aligned}\int \mathcal{D}\mathbf{h}\, e^{-U_h(\mathbf{v},\,\mathbf{h})} = &\sqrt{\frac{\left(2\pi\right)^n}{\prod_{\beta=1}^{n}\tilde{m}_\beta^2}} \exp\left(\frac{1}{2}\sum_{\alpha=1}^{n}\frac{c_\alpha^2}{\tilde{m}_\alpha^2}\right) \\ &\times \exp\left(\sum_{j=1}^{m} v_j \sum_{\alpha=1}^{n}\left[\left(\frac{c_\alpha \omega_{\alpha j}}{\tilde{m}_\alpha^2}\right) + \frac{1}{2}\sum_{k=1}^{m}\left(\frac{\omega_{\alpha j}\omega_{\alpha k}}{\tilde{m}_\alpha^2} v_k\right)\right]\right).\end{aligned} \tag{3.19}$$

This is the integral for the RBM that is equivalent to the integral that produced the effective action (3.14), but in one dimension. The visible node marginalised probability distribution picks up a linear and quadratic term. It also picks up a bilinear term between visible nodes as the integrals are no longer over the field values of a single lattice point. Following this, the visible node distribution then becomes

$$q\left(\mathbf{v}|\, \theta\right) = \frac{1}{Z'\left(\theta\right)} \exp\left(-E'\left(\mathbf{v};\, \theta\right)\right), \tag{3.20}$$

where

$$Z'\left(\theta\right) = Z\left(\theta\right)\sqrt{\frac{\prod_{\beta=1}^{n}\tilde{m}_\beta^2}{\left(2\pi\right)^n}} \exp\left(-\frac{1}{2}\sum_{\alpha=1}^{m}\frac{c_\alpha^2}{\tilde{m}_\alpha^2}\right), \tag{3.21}$$

and

$$E'\left(\mathbf{v};\, \theta\right) = -\sum_{j=1}^{m}\left[U_j\left(v_j\right) + v_j\sum_{\alpha=1}^{n}\frac{\omega_{\alpha j}}{\tilde{m}_\alpha^2}\left(c_\alpha + \frac{1}{2}\sum_{k=1}^{m}\omega_{\alpha k} v_k\right)\right], \tag{3.22}$$

which is the equivalent calculation as for the effective action in (3.14), but with additional cross terms due to the multiple nodes/lattice points considered.

3.3.1 Gaussian visible nodes

Following the result (3.22), if the hidden layer is chosen to be Gaussian and the mixing term is bilinear, then the visible layer picks up Gaussian and bias terms only. If then U_j is chosen to be only quadratic in each v_j, that is

$$U_j\left(v_j\right) = -\frac{1}{2}m_j^2 v_j^2 + b_j v_j, \tag{3.23}$$

then the marginalised probability distribution can be written in terms of the effective energy

$$E'(\mathbf{v};\,\theta) = -\left[-\frac{1}{2}\sum_{j=1}^{m}\sum_{k=1}^{m} M_{jk}^2 v_j v_k + \sum_{j=1}^{m} v_j B_j\right], \tag{3.24}$$

where

$$M_{jk}^2 = m_j^2\delta_{jk} - \sum_{\alpha=1}^{n}\frac{\omega_{\alpha j}\omega_{\alpha k}}{\tilde{m}_\alpha^2}, \tag{3.25}$$

and

$$B_j = b_j + \sum_{\alpha=1}^{n}\frac{\omega_{\alpha j}c_\alpha}{\tilde{m}_\alpha^2}. \tag{3.26}$$

This is an interesting point to pause, as the term $M_{jk}^2 v_j v_k$ forms a symmetric bi-linear form. Because M_{jk}^2 is symmetric, there exists an orthogonal matrix $\mathbf{O}$ that diagonalises M_{jk}^2. By writing

$$\mathbf{M}^2 = \mathbf{O}^T\cdot\mathbf{D}\cdot\mathbf{O}, \tag{3.27}$$

where D is the diagonal matrix of eigenvalues of $\mathbf{M}^2$, the effective energy for the visible layer can then be written

$$E(\mathbf{v};\,\theta) = -\sum_{j=1}^{m}\left(-\frac{1}{2}D_{jj}v'^2_j + B'_j v'_j\right), \tag{3.28}$$

where

$$\mathbf{B}' = \mathbf{B}\cdot\mathbf{O}^T, \tag{3.29}$$

and

$$\mathbf{v}' = \mathbf{O}\cdot\mathbf{v}. \tag{3.30}$$

This is a useful insight, as it means that regardless of whether the data represented by the visible nodes are interacting, the RBM treats the visible nodes as if linear combinations forming each $\mathbf{v}'$ are actually independent. Such an RBM is the generalisation of the model probability distribution $Q_2(x)$ discussed in 1.5 to multiple degrees of freedom. Such a generalisation would be useful in contexts where there actually is unclear independence between data hidden behind some linear transformation of variables; however, this means that choosing $U_j(v_j)$ as quadratic constrains the space of interactions representable to become superficial. As such, higher order choices of $U_j(v_j)$ need study.

3.3.2 Higher order visible nodes

Based on the discussions above, it is evident that RBMs with bilinear mixing and a Gaussian hidden layer cannot fully capture systems with non-trivial interactions if $U_j(v_j)$ is chosen to be purely Gaussian in each node. This is not a huge issue, however, as any polynomial of cubic order or higher in each v_j cannot have the diagonalisation procedure mentioned in 3.3.1 applied without the mixing of nodes at higher order, meaning the decoupling of the nodes cannot occur.

Upon choosing some non-trivial $U_j(v_j)$, the next issue with regards to interpreting RBM behaviour comes from the calculation of the partition function. As demonstrated in 1.5, calculations with only a single quartic term become heavy quickly. Aware of the significant complexity that could arise upon selecting the form of U_j, further inspiration is taken from QFT. A few rules must be kept in mind when constructing the Lagrangian of a quantum field theory. A selection of these rules are

1. The Lagrangian must be real-valued.

2. The field must be local in value and its derivatives.

The first two map over well to the mathematics describing RBMs considered here. The energies of all RBM configurations are real, and the energy (3.22) has at most bilinear interactions between visible nodes, which can be thought of as a set of nearest neighbour interactions for field values on a lattice. An additional rule that will be imposed for the RBM is that the energy function must be bounded from below in order to ensure that the probability distribution it describes is normalizable. Furthermore, in calculating the difference in KL divergence between two models, the partition function must be calculated for both models, and so the probability distributions being considered must be normalizable for progress to be made.

In the context of QFT, such a choice corresponds to the stability of the vacuum; a feature of very many QFTs that have been studied. One way of achieving this is to study theories with a Lagrangian that is strictly even in the highest power of the field ϕ, as for such a choice, contributions from e^{-S} are always finite for any field configuration. Note however that an even highest field power is not a requirement of QFT in general. There have been many studies of fields with an odd highest field power, such as in [46] and [47] where field theories where each field has a cubic highest power are explored.

A convenient way to hold the analogy between the RBMs studied and QFTs with an energy that is bounded below would be to consider only RBMs with even powers in the field (rather than constraining only the highest power to be even). This would introduce an additional Z_2 symmetry to the RBMs, which could spontaneously be broken, and focus the scope of work within QFT that could have relevant applications to this work to only QFTs that have a Z_2 symmetry. However, since the Gaussian hidden layer (with $c_i \neq 0$ in the RBMs of interest) necessarily introduces a linear dependence of the visible node effective energy on each v_j, this constraint cannot fully be applied. Therefore, recognising this, as well as the work in QFT as something that could potentially be leaned on later on down the line when making calculations, U_j is chosen to take the form

$$U_j\left(v_j,\, f_{j\alpha}^{(v)}\right) = -\sum_{\alpha=1,\,2,\,4,\,6,\,8,\,\dots}^{2N} \mu_j^{(\alpha)} v_j^{\alpha}, \tag{3.31}$$

where $\mu_j^{(\alpha)}$ denotes the parameter characterising the behaviour of the the α^{th} power of the j^{th} visible node. This fully defines the set of visible node parameters $f_{j\alpha}^{(v)}$ used. The single odd power v_j^1 is kept, as the if the hidden layer has non-zero c_i then linear dependence on each v_j will necessarily be introduced. Under this choice, such an RBM has an energy

$$E\left(\mathbf{v},\, \mathbf{h};\, \theta\right) = -\left(-\sum_{\alpha=1,\,2,\,4,\,6,\,8,\,\dots}^{2N}\sum_{j=1}^{m}\mu_j^{(\alpha)}v_j^{\alpha} + \sum_{i=1}^{n}\left(-\frac{1}{2}\tilde{m}_i^2 h_i^2 + c_i h_i\right) + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i \omega_{ij} v_j\right). \tag{3.32}$$

Given the choice of visible node energies (3.31), a substitution into (3.22) can be made to determine the effective energy for the visible nodes. By first introducing the new variables $\lambda_j^{(\alpha)}$ such that

$$\lambda_j^{(1)} = \mu_j^{(1)} - \sum_{i=1}^{n} \frac{\omega_{ij} c_i}{\tilde{m}_i^2} \tag{3.33}$$

and

$$\lambda_j^{(\alpha)} = \mu_j^{(\alpha)}, \text{ for } \alpha \geq 4, \tag{3.34}$$

as well as the new mass variables $\lambda_{jj'}^{(2)}$ such that

$$\lambda_{jj'}^{(2)} = 2\left(\mu_j^{(2)} \delta_{jj'} - \frac{1}{2}\sum_{i=1}^{n} \frac{\omega_{ij}\omega_{ij'}}{\tilde{m}_i^2}\right), \tag{3.35}$$

the marginalised probability distribution for the visible nodes can be written

$$Q_{2N}(\mathbf{v}) := \frac{\exp\left(-E'_{(2N)}(\mathbf{v})\right)}{Z_{2N}}, \tag{3.36}$$

with

$$E'_{2N}(\mathbf{v}) = \sum_{j=1}^{m} \sum_{\alpha\in\mathcal{P}_N} \lambda_j^{(\alpha)} v_j^{\alpha} + \frac{1}{2}\sum_{j=1}^{m}\sum_{j'=1}^{m} \lambda_{jj'}^{(2)} v_j v_{j'}. \tag{3.37}$$

Here, the label $2N$ denotes that the highest power of the model is $v_j^{(2N)}$ in each node. The set $\mathcal{P}_{2N}$ comprises all even numbers from 4 to $2N$, inclusive, along with the number 1:

$$\mathcal{P}_N = \{1, 4, 6, 8, \ldots, 2N\}, \tag{3.38}$$

The parameters $\lambda_j^{(\alpha)}$ will capture the behaviour of each node in each of these powers. The terms $\lambda_{jj'}^{(2)} v_j v_{j'}$ are written separately, as they contain cross-terms in pairs of different visible nodes due to the mixing introduced by the Gaussian hidden layer. With this set, the KL divergence of such a model with respect to a target P can be computed, as well as the gradient of the RBM log-likelihood.

3.4 KL divergence of a QFT inspired RBM

Recall the definition of the Kullback-divergence of a model P with respect to some target Q_{2N} as

$$D_{\mathrm{KL}}(P||\, Q_{2N}) = \mathbb{E}_P\left[\log\left(\frac{P}{Q_{2N}}\right)\right]. \tag{3.39}$$

Substitution of the marginalised visible node distribution (3.36) into this yields

$$D_{\mathrm{KL}}(P||\, Q_{2N}) = \mathbb{E}_P[\log(P)] + \log(Z_{2N}) + \mathbb{E}_P\left[E_{(2N)}(\mathbf{v})\right]. \tag{3.40}$$

Since P is a probability distribution in $\mathbf{v}$, it can always be written in the form of some Boltzmann distribution such that

$$P\left(\mathbf{v}\right)=\frac{\exp\left(-E_{P}\left(\mathbf{v}\right)\right)}{Z_{P}}, \tag{3.41}$$

because of the positivity of the exponential. Inserting this into the expression for the KL divergence returns

$$D_{\mathrm{KL}}\left(P||\,Q_{2N}\right)=\log\left(\frac{Z_{2N}}{Z_{P}}\right)+\mathbb{E}_{P}\left[E_{2N}'\left(\mathbf{v}\right)-E_{P}\left(\mathbf{v}\right)\right]. \tag{3.42}$$

Since a lower KL divergence represents a model Q_{2N} that better represents the target P, and since the KL divergence is non-negative, a "perfect" model can be interpreted as being one that has matched the free energy and internal energy of the target distribution, with worse models having a mismatch. The excess KL divergence is then given by

$$\Delta D_{\mathrm{KL}}\left(P||\,Q_{2N},\,P\right)=\log\left(Z_{2N}\right)+\mathbb{E}_{P}\left[E_{2N}\left(\mathbf{v}\right)\right]. \tag{3.43}$$

Expanding out the model energy and by the linearity of the integral, for the RBMs under consideration, this is then

$$\Delta D_{\mathrm{KL}}\left(P||\,Q_{2N},\,P\right)=\log\left(Z_{2N}\right)+\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N}\lambda_j^{(\alpha)}\,\mathbb{E}_{P}\left[v_j^{\alpha}\right]+\frac{1}{2}\sum_{j=1}^{m}\sum_{j'=1}^{m}\lambda_{jj'}^{(2)}\,\mathbb{E}_{P}\left[v_jv_{j'}\right]. \tag{3.44}$$

Since the form of P is generally unknown, each $\mathbb{E}_{P}\left[v_j^{\alpha}\right]$ and $\mathbb{E}_{P}\left[v_jv_{j'}\right]$ needs to be determined experimentally via the sampling of data. On the other hand, $\log\left(Z_{2N}\right)$ is not an explicit function of the target distribution as it is written in terms of only the parameters of the model. Given the choice of the RBM energies, techniques from QFT can now be employed to probe the form of $\log\left(Z_{2N}\right)$ in order to compare models. This is demonstrated in 4.3, where the performance of two models are contrasted. However, before proceeding with any calculations, it is worth taking relevant derivatives of the energy to understand how an implementation of the learning algorithm for such an RBM would be made.

As discussed in (2.21), there is only one interaction term between the layers, and so the only relevant derivative of weights is

$$\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial\omega_{ij}}=-h_iv_j. \tag{3.45}$$

The visible node biases $f_{j\alpha}^{(v)}$ are the parameters $\mu_j^{(\alpha)}$. The derivatives with respect to each of these are given by

$$\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial\mu_j^{(\alpha)}}=v_j^{\alpha}. \tag{3.46}$$

Correspondingly, the relevant hidden node variables are the mass parameters $\tilde{m}_i^2$ and the biases c_i. The relevant derivatives for these are

$$\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial\tilde{m}_i^2}=\frac{1}{2}h_i^2, \tag{3.47}$$

and

$$\frac{\partial E\left(\mathbf{v},\, \mathbf{h};\, \theta\right)}{\partial c_i} = h_i. \tag{3.48}$$

These can then be substituted into 2.19 to give the gradient of the log-likelihood. Note that the sum over the configurations of $\mathbf{h}$ is upgraded to an integral as each h_i is now a real variable. Also, each $q\left(\mathbf{h}|\mathbf{v};\, \theta\right)$ will require the calculation of the non-trivial integral $\int \mathcal{D}\mathbf{v}\, e^{-U_v(\mathbf{v},\,\mathbf{h})}$. Finally, it is interesting to note here that only the expectation value of local powers of h_i and v_j are needed, as well as the two-point function.

In order to set up a CD-like algorithm for RBMs of this sort, activation functions are required. These can be found by substitution of the energy into (2.41) and (2.43). Note now that the sum over allowed v_a and h_a will become an integral due to the continuous nature of those labels; as well as this, the $q\left(v_\ell|\, \mathbf{h}\right)$ and $q\left(h_\ell|\, \mathbf{v}\right)$ represent probability *densities* rather than full probabilities. The calculation for the hidden layer activation function is simple (shown in A.6), the probability density that the ℓ^{th} hidden node takes the value h_ℓ is given by

$$q\left(h_\ell|\, \mathbf{v}\right) = \sqrt{\frac{\tilde{m}_\ell^2}{2\pi}} \exp\left(-\frac{1}{2}\tilde{m}_\ell^2 \left(h_\ell - \frac{c_\ell + \sum_{j=1}^m \omega_{\ell j} v_j}{2\tilde{m}_\ell^2}\right)^2\right) \tag{3.49}$$

which is a Gaussian distribution centred at

$$h_\ell = \frac{c_\ell + \sum_{j=1}^m \omega_{\ell j} v_j}{2\tilde{m}_\ell^2}, \tag{3.50}$$

and with a variance $\tilde{m}_\ell^2$. The equivalent calculation for the visible node activation function yields

$$q\left(v_\ell|\, \mathbf{h}\right) = \frac{\exp\left(\sum_{i=1}^n h_i \omega_{i\ell} v_\ell - \sum_{\alpha=1,\,2,\,4,\,6,\,8,\,\dots}^{2N} \mu_\ell^{(\alpha)} v_\ell^\alpha\right)}{\int_{\mathbb{R}} \mathrm{d}v_a \, \exp\left(\sum_{i=1}^n h_i \omega_{i\ell} v_a - \sum_{\alpha=1,\,2,\,4,\,6,\,8,\,\dots}^{2N} \mu_\ell^{(\alpha)} v_a^\alpha\right)}. \tag{3.51}$$

This is a more involved expression as $2N$ is left arbitrary. The denominator in the expression is simply a normalising factor and has no dependence on v_ℓ, meaning that the probability density is proportional to the numerator of the expression for a given $\mathbf{h}$. Since the content of the exponential is still a polynomial of order $2N$ in v_ℓ, the general form of $q\left(v_\ell|\mathbf{h}\right)$ will be complex relative to the distributions previously discussed, with up to N local peaks in probability density.

Now that all housekeeping with regards to the RBMs under consideration has been set up, various models of these kinds can be contrasted.

Chapter 4

Calculation of sufficient conditions for RBM improvement

"Modern Physics has taught us that the nature of any system cannot be discovered by dividing it into its component parts and studying each part by itself... We must keep our attention fixed on the whole and on the interconnection between the parts."

-Max Planck

In this chapter, the discussions from previous chapters are utilized to establish a specific research question concerning the potential improvement of RBMs. Section 4.1 defines a research question given the insights from previous chapters. In Section 4.2, the methodology detailing how the question shall be answered is briefly discussed. Subsequently, the primary calculations are carried out in 4.3, and the obtained results are then discussed in 4.4.

4.1 The research question

Section 1.5 discussed the effect of introducing a quartic parameter to a single variable Gaussian model that was expressing some target P. The conclusion of this demonstration was that, in that case, it was generally advantageous to introduce the new parameter, as it was always possible to decrease the KL divergence of the model with respect to the target by doing so. Following the motivations of chapters 2 and 3 regarding the choice to study RBMs with a bilinear interaction between the hidden and visible layer, with a Gaussian hidden layer, and with a visible layer with only even contributions to the energy above quadratic, the questions from 1.5.4 come into play leaving the following question:

> Given such an RBM, when is it possible to improve on a trained model $Q_{2(N-1)}$ via the introduction of a new model Q_{2N} with an additional parameter *without* changing any of the already-trained parameters?

The answer to such a question will bring clarity as to when the inclusion of an additional model parameter can be worth the additional computational complexity of the algorithms discussed in 2.1.

4.2 Methodology

To approach the research question, define the model $Q_{2(N-1)}(\mathbf{v})$ as follows:

$$Q_{2(N-1)}(\mathbf{v}) = \frac{\exp(-E_{(2(N-1))}(\mathbf{v}))}{Z_{2(N-1)}}, \tag{4.1}$$

where the energy function $E_{(2(N-1))}(\mathbf{v})$ is given by:

$$E_{(2(N-1))}(\mathbf{v}) = \sum_{j=1}^{m} \sum_{\alpha \in \mathcal{P}_{N-1}} \tilde{\lambda}_j^{(\alpha)} v_j^\alpha + \frac{1}{2} \sum_{j=1}^{m} \sum_{j'=1}^{m} \tilde{\lambda}_{jj'}^{(2)} v_j v_{j'}. \tag{4.2}$$

Here, $\tilde{\lambda}_j^{(\alpha)}$ represents the parameters for the coefficients of α^{th} power of v_j, and $\tilde{\lambda}_{jj'}^{(2)}$ is the "mass term" parameterizing all mixing between the visible nodes and the quadratic contribution to the visible node configuration energy. The partition function $Z_{2(N-1)}$ is computed by integrating over all possible configurations of the model. Additionally, $\mathcal{P}_N$ denotes the set of all even numbers up to $2N$, excluding 2, and with the inclusion of 1:

$$\mathcal{P}_N = \{1,\, 4,\, 6,\, 8,\, \ldots,\, 2N\}. \tag{4.3}$$

Notice the change in sign convention to avoid confusion; each $\lambda_j^{(2(N-1))}$ must be positive for $Z_{2(N-1)}$ to be finite. Given a target distribution P, the model is trained and the parameters $\tilde{\lambda}$ adjusted such to minimise the KL divergence of $Q_{2(N-1)}$ with respect to P. A new model Q_{2N} is then introduced,

$$Q_{2N}(\mathbf{v}) := \frac{\exp\left(-E_{(2N)}(\mathbf{v})\right)}{Z_{2N}}, \tag{4.4}$$

with

$$E_{2N}(\mathbf{v}) = \sum_{j=1}^{m} \sum_{\alpha \in \mathcal{P}_N} \lambda_j^{(\alpha)} v_j^\alpha + \frac{1}{2} \sum_{j=1}^{m} \sum_{j'=1}^{m} \lambda_{jj'}^{(2)} v_j v_{j'}, \tag{4.5}$$

with its partition function calculated as usual. The parameter choices for the new model $\lambda_{jj'}^{(2)} = \tilde{\lambda}_{jj'}^{(2)}$ and $\lambda_j^{(\alpha)} = \tilde{\lambda}_j^{(\alpha)}$ for $\alpha < N$ are made, and each $\lambda_j^{(2N)}$ is restricted to be positive. Upon these choices, the difference in the KL divergence of the two models relative to P is

$$\Delta D_{\text{KL}}\left(P||\, Q_{2N},\, Q_{2(N-1)}\right) = D_{\text{KL}}\left(P||\, Q_{2N}\right) - D_{\text{KL}}\left(P||\, Q_{2(N-1)}\right), \tag{4.6}$$

which, due to the choice for each $\lambda_j^{(\alpha)}$, will simplify to become

$$\Delta D_{\text{KL}}\left(P||\, Q_{2N},\, Q_{2(N-1)}\right) = \log\left(\frac{Z_{2N}}{Z_{2(N-1)}}\right) + \sum_{j=1}^{m} \lambda_j^{(2N)} \mathbb{E}_P\left[v_j^{(2N)}\right]. \tag{4.7}$$

The model $Q_{(2N)}$ can be an improvement on $Q_{2(N-1)}$ if there exists a choice for the set of $\lambda_j^{(2N)}$ for which $\Delta D_{\text{KL}}\left(P||\, Q_{2N},\, Q_{2(N-1)}\right) < 0$. The method to determine whether this can be made negative will be via a linearisation of the ratio of the partition in each $\lambda_j^{(2N)}$ to get an approximate expression

$$\log\left(\frac{Z_{(2N)}}{Z_{(2(N-1))}}\right) \approx \sum_{j=1}^{m} \lambda_j^{(2N)} \mathcal{W}_j, \tag{4.8}$$

with each $\mathcal{W}_j$ a function of only the λ_j^{α} for which $\alpha < 2N$. The method to determine $\mathcal{W}_j$ will be presented in 4.3. This approximation results in the conclusion that the introduction of a new parameter $\lambda_j^{(2N)}$ will always be beneficial when

$$\mathbb{E}_P\left[v_j^{2N}\right] < -\mathcal{W}_j. \tag{4.9}$$

This is a useful expression, as the right-hand side of the equation is a function of only the trained values of the old model $Q_{2(N-1)}$, whereas the left-hand side of the equation is a function of only the target distribution itself, and not on either of the models. The right-hand side can be approximated by averaging over all data from P that has been sampled (provided that each $\mathbf{v}$ has been sampled in proportion to $P(\mathbf{v})$ via

$$\mathbb{E}_P\left[v_j^{2N}\right] = \frac{1}{|S|}\sum_{\mathbf{v}\in S} v_j^{2N}, \tag{4.10}$$

where S is the batch of all data drawn from P. In what follows, a method for calculating $\mathcal{W}_j$ is demonstrated.

4.3 Performing the calculation

Given the two RBMs Q_{2N} and $Q_{2(N-1)}$ discussed, the difference between the Kullback-Leibler divergences of the two models, as mentioned, is

$$\Delta D_{\mathrm{KL}}\left(P||\, Q_{2N},\, Q_{2(N-1)}\right) = \log\left(\frac{Z_{2N}}{Z_{2(N-1)}}\right) + \sum_{j=1}^{m} \lambda_j^{(2N)} \mathbb{E}_P\left[v_j^{(2N)}\right]. \tag{4.11}$$

To make progress in understanding the calculation of the right hand side of this equation, consider how the partition functions are calculated.

Usually, the partition function of a distribution is thought of in terms of an integral over all possible configurations of the probability distribution they are normalising. However, taking inspiration from Quantum field theory 1.6, the partition function representing a model of a given order can be thought of as a perturbative expansion in terms of models at a lower (particularly quadratic) order.

4.3.1 Expectation values of functions

To understand the perturbative expansion allowing for the approximate calculation of the partition function, first consider a probability density $\mu(\mathbf{x})$ which takes as input a vector $\mathbf{x} = (x_1,\, x_2,\, \ldots,\, x_n)$. Define the function

$$F(\mathbf{y}) = \int \mu(\mathbf{x})\, f(\mathbf{x}+\mathbf{y})\, d^n\mathbf{x} = \langle f(\mathbf{x}+\mathbf{y})\rangle_{\mu}, \tag{4.12}$$

for some function $f(\mathbf{x})$ on the probability space. The Taylor expansion of such a function $f(\mathbf{x}+\mathbf{y})$ can be written in terms of the exponential

$$f(\mathbf{x}+\mathbf{y}) = \exp(\mathbf{x}\cdot\nabla_{\mathbf{y}}) f(\mathbf{y}) \,. \tag{4.13}$$

Now, a short calculation shows that $f(\mathbf{y})$ can be factorised out of the integral:

$$\begin{aligned} F(\mathbf{y}) &= \int \mu(\mathbf{x}) f(\mathbf{x}+\mathbf{y})\, d^n\mathbf{x} \\ &= \int \mu(\mathbf{x}) \exp(\mathbf{x}\cdot\nabla_{\mathbf{y}}) f(\mathbf{y})\, d^n\mathbf{x} \\ &= \langle \exp(\mathbf{x}\cdot\nabla_{\mathbf{y}})\rangle_\mu f(\mathbf{y}) \\ &= M_\mu(\nabla_{\mathbf{y}}) f(\mathbf{y}) \,, \end{aligned} \tag{4.14}$$

where M_μ is the moment generating function for $\mathbf{m}$. By setting $\mathbf{y}=\mathbf{0}$, the result

$$F(\mathbf{0}) = \int \mu(\mathbf{x}) f(\mathbf{x})\; d^n\mathbf{x} = \langle f(\mathbf{x})\rangle_\mu \tag{4.15}$$

is obtained. This means that for a probability distribution $\mu(\mathbf{v})$, the expectation value of a function $f(\mathbf{x})$ can be written as

$$\langle f(\mathbf{x})\rangle_\mu = M_\mu(\nabla_{\mathbf{y}}) f(\mathbf{y}) \bigg|_{\mathbf{y}=\mathbf{0}} \,, \tag{4.16}$$

which is a perturbative expansion of $f(\mathbf{y})$. This is a useful result in the calculation of the partition function for an RBM with a Gaussian hidden layer, because as demonstrated, *all* RBMs with a Gaussian hidden layer have a term which is bilinear in each pair of visible nodes $(v_j, v_{j'})$, and quadratic in each individual node. As such, by choosing $\mu(\mathbf{x})$ in the result (4.16) to be a multivariate Gaussian distribution, the partition function can be instead considered as an expectation value of some function $f(\mathbf{x}) = \exp(-V(\mathbf{x}))$ and perturbatively expanded away from the quadratic partition function, which is calculable. This is identical to the process demonstrated in 1.6.3, where the partition function $\mathcal{Z}[J]$ for the quartic field theory was approximated via an expansion around the quadratic partition function $\mathcal{Z}_0[J]$. As discussed in the appendix A.7, given a multivariate Gaussian distribution

$$G(\mathbf{x};\mathbf{m},\mathbf{\Sigma}) = \frac{1}{\sqrt{(2\pi)^n \det(\mathbf{\Sigma})}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{m})^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mathbf{m})\right), \quad \mathbf{x}\in\mathbb{R}^n, \tag{4.17}$$

the moment generating function of that distribution can be calculated as

$$M_{G(\mathbf{x})}(t) = \exp\left(\mathbf{t}^T\mathbf{m} + \frac{1}{2}\mathbf{t}^T\mathbf{\Sigma}\mathbf{t}\right), \tag{4.18}$$

leaving that the expectation value of a function $f(\mathbf{x})$ over that distribution can be written as

$$\langle f((\mathbf{x})\rangle_{G(\mathbf{x})} = \exp\left(\sum_{i=1}^n m_i \frac{\partial}{\partial x_i} + \frac{1}{2}\sum_{i,j=1}^n \mathbf{\Sigma}_{ij}\frac{\partial}{\partial x_i}\frac{\partial}{\partial x_j}\right) f(\mathbf{x})\bigg|_{\mathbf{x}=\mathbf{0}} . \tag{4.19}$$

4.3.2 Writing the partition function as an expectation value

For the model distribution $Q_{2N}(\mathbf{v})$, the partition function can be written down by using the form of the energy (4.5) for the model as

$$Z_{2N} = \int \mathrm{d}\mathbf{v} \, \exp\left(-\left(\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N}\lambda_j^{(\alpha)}v_j^{\alpha} + \frac{1}{2}\sum_{j=1}^{m}\sum_{j'=1}^{m}\lambda_{jj'}^{(2)}v_j v_{j'}\right)\right). \tag{4.20}$$

By separating the quadratic part, this can instead be written as an expectation of the function

$$f(\mathbf{v}) = \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}}\exp\left(-\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N}\lambda_j^{(\alpha)}v_j^{\alpha}\right) \tag{4.21}$$

over the multivariate Gaussian distribution with a mean of zero

$$G\left(\mathbf{v},\mathbf{0},\lambda^{(2)}\right) = \sqrt{\frac{\det\left(\lambda^{(2)}\right)}{(2\pi)^m}}\exp\left(-\frac{1}{2}\sum_{j=1}^{m}\sum_{j'=1}^{m}\lambda_{jj'}^{(2)}v_j v_{j'}\right), \tag{4.22}$$

such that

$$Z_{2N} = \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}}\left\langle \exp\left(-\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N}\lambda_j^{(\alpha)}v_j^{\alpha}\right)\right\rangle_{G\left(\mathbf{v};\mathbf{0},\lambda^{(2)}\right)}. \tag{4.23}$$

By then using the result (4.19), this becomes

$$Z_{2N} = \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}}\exp\left(\frac{1}{2}\sum_{j,j'=1}^{m}\left(\lambda^{(2)}\right)_{jj'}^{-1}\frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)\exp\left(-E'(\mathbf{v})\right)\Bigg|_{\mathbf{v}=\mathbf{0}}, \tag{4.24}$$

where $\left(\lambda^{(2)}\right)_{jj'}^{-1}$ is component $j\,j'$ of the matrix inverse of $\lambda^{(2)}$, and with

$$E'(\mathbf{v}) = \sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N}\lambda_j^{(\alpha)}v_j^{\alpha}. \tag{4.25}$$

This is exactly what is needed ready to begin performing a perturbative expansion in terms of $\lambda_j^{(2N)}$ in order to approximate the partition function for small $\lambda_j^{(2N)}$. To proceed, two main tools are needed. These are

1. The multinomial theorem.

2. The Faà di Bruno formula.

In particular, these two tools are used to calculate the action of the exponential of the derivatives in (4.24). Firstly, the exponential can be expanded the usual way

$$\exp\left(\frac{1}{2}\sum_{j,j'=1}^{m}\lambda_{jj'}^{(2)}\frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right) = \sum_{r=0}^{\infty}\frac{1}{r!}\left(\frac{1}{2}\right)^r\left(\sum_{j,j'=1}^{m}\lambda_{jj'}^{(2)}\frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)^r, \tag{4.26}$$

on which the multinomial theorem can then be applied.

4.3.3 The multinomial theorem

The multinomial theorem states that the sum of m terms, labelled x_s, all raised to the power n can be written as

$$\left(\sum_{s=0}^{m} x_s\right)^n = \sum_{\{k\}\in\mathcal{K}_n} \frac{n!}{k_1!\,k_2!\,\dots k_m!} \prod_{t=1}^{m} x_t^{k_t}, \tag{4.27}$$

where here, the sum is over all tuples contained in the set $\mathcal{K}_n$, where $\mathcal{K}_n$ is the set of all tuples $\{k_1,\,k_2,\,\dots k_m\}$ for which

$$\sum_{s=1}^{m} k_s = n, \tag{4.28}$$

for $k_s \in \mathbb{N}$ [48]. This is a useful result, as it can be applied directly to the right hand side of (4.26). Following the appendix A.8, a first use of the multinomial theorem on the r^{th} term in the expansion of the exponential (4.26) yields

$$\left(\sum_{j,j'=1}^{m} \lambda^{(2)}_{jj'} \frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)^r = \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\dots k_m!} \prod_{t=1}^{m}\left[\left(\sum_{j'=1}^{m} (\lambda^{(2)})^{-1}_{tj'} \frac{\partial}{\partial v_{j'}}\right)^{k_t} \frac{\partial^{k_t}}{\partial v_t^{k_t}}\right]. \tag{4.29}$$

A second use then shows that

$$\left(\sum_{j'=1}^{m} (\lambda^{(2)})^{-1}_{tj'} \frac{\partial}{\partial v_{j'}}\right)^{k_t} = \sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\dots\ell_m!} \prod_{q=1}^{m}\left((\lambda^{(2)})^{-1}_{tq}\frac{\partial}{\partial v_q}\right)^{\ell_q}, \tag{4.30}$$

and so the r^{th} term of the expansion opens out as

$$\left(\sum_{j,j'=1}^{m} \lambda^{(2)}_{jj'} \frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)^r = \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\dots k_m!} \prod_{t=1}^{m}\left(\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\dots\ell_m!}\hat{D}_{(t,\{k\},\{\ell\})}\right), \tag{4.31}$$

with the differential operator $\hat{D}_{(t,\{k\},\{\ell\})}$ being a shorthand for

$$\hat{D}_{(t,\{k\},\{\ell\})} = \left[\prod_{q'=1}^{m}\left((\lambda^{(2)})^{-1}_{tq'}\right)^{\ell_q}\right] \frac{\partial^{k_t}}{\partial v_t^{k_1}} \left(\prod_{q=1}^{m} \frac{\partial^{\ell_q}}{\partial v_q^{\ell_q}}\right). \tag{4.32}$$

The takeaway from expressing the moment generating function this way is that it is now seen that to calculate the partition function (4.24), the behaviour of the term

$$\frac{\partial^{k_t}}{\partial v_t^{k_t}} \left(\prod_{q=1}^{m} \frac{\partial^{\ell_q}}{\partial v_q^{\ell_q}}\right) \exp\left(-\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N} \lambda^{(\alpha)}_j v_j^{\alpha}\right)\Bigg|_{\mathbf{v}=\mathbf{0}} \tag{4.33}$$

needs to be understood. Since this is the action of mixed partial derivatives on a multivariable function, in order to make general statements, use of the Faà di Bruno formula is required.

4.3.4 The Faà di Bruno formula

In order to understand the action of the differentiation in (4.33), the application with a general form of the Faà di Bruno formula can be used. The original formula states that for a composite function of a single variable $f(g(x))$, the n^{th} derivative with respect to x is given by

$$\frac{d^n}{dx^n} f\left(g\left(x\right)\right) = \sum_{\pi \in \Pi_n} f^{(|\pi|)}\left(g\left(x\right)\right) \cdot \prod_{B \in \pi} g^{(|B|)}\left(x\right), \tag{4.34}$$

where here, Π_n is the set of all partitions of the set $\{1, \ldots, n\}$, $B \in \pi$ is a variable which lists all possible "blocks" of the partition π [49]. Appendix A.9 demonstrates the partitioning of a set Π_n into blocks, and how the formula (4.34) can be used to find the fourth derivative of $\exp\left(-x^2\right)$.

The number elements of Π_n is given by the n^{th} Bell number, B_n [50]. In the example given in the appendix, the fourth derivative sets $n = 4$, and consequently, $B_4 = 15$ possible partitions are considered. For completeness, the number of partitions of Π_n given in terms of the sizes $|B_1|$, $|B_2|$, ..., $|B_l|$ is always given by the multinomial coefficient

$$\binom{n}{|B|_1, |B|_2, \ldots, |B|_l} = \frac{n!}{|B|_1! \cdot |B|_2! \cdot \ldots \cdot |B|_l!}. \tag{4.35}$$

This said, it turns out that the original formula (4.34) is not quite enough for application to the expression of interest (4.33), as the function being acted on is multivariate, and the derivatives being taken are mixed in these variables. The multivariate generalisation of (4.34) states that [51]

$$\frac{\partial^n}{\partial x_1 \cdots \partial x_n} f\left(y(\mathbf{x})\right) = \sum_{\pi \in \Pi_n} f^{(|\pi|)}(y) \cdot \prod_{B \in \pi} \frac{\partial^{|B|}}{\prod_{j \in B} \partial x_j} y(\mathbf{x}). \tag{4.36}$$

There are a few subtleties with this formula which require attention before moving on. Firstly, the formula works both when the derivatives ∂x_i are distinguishable and when they are indistinguishable. The consequence of this is that Π_n is no longer the *set* $\{1, \ldots, n\}$ but a *multiset* $\{1, \ldots, n\}$, where now repeated entries within Π_n can appear, and are not discarded when they do so. The product $\prod_{j \in B} \partial x_j$ simply means the product over all of the labels contained within the block B under consideration. Examples of calculating partial derivatives of various multivariable functions have been included in Appendix A.10 in order to help gain familiarity with the procedure.

Given the motivation to consider this formula, it is useful to note at this point that when $f(y(\mathbf{x})) = \exp\left(y(\mathbf{x})\right)$ is chosen (which will always be the case), the formula reduces to

$$\frac{\partial^n}{\partial x_1 \cdots \partial x_n} \exp\left(y(\mathbf{x})\right) = \exp\left(y(\mathbf{x})\right) \sum_{\pi \in \Pi_n} \prod_{B \in \pi} \frac{\partial^{|B|}}{\prod_{j \in B} \partial x_j} y(\mathbf{x}). \tag{4.37}$$

By writing $\mathbf{x} = \mathbf{v}$, and then choosing

$$y(\mathbf{v}) = -E'(\mathbf{v}) \equiv -\sum_{j=1}^{m} \sum_{\alpha \in \mathcal{P}_N} \lambda_j^{(\alpha)} v_j^{\alpha}, \tag{4.38}$$

it follows that

$$\frac{\partial^n}{\partial v_1 \cdots \partial v_n} \exp\left(-E'(\mathbf{v})\right) = \exp\left(-E'(\mathbf{v})\right) \sum_{\pi \in \Pi_n} \prod_{B \in \pi} \frac{\partial^{|B|}}{\prod_{j \in B} \partial v_j} \left(-E'(\mathbf{v})\right), \tag{4.39}$$

which, upon evaluating at $\mathbf{v} = \mathbf{0}$ simplifies to

$$\left.\frac{\partial^n}{\partial v_1 \cdots \partial v_n} \exp\left(-E'(\mathbf{v})\right)\right|_{\mathbf{v}=\mathbf{0}} = \sum_{\pi \in \Pi_n} \prod_{B \in \pi} (-1)^{|B|} \frac{\partial^{|B|}}{\prod_{j \in B} \partial v_j} \left.\left(\sum_{j=1}^{m} \sum_{\alpha \in \mathcal{P}_N} \lambda_j^{(\alpha)} v_j^{\alpha}\right)\right|_{\mathbf{v}=\mathbf{0}}. \tag{4.40}$$

For additional clarity needed later, each partition π of Π will pick up the same label as Π. For example, the multiset $\Pi_{\{\ell\}}$ will have partitions labelled here as $\pi_{\{\ell\}}$. Given this formula, the correct choice of partial derivatives needs to be made. Since (4.33) performs k_t differentiations with respect v_t, followed by ℓ_q differentiations with respect to each v_q, the correct choice of n for application to (4.40) is

$$n = k_t + \sum_{q=1}^{m} \ell_q = 2k_t, \tag{4.41}$$

and the multiset Π_n, which will be denoted from here as $\Pi_{\{\ell\}}$. This is then given by

$$\Pi_{\{\ell\}} = \left\{ \underbrace{t, \, t, \, \ldots, \, t}_{k_t \text{ times}}, \underbrace{1, \, 1, \, \ldots, \, 1}_{\ell_1 \text{ times}}, 2, \ldots, m-1, \underbrace{m, \, \ldots, \, m}_{\ell_m \text{ times}} \right\}. \tag{4.42}$$

There is no need for the additional labelling of k_t on $\Pi_{\{\ell\}}$ because

$$\sum_{q=1}^{m} \ell_q = k_t, \tag{4.43}$$

and so the value of k_t is implicit upon the specification of $\{\ell\}$ By choosing n and Π_n this way, the term (4.33) can be written instead as

$$\sum_{\pi \in \Pi_{\{\ell\}}} \prod_{B \in \pi} (-1)^{|B|} \frac{\partial^{|B|}}{\prod_{j \in B} \partial v_j} \left.\left(\sum_{j=1}^{m} \sum_{\alpha \in \mathcal{P}_N} \lambda_j^{(\alpha)} v_j^{\alpha}\right)\right|_{\mathbf{v}=\mathbf{0}}, \tag{4.44}$$

which is a great simplification, as the content of the exponential has now been extracted for differentiation. Now, to summarise everything so far, the partition function Z_{2N} can be written as

$$Z_{2N} = \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}} \exp\left(\frac{1}{2} \sum_{j,j'=1}^{m} \left(\lambda^{(2)}\right)^{-1}_{jj'} \frac{\partial}{\partial v_j} \frac{\partial}{\partial v_{j'}}\right) \left.\exp\left(-E'(\mathbf{v})\right)\right|_{\mathbf{v}=\mathbf{0}}, \tag{4.45}$$

which after two uses of the multinomial theorem, and use of the Faà di Bruno formula expands out to become

$$Z_{2N} = \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}} \sum_{r=0}^{\infty} \frac{1}{r!} \left(\frac{1}{2}\right)^r \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\ldots k_m!} \prod_{t=1}^{m} \left(\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \right. \tag{4.46}$$
$$\left. \left[\prod_{q'=1}^{m} \left(\left(\lambda^{(2)}\right)^{-1}_{tq'}\right)^{\ell_q}\right] \sum_{\pi_{\{\ell\}}\in\Pi_{\{\ell\}}} \prod_{B\in\pi} (-1)^{|B|} \frac{\partial^{|B|}}{\prod_{\beta\in B}\partial v_\beta} \left(\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N} \lambda_j^{(\alpha)} v_j^\alpha\right)\right)\Bigg|_{\mathbf{v}=\mathbf{0}} .$$

This is a very heavy formula, and its calculation in practice would be computationally intractable. However, by choosing the new $\lambda_j^{(2N)} \ll 1$, it is possible to proceed by linearising the result in terms of each $\lambda_j^{(2N)}$. First however, a discussion will be made regarding terms in the sum over partitions $\pi_{\{\ell\}}$ which will necessarily be zero, and hence can be bypassed in any attempt to make a calculation.

4.3.5 Finding the non-zero contributions to the partition function

By choosing the hidden layer to have Gaussian energies, all cross terms between any pair of visible nodes became bilinear. The consequence of this is that $E'(\mathbf{v})$ has no cross terms between visible nodes, and so it is simple to make deductions about derivatives of it. Firstly, only derivatives $\prod_{\beta\in B}\partial v_\beta$ which are not mixed will be non-zero. This means that any block whose elements are not all identical will have a vanishing derivative. As such, no contribution is made unless all $\beta \in B$ are identical. Knowing this, the derivatives simplify to

$$\frac{\partial^{|B|}}{\prod_{\beta\in B}\partial v_\beta} \left(\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N} \lambda_j^{(\alpha)} v_j^\alpha\right)\Bigg|_{\mathbf{v}=\mathbf{0}} = \sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N} \lambda_j^{(\alpha)} \frac{\partial^{|B|}}{\partial v_\beta^{|B|}} v_j^\alpha \Bigg|_{\mathbf{v}=\mathbf{0}} \tag{4.47}$$

when all elements of $\beta \in B$ are identical. The term evaluates to zero otherwise. To evaluate the above derivative, consider the n^{th} derivative of a polynomial x^k evaluated at $x = 0$;

$$\frac{d^n}{dx^n} x^k \Bigg|_{x=0} = \begin{cases} 0, & \text{if } n > k, \\ k!, & \text{if } n = k, \\ 0, & \text{if } n < k. \end{cases} \tag{4.48}$$

For a given k, all choices of n except $n = k$ vanish when the evaluation of the derivative is made, and a similar situation will apply to the derivatives in (4.47). Only blocks B with all elements being identical *and* with the multiplicity of those blocks being equal to some $\alpha \in \mathcal{P}_N$ will contribute non-zero amounts to the sum. This means that the expression (4.47) simplifies to

$$\sum_{j=1}^{m}\sum_{\alpha\in\mathcal{P}_N} \lambda_j^{(\alpha)} \frac{\partial^{|B|}}{\partial v_\beta^{|B|}} v_j^\alpha \Bigg|_{\mathbf{v}=\mathbf{0}} = \lambda_\beta^{(|B|)} \times |B|! \tag{4.49}$$

if B only has all identical elements *and* where those elements have a multiplicity $|B| \in \mathcal{P}_N$. The sum evaluates to zero for that block otherwise. This is a powerful result, because for a given partition π, a product is taken over all blocks of that partition in the calculation of the partition function (4.46), and if *any* of the blocks B either have any non-identical elements or have a size not equal to some $\alpha \in \mathcal{P}_N$, then that partition will contribute zero to the sum.

To conclude, the results of this section mean that the sum over all partitions $\pi_{\{\ell\}} \in \Pi_{\{\ell\}}$ can instead be thought of as a sum over particular partitions $\pi^*_{\{\ell\}}$, which are partitions of $\Pi_{\{\ell\}}$, which satisfy the additional properties that all blocks B of $\pi^*_{\{\ell\}}$ must:

1. Have all identical internal elements,
2. Have a size $|B| \in \mathcal{P}_N$.

By only summing over these particular partitions, and by noting that $\sum_{B\in\pi^*} |B| = |\pi^*_{\{\ell\}}|$, the expression for the partition function reduces to

$$Z_{2N} = \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}} \sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} \Bigg(\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \\ \left[\prod_{q'=1}^{m} \left(\left(\lambda^{(2)}\right)^{-1}_{tq'}\right)^{\ell_q}\right] \sum_{\pi^*_{\{\ell\}}\in\Pi_{\{\ell\}}} (-1)^{|\pi^*_{\{\ell\}}|} \prod_{B\in\pi^*} \lambda_\beta^{(|B|)} \times |B|! \Bigg), \tag{4.50}$$

with β being the label for the repeated element within each B. This is a useful milestone, as now that all of the differentiation has been performed, a linearisation procedure can be carried out.

4.3.6 Linearisation part 1

In consideration of the sum over partitions $\pi^*_{\{\ell\}}$ in (4.50), define for conciseness $X^{(2N)}_{\pi^*_{\{\ell\}}}$ as

$$X^{(2N)}_{\pi^*_{\{\ell\}}} := (-1)^{|\pi^*_{\{\ell\}}|} \prod_{B\in\pi^*} \lambda_\beta^{(|B|)} \times |B|!. \tag{4.51}$$

Since each $\pi^*_{\{\ell\}}$ is restricted such that for each $B \in \pi^*_{\{\ell\}}$, $|B| \in \mathcal{P}_N$, the largest possible block B is $2N$, as this is the largest element of $\mathcal{P}_N$. As such, for any partition $\pi^*_{\{\ell\}}$, either that partition has blocks of size $2N$ and less, or only blocks of size less than $2N$. Therefore, the sum over $X^{(2N)}_{\pi^*_{\{\ell\}}}$ can be split in terms of $\pi^*_{[0,\,\{\ell\}]}$, $\pi^*_{[1,\,\{\ell\}]}$, and $\pi^*_{[\geq 2,\,\{\ell\}]}$, where $\pi^*_{[i,\,\{\ell\}]}$ denotes a partition $\pi^*_{\{\ell\}}$ with i blocks of size $2N$. Following this, the sum over partitions decomposes as

$$\sum_{\pi^*_{\{\ell\}}\in\Pi_{\{\ell\}}} X^{(2N)}_{\pi^*_{\{\ell\}}} = \sum_{\pi^*_{[0,\,\{\ell\}]}\in\Pi_{\{\ell\}}} X^{(2N)}_{\pi^*_{[0,\,\{\ell\}]}} + \sum_{\pi^*_{[1,\,\{\ell\}]}\in\Pi_{\{\ell\}}} X^{(2N)}_{\pi^*_{[1,\,\{\ell\}]}} + \sum_{\pi^*_{[\geq 2,\,\{\ell\}]}\in\Pi_{\{\ell\}}} X^{(2N)}_{\pi^*_{[\geq 2,\,\{\ell\}]}}. \tag{4.52}$$

The final term on the right hand side of this expression is discarded for approximation, as it contains the product of at least two $\lambda_\beta^{(2N)}$. The validity of discarding these terms relies on the assumption that those terms to not diverge, meaning that arbitrarily small but non-zero choices of $\lambda_\beta^{(2N)}$ will cause these terms to become insignificant. Note here that the first term on the right hand side of this expression contains the full sum when $k_t < N$, as in this case, the largest partition of $\Pi_{\{\ell\}}$ is less than $2N$. Note also that since k_t is at most r, this means that the first contribution to the sum over $\pi^*_{[1,\,\{\ell\}]}$ occurs when $r = N$. So for example, for a quartic model (where $2N = 4$), the sum over $\pi^*_{[1,\,\{\ell\}]}$ begins on the $r = 2$ term of the original exponential expansion.

The terms from the sum over $\pi^*_{[1,\{\ell\}]}$ will in general be a sum over $\lambda_j^{(2N)}$ with some non-trivial coefficient. This is because, by definition, each partition $\pi^*_{[1,\{\ell\}]}$ has a single block with elements which will produce a $\lambda_\beta^{(2N)}$. By defining $\pi^{*(j)}_{[1,\{\ell\}]}$ as a partition of $\Pi_{\{\ell\}}$ with some properties, the sum (4.52) can be split and decomposed into a sum over each $\lambda_j^{(2N)}$. The properties of $\pi^*_{[1,\,\{\ell\}]}$ for this to be the case are:

1. $\pi^{*(j)}_{[1,\{\ell\}]}$ is a partition of $\Pi_{\{\ell\}}$
2. All blocks of $\pi^{*(j)}_{[1,\{\ell\}]}$ have identical elements internal to that block.
3. All blocks $B \in \pi^{*(j)}_{[1,\{\ell\}]}$ have a size $|B| \in \mathcal{P}_N$.
4. One (and only one) block of $\pi^{*(j)}_{[1,\{\ell\}]}$ has a size $2N$, and all of its elements are j.

With these properties, the second term on the right hand side of (4.52) can be re-written as

$$\sum_{\pi^*_{[1,\,\{\ell\}]}\in\Pi_{\{\ell\}}} X^{(2N)}_{\pi^*_{[1,\,\{\ell\}]}} = \sum_{j=1}^{m} \lambda_j^{(2N)} F_{\pi^{*(j)}_{[1,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right), \tag{4.53}$$

where

$$F_{\pi^{*(j)}_{[1,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right) = \sum_{\pi^{*(j)}_{[1,\,\{\ell\}]}\in\Pi_{\{\ell\}}} (-1)^{|\pi_1^{*(j)}|} \times (2N)! \times \prod_{\substack{B\in\pi^{*(j)}_{[1,\,\{\ell\}]}\\ |B|\neq 2N}} \lambda_\gamma^{|B|} \times |B|!, \tag{4.54}$$

and where γ is the dummy label for the repeated term in each B. Note that if the partition $\pi^{*(j)}_{[1,\{\ell\}]}$ is of size $2N$, then the product on the right hand side of (4.90) evaluates to 1. This is an important result, as it is the first instance in the approximation of the partition function where a clear sum over each $\lambda_j^{(2N)}$ has appeared. By then defining a similar shorthand

$$F_{\pi^*_{[0,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right) = \sum_{\pi_0^*\in\Pi_{\{\ell\}}} X^{(2N)}_{\pi^*_{[0,\,\{\ell\}]}}, \tag{4.55}$$

the sum over partitions of $\Pi_{\{\ell\}}$ can be approximated to first order in $\lambda^{(2N)}$ as

$$\sum_{\pi^*\in\Pi_{\{\ell\}}} X^{(2N)}_{\pi^*} \approx F_{\pi^*_{[0,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right) + \sum_{j=1}^{m} \lambda_j^{(2N)} F_{\pi^{*(j)}_{[1,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right). \tag{4.56}$$

This then appears in the partition function as

$$Z_{2N} \approx \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}} \sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m}\left(\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \right.$$
$$\left. \left[\prod_{q'=1}^{m}\left(\left(\lambda^{(2)}\right)^{-1}_{tq'}\right)^{\ell_q}\right] \left(F_{\pi^*_{[0,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right) + \sum_{j=1}^{m} \lambda_j^{(2N)} F_{\pi^{*(j)}_{[1,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right) \right) \right), \tag{4.57}$$

which now makes explicit how the linearisation in each $\lambda_j^{(2N)}$ will occur.

4.3.7 Linearisation part 2

The next stage in simplifying the approximate form of the partition function Z_{2N} is simply the repackaging of various terms, and the further linearisation of products of these terms. By first setting up the two functions

$$G_{k_t}^{(0)}\left(\lambda^{(2(N-1))}\right) := \sum_{\{\ell\}\in K_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\dots\ell_m!}\left(\prod_{q'=1}^{m}\left(\left(\lambda^{(2)}\right)^{-1}_{q't}\right)^{\ell_{q'}}\right) F_{\pi^*_{[0,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right), \tag{4.58}$$

and

$$G_{j\,k_t}^{(1)}\left(\lambda^{(2(N-1))}\right) := \sum_{\{\ell\}\in K_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\dots\ell_m!}\left(\prod_{q'=1}^{m}\left(\left(\lambda^{(2)}\right)^{-1}_{q't}\right)^{\ell_{q'}}\right) F_{\pi^{*(j)}_{[1,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right), \tag{4.59}$$

the partition function can be expressed as

$$Z_{2N} \approx \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}}\sum_{r=0}^{\infty}\frac{1}{r!}\left(\frac{1}{2}\right)^r \sum_{\{k\}\in\mathcal{K}_r}\frac{r!}{k_1!\,k_2!\,\dots k_m!}\Bigg[\prod_{t=1}^{m}\left(G_{k_t}^{(0)}\left(\lambda^{(2(N-1))}\right)+\sum_{j=1}^{m}\lambda_j^{(2N)}G_{j\,k_t}^{(1)}\left(\lambda^{(2(N-1))}\right)\right)\Bigg]. \tag{4.60}$$

By noting then that

$$\prod_{t=1}^{m}\left(a_t+\sum_{j=1}^{m}b_{tj}x_j\right) = \prod_{t=1}^{m}a_t + \sum_{j=1}^{m}\left[\sum_{t=1}^{m}b_{tj}\prod_{s\neq t}a_s\right]x_j + \mathcal{O}\left(x^2\right), \tag{4.61}$$

and by relabeling

$$\begin{aligned} a_t &\to G_{k_t}^{(0)}\left(\lambda^{(2(N-1))}\right),\\ b_{tj} &\to G_{j\,k_t}^{(1)}\left(\lambda^{(2(N-1))}\right),\ \text{and}\\ x_j &\to \lambda_j^{(2N)}, \end{aligned} \tag{4.62}$$

the product over t in (4.60), can be approximated to linear order in $\lambda_j^{(2N)}$ as

$$\prod_{t=1}^{m}G_{k_t}^{(0)}\left(\lambda^{(2(N-1))}\right)+\sum_{j=1}^{m}\left[\sum_{t=1}^{m}G_{j\,k_t}^{(1)}\left(\lambda^{(2(N-1))}\right)\prod_{s\neq t}G_{k_s}^{(0)}\left(\lambda^{(2(N-1))}\right)\right]\lambda_j^{(2N)}. \tag{4.63}$$

Next, by defining the functions

$$H_{(\{k\},m)}^{(0)}\left(\lambda^{(2(N-1))}\right) := \prod_{t=1}^{m}G_{k_t}^{(0)}\left(\lambda^{(2(N-1))}\right) \tag{4.64}$$

and

$$H^{(1)}_{j\,(\{k\},\,m)}\left(\lambda^{(2(N-1))}\right) := \sum_{t=1}^{m} G^{(1)}_{j\,k_t}\left(\lambda^{(2(N-1))}\right) \prod_{s\neq t} G^{(0)}_{k_s}\left(\lambda^{(2(N-1))}\right), \tag{4.65}$$

and then the functions

$$Y^{(0)}_{r}\left(\lambda^{(2(N-1))}\right) := \sum_{\{k\}\in K_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} H^{(0)}_{(\{k\},\,m)}\left(\lambda^{(2(N-1))}\right), \tag{4.66}$$

and

$$Y^{(1)}_{j\,r}\left(\lambda^{(2(N-1))}\right) := \sum_{\{k\}\in K_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} H^{(1)}_{j\,(\{k\},\,m)}\left(\lambda^{(2(N-1))}\right), \tag{4.67}$$

the partition function can be approximated as

$$Z_{2N} \approx \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}} \sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r \left[Y^{(0)}_r + \sum_{j=1}^{m} \lambda^{(2N)}_j Y^{(1)}_{jr}\right]. \tag{4.68}$$

Note that the product in equation (4.65) should be set to 1 when $m = 1$ for consistency, as in this case the product over t in (4.60) drops out. The formula (4.68) is almost where things need to be in order to factorise out terms proportional to each $\lambda^{(2N)}_j$ in the second term. This is, however, not as non-trivial as it might seem, due to the infinite sum over r. Interchanging the order of summations

$$\sum_{r=0}^{\infty}\sum_{j=1}^{m} \frac{1}{r!}\left(\frac{1}{2}\right)^r \lambda^{(2N)}_j Y^{(1)}_{jr} = \sum_{j=1}^{m}\sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r \lambda^{(2N)}_j Y^{(1)}_{jr} \tag{4.69}$$

requires the consideration of issues of convergence of the two series. These issues are disregarded here and the interchange is made without further consideration, other than highlighting that if such an interchange is not possible, that the results following will be invalidated.

Having performed this interchange, the functions

$$W^{(0)}\left(\lambda^{(2(N-1))}\right) := \lim_{R\to\infty} \sum_{r=0}^{R} \frac{1}{r!}\left(\frac{1}{2}\right)^r Y^{(0)}_r\left(\lambda^{(2(N-1))}\right), \tag{4.70}$$

and

$$W^{(1)}_j\left(\lambda^{(2(N-1))}\right) := \lim_{R\to\infty} \sum_{r=0}^{R} \frac{1}{r!}\left(\frac{1}{2}\right)^r Y^{(1)}_{j\,r}\left(\lambda^{(2(N-1))}\right), \tag{4.71}$$

are defined, allowing the partition function to be expressed as

$$Z_{2N} \approx \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}} \left[W^{(0)} + \sum_{j=1}^{m} \lambda^{(2N)}_j W^{(1)}_j\right]. \tag{4.72}$$

This is the main result with regards to the linearisation of the partition function; its use will stem from the fact that the approximation for Z_{2N} is in the linearisation of each $\lambda^{(2N)}_j$ but not in any $\lambda^{(\alpha)}_j$ for $\alpha < 2N$.

4.3.8 Change in the Kullback-Lebiler divergence result

Since the expression for Z_{2N} is only a linearisation in $\lambda_j^{(2N)}$ but not in any $\lambda_j^{(\alpha)}$ for $\alpha < 2N$, the calculation for the partition function of the older model $Z_{2(N-1)}$ will follow the exact same procedure, but without any approximations needed, as there was no $\lambda_j^{(2N)}$ parameter for that model. Using this, the partition function for the older model can be written

$$Z_{2(N-1)} = \sqrt{\frac{(2\pi)^m}{\det\left(\lambda^{(2)}\right)}} W^{(0)}. \tag{4.73}$$

Insertion of the two expressions for the partition functions Z_{2N} and $Z_{2(N-1)}$ into the formula (4.7) for the difference in the Kullback-Leibler divergence of the two models relative to the target yields

$$\Delta D_{\text{KL}}\left(P || \, Q_{2N}, \, Q_{2(N-1)}\right) \approx \log\left(1 + \frac{1}{W^{(0)}} \sum_{j=1}^{m} \lambda_j^{(2N)} W_j^{(1)}\right) + \sum_{j=1}^{m} \lambda_j^{(2N)} \mathbb{E}_P\left[v_j^{(2N)}\right], \tag{4.74}$$

Upon making a final linearisation by approximating the logarithm via

$$\log(1+x) \approx x, \tag{4.75}$$

and by defining

$$\mathcal{W}_j = \frac{W_j^{(1)}}{W^{(0)}}, \tag{4.76}$$

the change in the Kullback-Leibler divergence upon introduction of the new model can be expressed as

$$\Delta D_{\text{KL}}\left(P || \, Q_{2N}, \, Q_{2(N-1)}\right) \approx \sum_{j=1}^{m} \lambda_j^{(2N)} \left[\mathcal{W}_j + \mathbb{E}_P\left[v_j^{2N}\right]\right]. \tag{4.77}$$

Overall, since $\lambda_j^{(2N)} > 0$ is required, if for *any* visible node j, the model dependent function $-W_j$ is greater than the data dependent quantity $\mathbb{E}_P\left[v_j^{2N}\right]$, then the introduction of a new parameter $\lambda_j^{(2N)}$ will decrease the Kullback-Leibler divergence of the model used overall immediately. That is, there exists a choice of $\lambda_j^{(2N)}$ for which $\Delta D_{\text{KL}} < 0$.

4.4 Verifying the linearisation procedure

As a check of the above working, consider the case examples of two models Q_A and Q_B given by

$$Q_A(v) = \frac{e^{-\frac{1}{2}m^2v^2 - bv}}{Z_A} \quad \text{with } Z_A = \sqrt{\frac{2\pi}{m^2}} e^{\frac{b^2}{2m^2}}, \tag{4.78}$$

and

$$Q_B(v) = \frac{e^{-\lambda v^4 - \frac{1}{2}m^2v^2 - bv}}{Z_B}, \tag{4.79}$$

and with Z_B being without a closed form

$$Z_B = \int \mathrm{d}v\, e^{-\lambda v^4 - \frac{1}{2}m^2 v^2 - bv}. \tag{4.80}$$

Understanding these two models as RBMs with a single visible node, a Gaussian hidden layer, and a bilinear interaction terms, equation (4.77) can be applied. Assuming the validity of section 4.3, it is stated that Q_B can necessarily outperform Q_A when

$$\mathbb{E}_P\left[v_j^{2N}\right] < -\mathcal{W}_j, \tag{4.81}$$

where the subscript 1 is as a replacement for j, as there is only a single visible node here. Here, the newer model has a quartic parameter, and so $2N = 4$. Since $W^{(0)}$ can be given in terms of the simpler model's partition function $Z_{2(N-1)}$, which here is calculable, $W^{(0)}$ can be expressed as

$$W^{(0)} = e^{\frac{b^2}{2m^2}}. \tag{4.82}$$

Substituting this into the expression for $\mathcal{W}_1$ then gives

$$\mathcal{W}_1 = e^{-\frac{b^2}{2m^2}} W_1^{(1)}. \tag{4.83}$$

Setting $j = 1$ in (4.71) then gives

$$\mathcal{W}_1 = e^{-\frac{b^2}{2m^2}} \sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r Y_{1r}^{(1)}. \tag{4.84}$$

When inserting the definition of $Y_{jr}^{(1)}$ from equation (4.67), it should be noted that $K_r = \{\{r\}\}$ only, as $m = 1$. Therefore, the expression simplifies to

$$\mathcal{W}_1 = e^{-\frac{b^2}{2m^2}} \sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r H_{1\,(\{r\},\,1)}^{(1)}. \tag{4.85}$$

Following this, since the product in (4.66) evaluates to 1 when $m = 1$, the expression for $\mathcal{W}_1$ further simplifies to

$$\mathcal{W}_1 = e^{-\frac{b^2}{2m^2}} \sum_{r=0}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r G_{1r}^{(1)}. \tag{4.86}$$

As a reminder, the general expression for each $G_{j\,k_t}^{(1)}$ is

$$G_{j\,k_t}^{(1)}\left(\lambda^{(2(N-1))}\right) := \sum_{\{\ell\}\in K_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!}\left(\prod_{q'=1}^{m}\left(\left(\lambda^{(2)}\right)^{-1}_{q't}\right)^{\ell_{q'}}\right) F_{\pi^{*(j)}_{[1,\,\{\ell\}]}}\left(\lambda^{(2(N-1))}\right). \tag{4.87}$$

The relevant choices here are $j = 1$ and $k_t = r$. Once again, since $m = 1$, $K_{k_t} = \{\{r\}\}$ only. Also, here $\lambda^{(2)}$ is denoted m^2, and so

$$G_{1r}^{(1)} = \frac{r!}{r!}\left(\left(m^2\right)^{-1}\right)^r F_{\pi^{*(1)}_{[1,\,\{r\}]}}, \tag{4.88}$$

leaving

$$\mathcal{W}_1 = e^{-\frac{b^2}{2m^2}} \sum_{r=0}^{\infty} \frac{1}{r!} \left(\frac{1}{2}\right)^r \frac{1}{(m^2)^r} F_{\pi^{*(1)}_{[1,\,\{r\}]}}. \tag{4.89}$$

Each contribution $F_{\pi^{*(j)}_{[1,\,\{r\}]}}\left(\lambda^{(2(N-1))}\right)$ is given by

$$F_{\pi^{*(j)}_{[1,\,\{r\}]}}\left(\lambda^{(2(N-1))}\right) = \sum_{\pi^{*(j)}_{[1,\,\{r\}]} \in \Pi_{\{\ell\}}} (-1)^{|\pi^{*(j)}_{[1,\,\{r\}]}|} \times (2N)! \times \prod_{\substack{B \in \pi^{*(j)}_1 \\ |B| \neq 2N}} \lambda_\gamma^{|B|} \times |B|!. \tag{4.90}$$

Now, all blocks B of $\pi^{*(1)}_{[1,\,\{r\}]}$ each have a size $|B| \in \mathcal{P}_N$. Given the setup, $2N = 4$, and so $\mathcal{P}_N = \{1,\, 4\}$, meaning all blocks B considered are of size either 1 or 4. On top of this, each partition $\pi^{*(1)}_{[1,\,\{r\}]}$ has at most one block of size $2N = 4$. The multiset $\Pi_{\{r\}}$ from which partitions are taken, is given by

$$\Pi_{\{r\}} = \underbrace{\{1, 1, \ldots, 1\}}_{2r \text{ times}}, \tag{4.91}$$

and so for $r \geq 2$, the *only* partitions of this satisfying the above conditions are partitions of the form

$$\pi^{*(1)}_{[1,\,\{r\}]} = \left\{ \{1,\, 1,\, 1,\, 1\},\, \underbrace{\{1\},\, \{1\},\, \ldots,\, \{1\}}_{2r-4 \text{ of times}} \right\}. \tag{4.92}$$

At this point, a simple mistake to make would be in the under-counting of the number of partitions $\pi^{*(1)}_{[1,\,\{r\}]}$ that take this form. As explored in the appendix A.10, the ordering of the blocks of a partition do matter, but the content of the blocks of a partition do matter. Since $\Pi_{\{r\}}$ has $2r$ elements which are all identical, it means that there are $(2r)!$ ways of constructing the partitions $\pi^{*(1)}_{[1,\,\{r\}]}$ of the form (4.92). A simple mistake would be in omitting this and to under-count by only considering a single partition $\pi^{*(1)}_{[1,\,\{r\}]}$ at this stage of the sum. This leads to misleading conclusions regarding the success of the linearisation procedure. The implications of such an error are discussed further in 4.4.2.

4.4.1 Case 1: Counting properly

For $r = 0$ or 1, r is not large enough to construct such partitions and so no contributions to the sum are made from these. In all non-zero contributions, $|\pi^{*(1)}_{[1,\,\{r\}]}| = 1 + 2r - 4 = 2r - 3$ which is odd, and so $(-1)^{|\pi^{*(1)}_{[1,\,\{r\}]}|} = -1$ for all r. With this,

$$F_{\pi^{*(1)}_{[1,\,\{r\}]}} = -4! \times (2r)! \times \prod_{\substack{B \in \pi^{*(j)}_1 \\ |B| \neq 2N}} \lambda_\gamma^{|B|} \times |B|!, \tag{4.93}$$

leaving that

$$\boxed{\mathcal{W}_1 = -4! \times e^{-\frac{b^2}{2m^2}} \sum_{r=2}^{\infty} \frac{(2r)!}{r!} \left(\frac{1}{2}\right)^r \frac{b^{2r-4}}{(m^2)^r}.} \tag{4.94}$$

By performing a ratio test, this is quickly seen as divergent for $b^2 \neq 0$. This means that contributions to linear order in λ are divergent, despite the fact that all contributions on whole will lead to a finite partition function. This points to the invalidity of dropping terms of quadratic order or higher in (4.51), as the coefficients of higher order terms in λ were potentially divergent. Taking note of this, and restraining only to the case where $b^2 = 0$, the expression for $\mathcal{W}_1$ reduces to

$$\mathcal{W}_1 = -4! \times \frac{3}{\left(m^2\right)^2}. \tag{4.95}$$

Finally this leads to a constraint on the fourth moment of P in order for Q_B to automatically be an improvement on Q_A when

$$\mathbb{E}_P \left[v_1^4\right] < 4! \times \frac{3}{\left(m^2\right)^2}. \tag{4.96}$$

This is in disagreement with the results from the discussion in 1.5. This can be seen by the fact that since Q_B is Gaussian, its fourth moment can easily be calculated

$$\mathbb{E}_{Q_B} \left[v_1^4\right] = \sqrt{\frac{m^2}{2\pi}} \int \mathrm{d}v_1 \, v_1^4 e^{-\frac{1}{2}m^2 v_1^4}, \tag{4.97}$$

which evaluates to

$$\mathbb{E}_{Q_B} \left[v_1^4\right] = \frac{3}{\left(m^2\right)^2}. \tag{4.98}$$

This means that the condition (4.95) becomes

$$\mathbb{E}_P \left[v_1^4\right] < 4! \times \mathbb{E}_{Q_A} \left[v_1^4\right], \tag{4.99}$$

in disagreement with the previous results. This is an issue, as if the calculations of section 1.5 are taken as correct, then the linearisation procedure will suggest the possibility of improvement of models Q_A via introduction of a new model Q_B without the re-training of older parameters even when the fourth moment of P is such that

$$\mathbb{E}_{Q_A} \left[v_1^4\right] < \mathbb{E}_P \left[v_1^4\right] < 4! \times \mathbb{E}_{Q_A} \left[v_1^4\right], \tag{4.100}$$

in contradiction with equation (1.5.1). Overall this indicates the failure of the linearisation procedure to act as a useful method for determining the improvement potential of the Gaussian RBM.

4.4.2 Case 2: Counting improperly

The remainder of this section is intended to highlight potential mistakes others attempting this linearisation procedure should be wary of, as a simple miscounting of the number of partitions $\pi^{*(1)}_{[1,\{r\}]}$ leads to conclusions that appear in agreement with test developed earlier (1.5.1).

When the $(2r)!$ ways of constructing the partition (4.92) are (improperly) omitted, the factor of $(2r)!$ drops out of each term in the summand for $F_{\pi^{*(1)}_{[1,\{r\}]}}$. It would then follow that

$$F_{\pi^{*(1)}_{[1,\,\{r\}]}} = -4! \prod_{\substack{B\in\pi_1^{*(j)} \\ |B|\neq 2N}} \lambda_\gamma^{|B|} \times |B|! \tag{4.101}$$

Each block of $\pi^{*(1)}_{[1,\{r\}]}$ which does not have four elements has a single element, and so $\lambda_\gamma^{|B|} = \lambda_\gamma^{(1)}$. Since there is only one node, $\gamma = 1$ only, leaving $\lambda_\gamma^{|B|} = b$ for each term in the product. Overall this would give

$$F_{\pi^{*(1)}_{[1,\,\{r\}]}} = -4! \times b^{2r-4}, \tag{4.102}$$

substitution of which into (4.89) leaves

$$\mathcal{W}_1 = -4! \times e^{-\frac{b^2}{2m^2}} \sum_{r=2}^{\infty} \frac{1}{r!}\left(\frac{1}{2}\right)^r \frac{b^{2r-4}}{(m^2)^r}. \tag{4.103}$$

Expanding out the b independent term $r = 2$ gives

$$\mathcal{W}_1 = -4! \times e^{-\frac{b^2}{2m^2}} \left[\frac{1}{8\,(m^2)^2} + \sum_{r=3}^{\infty} \frac{1}{r!}\left(\frac{1}{2m^2}\right)^r b^{2r-4}\right], \tag{4.104}$$

meaning it would be concluded that Q_B can be introduced without the modification of older parameters when

$$\boxed{\mathbb{E}_P\left[v_1^4\right] < 4! \times e^{-\frac{b^2}{2m^2}} \left[\frac{1}{8\,(m^2)^2} + \sum_{r=3}^{\infty} \frac{1}{r!}\left(\frac{1}{2m^2}\right)^r b^{2r-4}\right].} \tag{4.105}$$

The appearance of such a formula would motivate individuals making the mistake of miscounting the partitions $\pi^{*(1)}_{[1,\{r\}]}$ to consider the two cases of $b = 0$ and $b \neq 0$.

When $b = 0$

A Gaussian hidden layer without biases (that is, where $c_i = 0$ in (3.16)) would not pick up a linear bias in the visible nodes automatically. In these contexts, $b = 0$ can be chosen beforehand to reduce the number of parameters being trained. In this context, the miscounted $\mathcal{W}_1$ simplifies to become

$$\mathcal{W}_1 = -\frac{3}{(m^2)^2}, \tag{4.106}$$

and so equation (4.105) becomes

$$\mathbb{E}_P\left[v_1^4\right] < \frac{3}{(m^2)^2}. \tag{4.107}$$

This means that when $b = 0$, the new model Q_B is an improvement on Q_A when

$$\mathbb{E}_P\left[v_1^4\right] < \mathbb{E}_{Q_A}\left[v_1^4\right], \tag{4.108}$$

in agreement with the result (1.64).

When $b \neq 0$

When $b \neq 0$, the expression (4.105) can be simplified by instead running the sum from $r = 0$ and subtracting away the additional terms introduced. This allows the sum to be replaced with an exponential as

$$\mathbb{E}_P\left[v_1^4\right] < 4! \times \frac{e^{-\frac{b^2}{2m^2}}}{b^4}\left[e^{\frac{b^2}{2m^2}} - \left(1 + \frac{b^2}{2m^2}\right)\right] \tag{4.109}$$

This result provides a useful generalisation to the result in 1.5. Given a trained single node RBM such that b and m^2 have been set as optimal for representing the distribution P, then the right hand side of 4.109 can be easily calculated to determine whether the introduction of Q_B will be beneficial without the re-training of older parameters. This is a sensible function, and although the right-hand side is not defined for $b = 0$, the limit

$$\lim_{b\to 0} 4! \times \frac{e^{-\frac{b^2}{2m^2}}}{b^4}\left[e^{\frac{b^2}{2m^2}} - \left(1 + \frac{b^2}{2m^2}\right)\right] = \frac{3}{(m^2)^2}, \tag{4.110}$$

which is again in agreement with (4.108).

4.4.3 Summary of verification

As demonstrated in 4.4.1, the coefficient of the linear term diverges when $b \neq 0$, and gives answers in contradiction with the earlier test developed when $b = 0$. This hints at one of

1. An issue with the earlier test developed.
2. Issues with the underlying assumptions regarding the possibility of linearisation.
3. A miscalculation on behalf of the author during the linearisation procedure.

Following this, it was demonstrated in 4.4.2 that a simple misunderstanding of the differentiation procedure leading to a miscounting of the number of partitions $\pi^{*(1)}_{[1,\{r\}]}$ of $\Pi_{\{r\}}$ leads to a misleading confirmation of the linearisation procedure with the previously developed test. To emphasize the significance of this finding and to serve as a cautionary note for future researchers, the author wishes to highlight this potential pitfall. This miscalculation serves as a reminder that errors in mathematical derivations can lead to non-obviously false conclusions and erroneous interpretations of results. The importance of meticulousness and transparency in presenting and verifying calculations cannot be overstated.

4.5 Reflection on methodology and findings

In reconsidering the methodology employed in section 4.3, an alternative, more focused strategy for addressing the core problem at hand could have been pursued. A potentially more efficient path might have involved exploring the possibilities of linearization, which was central to the calculations, more early in the calculation. The decision to expand the exponential, employ the multinomial theorem, and utilize the faa di Bruno formula before any linearization was driven by uncertainties about potential issues with early linearization. Hindsight reveals that the complexity introduced at each step stemmed from a seemingly natural progression without a thorough evaluation of the rationale behind each decision. A more deliberate and cautious examination of the methodology, weighing each step's pros and cons, could have led to a more streamlined process.

The primary challenge surfaced in the initial definition of the research question. What was initially anticipated as a brief exploration—whether it is always possible to improve an RBM by adding new parameters—evolved into the most significant contribution to the thesis. The approach began with a simplified model, but as generality increased, so did the complexity. A key lesson was the late discovery of a mistake in the verification process. Despite success in one avenue, section 1.5, the substantial mathematical weight and the need for constant checks in later work presented formidable challenges.

Key findings suggest that the methodology likely approaches plausibility, as evidenced by checks reducing to a correct scenario when a slip-up is made. The decision to retain the error in the final draft was driven by its elusiveness - a mistake that was almost let through during the writing of this thesis. The conclusion that there might be a more sensible approach, with calculated steps rather than leaps of faith, suggests clear room for improvement and potential for a more certain outcome to the investigation.

Exploration of other methods not documented in the thesis exposed the limitations of comparing two distinctly different RBMs without controlling related parameters. Adjusting the approach to fix related parameters and framing the comparison as an incremental improvement highlighted the inadequacies of older, unrecorded methods and the strength behind the new method.

The most notable weakness in the approach taken was a failure to recognize opportunities for simplification. The complexity of the problem, particularly in section 4.3, where the exponential of a multivariate polynomial was involved, could have been reduced by explicitly linearizing this composite function in terms of the new parameters before expanding the resulting multinomial object. In particular, Taylor's theorem guarantees that an analytic function can be expressed as an infinite convergent series in the form of a Taylor series; if this condition is met (which it is for these expansions), linearizing before the expansion is generally valid.

The practical application of this work lies in providing a method that likely works but demands more attention to simplifications. Future work should explore the process further, considering the comments made, and test its efficacy by training an RBM on relevant data.

Chapter 5

Conclusions

In this thesis, an exploration of Restricted Boltzmann Machines was embarked on in light of their connections with Quantum Field Theory. The primary research goal guiding the investigation was to understand when RBMs with continuous labels could be improved upon via the introduction of new parameters but without the re-training of older parameters. The aim was to establish a deeper understanding of the relationship between RBMs and QFT and to explore the applicability and potential improvements of RBMs in modelling complex systems with continuous variables.

The work began by understanding the Kullback-Leibler divergence from an information theory perspective and connecting this to thermodynamics. Following this, a simple test case of using a Gaussian distribution to model some target P was explored along with the conditions for improvement via the introduction of a quartic parameter to the model, but without changing the variance of the Gaussian, was set up. From here, quantities of interest were expressed as a "field integral" to help concrete the link between RBMs and statistical field theory. Then, the gradient of the log-likelihood for a generic RBM with only its inter-layer connections specified was derived, followed by a calculation of the activation functions for individual nodes in these networks. The work of [42] was then reproduced by choosing all nodes to take on binary values and choosing the energy function to take on only linear contributions in each node. After verifying the produced equations, the corresponding RBM was trained on data representing configurations of the Ising model with six lattice sites. The results were in agreement with those of in the reference paper.

From here, motivations to study RBMs through the lens of QFT were discussed in light of the ability of RBMs to well-represent the physics of the Ising model. Following this, essential precautions learned in QFT were brought to light to ensure the investigation did not consider RBMs with overly complicated generality. In particular, the result of calculations of effective actions for specific scalar field theories was demonstrated, and the complexity that arises when the auxiliary field is too general was shown. Given these considerations, the class of RBM to be studied was defined.

Upon the choice of RBM class, the issue of reducing the KL divergence of the RBM relative to a target distribution P via the introduction of a new parameter but without the re-training of older parameters was set up mathematically. Given this, a linearisation procedure was employed to understand how the RBMs KL divergence relative to the target changes upon introducing such a parameter. The linearisation procedure employed the use of moment-generating functions, the multinomial theorem, and Faà di Bruno's formula. The result claimed that if the model-dependent quantity $-W_j$ exceeded the data-dependent quantity

$\mathbb{E}_P\left[v_j^{2N}\right]$ for any of the visible nodes j, then the model could be improved upon. However, this encountered challenges when the procedure was used to check when a Gaussian RBM could be improved via the introduction of a quartic parameter and disagreed with the test developed earlier; this was demonstrated at the end of this thesis. This finding proposed a revisiting of the approach used and a scrutinisation of the assumptions underlying the procedure. In particular, it is suspected that the linearisation procedure is invalid due to the divergent nature of the individual coefficients of the new parameter in the expansion of the partition function of the new model. This said, an accidental miscalculation involving the miscounting of partitions of a set produced results that coincided with the results of the earlier calculations. Meticulous review does not give any indication that this way of counting the set partitions is by any way correct; however, the coincidental agreement with the results of this calculation serves as a reminder of the ease by which non-obviously false conclusions can be made due to mathematical errors. A potential direction for future research would be to take techniques from quantum field theory used to deal with the issue corresponding closely to this, such as the work [52], and apply it to the context of the RBM.

Through this investigation, several important contributions and implications have been identified:

1. **Advancing Understanding of RBMs and QFT Connections:** The exploration of RBMs through the lens of QFT has provided insights into the relationship between these two domains. By connecting the field integrals of statistical field theory to the marginalised distributions of RBM visible layers, a relationship between the two domains has been concreted.

2. **Applicability of RBMs to Complex Systems:** The successful reproduction of RBMs in modeling the Ising model showcases their efficacy in capturing essential features of complex systems. This further demonstrates that RBMs can be valuable tools for understanding a wide range of real-world phenomena beyond the Ising model.

3. **Challenges in Linearization Procedures:** The challenges encountered in the linearization procedure and the disagreement with earlier tests underscore the need for further investigation and careful scrutiny of assumptions in mathematical approaches. This observation calls for future research to explore alternative methods or improvements to linearization techniques for RBMs.

4. **Improving Machine Learning Algorithms:** Insights gained from understanding the conditions for improving RBMs with the introduction of new parameters could lead to the development of enhanced learning strategies and more efficient machine learning algorithms.

5. **Potential Applications in Quantum Computing:** As RBMs and QFT share common mathematical structures, this research may have implications for quantum computing. The insights gained from studying the connections between RBMs and QFT could inspire the development of quantum computing algorithms for machine learning and optimization tasks.

In conclusion, this thesis has contributed to a deeper understanding of the relationship between RBMs and QFT, highlighting the applicability of RBMs in modeling complex systems and offering potential avenues for future research and applications. The investigations undertaken here could advance the field of machine learning. As the quest for better machine learning algorithms and representations of physical phenomena continues, the connections between RBMs and QFT remain an intriguing and promising avenue for further exploration.

Appendix A

A.1 Positivity of the Kullback-Leibler divergence

This positivity of $D_{KL}(P\,||\,Q)$ can be seen by writing the function as an integral over Ω, the probability space of P, and then using the inequality (1.17). It follows that

$$\begin{aligned} D_{KL}(P\,||\,Q) &= \int_\Omega P(x)\log\left(\frac{P(x)}{Q(x)}\right)dx \\ &= -\int_\Omega P(x)\log\left(\frac{Q(x)}{P(x)}\right)dx \\ &\geq -\int_\Omega P(x)\left(\frac{Q(x)}{P(x)}-1\right)dx \\ &= 1-\int_\Omega Q(x)\,dx \\ &\geq 0, \end{aligned} \tag{A.1}$$

where in the final inequality, it was used that $\int_\Omega Q(x)\,dx \leq 1$ because Ω is the probability space of P and not Q, meaning there is a potential that the integral excluded some region of non-zero probability from the domain of Q, leaving the integral less than 1 unless $Q(x) = P(x)$ everywhere.

A.2 Partition function of quartic scalar field theory

The aim of this appendix is to demonstrate that the partition function of $\mathcal{Z}[J]$ of quartic scalar field theory

$$\mathcal{Z}[J] = \int \mathcal{D}\phi \exp\left(\frac{i}{2}\int d^4x\left\{(\partial_\mu\phi)^2 - \frac{\lambda}{4!}\phi^4 - m^2\phi^2\right\} + i\int \mathrm{d}^4xJ\phi\right), \tag{A.2}$$

can be expressed in terms of the partition function of free scalar field theory

$$\mathcal{Z}_0[J] = \int \mathcal{D}\phi \exp\left(\frac{i}{2}\int d^4x\left\{(\partial_\mu\phi)^2 - m^2\phi^2\right\} + i\int d^4xJ\phi\right), \tag{A.3}$$

via the relation

$$\mathcal{Z}[J] = \left[\exp\left(-\frac{i\lambda}{4!}\int d^4z\,\frac{\delta^4}{\delta J(z)^4}\right)\right]\mathcal{Z}_0[J]. \tag{A.4}$$

This demonstration starts by first expanding the exponential

$$\exp\left(-\frac{i\lambda}{4!}\int \mathrm{d}^4z \frac{\delta^4}{\delta J(z)^4}\right)\mathcal{Z}_0\left[J\right] = \sum_{n=0}^{\infty}\frac{1}{n!}\left(-\frac{i\lambda}{4!}\int \mathrm{d}^4z\frac{\delta^4}{\delta J(z)^4}\right)^n \mathcal{Z}_0\left[J\right]. \tag{A.5}$$

By taking the variation into the integral for $\mathcal{Z}_0\left[J\right]$, the partition function becomes

$$\begin{aligned}\mathcal{Z}\left[J\right] = \int \mathcal{D}\phi\Bigg[&\exp\left(\frac{i}{2}\int \mathrm{d}^4x\left((\partial_\mu\phi)^2 - m^2\phi^2\right)\right)\\ &\sum_{n=0}^{\infty}\frac{1}{n!}\left(-\frac{i\lambda}{4!}\int \mathrm{d}^4z\frac{\delta^4}{\delta J(z)^4}\right)^n \exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right)\Bigg],\end{aligned} \tag{A.6}$$

necessitating the calculation of a term like

$$\frac{\delta^4}{\delta J(z)^4}\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right).$$

To compute this, consider the first order variation

$$\begin{aligned}\frac{\delta}{\delta J(z)}\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right) &= \frac{\delta}{\delta J(z)}\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right)\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right)\\ &= \left(-i\int \mathrm{d}^4x\,\delta\left(x-z\right)\phi(x)\right)\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right)\\ &= -i\phi(z)\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right).\end{aligned} \tag{A.7}$$

Applied four times, this yields

$$\frac{\delta^4}{\delta J(z)^4}\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right) = \phi(z)^4\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right). \tag{A.8}$$

As such, the exponential expansion is then

$$\begin{aligned}&\sum_{n=0}^{\infty}\frac{1}{n!}\left(-\frac{i\lambda}{4!}\int \mathrm{d}^4z\frac{\delta^4}{\delta J(z)^4}\right)^n\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right)\\ &= \sum_{n=0}^{4}\frac{1}{n!}\left(-\frac{i\lambda}{4!}\right)^4\left(\int \mathrm{d}^4z\,\phi(z)^4\right)^n\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right)\\ &= \exp\left(-\frac{i\lambda}{4!}\int \mathrm{d}^4z\,\phi(z)^4\right)\exp\left(-i\int \mathrm{d}^4x\, J(x)\phi(x)\right),\end{aligned} \tag{A.9}$$

which, upon insertion into A.5 completes the demonstration.

A.3 Gaussian integrals

Starting with an $N-$dimensional vector $\mathbf{x}$, with components x_j, consider the integral

$$\mathcal{J} = \int \mathrm{d}x_1\,\mathrm{d}x_2\ldots\mathrm{d}x_N \exp\left(-\frac{1}{2}x_i A_{ij} x_j\right). \tag{A.10}$$

In shorthand this can be written

$$\mathcal{J} = \int \mathrm{d}^N x\, \exp\left(-\frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x}\right), \tag{A.11}$$

where $\mathbf{A}$ is a symmetric, real matrix. Since $\mathbf{A}$ is a symmetric matrix, it can be diagonalised by the orthogonal matrix $\mathbf{O}$, such that $\mathbf{A} = \mathbf{O}^T\mathbf{D}\mathbf{O}$, where $\mathbf{D}$ is diagonal. Under this, the integral is then

$$\mathcal{J} = \int \mathrm{d}^N x \exp\left(-\frac{1}{2}\mathbf{x}^T\mathbf{O}^T\mathbf{D}\mathbf{O}\mathbf{x}\right). \tag{A.12}$$

Next, perform the substitution $\mathbf{y} = \mathbf{O}\mathbf{x}$. Since $\mathbf{O}$ was chosen to be orthogonal, the Jacobian representing the change of variables is 1, and so the measure $\mathrm{d}^N\mathbf{y} = |\det(\mathbf{O})|^N \mathrm{d}^N\mathbf{x} = d^N\mathbf{x}$. Now, the integral has become

$$\mathcal{J} = \int \mathrm{d}^N\mathbf{y} \exp\left(-\frac{1}{2}\mathbf{y}^T\mathbf{D}\mathbf{y}\right). \tag{A.13}$$

$\mathbf{D}$ being diagonal means that $\mathbf{y}^T\mathbf{D}\mathbf{y} = \sum_{i=1}^N D_{ii}y_i^2$, allowing each dimension of the integral to be factorised:

$$\begin{aligned}
\mathcal{J} &= \int \mathrm{d}^N\mathbf{x} \exp\left(-\frac{1}{2}\sum_{i=1}^N D_{ii}y_i^2\right) \\
&= \prod_{i=1}^N \int \mathrm{d}x_i \exp\left(-\frac{1}{2}D_{ii}y_i^2\right) \\
&= \prod_{i=1}^N \left(\frac{2\pi}{D_{ii}}\right)^{\frac{1}{2}} \\
&= \left(\frac{(2\pi)^N}{\det\mathbf{D}}\right) \\
&= \left(\frac{(2\pi)^N}{\det\mathbf{A}}\right),
\end{aligned} \tag{A.14}$$

where in the third equality, the standard single-dimensional Gaussian integral

$$\int_{-\infty}^{\infty} e^{-ax^2} = \sqrt{\frac{\pi}{a}} \tag{A.15}$$

was used.

A.4 Gradient of the RBM log-likelihood

This section follows the derivation by [53]. Start by writing the marginalised distribution for the visible layer as

$$q\left(\mathbf{v}\middle|\,\theta\right) = \sum_{\mathbf{h}} \frac{1}{Z} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}, \tag{A.16}$$

with the partition function

$$Z = \sum_{\mathbf{v},\,\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}. \tag{A.17}$$

The marginalised distribution for the hidden layer can be written in terms of this by use of Bayes' theorem. That is

$$q\left(\mathbf{h}\middle|\,\mathbf{v};\,\theta\right) = \frac{q\left(\mathbf{v},\,\mathbf{h}\middle|\,\theta\right)}{q\left(\mathbf{v}\middle|\,\theta\right)}. \tag{A.18}$$

By inserting the definitions of the RBM probability distribution, this becomes

$$q\left(\mathbf{h}|\mathbf{v};\,\theta\right) = \frac{\frac{1}{Z}e^{-E(\mathbf{v}\,\mathbf{h};\,\theta)}}{\frac{1}{Z}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}}, \tag{A.19}$$

which simplifies to

$$q\left(\mathbf{h}\middle|\,\mathbf{v};\,\theta\right) = \frac{e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}}{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}}. \tag{A.20}$$

This is a result to be used in the calculation of the log-likelihood gradient. The gradient with respect to the i^{th} model parameter can be written by writing the partition function as a sum over all possible configurations:

$$\begin{aligned}
\frac{\partial}{\partial\theta_i}\log\left(q\left(\mathbf{v}\middle|\,\theta\right)\right) &= \frac{\partial}{\partial\theta_i}\log\left(\frac{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}}{\sum_{\mathbf{v},\,\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}}\right) \\
&= \frac{\partial}{\partial\theta_i}\left(\log\left(\sum_{\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}\right)\right) - \frac{\partial}{\partial\theta_i}\log\left(\sum_{\mathbf{v}',\,\mathbf{h}} e^{-E(\mathbf{v}',\,\mathbf{h};\,\theta)}\right) \\
&= \frac{-1}{\sum_{\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}}\sum_{\mathbf{h}} e^{-E(\mathbf{v},\,\mathbf{h};\,\theta)}\frac{\partial E\left(\mathbf{v},\,\mathbf{h};\,\theta\right)}{\partial\theta_i} \\
&+ \frac{1}{\sum_{\mathbf{v}'',\,\mathbf{h}} e^{-E(\mathbf{v}'',\,\mathbf{h};\,\theta)}}\sum_{\mathbf{v}',\,\mathbf{h}} e^{-E(\mathbf{v}',\,\mathbf{h};\,\theta)}\frac{\partial E\left(\mathbf{v}',\,\mathbf{h};\,\theta\right)}{\partial\theta_i}.
\end{aligned} \tag{A.21}$$

By using (A.20), this can then be written as

$$\frac{\partial}{\partial\theta_i}\log\left(q\left(\mathbf{v}\middle|\,\theta\right)\right) = -\sum_{\mathbf{h}} q\left(\mathbf{h}\middle|\,\mathbf{v}\right)\frac{\partial E\left(\mathbf{v},\,\mathbf{h}\middle|\,\theta\right)}{\partial\theta_i} + \sum_{\mathbf{v}',\,\mathbf{h}} q\left(\mathbf{v}',\,\mathbf{h};\,\theta\right)\frac{\partial E\left(\mathbf{v}',\,\mathbf{h};\,\theta\right)}{\partial\theta_i}, \tag{A.22}$$

which is the result that will be used.

A.5 Field integrals

The partition function

$$Z(\theta) = \int_{V\times H} \mathrm{d}\mathbf{v}\,\mathrm{d}\mathbf{h}\,\exp\left(\sum_{j=1}^{m} U_j(v_j) + \sum_{i=1}^{n} \tilde{U}_i(h_i) + \sum_{i=1}^{n}\sum_{j=1}^{m} h_i\omega_{ij}v_j\right) \tag{A.23}$$

can be re-expressed in terms of a field integral by breaking up the multidimensional integral into products of pairs of integrals.

$$\begin{aligned} Z(\theta) &= \int_{V\times H} \mathrm{d}\mathbf{v}\,\mathrm{d}\mathbf{h}\,\exp\left(\sum_{i=1}^{n}\sum_{j=1}^{m}\left(\frac{U_j(v_j)}{n} + \frac{\tilde{U}_i(h_i)}{m} + h_i\omega_{ij}v_j\right)\right) \\ &= \int_{V\times H} \mathrm{d}\mathbf{v}\,\mathrm{d}\mathbf{h}\,\prod_{i=1}^{n}\prod_{j=1}^{m}\exp\left(\frac{U_j(v_j)}{n} + \frac{\tilde{U}_i(h_i)}{m} + h_i\omega_{ij}v_j\right) \\ &= \int_{V\times H} \mathrm{d}v_1\,\mathrm{d}v_2\ldots\mathrm{d}v_m\,\mathrm{d}h_1\,\mathrm{d}h_2\ldots\mathrm{d}h_n\,\prod_{i=1}^{n}\prod_{j=1}^{m}\exp\left(\frac{U_j(v_j)}{n} + \frac{\tilde{U}_i(h_i)}{m} + h_i\omega_{ij}v_j\right) \\ &= \prod_{i=1}^{n}\prod_{j=1}^{m}\int_{V^{(j)}\times H^{(i)}} \mathrm{d}v_j\,\mathrm{d}h_i\,\exp\left(\frac{U_j(v_j)}{n} + \frac{\tilde{U}_i(h_i)}{m} + h_i\omega_{ij}v_j\right), \end{aligned} \tag{A.24}$$

where $V^{(j)}$ and $H^{(i)}$ denote the configuration spaces of the j^{th} visible and i^{th} hidden nodes respectively. Upon introduction of the field integral of a function $f(\mathbf{x})$

$$\int \mathcal{D}\mathbf{x}\, f(\mathbf{x}) = \prod_{\alpha=1}^{\dim(\mathbf{x})} \int_{\mathrm{Dom}(x_\alpha)} \mathrm{d}x_\alpha\, f_\alpha(x_\alpha), \tag{A.25}$$

with

$$f(\mathbf{x}) = \prod_{\alpha=1}^{\dim(\mathbf{x})} f_\alpha(x_\alpha), \tag{A.26}$$

the path integral can then be written as

$$Z(\theta) = \iint \mathcal{D}\mathbf{v}\,\mathcal{D}\mathbf{h}\, e^{-U_{v,h}(\mathbf{v},\,\mathbf{h};\,\theta)}, \tag{A.27}$$

with

$$U_{v,h}(\mathbf{v},\,\mathbf{h};\,\theta) = -\sum_{j=1}^{m}\sum_{i=1}^{n}\left(\frac{U_j(v_j)}{n} + \frac{U_i(h_i)}{m} + h_i\omega_{ij}v_j\right). \tag{A.28}$$

The notation $\mathcal{D}\mathbf{v}\,\mathcal{D}\mathbf{h}$ is simply a shorthand for the field integral $\mathcal{D}\mathbf{x}$, with $\mathbf{x} = (\mathbf{v},\,\mathbf{h})$.

A.6 Activation functions

By inserting the hidden node energy (3.16) into the expression for the hidden node activation function (2.44), and by taking the limit of the sum over allowed h_a as an integral over the real line, the hidden node activation function can be expressed via

$$\begin{aligned}\frac{1}{q\left(h_\ell|\,\mathbf{v}\right)} &= \int_{\mathbb{R}} \mathrm{d}h_a \,\exp\left(\sum_{j=1}^{m} v_j\omega_{\ell j}\left(h_a-h_\ell\right)+\tilde{U}_\ell\left(h_a\right)-\tilde{U}_\ell\left(h_\ell\right)\right)\\ &= \int_{\mathbb{R}} \mathrm{d}h_a \,\exp\left(\sum_{j=1}^{m} v_j\omega_{\ell j}\left(h_a-h_\ell\right)-\frac{1}{2}\tilde{m}_\ell^2\left(h_a^2-h_\ell^2\right)+c_\ell\left(h_a-h_\ell\right)\right). \qquad (A.29)\end{aligned}$$

Factorising out where possible gives

$$\begin{aligned}\frac{1}{q\left(h_\ell|\,\mathbf{v}\right)} &= e^{-h_\ell\left(c_\ell+\sum_{j=1}^m \omega_{\ell j}v_j\right)+\frac{1}{2}\tilde{m}_\ell^2 h_\ell^2}\\ &\times \int_{\mathbb{R}} \mathrm{d}h_a \,\exp\left(-\frac{1}{2}\tilde{m}_\ell^2 h_a^2+h_a\left(c_\ell+\sum_{j=1}^{m}\omega_{\ell j}v_j\right)\right). \qquad (A.30)\end{aligned}$$

A simple Gaussian integral then yields

$$\frac{1}{q\left(h_\ell|\,\mathbf{v}\right)} = e^{-h_\ell\left(c_\ell+\sum_{j=1}^m \omega_{\ell j}v_j\right)+\frac{1}{2}\tilde{m}_\ell^2 h_\ell^2}\sqrt{\frac{2\pi}{\tilde{m}_\ell^2}}\exp\left(\frac{\left(c_\ell+\sum_{j=1}^m \omega_{\ell j}v_j\right)^2}{2\tilde{m}_\ell^2}\right). \qquad (A.31)$$

Inverting this, followed by completing the square in h_ℓ then gives the result

$$q\left(h_\ell|\,\mathbf{v}\right) = \sqrt{\frac{\tilde{m}_\ell^2}{2\pi}}\exp\left(-\frac{1}{2}\tilde{m}_\ell^2\left(h_\ell-\frac{c_\ell+\sum_{j=1}^m \omega_{\ell j}v_j}{2\tilde{m}_\ell^2}\right)^2\right), \qquad (A.32)$$

which is a (correctly normalised) Gaussian in v_ℓ. A similar procedure for the visible node activation function gives

$$\begin{aligned}\frac{1}{q\left(v_\ell|\,\mathbf{h}\right)} &= \int_{\mathbb{R}} \mathrm{d}v_a \,\exp\left(\sum_{i=1}^{n} h_i\omega_{i\ell}\left(v_a-v_\ell\right)+U_\ell\left(v_a\right)-U_\ell\left(v_\ell\right)\right)\\ &= \int_{\mathbb{R}} \mathrm{d}v_a \,\exp\left(\sum_{i=1}^{n} h_i\omega_{i\ell}\left(v_a-v_\ell\right)+U_\ell\left(v_a\right)-U_\ell\left(v_\ell\right)\right)\\ &= \int_{\mathbb{R}} \mathrm{d}v_a \,\exp\left(\sum_{i=1}^{n} h_i\omega_{i\ell}\left(v_a-v_\ell\right)-\sum_{\alpha=1,2,4,6,8,\dots}^{2N}\mu_\ell^{(\alpha)}\left(v_a^\alpha-v_\ell^\alpha\right)\right), \qquad (A.33)\end{aligned}$$

which upon factorisation of possible terms out of the integral gives

$$q\left(v_\ell|\,\mathbf{h}\right) = \frac{\exp\left(\sum_{i=1}^n h_i\omega_{i\ell}v_\ell-\sum_{\alpha=1,2,4,6,8,\dots}^{2N}\mu_\ell^{(\alpha)}v_\ell^\alpha\right)}{\int_{\mathbb{R}}\mathrm{d}v_a\,\exp\left(\sum_{i=1}^n h_i\omega_{i\ell}v_a-\sum_{\alpha=1,2,4,6,8,\dots}^{2N}\mu_\ell^{(\alpha)}v_a^\alpha\right)}, \qquad (A.34)$$

which is also clearly normalised.

A.7 Moment generating function of a Gaussian

This derivation follows [54]. Consider a Gaussian distribution $\mathbf{X}$ with probability density function

$$\mu_{\mathbf{X}}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right). \tag{A.35}$$

The moment generating function $M_{\mathbf{X}}(t)$ of this distribution is given by

$$M_{\mathbf{X}}(t) = \mathbb{E}_\mu\left(e^{tx}\right) = \int_{-\infty}^{\infty} e^{tx}\mu_{\mathbf{X}}(x)\,\mathrm{d}x. \tag{A.36}$$

Substitution of μ into this integral yields

$$M_{\mathbf{X}}(t) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(tx - \frac{(x-\mu)^2}{2\sigma^2}\right)\mathrm{d}x. \tag{A.37}$$

By the substitution $u = \frac{x-\mu}{\sqrt{2}\sigma}$, it then follows that

$$\begin{aligned} M_{\mathbf{X}}(t) &= \frac{\sqrt{2}\sigma}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} \exp\left(\left(\sqrt{2}\sigma u + \mu\right)t - u^2\right)\mathrm{d}u \\ &= \frac{\exp(\mu t)}{\sqrt{\pi}} \int_{-\infty}^{\infty} \exp\left(-\left(u^2 - \sqrt{2}\sigma u t\right)\right)\mathrm{d}u \\ &= \frac{\exp(\mu t)}{\sqrt{\pi}} \int_{-\infty}^{\infty} \exp\left(-\left(u - \frac{\sqrt{2}}{2}\sigma t\right)^2 + \frac{1}{2}\sigma^2 t^2\right)\mathrm{d}u, \end{aligned} \tag{A.38}$$

where the last equality was simply completing the square. Then, by a substitution $v = u - \frac{\sqrt{2}}{2}\sigma t$,

$$\begin{aligned} M_{\mathbf{X}}(t) &= \frac{\exp\left(\mu t + \frac{1}{2}\sigma^2 t^2\right)}{\sqrt{\pi}} \int_{-\infty}^{\infty} \exp\left(-v^2\right)\mathrm{d}x \\ &= \frac{\sqrt{\pi}}{\sqrt{\pi}} \exp\left(\mu t + \frac{1}{2}\sigma^2 t^2\right), \end{aligned} \tag{A.39}$$

where the final equality employed the result of the Gaussian integral demonstrated earlier. This gives the final result

$$M_{\mathbf{X}}(t) = \exp\left(\mu t + \frac{1}{2}\sigma^2 t^2\right). \tag{A.40}$$

This is the base case of a more general result which states that if a probability distribution $G(\mathbf{v}) \sim \mathcal{N}(\mu, \mathbf{\Sigma})$, i.e.

$$G(\mathbf{x};\, \mu,\, \mathbf{\Sigma}) = \frac{1}{\sqrt{(2\pi)^n \det(\mathbf{\Sigma})}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T \mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)\right), \quad \mathbf{x} \in \mathbb{R}^n, \tag{A.41}$$

then the moment generating function of that distribution is

$$M_{G(\mathbf{x})}(t) = \exp\left(\mathbf{t}^T\mu + \frac{1}{2}\mathbf{t}^T\mathbf{\Sigma}\mathbf{t}\right). \tag{A.42}$$

By letting the probability distribution μ be a multivariate Gaussian distribution with a mean of zero and with $\mathbf{\Sigma}^{-1} = \mathbf{A}$, it then follows that the expectation value of a function $f(\mathbf{x})$ over that distribution is given by

$$\begin{aligned}\langle f((\mathbf{x})\rangle_\mu &= \sqrt{\frac{1}{(2\pi)^n \det\left(\mathbf{A}^{-1}\right)}}\int f(\mathbf{x})\exp\left(-\frac{1}{2}\sum_{i=1}^{n}A_{ij}x_ix_j\right)d^n\mathbf{x}\\ &= \exp\left(\frac{1}{2}\sum_{i,j=1}^{n}\left(A^{-1}\right)_{ij}\frac{\partial}{\partial x_i}\frac{\partial}{\partial x_j}\right)f(\mathbf{x})\Bigg|_{\mathbf{x}=\mathbf{0}}\end{aligned} \tag{A.43}$$

for functions $f(\mathbf{x})$ in general.

A.8 Multinomial theorem for the partition function

The term under consideration for expansion using the multinomial theorem is

$$\exp\left(\frac{1}{2}\sum_{j,j'=1}^{n}\left(\lambda^{(2)}\right)^{-1}_{jj'}\frac{\partial}{\partial v_i}\frac{\partial}{\partial v_j}\right) = \sum_{r=0}^{\infty}\frac{1}{r!}\left(\frac{1}{2}\sum_{j,j'=1}^{n}\left(\lambda^{(2)}\right)^{-1}_{jj'}\frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)^r. \tag{A.44}$$

The multinomial theorem then states that the expansion of the sum of m terms, labelled x_s, raised to the power n can be written.

$$\left(\sum_{s=0}^{m}x_s\right)^n = \sum_{\{k\}\in\mathcal{K}_n}\frac{n!}{k_1!\,k_2!\ldots k_m!}\prod_{t=1}^{m}x_t^{k_t}, \tag{A.45}$$

To apply this to (A.44), consider first the simplification of the double sum into a single sum, with the second sum packaged inside each term. By defining $X_j = \left(\sum_{i=1}\left(\lambda^{(2)}\right)^{-1}_{jj'}\frac{\partial}{\partial v_{j'}}\right)\frac{\partial}{v_j}$, the double sum is repackaged as

$$\begin{aligned}\sum_{jj'}\left(\lambda^{(2)}\right)^{-1}_{jj'}\frac{\partial}{\partial v_{j'}}\frac{\partial}{\partial v_j} &= \sum_{j=1}\left(\sum_{j'=1}\left(\lambda^{(2)}\right)^{-1}_{jj'}\frac{\partial}{\partial v_{j'}}\right)\frac{\partial}{\partial v_j}\\ &= \sum_{j=1}^{n}X_j,\end{aligned} \tag{A.46}$$

from which it follows that

$$\left(\sum_{ij}^{m}\left(\lambda^{(2)}\right)^{-1}_{jj'}\frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)^r = \left(\sum_{j=1}^{m}X_j\right)^r. \tag{A.47}$$

This can now more easily have the multinomial theorem applied to it. Using (A.45), this expands as

$$\begin{aligned}\left(\sum_{j=1}^{m} X_j\right)^r &= \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} X_t^{k_t} \\ &= \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} \left(\left(\sum_{j'=1}^{m} \left(\lambda^{(2)}\right)^{-1}_{tj} \frac{\partial}{\partial v_{j'}}\right) \frac{\partial}{\partial v_t}\right)^{k_t} \\ &= \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} \left[\left(\sum_{j'=1}^{m} \left(\lambda^{(2)}\right)^{-1}_{tj'} \frac{\partial}{\partial v_{j'}}\right)^{k_t} \left(\frac{\partial}{\partial v_t}\right)^{k_t}\right] .\end{aligned} \tag{A.48}$$

At this stage, a second use of the multinomial expansion is required:

$$\left(\sum_{j'=1}^{m} \left(\lambda^{(2)}\right)^{-1}_{tj'} \frac{\partial}{\partial v_{j'}}\right)^{k_t} = \sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \prod_{q=1}^{m} \left(\left(\lambda^{(2)}\right)^{-1}_{tq} \frac{\partial}{\partial v_q}\right)^{\ell_q} . \tag{A.49}$$

Using this,

$$\begin{aligned}&\left(\sum_{j,j'=1}^{m} \left(\lambda^{(2)}\right)^{-1}_{jj'} \frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)^r \qquad (A.50)\\ &= \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} \left[\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \left(\frac{\partial}{\partial v_t}\right)^{k_t} \prod_{q=1}^{m} \left(\left(\lambda^{(2)}\right)^{-1}_{qt} \frac{\partial}{\partial v_q}\right)^{\ell_q}\right] \\ &= \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} \left[\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \frac{\partial^{k_t}}{\partial v_t^{k_t}} \prod_{q=1}^{m} \left(\left(\left(\lambda^{(2)}\right)^{-1}_{qt}\right)^{\ell_q} \frac{\partial^{\ell_q}}{\partial v_q^{\ell_q}}\right)\right] \\ &= \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} \left[\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \left(\prod_{q'=1}^{m} \left(\left(\lambda^{(2)}\right)^{-1}_{q't}\right)^{\ell_{q'}}\right) \left(\frac{\partial^{k_t}}{\partial v_t^{k_t}} \prod_{q=1}^{m} \left(\frac{\partial^{\ell_q}}{\partial v_q^{\ell_q}}\right)\right)\right] .\end{aligned}$$

This can be written conceptually compactly as

$$\left(\sum_{j,j'=1}^{m} \lambda^{(2)}_{jj'} \frac{\partial}{\partial v_j}\frac{\partial}{\partial v_{j'}}\right)^r = \sum_{\{k\}\in\mathcal{K}_r} \frac{r!}{k_1!\,k_2!\,\ldots k_m!} \prod_{t=1}^{m} \left(\sum_{\{\ell\}\in\mathcal{K}_{k_t}} \frac{k_t!}{\ell_1!\ell_2!\ldots\ell_m!} \hat{D}_{(t,\{k\},\{\ell\})}\right) , \tag{A.51}$$

with

$$\hat{D}_{(t,\{k\},\{\ell\})} = \left[\prod_{q'=1}^{m} \left(\left(\lambda^{(2)}\right)^{-1}_{tq'}\right)^{\ell_q}\right] \frac{\partial^{k_t}}{\partial v_t^{k_1}} \left(\prod_{q=1}^{m} \frac{\partial^{\ell_q}}{\partial v_q^{\ell_q}}\right) . \tag{A.52}$$

A.9 Example of using Faà di Bruno's formula

The Faà di Bruno's formula is

$$\frac{d^n}{dx^n} f\left(g\left(x\right)\right) = \sum_{\pi \in \Pi_n} f^{(|\pi|)}\left(g\left(x\right)\right) \cdot \prod_{B \in \pi} g^{(|B|)}\left(x\right), \tag{A.53}$$

where here, Π_n is the set of all partitions of the set $\{1, \ldots, n\}$, $B \in \pi$ is a variable which lists all possible "blocks" of the partition π. To understand this formula, consider the illustrative example of the Gaussian,

$$y = \exp\left(-x^2\right). \tag{A.54}$$

To find, for example, the fourth derivative of this function, the usual method employed would be repeated differentiation, giving

$$\begin{aligned}
\frac{dy}{dx} &= -x \exp\left(-\frac{1}{2}x^2\right), \\
\frac{d^2y}{dx^2} &= \left(x^2 - 1\right) \exp\left(-\frac{1}{2}x^2\right), \\
\frac{d^3y}{dx^3} &= \left(3x - x^3\right) \exp\left(-\frac{1}{2}x^2\right), \\
\frac{d^4y}{dx^4} &= \left(x^4 - 6x^2 + 3\right) \exp\left(-\frac{1}{2}x^2\right).
\end{aligned} \tag{A.55}$$

To instead use (A.53) to bypass the process of calculating sequential derivatives, start by writing $y(x) = f(g(x))$, and insert $n = 4$ into the formula to find that

$$\frac{d^4y}{dx^4} = \sum_{\pi \in \Pi_n} f^{(|\pi|)}\left(g\left(x\right)\right) \cdot \prod_{B \in \pi} g^{(|B|)}\left(x\right). \tag{A.56}$$

Now, $y(x)$ can be correctly composed by choosing $f(g) = \exp(g)$ and $g(x) = -\frac{1}{2}x^2$. Noting then that $f^{(|\pi|)}(g) = f(g)$ for any partition π, the formula simplifies to

$$\frac{d^4y}{dx^4} = f\left(g\left(x\right)\right) \sum_{\pi \in \Pi_n} \cdot \prod_{B \in \pi} g^{(|B|)}\left(x\right). \tag{A.57}$$

Now, Π_n is the set of all partitions of $\{1, 2, 3, 4\}$, and the sum over π is a sum over each of these partitions. These can be listed out into fifteen possible partitions and organised by block size into five groups:

1. $(\{1, 2, 3, 4\})$
2. $(\{1\}, \{2, 3, 4\}), (\{2\}, \{1, 3, 4\}), (\{3\}, \{1, 2, 4\}), (\{4\}, \{1, 2, 3\}),$
3. $(\{1, 2\}, \{3, 4\}), (\{1, 3\}, \{2, 4\}), (\{1, 4\}, \{2, 3\}),$
4. $(\{1\}, \{2\}, \{3, 4\}), (\{1\}, \{3\}, \{2, 4\}), (\{1\}, \{4\}, \{2, 3\}),$
   $(\{2\}, \{3\}, \{1, 4\}), (\{2\}, \{4\}, \{1, 3\}), (\{3\}, \{4\}, \{1, 2\}),$
5. $(\{1\}, \{2\}, \{3\}, \{4\})$.

The blocks of the partitions in each section are then of sizes (4), (1, 3), (2, 2), (1, 1, 2), and (1, 1, 1, 1). Given these sizes, the relevant derivatives of $g(x)$ are calculated as

$$\begin{aligned} g^{(1)}(x) &= -x, \\ g^{(2)}(x) &= -1, \\ g^{(3)}(x) &= 0, \\ g^{(4)}(x) &= 0. \end{aligned} \tag{A.58}$$

Since the derivatives of third order or higher are vanishing, all partitions with any blocks B with a size $|B| \geq 3$ can be omitted. Thus the relevant partitions can be summarised as

1. Three partitions with two blocks both of size 2.
2. Six partitions with three blocks, where two blocks have size 1 and one block has size 2.
3. One partition with four blocks all of size 1.

With this, equation (A.57) reduces to

$$\begin{aligned} \frac{d^4 y}{dx^4} &= \exp\left(g(x)\right)\left[3g^{(2)}(x)g^{(2)}(x) + 6\left(g^{(1)}(x)\right)^2 g^{(2)}(x) + 1\left(g^{(1)}(x)\right)^4\right] \\ &= \left(x^4 - 6x^2 + 3\right)\exp\left(-\frac{1}{2}x^2\right), \end{aligned} \tag{A.59}$$

which is in agreement with the calculation made using the usual method (A.55).

A.10 Partial derivatives with Faà di Bruno

Given the motivation to consider the multivariate version of the Faà di Bruno formula, it is useful to note at this point that when $f(y(\mathbf{x})) = \exp(y(\mathbf{x}))$ is chosen (which will always be the case), the formula reduces to

$$\frac{\partial^n}{\partial x_1 \cdots \partial x_n}\exp\left(y(\mathbf{x})\right) = \exp\left(y(\mathbf{x})\right)\sum_{\pi\in\Pi_n}\prod_{B\in\pi}\frac{\partial^{|B|}}{\prod_{j\in B}\partial x_j}y(\mathbf{x}). \tag{A.60}$$

To gain familiarity with this formula, and to see how it behaves when the x_i are distinguishable or indistinguishable, consider the two examples below of taking the partial derivative of y with respect to three coordinates.

A.10.1 Example 1: $\partial x_1 \partial x_2 \partial x_3$

Consider the calculation of

$$\frac{\partial^3 \exp\left(y\left(\mathbf{x}\right)\right)}{\partial x_1 \partial x_2 \partial x_3} = \exp\left(y(\mathbf{x})\right)\sum_{\pi\in\Pi_3}\prod_{B\in\pi}\frac{\partial^{|B|}}{\prod_{j\in B}\partial x_j}y(\mathbf{x}), \tag{A.61}$$

where $y\left(\mathbf{x}\right)$ is unspecified. Here, the partitions Π_3 are of $\{1,\ 2,\ 3\}$, and can be listed down into three groups of block sizes as

1. $(\{1,\, 2,\, 3\})$
2. $(\{1,\, 2\}, \{3\}), (\{1,\, 3\}, \{2\}), (\{2,\, 3\}, \{1\}),$
3. $(\{1\}, \{2\}, \{3\})$.

Note here that the ordering of the blocks within a partition is not important, only the content of the blocks themselves. For the first case, where $\pi = (\{1,\, 2,\, 3\})$, $B = \{1,\, 2,\, 3\}$, and so $|B| = 3$. Here, $\prod_{j\in B} \partial x_j = \partial x_1 \partial x_2 \partial x_3$, and therefore for this choice of π,

$$\prod_{B\in\pi} \frac{\partial^{|B|} y(\mathbf{x})}{\prod_{j\in B} \partial x_j} = \frac{\partial^3 y(\mathbf{x})}{\partial x_1 \partial x_2 \partial x_3}. \tag{A.62}$$

The calculation is similar for the other partitions, for example, for $\pi = (\{1,\, 2\}, \{3\})$, there are now two blocks to consider; $B = \{1,\, 2\}$ which has $|B| = 2$, and $B = \{3\}$ which has $|B| = 1$. As such, this partition's contribution to the derivative is

$$\prod_{B\in\pi} \frac{\partial^{|B|} y(\mathbf{x})}{\prod_{j\in B} \partial x_j} = \frac{\partial^2 y(\mathbf{x})}{\partial x_1 \partial x_2} \cdot \frac{\partial y(\mathbf{x})}{\partial x_3}. \tag{A.63}$$

Similar results follow by exchange for $\pi = (\{1,\, 3\}, \{2\})$ and $\pi = (\{2,\, 3\}, \{1\})$. The final partition $\pi = (\{1\}, \{2\}, \{3\})$ yields a product of three first derivatives

$$\prod_{B\in\pi} \frac{\partial^{|B|} y(\mathbf{x})}{\prod_{j\in B} \partial x_j} = \frac{\partial y(\mathbf{x})}{\partial x_1} \cdot \frac{\partial y(\mathbf{x})}{\partial x_2} \cdot \frac{\partial y(\mathbf{x})}{\partial x_3}. \tag{A.64}$$

Putting all of these results together then gives

$$\begin{aligned} \frac{\partial^3 \exp(y(\mathbf{x}))}{\partial x_1 \partial x_2 \partial x_3} &= \exp(y(\mathbf{x})) \qquad\qquad (\text{A.65}) \\ \times \left[\frac{\partial^3 y}{\partial x_1 \partial x_2 \partial x_3} + \frac{\partial^2 y}{\partial x_1 \partial x_2} \cdot \frac{\partial y}{\partial x_3} + \frac{\partial^2 y}{\partial x_1 \partial x_3} \cdot \frac{\partial y}{\partial x_2} + \frac{\partial^2 y}{\partial x_2 \partial x_3} \cdot \frac{\partial y}{\partial x_1} + \frac{\partial y}{\partial x_1} \cdot \frac{\partial y}{\partial x_1} \cdot \frac{\partial y}{\partial x_1} \right]&, \end{aligned}$$

which is in agreement with the result of the calculation by sequential differentiation.

A.10.2 Example 2: $\partial x_1 \partial x_2^2$

Consider a similar example to that above, but this time with a repeated derivative. In particular, consider the calculation of

$$\frac{\partial^3 \exp(y(\mathbf{x}))}{\partial x_1 \partial x_2^2} = \exp(y(\mathbf{x})) \sum_{\pi\in\Pi_3} \prod_{B\in\pi} \frac{\partial^{|B|}}{\prod_{j\in B} \partial x_j} y(\mathbf{x}). \tag{A.66}$$

This is almost identical to the calculation (A.61), but with x_3 set to x_2. The calculation which follows is identical, but only if the repeated entry in $\Pi_3 = \{1, 2, 2\}$ is not omitted. By following the same procedure as above, but with each index $j = 3$ replaced with a 2, it then follows that

$$\frac{\partial^3 \exp\left(y\left(\mathbf{x}\right)\right)}{\partial x_1 \partial x_2^2} = \exp\left(y\left(\mathbf{x}\right)\right)\left[\frac{\partial^3 y}{\partial x_1 \partial x_2^3} + 2\frac{\partial^2 y}{\partial x_1 \partial x_2} \cdot \frac{\partial y}{\partial x_2} + \frac{\partial^2 y}{\partial x_2^2} \cdot \frac{\partial y}{\partial x_1} + \frac{\partial y}{\partial x_1} \cdot \left(\frac{\partial y}{\partial x_2}\right)^2\right],$$

which is, once again, in exact agreement with the formula which follows from application of sequential differentiation.

Bibliography

[1] A. Zamir, H. U. Khan, W. Mehmood, T. Iqbal, and A. U. Akram, “A feature-centric spam email detection model using diverse supervised machine learning algorithms,” *The Electronic Library*, vol. 38, no. 3, pp. 633–657, 2020.

[2] R. Krishnan, P. Rajpurkar, and E. J. Topol, “Self-supervised learning in medicine and healthcare,” *Nature Biomedical Engineering*, vol. 6, no. 12, pp. 1346–1352, 2022.

[3] F. Shen, Z. Yang, X. Zhao, and D. Lan, “Reject inference in credit scoring using a three-way decision and safe semi-supervised support vector machine,” *Information sciences*, vol. 606, pp. 614–627, 2022.

[4] N. Sharma and V. Ranjan, “Credit card fraud detection: A hybrid of pso and k-means clustering unsupervised approach,” in *2023 13th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, IEEE, 2023, pp. 445–450.

[5] G Ganesh, K. N. Reddy, and M. Tripathi, “Characterising online purchasing behaviour,” 2023.

[6] C. Liu, J. Trmal, M. Wiesner, C. Harman, and S. Khudanpur, “Topic identification for speech without asr,” *arXiv preprint arXiv:1703.07476*, 2017.

[7] J. Homolak, “Opportunities and risks of chatgpt in medicine, science, and academic publishing: A modern promethean dilemma,” *Croatian Medical Journal*, vol. 64, no. 1, p. 1, 2023.

[8] P. Saraf, A. Sarkar, and A. Javed, “Terrain adaptive gait transitioning for a quadruped robot using model predictive control,” in *2021 26th International Conference on Automation and Computing (ICAC)*, IEEE, 2021, pp. 1–6.

[9] S. D. Holcomb, W. K. Porter, S. V. Ault, G. Mao, and J. Wang, “Overview on deepmind and its alphago zero ai,” in *Proceedings of the 2018 international conference on big data and education*, 2018, pp. 67–71.

[10] X. C. Vidal, L. D. Maroñas, and Á. D. Suárez, “How to use machine learning to improve the discrimination between signal and background at particle colliders,” *Applied Sciences*, vol. 11, no. 22, p. 11 076, 2021.

[11] A. Aurisano, A. Radovic, D Rocco, *et al.*, “A convolutional neural network neutrino event classifier,” *Journal of Instrumentation*, vol. 11, no. 09, P09001, 2016.

[12] E. Govorkova, E. Puljak, T. Aarrestad, *et al.*, “Autoencoders on field-programmable gate arrays for real-time, unsupervised new physics detection at 40 mhz at the large hadron collider,” *Nature Machine Intelligence*, vol. 4, no. 2, pp. 154–161, 2022.

[13] S. Gharat and Y. Dandawate, “Galaxy classification: A deep learning approach for classifying sloan digital sky survey images,” *Monthly Notices of the Royal Astronomical Society*, vol. 511, no. 4, pp. 5120–5124, 2022.

[14] A. Halder, A. Ghosh, and T. S. Dasgupta, "Machine-learning-assisted prediction of magnetic double perovskites," *Physical Review Materials*, vol. 3, no. 8, p. 084 418, 2019.

[15] D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer, "Machine learning–accelerated computational fluid dynamics," *Proceedings of the National Academy of Sciences*, vol. 118, no. 21, e2101784118, 2021.

[16] W. Wang and J. Gang, "Application of convolutional neural network in natural language processing," in *2018 international conference on information Systems and computer aided education (ICISCAE)*, IEEE, 2018, pp. 64–70.

[17] S. Parveen and P. Green, "Speech recognition with missing data using recurrent neural nets," *Advances in Neural Information Processing Systems*, vol. 14, 2001.

[18] Y. E. UNUTMAZ, A. Demirci, S. M. Tercan, and R. Yumurtaci, "Electrical energy demand forecasting using artificial neural network," in *2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, IEEE, 2021, pp. 1–6.

[19] ANALYTIXLABS, *Fundamentals concepts of neural networks & deep learning*, https://www.analytixlabs.co.in/blog/fundamentals-of-neural-networks/, Accessed on August 31, 2023.

[20] A. L. Hodgkin and A. F. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *The Journal of physiology*, vol. 117, no. 4, p. 500, 1952.

[21] A. Koubaa, W. Boulila, L. Ghouti, A. Alzahem, and S. Latif, "Exploring chatgpt capabilities and limitations: A critical review of the nlp game changer," 2023.

[22] S. Salman and X. Liu, "Overfitting mechanism and avoidance in deep neural networks," *arXiv preprint arXiv:1901.06566*, 2019.

[23] J. Cao, J. Li, X. Hu, X. Wu, and M. Tan, "Towards interpreting deep neural networks via layer behavior understanding," *Machine Learning*, vol. 111, no. 3, pp. 1159–1179, 2022.

[24] X. Guo, H. Huang, and J. Zhang, "Comparison of different variants of restricted boltzmann machines," in *Proceedings of 2nd International Conference on Information Technology and Electronic Commerce*, IEEE, 2014, pp. 239–242.

[25] S. Kamada and T. Ichimura, "A structural learning method of restricted boltzmann machine by neuron generation and annihilation algorithm," in *Neural Information Processing: 23rd International Conference, ICONIP 2016, Kyoto, Japan, October 16–21, 2016, Proceedings, Part IV 23*, Springer, 2016, pp. 372–380.

[26] R. Salakhutdinov, A. Mnih, and G. Hinton, "Restricted boltzmann machines for collaborative filtering," in *Proceedings of the 24th international conference on Machine learning*, 2007, pp. 791–798.

[27] D. Bachtis, G. Aarts, and B. Lucini, "Quantum field-theoretic machine learning," *Physical Review D*, vol. 103, no. 7, p. 074 510, 2021.

[28] O. Krause, A. Fischer, T. Glasmachers, and C. Igel, "Approximation properties of dbns with binary hidden units and real-valued visible units," in *International conference on machine learning*, PMLR, 2013, pp. 419–426.

[29] DataDrivenInvestor, *An intuitive introduction of restricted boltzmann machine (rbm)*, https://medium.datadriveninvestor.com/an-intuitive-introduction-of-restricted-boltzmann-machine-rbm-14f4382a0dbb, Accessed on July 20, 2023.

[30] A Sellami and I. Farah, "Spectra-spatial graph-based deep restricted boltzmann networks for hyperspectral image classification," in *2019 PhotonIcs & Electromagnetics Research Symposium-Spring (PIERS-Spring)*, IEEE, 2019, pp. 1055–1062.

[31] G. Montúfar, "Restricted boltzmann machines: Introduction and review," in *Information Geometry and Its Applications: On the Occasion of Shun-ichi Amari's 80th Birthday, IGAIA IV Liblice, Czech Republic, June 2016*, Springer, 2018, pp. 75–115.

[32] M. J. Golin and H.-S. Na, "Generalizing the kraft-mcmillan inequality to restricted languages," in *Data Compression Conference*, IEEE, 2005, pp. 163–172.

[33] T. Lancaster and S. J. Blundell, *Quantum field theory for the gifted amateur*. OUP Oxford, 2014.

[34] J. Bernstein, "Spontaneous symmetry breaking, gauge theories, the higgs mechanism and all that," *Reviews of modern physics*, vol. 46, no. 1, p. 7, 1974.

[35] D. Tong, "Statistical field theory," *Lecture Notes*, 2017.

[36] S. Weinberg, "High-energy behavior in quantum field theory," *Physical Review*, vol. 118, no. 3, p. 838, 1960.

[37] D. Tong, "Gauge theory," *Lecture notes, DAMTP Cambridge*, vol. 10, 2018.

[38] T. Farrelly and J. Streich, "Discretizing quantum field theories for quantum simulation," *arXiv preprint arXiv:2002.02643*, 2020.

[39] A. Strominger, "Quantum gravity and string theory, what have we learned?" *arXiv preprint hep-th/9110011*, 1991.

[40] G. Münster, "Lattice quantum field theory," *Scholarpedia*, vol. 5, no. 12, p. 8613, 2010.

[41] C. Hamer and M. N. Barber, "Finite-lattice methods in quantum hamiltonian field theory. i. o (2) and o (3) heisenberg models," *Journal of Physics A: Mathematical and General*, vol. 14, no. 1, p. 259, 1981.

[42] G. Cossu, L. Del Debbio, T. Giani, A. Khamseh, and M. Wilson, "Machine learning determination of dynamical parameters: The ising model case," *Physical Review B*, vol. 100, no. 6, p. 064 304, 2019.

[43] s9w, *Magneto: 2d ising model in c++*, https://github.com/s9w/magneto, Accessed: July 2023.

[44] M. Thomas, *Rbm-qft*, https://github.com/markdathomas/RBM-QFT/tree/main, Accessed on December 11, 2023.

[45] D. Skinner, "Quantum field theory ii," *Lecture notes, Part III of the Mathematical Tripos, University of Cambridge*, 2018.

[46] K. G. Zloshchastiev, "Nonminimal particle-like solutions in cubic scalar field theory," *Physics Letters B*, vol. 450, no. 4, pp. 397–404, 1999.

[47] S. Kruglov, "Effective lagrangian at cubic order in electromagnetic fields and vacuum birefringence," *Physics Letters B*, vol. 652, no. 2-3, pp. 146–149, 2007.

[48] F. Buet-Golfouse *et al.*, “A multinomial theorem for hermite polynomials and financial applications,” *Applied Mathematics*, vol. 6, no. 06, p. 1017, 2015.

[49] S. G. Johnston and J. Prochno, “Faà di bruno’s formula and inversion of power series,” *Advances in Mathematics*, vol. 395, p. 108 080, 2022.

[50] S. M. Tanny, “On some numbers related to the bell numbers,” *Canadian Mathematical Bulletin*, vol. 17, no. 5, pp. 733–738, 1975.

[51] M. Hardy, “Combinatorics of partial derivatives,” *arXiv preprint math/0601149*, 2006.

[52] L. Lin and M. Lindsey, “Bold feynman diagrams and the luttinger–ward formalism via gibbs measures: Non-perturbative analysis,” *Archive for Rational Mechanics and Analysis*, vol. 242, pp. 527–579, 2021.

[53] A. Fischer and C. Igel, “An introduction to restricted boltzmann machines,” in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 17th Iberoamerican Congress, CIARP 2012, Buenos Aires, Argentina, September 3-6, 2012. Proceedings 17*, Springer, 2012, pp. 14–36.

[54] S. G. Kwak and J. H. Kim, “Central limit theorem: The cornerstone of modern statistics,” *Korean journal of anesthesiology*, vol. 70, no. 2, pp. 144–156, 2017.