From statistical mechanics to machine learning: effective models for neural activity

Bram Schönfeldt
SCHABR004

Supervised by
Dr Christian Rohwer
&
Associate professor Jonathan Shock

February 13, 2022

Abstract

In the retina, the activity of ganglion cells, which feed information through the optic nerve to the rest of the brain, is all that our brain will ever know about the visual world. The interactions between many neurons are essential to processing visual information and a growing body of evidence suggests that the activity of populations of retinal ganglion cells cannot be understood from knowledge of the individual cells alone. Modelling the probability of which cells in a population will fire or remain silent at any moment in time is a difficult problem because of the exponentially many possible states that can arise, many of which we will never even observe in finite recordings of retinal activity. To model this activity, maximum entropy models have been proposed which provide probabilistic descriptions over all possible states but can be fitted using relatively few well-sampled statistics. Maximum entropy models have the appealing property of being the least biased explanation of the available information, in the sense that they maximise the information theoretic entropy. We investigate this use of maximum entropy models and examine the population sizes and constraints that they require in order to learn nontrivial insights from finite data. Going beyond maximum entropy models, we investigate autoencoders, which provide computationally efficient means of simplifying the activity of retinal ganglion cells.

CONTENTS

ACKNOWLEDGEMENTS

I would like to thank my two brilliant supervisors for helping me through the daunting task of writing a thesis. This was especially difficult during a pandemic where a large portion of the writing of this thesis was undertaken in isolation, and their checkups and encouragements were greatly appreciated. I am very grateful for their support, and am confident that they will continue to play a role in my life going forward. I would also like to thank Yasser Roudi and Elena Agliari for promptly responding when I reached out to them and for their valuable responses to my questions about strange phrases like ”the perturbative regime” and ”importance sampling”. In addition, I would like to thank Alex Antrobus for our helpful discussion which helped put this research into the context of the broader field of computational neuroscience. On a more familiar note, thank you to everyone at the Vicarage, for always sharing such warmth and delicious meals with me during my stay there and now during my frequent visits. To my incredible significant other, Joséphine, thank you for all the emotional support and happiness you have brought me during this period. Lastly, to my now physically distant family, you will always remain close to my heart.

1. INTRODUCTION

The neuron doctrine, first enunciated by Ramón y Cajal, 1888 and Sherrington, 1906, states that the neuron is the structural and functional unit of the nervous system and has served as a conceptual foundation for neuroscience for over a century (Yuste, 2015). Though this doctrine has helped guide progress, its methodological focus on the activity of single cells has left emergent properties of populations of cells largely unexplored. While significant progress has been made in understanding how single neurons process signals, we are still far from understanding how populations of neurons collectively process information (Adrianna Renee Loback, 2018). Could statistical models provide evidence that there are emergent properties of populations of cells that cannot be explained by our understanding of the individual cells that comprise them, and furthermore, could these models aid our understanding of these emergent properties?

Savin and Tkačik, 2017 define *collective behavior* as occurring when the distribution of activity configurations or states explored by a population of cells has nontrivial structure that cannot be explained by the statistical properties of individual cells alone. With the application of new recording techniques that facilitate long, stable recordings from larger populations of neurons and the creation of spike sorting software for automatically detecting action potentials or 'spikes' from electrical signals, we are now able to better study this collective behaviour (Shlens et al., 2006; Olivier Marre, Amodei, et al., 2012; Ahrens et al., 2013; Golub, Byron, and Chase, 2015). We focus on work done in understanding the behaviour of populations of retinal ganglion cells (RGCs) - the earliest circuit involved in visual processing (Adrianna Renee Loback, 2018). The motivation behind studying RGCs goes beyond the fact that these cells are experimentally accessible and that it is possible to record the activity from all relevant cells within this local circuit, which in itself is an extraordinary feat of engineering. A growing body of evidence suggests that even at this early stage in visual processing, RGCs exhibit collective behaviour. For instance, a model which assumes the firing of RGCs as being independent falls short of reproducing how often pairs of cells fire together (Schneidman, Berry, et al., 2006).

There has now been almost two decades of work (Schneidman, Berry, et al., 2006; Tkačik, Olivier Marre, Amodei, et al., 2014; Berry II and Tkačik, 2020) in trying to understand this collective behaviour, and one of the tools that has proved useful in this body of work is the maximum entropy (MaxEnt) principle (Jaynes, 1957a). The MaxEnt principle provides a framework for building probabilistic models of systems that are consistent with what information we know about the system, but that otherwise makes as few assumptions as possible about its behaviour. Fitting these models can be framed as a constrained optimisation problem where we attempt to maximise the information theoretic entropy, a measure of the average information or surprise in the system. The result of this optimisation is a probabilistic model that reproduces the constrained statistics that can then be further used to investigate how well the information from the constraints accounts for other statistical features of the data. For instance, the distribution that maximises the entropy while obeying the constraint that the probabilities assigned to each event sum to one is the uniform distribution, which can easily be derived from the MaxEnt principle and is illustrated in Section A.1.2.

In this thesis, we start by introducing RGCs and explain their role in the visual pathway. We then dwell on the implications of representing continuous recordings from RGCs as discrete binary vectors. Having traced how the signals from RGCs get simplified representations as vectors, we explain how we might build probabilistic models over these simplified representations. We start by introducing the maximum entropy principle and explain how we derive the maximum entropy distribution consistent with a set of constraints. We then analyse the convexity of this problem and derive conditions under which the parameterisation of the maximum entropy distribution is unique. Although finding the distribution's parameters is often a strictly convex problem, it is not trivial to find these parameters. Thus, we explain how we might numerically determine them using gradient ascent. As we see, this process involves calculating expectations involving summations over exponentially many states, which motivates our introduction of Monte-Carlo methods where we estimate expectations by sampling from the distribution.

Having established how we derive the maximum entropy description from experimental data, we then note its connection to the Boltzmann distribution from statistical physics. We define various quantities such as the microcanonical Gibbs entropy and the heat capacity and derive them from the Boltzmann distribution. These quantities are later used to characterise the properties of the distribution specified by our MaxEnt models, and support the hypotheses that the states we see in retinal ganglion cells can be grouped into discrete clusters with error-correcting properties.

From a general handling of MaxEnt models, we then introduce some of the specific MaxEnt models that have been used to model RGCs and discuss how we might assess their goodness of fit, including whether certain MaxEnt descriptions of small populations of neurons might be trivial. We propose that one way that we can learn from MaxEnt models is by using them as null models in hypothesis tests to identify the significant statistical features of neural data. Returning to the equivalence of MaxEnt models and the Boltzmann distribution, we also look at what we can learn about the activity through drawing on concepts from statistical physics such as criticality and basins of attraction.

Having covered the theoretical groundwork, we then shift our attention to applying MaxEnt models to actual experimental data obtained from recording the responses of a population of RGCs to a video clip. We first demonstrate our understanding of MaxEnt models by writing code that fits them to the activity of a small number of cells and we show that our implementation learns the same distribution and weights as an existing implementation. Using a data-set that comes from a recording of the activity of 160 salamander retinal ganglion cells, we then fit MaxEnt models to sub-populations of different sizes, constraining these models to reproduce lower-order expectations such as the average probability of individual and pairs of cells firing. We examine how well these models reproduce statistics in the data-set that they were not trained on and investigate the population sizes at which the differences between the empirical and predicted expectations become non-trivial. MaxEnt models that only reproduce lower-order moments present one means of trying to find a simplified description of the activity of this population of cells. As an alternative, we investigate auto-encoders as a means of mapping the activity to condensed latent representations.

2. FROM NEURAL ACTIVITY TO BINARY WORDS

Retinal ganglion cells lie towards the beginning of the visual pathway and feed information to the rest of the brain through the optic nerve. The states of these populations are represented as binary vectors, also referred to as binary words, which indicate which cells fire, denoted 1, and which do not, denoted 0, within a window of time. As an example, let us say we record 4 cells for 20 ms. Then, the state $\boldsymbol{\sigma} = (1101)$ indicates that the 1^{st}, 2^{nd} and 4^{th} cells generated an action potential, or fired, at least once, whereas the 3^{rd} cell remained silent within the 20 ms window of time. We will focus on the probability of observing different states $p(\boldsymbol{\sigma})$ within a recording, where a recording is made by projecting a visual stimulus onto the retina and recording the response from the retinal ganglion cells. We do not look at the probability of observing a state given the stimulus $p(\boldsymbol{\sigma}|\text{stimulus})$, nor do we try model temporal dynamics $p(\boldsymbol{\sigma}_t|\boldsymbol{\sigma}_{t-1}, ...)$. To make an analogy to trying to model language, we are trying to model the frequencies of different words, as opposed to modelling how they get strung together or what inspired a particular choice of words.

Though we start with the above simplified, mathematical description of the state of a population of cells, the signals that biological cells emit are not neat strings of 0s and 1s. Furthermore, chopping up time into discrete windows is another simplification we make and different sizes of windows will have implications for the states that we see. Though our starting point is a pre-processed data-set where someone else has already decided on how to make these simplifications, it is worth reviewing this process.

2.1. Sources of data

Whereas it was once difficult to record the activity of even a handful neurons, as technology has advanced, experimental techniques such as multi-electrode array technology (Olivier Marre, Amodei, et al., 2012; Shlens et al., 2006), light sheet fluorescence microscopy (Ahrens et al., 2013) and *in vivo* two-photon excitation microscopy (Greenberg, Houweling, and Kerr, 2008) have made recording from populations of neurons

in various regions of the brain possible. Out of the various regions in the brain that could be studied, a lot of the literature on modelling populations of neurons centers around populations of RGCs (Puchalla et al., 2005; Schneidman, Berry, et al., 2006; Tkačik, Schneidman, et al., 2009; Tkačik, Olivier Marre, Amodei, et al., 2014; Prentice et al., 2016; Mark L. Ioffe and Berry II, 2017; Berry II and Tkačik, 2020), which are the earliest neural circuit involved in visual processing. We try and motivate what RGCs are and why they are worth studying. We start our attempt to place RGCs in context by explaining what comprises the retina and how visual information flows through the *neural* retina.

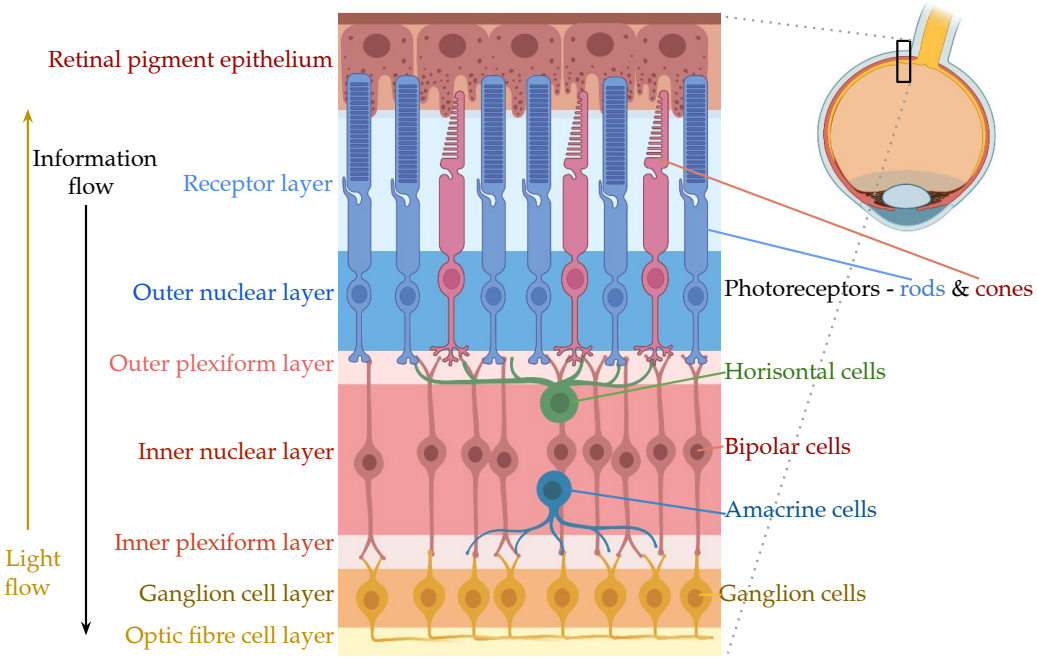


Figure 2.1: Cartoon showing a cross-section of the retinal layers, with the retinal ganglion cells appearing in the ganglion cell layer. The direct pathway of transmission of visual information is photoreceptors → bipolar cells → retinal ganglion cells. The horizontal cells modulate synaptic activity of the photoreceptors and amacrine cells modulate synaptic activity of bipolar and ganglion cells. Created using BioRender.com based on the figure in Adrianna Renee Loback, 2018.

The retina is the innermost layer of the eye and comprises the retinal pigment epithelium and the neural retina, depicted in Figure 2.1. When light makes its way into the eye through the cornea, the lens and then through the vitreous chamber, it, perhaps counter-intuitively, first passes through most of the retinal layers before it reaches the

light sensitive photoreceptors. When light strikes the photoreceptor cells, it initiates a biochemical process which reduces the release of glutamate from the axons of the photoreceptors. The reduced glutamate affects the activity of bipolar and horizontal cells which synapse with the photoreceptors. Though different bipolar cells will react differently to signals from the photoreceptors, bipolar cells may in turn release glutamate affecting the activity of ganglion and amacrine cells. For instance, *off* bipolar cells will depolarise when the photoreceptors that they synapse with are in the dark. Glutamate is excitatory to ganglion cells and causes them to depolarise. In comparison to the photoreceptors and bipolar cells, ganglion cells generate action potentials. The axons of the ganglion cells then exit the eye as the optic nerve transmitting information to downstream brain areas. The direct flow of information, which happens to flow in the opposite direction to light passing through the retinal layers, passes from the photoreceptors to the bipolar cells to the ganglion cells and then through the optic nerve to the brain. The horizontal cells and amacrine cells provide lateral connectivity along the photoreceptors, and bipolar and ganglion cells respectively and modulate the activity of the cells they synapse with, indirectly affecting the transmission of visual information.

This is a rather terse description of how the retina processes visual information which skips over many details. One such detail that should be mentioned is that there are a number of different types of RGCs – one paper (Sanes and Masland, 2015) arguing that there are around 30 – and there is an ongoing effort to classify them based on physiological, morphological, and molecular criteria. For instance, ON-OFF directionally selective ganglion cells, initially described in rabbits by Barlow, Hill, and Levick, 1964, respond to both increases and decreases in light intensity (ON and OFF responses, respectively), and they respond best to motion of a stimulus in a particular direction (Sanes and Masland, 2015). There are also ON directionally selective ganglion cells, intrinsically photosensitive melanopsin-containing RGCs, and local edge detectors, to name a few. We would recommend that those who are interested in finding out what is known about these types of RGCs refer to Sanes and Masland, 2015 for further reading.

Hopefully, the variety and functionally-derived names of these cells draws your attention to the fact that groups of RGCs are not merely light detectors but feature detectors that perform nontrivial transformations on their inputs. Additional evidence of the complexity of RGCs includes evidence that they are involved in predictive computation for moving stimuli (Berry et al., 1999), and periodic temporal patterns (Schwartz et al., 2007), that they exhibit exaggerated responses to unexpected inputs (Deshmukh,

2015) and differential responses to local and global motion (Ölveczky, Baccus, and Meister, 2003). Importantly, since signals from ganglion cells feed into the brain via the optic nerve, all that we will ever know about the visual world starts with the activity of these cells (Sanes and Masland, 2015).

Beyond furthering our understand of their functionality, there are a number of benefits to studying retinal ganglion cells. Firstly, from a practical standpoint, nearly all of the relevant cells in a population of RGCs are experimentally accessible and can be recorded from, using dense multi-electrode arrays, which we explain further in the following section. Secondly, whereas it can be difficult to study populations of neurons in other parts of the brain since their behaviour can be effected by unobserved parts of the brain, there is little known about neural feedback from other parts of the brain that effects the behaviour of retinal ganglion cells (Adrianna Renee Loback, 2018). Finally, to our benefit, a lot of the data used in studies of retinal ganglion cell has been made available online (Tkačik, Olivier Marre, Amodei, et al., 2014; Prentice et al., 2016; Mark L. Ioffe and Berry II, 2017), which saves us the hassle of having to record the data ourselves. This also raises the opportunity to look into the reproducibility of previous results.

As a final note, though we have talked about RGCs at length, the approach to modelling that we take here is not restricted to them, and has also been applied to modelling populations of cells in other parts of the brain, such as the primary visual cortex in macaque monkeys (Ohiorhenuan et al., 2010), and the hippocampus in mice (Meshulam et al., 2021).

2.2. Processing data

2.2.1. Recording

We briefly explain how recordings are obtained from retinal ganglion cells, though we recommend referring to Olivier Marre, Amodei, et al., 2012 or section 2 of Mark L Ioffe, 2017 for further information. The retina is detached from the eye in darkness and kept in oxygenated Ringer's solution in order to prevent it from deteriorating. Once the pigment epithelium is removed, the retina is attached to a semipermeable membrane bathed in poly-L-lysine and then pressed, ganglion cell side down, against a multi-electrode array. The stimulus is then presented to the retina and the raw voltage traces from the array are then digitised and stored using a 252-channel preamplifier for off-line analysis.

Though we now have a digitised recording of the signals detected by the multi-electrode array, we are faced with the difficult task of sifting out the noise, detecting the spikes and finally clustering groups of spikes with similar features corresponding to different neurons. Overall, this process is called *spike sorting* (Rossant et al., 2016), a topic worthy of a thesis in itself. The main steps in spike sorting are:

- *Pre-processing* where we filter out low-frequency activity, for instance using a band pass filter (Rossant et al., 2016) or by estimating the low-frequency activity and then subtracting it from the signal (Olivier Marre, Amodei, et al., 2012), while trying not to alter the spike waveforms .
- *Spike detection* where we detect where and when the spikes are, for instance based on finding all signals with local minima which exceed a certain threshold.
- *Feature extraction* where we extract meaningful features about the shape of the spikes, reducing the dimensionality of the candidate spikes for downstream clustering.
- *Clustering* where we groups spikes with similar features together which we assume correspond to the activity of a particular cell.
- *Manual curation* is typically also involved in this process, for instance in identifying incorrectly clustered groups of spikes, or manually tweaking the threshold for spike detection.

The data we later consider was processed by a spike sorting pipeline developed for the dense multi-electrode array, documented in Olivier Marre, Amodei, et al., 2012. Beyond the spike sorting process, the data-set we consider has additionally been discretised, which we discuss in the next section. Though our starting point is data that has already undergone spike sorting, it should be noted that spike sorting has its own issues, for instance varying standards for how spike sorting should be performed (Wood et al., 2004; Rey, Pedreira, and Quiroga, 2015).

2.2.2. What happens when we discretise?

An important step in moving from *spike trains*, which we can imagine as a sequence of the times at which individual neurons spiked, to a collection of states consisting of 0s and 1s is choosing the width of the time bins that we use use to discretise the spike trains. If we choose bins that are too large, we end up grouping together distinct responses, but if we choose bins that are too narrow, meaningful correlations may end up being spread over multiple bins (Tkačik, Mora, et al., 2015). Though the exact width

of the time bins may be somewhat arbitrary, it is worth identifying sensible ranges for their widths, as well as reflecting on the implications of modelling the instantaneous state of neural data.

Berry II and Tkačik, 2020 point to 3 criteria for choosing the time bin: the temporal precision, the time-scale of noise correlations and the time-scale that it takes downstream neurons to process incoming signals. Intuitively, we want to pick time bins that are less than the time-scale that it takes downstream neurons to process incoming signals, but that are wide enough to encompass the time-scale of the noise correlations and account for the combinatorial nature of neural activity. In the case of the retina, Berry II and Tkačik, 2020 point to 10-20ms as being the correct choice for time bins (Berry II and Tkačik, 2020). If we decrease the width of the time bins, though it may reduce the total number of spikes we observe within each bin and simplify the combinations of firing that we observe, adjacent bins become highly correlated and the assumption of temporal independence becomes increasingly worse (Roudi, Nirenberg, and P. E. Latham, 2009).

Even if we choose this established width of 10-20ms, there are temporal correlations that exist across multiple time bins. In general, knowing that models are successful in modelling spike trains under the assumption of temporal independence does not imply that they will be successful in describing temporally correlated spike trains (Roudi, Nirenberg, and P. E. Latham, 2009). This is something that we must keep in mind from the onset. There are contexts where the assumption of temporal independence may be justified, such as recordings of in vitro cells, or anaesthetised animals. However, when modelling activity in awake animals, the activity of neurons may be affected by the stimulus and the internal state (Donner, Obermayer, and Shimazaki, 2017). Certain stimuli, such as recordings of natural scenes, have long range correlations which may also be present in the recorded spike trains (Roudi, Nirenberg, and P. E. Latham, 2009). One work that used a hidden Markov model to model neural activity across time bins in the retina, found that the transition matrix was dominated by self-transitions, indicating that temporal correlations are dominated by the same state persisting over 3-5 consecutive time bins (Prentice et al., 2016).

Though this is not a question we will answer in this work, one does have to ask to what extent correlations are inherited from either correlations in the stimulus or the influence of top-down processes from elsewhere in the brain. The activity of the cells that we study is clearly not modulated by other parts of the brain since it is isolated from the eye, but correlations could purely be inherited from the stimulus, especially since

retinal ganglion cells are so close to the beginning of visual processing. There has been work done into explicitly developing models that distinguish between the intrinsic and extrinsic interactions (Ferrari et al., 2018), as well as attempts at using MaxEnt models to model temporal correlations (Tang et al., 2008; O. Marre et al., 2009; Vasquez et al., 2012; Mora, Deny, and Olivier Marre, 2015; Donner, Obermayer, and Shimazaki, 2017), though fitting these models to large populations becomes challenging even when using approximate methods (Nguyen, Zecchina, and Berg, 2017).

2.3. Defining observables

We have briefly looked at the process of recording signals from retinal ganglion cells, performing spike sorting and simplifying their representation by discretising the data into binary vectors, which we refer to as states. We now define the quantities we are interested in deriving from these collections of binary words. Suppose we have a dataset $\boldsymbol{D}$ which comprises $|D|$ observations, $\boldsymbol{D} = \{\boldsymbol{\sigma}^{(1)}, ..., \boldsymbol{\sigma}^{(|D|)}\}$. We will typically use the superscript to index the time bin of the observation, and $\boldsymbol{D}$ is ordered in that sense. However, when we compute expectations, we do so over time, which means that we lose temporal information and often we will simply write $\boldsymbol{\sigma}$ without the superscript which represents that state at some random moment in time. Each state, $\boldsymbol{\sigma}^{(t)}$, is an N dimensional vector where N is the number of cells that we model, and each entry of the state is a binary variable $\sigma_i^{(t)} \in \{0, 1\}$ representing whether cell i fired within time bin t. We denote the empirical expectation of some observable $O(\boldsymbol{\sigma})$ as

$$\langle O(\boldsymbol{\sigma}) \rangle_D \doteq \frac{1}{|D|} \sum_{\mu=1}^{|D|} O(\boldsymbol{\sigma}^{(\mu)}),$$

where $\doteq$ means "is defined as", and $\langle \cdot \rangle_D$ denotes a statistic measured from the data as the average over $|D|$ sample elements. For instance, we can define the empirical *average* state of cell i as

$$\langle \sigma_i \rangle_D \doteq \frac{1}{|D|} \sum_{\mu=1}^{|D|} \sigma_i^{(\mu)},$$

which we can intuitively think of as counting the number of states that cell i fired in and dividing by the number of observed states.

Similarly, we can define the pairwise correlation between cell σ_i and cell σ_j as

$$\langle \sigma_i \sigma_j \rangle_D \doteq \frac{1}{|D|} \sum_{\mu=1}^{|D|} \sigma_i^{(\mu)} \sigma_j^{(\mu)},$$

which, since $\sigma_i \in \{0, 1\}$, we can intuitively think of as counting the number of times cell i and cell j fired in the same state and dividing by the number of observed states $|D|$. Analogous definitions exist for higher order correlations, where the C^{th} order correlation will count how many times C specific cells fired together and divide by the number of observed states.

We are also interested in the empirical *population count* distribution $p^{(D)}(K)$, also called the population spike or the spike synchrony distribution. This distribution counts how many times we observed states where K neurons fire and divides by the total number of observed states,

$$p^{(D)}(K) = \frac{1}{|D|} \sum_{\mu=1}^{|D|} \delta \left(\sum_{i=1}^{N} \sigma_i^{(\mu)} - K \right),$$

where $\delta(0) = 1$ and 0 otherwise. As you can see, we are interested in relatively intuitive statistics from the data-set, and the approach to modelling that we take in the next section explores how making use of reliable estimated statistics from the data can give rise to probabilistic models for the data.

From a statistical physics perspective, we emphasize that the observables defined here are *equilibrium* observables in the sense that all temporal realisations of states are treated as equally likely (Kardar, 2007). Depending on the type of stimulus, this assumption of temporal independence may not be valid. For instance, RGC responses to natural scenes are likely to have long range temporal correlations (Roudi, Nirenberg, and P. E. Latham, 2009). However, maximum entropy models that include temporal dynamics are practically unfeasible to fit to even moderate sized populations of RGCs (Gardella, Olivier Marre, and Mora, 2019), and the above observables still provide interesting insights into the behaviour of RGCs.

3. MODELLING BINARY WORDS

3.1. Maximum entropy models

Having explained the context of the data, we now outline the framework we will use to model it. In particular, we look at the Maximum Entropy principle for building probabilistic models from experimental data. We want to find a probability distribution over the set of states that N neurons can take on, which we represent as the set of binary vectors of length N. As the length of the binary vectors grows, the number of possible vectors increases exponentially as 2^N. Estimating the probability distribution over these exponentially many states directly from the data is unrealistic since many of these states may not even appear in neural recordings of finite duration. The question then becomes, "How do we choose a model given that many of the higher-order moments of our distribution are unknown?" The maximum entropy principle provides a means of making use of the statistics that can be reliably estimated from data to build probabilistic models.

However, the maximum entropy principle is not just a means of building a probabilistic model. It also allows us to find the probability distribution that avoids bias while agreeing with the available information. From information theory, the concept of entropy represents a unique measure of the amount of uncertainty represented by a probability distribution. We can identify searching for the least bias explanation of the available information with searching for the distribution that maximises the entropy subject to the available information. This is the maximum entropy principle. It not only is a means of building a probabilistic model consistent with statistics that we can reliably estimate, but a means of finding the probabilistic model which makes as few assumptions as possible, or that has maximum entropy, subject to the available information (Jaynes, 1957a).

One can view this principle as a bridge between a frequentist and a Bayesian approach to investigating neural activity (Savin and Tkačik, 2017). The frequentist approach involves finding salient features of neural computation in the form of summary statistics.

In order to investigate the significance of these features, one might perform hypothesis testing where the data is compared to a control in which some underlying structure that is hypothesised to give rise to these features is disrupted by some means of shuffling the data. For instance, the significance of temporal dependence in neural activity could be investigated by comparing the temporal correlations in data to the temporal correlations obtained in data where the times of spikes are randomly shuffled. On the other hand, the Bayesian approach involves building probabilistic models over the full set of states of the neural activity. In this approach the emphasis is on goodness of fit of the model. MaxEnt models bridge these two approaches in being probabilistic models that can additionally serve as null models that can be viewed as generalisations of frequentist shuffles (Savin and Tkačik, 2017).

3.1.1. Maximum entropy through Lagrange multipliers

We want to find a model of neural activity that reproduces certain observable quantities, but otherwise makes as few assumptions as possible. A way of achieving this is by maximising the entropy of our model, while imposing certain constraints on it (Jaynes, 1957a; Jaynes, 1957b). Information theoretic entropy is a measure of the uncertainty of a system and for discrete systems can be defined as $S \doteq -\sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}) \ln p(\boldsymbol{\sigma})$. The entropy is maximised when a system follows the uniform distribution and is minimized when it only takes on a single state. In order to maximise the entropy while satisfying certain constraints, we use the method of Lagrange multipliers, which is a strategy for finding the extrema of a function that also has to satisfy certain equations exactly.

Suppose we have K observables $O_1(\boldsymbol{\sigma}), ..., O_K(\boldsymbol{\sigma})$ and we want to find the probability mass function $p(\boldsymbol{\sigma})$ such that expectations of the observables match the empirically measured expectations:

$$\langle O_i \rangle \doteq \sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}) O_i(\boldsymbol{\sigma}) = \langle O_i \rangle_D.$$

If we associate the probability of each state with a variable $p(\boldsymbol{\sigma}_i) \to p_i$, then this constrained optimisation problem involves determining the set of values of $\boldsymbol{p} \doteq \{p_i\}_{i=1}^{2^N}$ that satisfies all the constraints, and that maximises the entropy. This corresponds to finding the point $\boldsymbol{p}$ where the gradient of the entropy with respect to $\boldsymbol{p}$ is in the span of the gradients of the constraints. A more detailed explanation of this is presented in the appendix in Section A.1.1. We can formulate an auxiliary function, called the Lagrangian, by introducing the Lagrange multipliers $\lambda_0, ..., \lambda_K$. Maximising the entropy

while satisfying our constraints can then be identified with maximising the Lagrangian with respect to $\boldsymbol{p}, \boldsymbol{\lambda}$:

$$\mathcal{L}(\boldsymbol{p}, \boldsymbol{\lambda}) = -\sum_{i=1}^{2^N} p_i \ln p_i - \lambda_0 \left(\sum_{i=1}^{2^N} p_i - 1 \right) - \sum_{i=1}^{K} \lambda_i \left(\sum_{j=1}^{2^N} p_j O_i(\boldsymbol{\sigma}_j) - \langle O_i \rangle_D \right). \quad (3.1)$$

Here, the constraint multiplied by λ_0 specifies that the probability distribution is normalised. To find the extrema of the Lagrangian, we take partial derivatives of it with respect to $\boldsymbol{p}$ and $\boldsymbol{\lambda}$ and equate it to zero. We start by taking the partial derivative with respect to p_k:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p_k} =& \frac{\partial}{\partial p_k} \text{Entropy} - \frac{\partial}{\partial p_k} \boldsymbol{\lambda}^\top \text{Constraints} = 0 \\ =& -\ln p_k - 1 - \lambda_0 - \sum_{i=1}^{K} \lambda_i O_i(\boldsymbol{\sigma}_k) = 0 \\ p_k =& \frac{1}{Z} \exp\left(-\sum_{i=1}^{K} \lambda_i O_i(\boldsymbol{\sigma}_k) \right). \end{aligned} \quad (3.2)$$

Note, we absorb $\exp(-1 - \lambda_0)$ into the normalisation factor Z^{-1}. If we define the argument of the exponential as the energy E, then this can be identified as the Boltzmann distribution, with Boltzmann's constant and the temperature set equal to 1, $k_B T = 1$:

$$p(\boldsymbol{\sigma}) = \frac{1}{Z} e^{-E(\boldsymbol{\sigma})}, \quad E(\boldsymbol{\sigma}) = \sum_{i=1}^{K} \lambda_i O_i(\boldsymbol{\sigma}).$$

This exponential form arises from taking the partial derivatives of the Lagrangian with respect to $\boldsymbol{p}$. By taking the partial derivatives of the Lagrangian (eq. 3.1) with respect to the Lagrange multipliers and equating this to zero, we recover the constraints:

$$\frac{\partial \mathcal{L}}{\partial \lambda_k} = \langle O_k \rangle - \langle O_k \rangle_D = 0. \quad (3.3)$$

Intuitively, at the maximum of the Lagrangian, the expectations of the observables produced by our model match the expectations measured empirically, and the probability distribution takes takes the form of the Boltzmann distribution. What remains is to find the Lagrange multipliers $\boldsymbol{\lambda}$ which satisfy equation 3.3, which can be done analytically in special cases, but is often done numerically using an optimisation algorithm. We talk about fitting maximum entropy models in Section 3.2.

3.1.2. Uniqueness of the solution

Finding the distribution which maximises the entropy while satisfying a set of constraints using the method of Lagrange multipliers is equivalent to maximising the *likelihood* of the data under the assumption that the observations in the data set are independent. Likelihood refers to $p(\boldsymbol{\sigma}^{(1)}, ..., \boldsymbol{\sigma}^{(|D|)}|\boldsymbol{\lambda})$ as a function of $\boldsymbol{\lambda}$ for a fixed set of observations $\{\boldsymbol{\sigma}^{(\mu)}\}_{\mu=1}^{|D|}$. We typically work with the logarithm of the likelihood, or the 'log-likelihood' which is defined as

$$\begin{aligned} L_D =& \frac{1}{|D|} \ln \prod_{\mu=1}^{|D|} p(\boldsymbol{\sigma}^{(\mu)}|\boldsymbol{\lambda}) \\ =& - \left[\frac{1}{|D|} \sum_{\mu=1}^{|D|} \sum_{i=1}^{K} \lambda_i O_i(\boldsymbol{\sigma}^{(\mu)}) \right] - \ln Z. \end{aligned} \tag{3.4}$$

Note that taking the partial derivative of the term in the brackets in equation 3.4 with respect to λ_i returns the empirical estimate of the expectation $\langle O_i \rangle_D$ and taking the partial derivative of the logarithm of the partition function returns the negative expectation $-\langle O_i \rangle$. Hence, we have

$$\frac{\partial L_D}{\partial \lambda_i} = \langle O_i \rangle - \langle O_i \rangle_D.$$

It easy to see that these two optimisation problems, maximising the entropy subject to certain constraints and maximising the log-likelihood of the data given that our model takes the form of the Boltzmann distribution, are equivalent by noting that the equations of the first partial derivatives in terms of the model parameters or Lagrange multipliers (eq. 3.3) are equivalent in both cases. Hence, any maxima, if they exist, will be the same for both problems. In order for the log-likelihood, or equivalently the Lagrangian, to have a unique maximum, we require the existence of a maximum and the log-likelihood to be a strictly concave function of the parameters $\boldsymbol{\lambda}$. One way of determining the concavity of our objective function is by looking at the Hessian of second partial derivatives, $\frac{\partial L_D}{\partial \lambda_i \lambda_j}$. If the Hessian is negative definite, then the function will be strictly concave and, if the Hessian is only negative semi-definite, the function will only be concave. As a quick illustration of the difference between concavity and strict concavity, a parabola $\cap$ is strictly concave whereas a plateau $\sqcap$ is concave but not strictly concave. Working with the log-likelihood and taking its second partial derivatives we have

$$\frac{\partial L_D}{\partial \lambda_i \lambda_j} = - \left(\langle O_i O_j \rangle - \langle O_i \rangle \langle O_j \rangle \right).$$

A matrix $\boldsymbol{M}$ is negative definite (negative semi-definite) if for any non-negative real column vector $\boldsymbol{a}$, the real number $\boldsymbol{a}^\top \boldsymbol{M} \boldsymbol{a}$ is negative (non-positive). Defining $\mathbb{L}_{ij} \doteq$

$-\frac{\partial L_D}{\partial \lambda_i \lambda_j}$, notice that for all a_i,

$$\begin{aligned}
\boldsymbol{a}^\top \mathbb{L} \boldsymbol{a} &= \sum_{i,j} a_i a_j \left(\langle O_i O_j \rangle - \langle O_i \rangle \langle O_j \rangle \right) \\
&= \left\langle \left[\sum_i a_i O_i - \langle a_i O_i \rangle \right] \left[\sum_j a_j O_j - \langle a_j O_j \rangle \right] \right\rangle \\
&= \left\langle \left[\sum_i a_i O_i - \langle a_i O_i \rangle \right]^2 \right\rangle \geq 0,
\end{aligned}$$

thus, the second partial derivatives are all non-positive (recall we consider $\frac{\partial L_D}{\partial \lambda_i \lambda_j} = -\left(\langle O_i O_j \rangle - \langle O_i \rangle \langle O_j \rangle \right)$ which has a minus sign up front) and the Hessian is negative semi-definite. This implies that the log-likelihood is at least a concave function of the parameters $\boldsymbol{\lambda}$. Note that this does not imply strict concavity. In order for the function to be strictly concave, we require the Hessian to be negative definite, which is the case if no non-trivial linear combination of the observables O_i has vanishing fluctuations,

$$\left\langle \left[\sum_i a_i O_i - \langle a_i O_i \rangle \right]^2 \right\rangle \neq 0.$$

Thus if the maximum of the log-likelihood exists and no non-trivial linear combination of the observables O_i has vanishing fluctuations, then this maximum is unique and the MaxEnt distribution has a unique parameterisation. When some of our constraints are linearly related, as is the case with the K-pairwise model, which we introduce in Section 3.4.5, we end up with multiple parameterisations that specify the same distribution and the parameters have to be set according some convention (Gardella, Olivier Marre, and Mora, 2016).

In summary, the model that reproduces the empirically measured expectations $\langle O_k \rangle_D$ but otherwise makes as few assumptions about the underlying distribution takes the form of the Boltzmann distribution. We found this by maximising an auxiliary function called the Lagrangian, corresponding to maximising the entropy while satisfying certain constraints. We also found that at an extremum of the Lagrangian or equivalently the log-likelihood, the expectations produced by our model have to match the empirically measured expectations. If the maximum exists, and no non-trivial linear combination of the observables O_i has vanishing fluctuations, then this maximum is unique. In the next section we discuss how to determine the parameters $\boldsymbol{\lambda}$ so that our model produces the constrained expectations.

3.2. Fitting maximum entropy models

As opposed to deriving observable quantities from microscopic laws governing the constituents of the system as is typical in statistical physics, we start with observable quantities of a system and try discover the microscopic parameters that give rise to these quantities. We call such problems inverse statistical problems (Nguyen, Zecchina, and Berg, 2017). Given the distribution $p(\boldsymbol{\sigma}) \propto \exp\left(-\sum_{i=1}^{K} \lambda_i O_i(\boldsymbol{\sigma})\right)$, how do we determine the parameters $\lambda_1, ..., \lambda_K$ such that the distribution reproduces the empirically measured expectations? In certain cases such as the independent model, which we define in Section 3.4.2, we can easily derive analytical expressions for $\boldsymbol{\lambda}$ in terms of the observed expectations. In others, such as the pairwise model, determining the model parameters becomes increasingly difficult with increasing system size, and we have to turn to sampling methods. Where applicable, we mention the analytic solutions to specific models in Section 3.4. Here, we discuss general methods that can be used to fit maximum entropy models.

3.2.1. Gradient ascent

We want to determine the parameters $\boldsymbol{\lambda}$ which maximise the Lagrangian (Eq. 3.1), or equivalently the log-likelihood of the observed data (Eq. 3.4). Previously, we determined that the partial derivative of the Lagrangian with respect to parameter λ_k is $\frac{\partial \mathcal{L}}{\partial \lambda_k} = \langle O_k \rangle - \langle O_k \rangle_D$. This suggests the following gradient ascent update rule:

$$\lambda_k^{(t+1)} = \lambda_k^{(t)} + \mu \left(\langle O_k \rangle - \langle O_k \rangle_D \right). \tag{3.5}$$

where μ is the learning rate. Updating each parameter λ_k involves computing the expectation $\langle O_k \rangle = \sum_{\boldsymbol{\sigma}} O_k(\boldsymbol{\sigma}) p(\boldsymbol{\sigma})$, which is a summation over 2^N states. For small systems consisting of around 15 or fewer binary variables, naïve gradient ascent is feasible with standard personal computers, but for larger systems computing these expectations explicitly becomes unfeasible. This is where importance sampling comes into play.

3.2.2. Histogram Monte Carlo

In order to avoid working out expectations by summing over 2^N states, we can use Monte-Carlo Markov Chain (MCMC) techniques to generate samples from our distribution, and then approximate these expectations as averages over these samples. In MCMC techniques, we define an appropriate transition operator, defining the probability of moving from one state to another, such that the stationary distribution that

the Markov chain converges to is the probability distribution from which we want to sample (Newman and Barkema, 1999; Liu, 2008; Binder et al., 2012).

A relatively straightforward learning algorithm that makes use of sampling methods to fit maximum entropy models on moderately sized populations of neurons was proposed by Broderick et al., 2007. Given a learning rule that updates the model parameters and requires calculating expectations, such as the gradient ascent update rule in Equation 3.5, Broderick et al. used ideas from histogram Monte Carlo (Ferrenberg and Swendsen, 1988) to recycle samples over multiple updates of the model parameters. This is in contrast to generating samples following each update, which would be more computationally expensive.

Though more sophisticated sampling techniques exist, they propose Gibbs sampling (S. Geman and D. Geman, 1984; Liu, 2008), where we update the i^{th} dimension of the current sample by sampling from the conditional distributions $p(\sigma_i|\sigma_1, ..., \sigma_{i-1}, \sigma_{i+1}, ..., \sigma_N)$. These N conditional expressions are straightforward to obtain in the case of binary variables, and are

$$p(\sigma_i|\sigma_1, ..., \sigma_{i-1}, \sigma_{i+1}, ..., \sigma_N) = \frac{p(\boldsymbol{\sigma})}{\sum_{\sigma_i=0,1} p(\boldsymbol{\sigma})} = \frac{\exp(-E(\boldsymbol{\sigma}))}{\sum_{\sigma_i=0,1} \exp(-E(\boldsymbol{\sigma}))}.$$

Notice that the conditional probability distribution does not require calculating the partition function Z. In contrast to the Metropolis-Hastings algorithm where we propose samples from a proposal distribution which are then either accepted or rejected based on an acceptance ratio (Hastings, 1970), in sampling directly from the conditional distributions we always accept the proposed samples and thus avoid unnecessary computations. More recent sampling techniques have been proposed to fit maximum entropy models (Ferrari, 2016).

Once we have decided on a means of generating samples, we can reuse these samples for multiple updates as follows. We denote the probability distribution parameterised by $\boldsymbol{\lambda}$ as $p(\boldsymbol{\sigma}|\boldsymbol{\lambda})$, and the expectation of some observable $\Phi(\boldsymbol{\sigma})$ over this distribution by $\langle\Phi\rangle_{\boldsymbol{\lambda}}$. We can use the samples generated by the distribution parameterised by $\boldsymbol{\lambda}$, the outdated parameters, to estimate expectations over the distribution parameterised by

$\boldsymbol{\lambda}'$, the current parameters, as follows:

$$\begin{aligned}
\langle \Phi \rangle_{\boldsymbol{\lambda}'} &\doteq \sum_{\boldsymbol{\sigma}} \Phi(\boldsymbol{\sigma}) p(\boldsymbol{\sigma}|\boldsymbol{\lambda}') \\
&= \sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}|\boldsymbol{\lambda}) \left[\Phi(\boldsymbol{\sigma}) \frac{p(\boldsymbol{\sigma}|\boldsymbol{\lambda}')}{p(\boldsymbol{\sigma}|\boldsymbol{\lambda})} \right] \\
&= \left\langle \Phi(\boldsymbol{\sigma}) \frac{p(\boldsymbol{\sigma}|\boldsymbol{\lambda}')}{p(\boldsymbol{\sigma}|\boldsymbol{\lambda})} \right\rangle_{\boldsymbol{\lambda}} \\
&= \left\langle \Phi(\boldsymbol{\sigma}) \frac{\exp[\boldsymbol{\lambda}' \cdot \boldsymbol{O}(\boldsymbol{\sigma})]}{Z(\boldsymbol{\lambda}')} \frac{Z(\boldsymbol{\lambda})}{\exp[\boldsymbol{\lambda} \cdot \boldsymbol{O}(\boldsymbol{\sigma})]} \right\rangle_{\boldsymbol{\lambda}} \\
&= \frac{\langle \Phi(\boldsymbol{\sigma}) \exp[(\boldsymbol{\lambda}' - \boldsymbol{\lambda}) \cdot \boldsymbol{O}(\boldsymbol{\sigma})] \rangle_{\boldsymbol{\lambda}}}{\langle \exp[(\boldsymbol{\lambda}' - \boldsymbol{\lambda}) \cdot \boldsymbol{O}(\boldsymbol{\sigma})] \rangle_{\boldsymbol{\lambda}}} \\
&\approx \frac{\langle \Phi(\boldsymbol{\sigma}) \exp[(\boldsymbol{\lambda}' - \boldsymbol{\lambda}) \cdot \boldsymbol{O}(\boldsymbol{\sigma})] \rangle_{MC\boldsymbol{\lambda}}}{\langle \exp[(\boldsymbol{\lambda}' - \boldsymbol{\lambda}) \cdot \boldsymbol{O}(\boldsymbol{\sigma})] \rangle_{MC\boldsymbol{\lambda}}},
\end{aligned}$$

where the dot product between the model parameters $\boldsymbol{\lambda}$ and the constrained observables $\boldsymbol{O}(\boldsymbol{\sigma})$ is denoted '$\cdot$' and the expectation calculated over samples obtained via some Monte Carlo method from the model parameterised by $\boldsymbol{\lambda}$ is denoted $\langle \cdot \rangle_{MC\boldsymbol{\lambda}}$. In the last line, each of the expectations in the fraction can be approximated as expectations over the samples generated from the model parameterised by $\boldsymbol{\lambda}$. Thus, in a learning algorithm where updating our model parameters involves computing expectations, such as gradient ascent, we can estimate these expectations as expectations over samples generated by our model with outdated parameters and reuse these samples for multiple updates.

3.3. The Boltzmann distribution

Maximum entropy models represent the most unbiased representation of our knowledge of a system (Jaynes, 1957b), where in practice, 'knowledge' refers to the constrained quantities that we can reliably measure from data. They also happen to map neatly onto the Boltzmann distribution, a familiar distribution in statistical mechanics which describes physical systems in thermal equilibrium (Schneidman, Berry, et al., 2006). Much of the use of maximum entropy models as a tool to study populations of neurons is motivated by trying to transfer concepts from statistical mechanics that arise from the Boltzmann distribution and use them to characterise the behaviour of populations of neurons (Schneidman, Berry, et al., 2006; Tkačik, Olivier Marre, Amodei, et al., 2014; Tkačik, Mora, et al., 2015). We now devote some attention to introducing the Boltzmann distribution and illustrating its relationship to quantities such as energy, entropy (which has a different definition in statistical mechanics to the one in infor-

mation theory), free energy, temperature, heat capacity and specific heat so that we can later facilitate a discussion on how we might bring ideas from statistical mechanics into analysing neural data.

The Boltzmann distribution over a number of states $\boldsymbol{\sigma} = (\sigma_1, ..., \sigma_N)$ takes the form:

$$p(\boldsymbol{\sigma}) = \frac{1}{Z} \exp\left(-\frac{1}{k_B T} E(\boldsymbol{\sigma})\right), \quad Z = \sum_{\boldsymbol{\sigma}} \exp\left(-\frac{1}{k_B T} E(\boldsymbol{\sigma})\right),$$

where $E \in [0, \infty)$ is the energy of each state, k_B is Boltzmann's constant and T is the temperature. The energy function can take various forms depending on the system in question. However, it is worth noting that states with low energy are associated with a high probability and states with high energy are associated with low probabilities. This will later help motivate our definition of energy when we consider models that do not arise from statistical mechanics.

The partition function Z, which normalises the probability distribution, additionally encodes many physical quantities. Specifically, we are going to define the free energy and heat capacity, among other quantities, in relation to it. We start by expressing Z as an integral over energies:

$$\begin{aligned} Z &= \sum_{\boldsymbol{\sigma}} e^{-E(\boldsymbol{\sigma})/k_B T} \\ &= \int_0^\infty dE e^{-E/k_B T} \underbrace{\sum_{\boldsymbol{\sigma}} \delta\big(E - E(\boldsymbol{\sigma})\big)}_{\doteq n(E)} \\ &= \frac{1}{k_B T} \int_0^\infty dE e^{-E/k_B T} \underbrace{\sum_{\boldsymbol{\sigma}} \Theta\big(E - E(\boldsymbol{\sigma})\big)}_{\doteq \mathcal{N}(E)}. \end{aligned}$$

In the above, we integrate by parts, noting that the integral of the delta function $\delta(x)$ is the Heaviside step function $\Theta(x < 0) = 0$, $\Theta(x > 0) = 1$. We also define the density of states $n(E)$, which counts the number of states with energy E, as well as the cumulative density of states which counts the number of states with energy less than E. When working with finite data, it is more convenient to work with the cumulative count than the density of states, since the latter requires defining somewhat arbitrary energy bins so that we can count how many states have approximately a certain energy.

We can define the microcanonical Gibbs entropy in terms of the cumulative density of states as

$$S(E) \doteq \ln \mathcal{N}(E).$$

Looking at the logarithm of the cumulative count is in part motivated by the exponentially many states that arise from N variables (Tkačik, Mora, et al., 2015). It should be noted that there is also an alternative definition of the microcanonical entropy in terms of the density of states which is discussed in Franzosi, 2018. This conception of entropy asks, "As we change the energy of the system, how many states become available to us?" It is important to keep in mind that n and $\mathcal{N}$ are not smooth functions for finite N, and thus taking derivatives of the entropy $S = \ln \mathcal{N}$ is problematic unless we are in the 'thermodynamic limit' where N tends towards infinity. This is one of the things one has to be wary of when adapting these ideas to finite populations of neurons.

As the size of our system N increases, we expect the energy E and entropy S to increase proportionally, which defines them as *extensive* quantities. However, we may be interested in seeing what happens to the energy and entropy per unit (i.e. per unit in our system of size N), or what we call *intensive* quantities. We define the energy per unit as $\epsilon \doteq E/N$, and the entropy per unit as $s(\epsilon) \doteq S(N\epsilon)/N$. We can substitute these definitions, along with the definition of entropy into the partition function:

$$\begin{aligned} Z =& \frac{1}{k_B T} \int_0^\infty dE e^{-E/k_B T} \mathcal{N}(E) \\ =& \frac{1}{k_B T} \int_0^\infty dE \exp\Big(-\frac{1}{k_B T} E + S(E) \Big) \\ =& \frac{N}{k_B T} \int_0^\infty d\epsilon \exp\Big(-\frac{N}{k_B T} \epsilon + N s(\epsilon) \Big), \end{aligned}$$

where we made the change of variables $E \to \epsilon = E/N$ in the third line. Defining the free energy $f(\epsilon) \doteq \epsilon - k_B T s(\epsilon)$, we arrive at:

$$Z = \frac{N}{k_B T} \int_0^\infty d\epsilon \exp\Big(-\frac{N}{k_B T} f(\epsilon) \Big). \tag{3.6}$$

We expect this integral to be dominated by the energies close to the minimum ϵ^* of the free energy $f(\epsilon)$. At the minimum of the free energy, we have:

$$\frac{df(\epsilon)}{d\epsilon}\Big|_{\epsilon^*} = 0 \implies \frac{1}{k_B T} = \frac{ds(\epsilon)}{d\epsilon}\Big|_{\epsilon^*}, \tag{3.7}$$

where we use $f'(\epsilon)|_{\epsilon^*}$ to denote the derivative of $f(\epsilon)$ evaluated at ϵ^*. If we Taylor expand the free energy around ϵ^* up to the second order and plug this into the partition function, we obtain:

$$\begin{aligned} Z \approx& \frac{N}{k_B T} \int_0^\infty d\epsilon \exp\Big[-\frac{N}{k_B T} \Big(f(\epsilon^*) + \frac{1}{2} \frac{d^2 f(\epsilon)}{d\epsilon^2}\Big|_{\epsilon^*} (\epsilon - \epsilon^*)^2 \Big) \Big] \\ =& \frac{N}{k_B T} \exp\Big(\frac{-N}{k_B T} f(\epsilon *) \Big) \int_0^\infty d\epsilon \exp\Big(\frac{N}{2} \frac{d^2 s(\epsilon)}{d\epsilon^2}\Big|_{\epsilon^*} (\epsilon - \epsilon^*)^2 \Big). \end{aligned}$$

Note, the first derivative of $f(\epsilon)$ is 0 at ϵ^* by definition. In this expression for the partition function Z, we integrate over all values of the energy per unit of our system ϵ. We can interpret this expression as showing us how the energy per unit ϵ is distributed and it suggests that it follows a Gaussian distribution with mean $\langle\epsilon\rangle = \epsilon^*$, and variance:

$$\langle(\epsilon - \langle\epsilon\rangle)^2\rangle = -\frac{1}{N}\left[\frac{d^2 s(\epsilon)}{d\epsilon^2}\right]^{-1}_{\epsilon^*}. \tag{3.8}$$

For reference, the probability density function $f(x)$ of the Gaussian distribution with mean μ and standard deviation σ takes the form

$$f(x) \propto \exp\left(-\frac{1}{2}\Big(\frac{x-\mu}{\sigma}\Big)^2\right).$$

The notion of ϵ^*, the minimum of the free energy, being the mean of the distribution ties in to the earlier suggestion that the partition function will be dominated by ϵ around ϵ^*.

Finally, to introduce the heat capacity C, and its normalised counterpart, the specific heat C/N, we observe how the mean energy ϵ^* changes as we change the temperature T. Starting from Equation 3.7 and rearranging it in terms of T, we have:

$$\begin{aligned} T =& \frac{1}{k_B}\left[\frac{ds(\epsilon)}{d\epsilon}\right]^{-1}_{\epsilon*} \\ \frac{dT}{d\epsilon^*} =& -\frac{1}{k_B T}\left[\frac{ds(\epsilon)}{d\epsilon}\right]^{-2}_{\epsilon*}\left[\frac{d^2 s(\epsilon)}{d\epsilon^2}\right]_{\epsilon*}. \end{aligned}$$

Using the expression for T from the first line, we arrive at the following definition for heat capacity:

$$C(T) \doteq \frac{d\epsilon^*}{dT} = -\frac{1}{k_B T^2}\left[\frac{d^2 s(\epsilon)}{d\epsilon^2}\right]^{-1}_{\epsilon*}. \tag{3.9}$$

By making use of the equation for variance (3.8), we define the specific heat C/N as:

$$\frac{C}{N} = \frac{1}{N}\frac{d\epsilon^*}{dT} = \frac{1}{k_B T}\langle(\epsilon - \langle\epsilon\rangle)^2\rangle. \tag{3.10}$$

Notice how, when $d^2 s(\epsilon)/d\epsilon^2 = 0$, the variance (3.8), heat capacity (3.9) and specific heat (3.10) all become infinite. This is referred to as a critical point. One typically finds the critical point of a canonical system by varying the temperature T of the system and looking for a divergence in the heat capacity, and this critical point is indicative that our system undergoes a phase transition.

This should act as a fairly self-contained introduction to the Boltzmann distribution. What we now have to address is to which extent these definitions generalise to MaxEnt models for finite systems of neurons. When modelling finite systems, we are no

longer in the thermodynamic limit $\lim N \to \infty$. It would seem that the microcanonical Gibbs entropy, $S(E) = \ln \mathcal{N}(E)$, which takes the logarithm of the number of states with energy less than E, is no longer a smooth function when we deal with systems of finite size. Thus it would seem that taking derivatives of the entropy is no longer properly defined. It is possible, however, to work out the variance in the energy per unit $\langle(\epsilon - \langle\epsilon\rangle)^2\rangle$, thus we can still work out the specific heat as

$$\frac{C(T{=}1)}{N} = \langle(\epsilon - \langle\epsilon\rangle)^2\rangle.$$

We of course have to keep in mind that in MaxEnt models, although we can identify the argument of the exponential as the energy E, we do not have a temperature nor Boltzmann's constant k_B in the argument, which implies $k_B T = 1$, and we can only talk about the heat capacity and specific heat at $T = 1$. We could introduce a fictitious temperature T into the exponential, and then varying the temperature while keeping all other parameters constant. This would correspond to rescaling the model parameters by a scalar, though only the unscaled parameters would correspond to a model of the actual data, since the rescaled models would no longer reproduce the constraints. Rescaling the parameters is then a means of placing the actual model into the context of a broader family of exponential models, because we compare quantities calculated from our actual model to the equivalent quantities calculated from the models in the models with rescaled parameters.

If we did decide to introduce a temperature into our model and we monitored the heat capacity over a range of temperatures, we would no longer expect to see a divergence in the heat capacity as a signature of a critical point. Instead, a peak in the heat capacity is an indication of criticality in systems of finite sizes (Berry II and Tkačik, 2020). We dwell on what this might suggest in section 3.6. In summary, there is a mathematical equivalence between the Boltzmann distribution from statistical physics and MaxEnt models when $k_B T = 1$, though we have to be careful in using concepts that apply to systems in the thermodynamic limit to models of finite systems. We can still define the heat capacity and specific heat by examining the variance in the energy, or equivalently the log probability, though since our system does not have a temperature, we would have to introduce a fictitious temperature to examining these quantities at temperatures other than $T = 1$. Note though, that although the variance of the energy is equivalent to the variance of the log probability, the energy and log probability are not. Introducing a temperature T into the exponential and varying it can be seen as rescaling the model parameters and could be seen as a means of placing the actual model into the context of a family of exponential models.

3.4. Models used in the literature

We have introduced maximum entropy models and illustrated their link to the Boltzmann distribution from statistical mechanics. Depending on which observables $O(\boldsymbol{\sigma})$ we constrain, maximum entropy models can take different forms. Often the choice of model is dictated by which observables can be reliably estimated from the data. Thus, we often see constraints on the averages $\langle\sigma_i\rangle$, the pairwise correlations $\langle\sigma_i\sigma_j\rangle$ and on the probability of observing K neurons firing. We start by introducing the full log linear model, which can reproduce arbitrarily high-order correlations, but assumes that we can reliably measure arbitrarily high-order correlations from experimental data which is unfeasible in practice. From here, we introduce the independent, pairwise, population count and K-pairwise models which can be seen as special cases of the full log linear model. For quick reference, we briefly summarise the models we will look at in Table 3.1.

Model	Constraints	Parameters
Independent, $p^{(1)}$ Schneidman, Berry, et al., 2006	$\langle\sigma_i\rangle$	N
Population count, $p^{(K)}$ Tkačik, Olivier Marre, Mora, et al., 2013	$p(K)$	$N+1$
Pairwise, $p^{(2)}$ Schneidman, Berry, et al., 2006	$\langle\sigma_i\rangle$, $\langle\sigma_i\sigma_j\rangle$	$N(N+1)/2$
K-pairwise, $p^{(2,K)}$ Tkačik, Olivier Marre, Amodei, et al., 2014	$\langle\sigma_i\rangle$, $\langle\sigma_i\sigma_j\rangle$, $p(K)$	$(N+1)(N+2)/2$

Table 3.1: Summary of different MaxEnt models, along with seminal references for their use in modeling RGCs.

3.4.1. Full log linear model $p^{(N)}$

The full log linear model, which can represent any probability distribution over binary random variables (Gardella, Olivier Marre, and Mora, 2019), takes the form

$$p^{(N)}(\boldsymbol{\sigma}) = Z^{-1}e^{-E(\boldsymbol{\sigma})}, \tag{3.11}$$

$$E(\boldsymbol{\sigma}) = \sum_{i_1}\lambda_{i_1}\sigma_{i_1} + \sum_{i_1<i_2}\lambda_{i_1 i_2}\sigma_{i_1}\sigma_{i_2} + ... + \lambda_{12...N}\sigma_1\sigma_2...\sigma_N, \tag{3.12}$$

where the parameters $\boldsymbol{\lambda}$ can be identified as the Lagrange multipliers, though we will frequently refer to them as the *interaction* parameters. The order of the parameter $\lambda_{i_1...i_M}$ is denoted by the number of subscripts. For instance, $\lambda_{i_1...i_M}$ denotes an M^{th}-order interaction parameter. The superscript in brackets $p^{(X)}$ relates to the expectations

that the model reproduces. For instance, the pairwise model, which we define in Section 3.4.3, reproduces pairwise correlations $\langle \sigma_i \sigma_j \rangle$ and hence is denoted $p^{(2)}$. The full log linear model, which can reproduce up to N^{th}-order correlations $\langle \sigma_1 ... \sigma_N \rangle$ in a system of N neurons is denoted $p^{(N)}$. Given, somewhat unrealistically, estimates of the probabilities of each of the 2^N states, we can determine the 2^N parameters of $p^{(N)}$ by substituting each state into Equation 3.12 and equating it to its negative log probability. This gives us a system of 2^N linear equations from which we can then solve for the parameters $\boldsymbol{\lambda}$ (Martignon et al., 1995). The issue is that we can seldom reliably estimate the full probability distribution from experimental data.

Before we go on, we should clarify that there is a difference between the *interaction* parameters $\lambda_{i_1 i_2 .. i_M}$ and the *correlations* $\langle \sigma_{i_1} \sigma_{i_2} ... \sigma_{i_M} \rangle$. The most trivial difference is that interaction parameters parameterise our model whereas correlations are obtained as expectations from the model. There is often not an obvious mapping between interaction parameters and correlations, though we will elaborate on how to interpret the interaction parameters in the following paragraph. However, a useful example at this stage is that we can calculate third-order correlations from models that include only up to second-order interactions. Where third-order interaction parameters come into play is that there are certain third-order correlations which can arise that cannot be produced by models that include only up to second-order interactions, regardless of how you choose the second-order interaction parameters. Also, it should also be emphasized that interactions are not interpretable as physical connections between neurons (Nguyen, Zecchina, and Berg, 2017).

The full log-linear model has been used in a number of early works (Martignon et al., 1995; Schneidman, Still, et al., 2003; Yu et al., 2011 and others), and despite the experimental impracticality of determining all the coefficients from empirical data of large populations, this model provides a theoretical means of exploring the correlation structure of populations of neurons. The full log linear model represents a decomposition of the correlations between N neurons where the parameter $\lambda_{i_1 ... i_M}$ relates to a correction to the correlation $\langle \sigma_{i_1} ... \sigma_{i_M} \rangle$ that cannot be deduced from knowledge of the marginal distributions over $L < M$ neurons, or equivalently, knowledge of the L^{th}-order correlations (Nakahara and Amari, 2002; Amari et al., 2003). To try phrase it more succinctly, a non-zero interaction parameter of M^{th}-order $\lambda_{i_1 ... i_M}$ provides a correction to the produced M^{th}-order correlation that that can only be deduced from knowing the states of the M neurons $\sigma_{i_1}, ..., \sigma_{i_M}$. To unpack this statement more rigorously requires a dive into information geometry, which we leave up to the interested reader – see Nakahara and Amari, 2002 specifically.

However, as intuition, consider a system of two neurons firing, where their states are represented by σ_1, σ_2. The log linear description of this system would be

$$p^{(2)}(\sigma_1, \sigma_2) = \frac{1}{Z} \exp\left(-\lambda_1 \sigma_1 - \lambda_2 \sigma_2 - \lambda_{12} \sigma_1 \sigma_2\right). \tag{3.13}$$

Now, the marginal distributions $p(\sigma_1)$ and $p(\sigma_2)$ tell us the probability of neuron 1 and neuron 2, respectively, either firing or remaining silent. Without any further knowledge we might assume that these neurons fire independently (and indeed this would be the assumption that maximises the entropy) which would then lead to us modelling their joint probability distribution as

$$p^{(1)}(\sigma_1, \sigma_2) = \frac{1}{Z} \exp\left(-\lambda_1' \sigma_1 - \lambda_2' \sigma_2\right) = \frac{1}{Z} \exp\left(-\lambda_1' \sigma_1\right) \exp\left(-\lambda_2' \sigma_2\right),$$

which is identical to Equation 3.13 when $\lambda_{12} = 0$. This model would predict the correlation between σ_1 and σ_2 to be $\langle \sigma_1 \sigma_2 \rangle = \langle \sigma_1 \rangle \langle \sigma_2 \rangle = p(\sigma_1{=}1) \cdot p(\sigma_2{=}1)$. Clearly this model does not account for a variety of scenarios, starting with neuron 1 and neuron 2 never firing together and ending with them always firing together. The only scenario that this model does account for is the one in which the neurons fire together with probability $p(\sigma_1{=}1) \cdot p(\sigma_2{=}1)$. The interaction parameter λ_{12} in Equation 3.13 thus provides the correction that would allow us to model all these other scenarios ranging from them never firing together, which would be modelled by $\lambda_{12} = -\lambda_1 - \lambda_2$, to them always firing together, which would be modelled by $\lambda_1, \lambda_2 = 0$.

Returning to the full log linear model, by varying the order of the parameters that we include in the model, we can define a hierarchy over the maximum entropy models where the model that includes all parameters up to order M will be able to reproduce up to M^{th}-order correlations (Schneidman, Still, et al., 2003). Although correlations of all orders in a distribution over N variables can be trivially modelled by a model with N^{th}-order interactions, it is interesting when we find that they can still be modelled with only M^{th}-order ($M < N$) interactions, which suggests a simpler description of the distribution. More precisely, it suffices to know the marginal distributions over M neurons to characterise the full distribution over N neurons.

3.4.2. Independent model $p^{(1)}$

If we take the full log linear model and keep only the first-order coefficients, we arrive at the independent model:

$$p^{(1)}(\boldsymbol{\sigma}) = \frac{1}{Z} \exp\left(-\sum_i h_i \sigma_i\right) = \frac{\exp\left(-\sum_i h_i \sigma_i\right)}{\prod_i \left(1 + e^{-h_i}\right)}.$$

Our change of notation from $\boldsymbol{\lambda}$ as the model parameters to $\boldsymbol{h}$ is inspired by the Ising model, which we discuss in Section 3.4.3. The independent model corresponds to the maximum entropy model that reproduces the averages $\langle\sigma_i\rangle$. Given the empirical averages $\langle\sigma_i\rangle_D$, we can analytically solve for h_i, to obtain

$$h_i = -\ln\left[\frac{\langle\sigma_i\rangle_D}{1-\langle\sigma_i\rangle_D}\right].$$

Intuitively, the probability of viewing state $\boldsymbol{\sigma}$ is just the product of the probabilities of observing each neuron σ_i fire or remain silent, $p(\boldsymbol{\sigma}) = \prod_i p(\sigma_i)$. Since we use the convention $\sigma_i \in \{0,1\}$, the probability of neuron i firing is just its expected value $p(\sigma_i{=}1) = \langle\sigma_i\rangle_D$ and similarly, $p(\sigma_i{=}0) = 1 - \langle\sigma_i\rangle_D$.
The independent model is used as a baseline in a number of works (Schneidman, Berry, et al., 2006; Köster et al., 2014, etc.) and as a point of departure, we often observe that this model fails to accurately model features of the data, suggesting the importance of higher-order interactions in modelling retinal activity.

3.4.3. Pairwise model $p^{(2)}$

We now consider including the second-order coefficients. This corresponds to the maximum entropy model that reproduces the averages $\langle\sigma_i\rangle$ and pairwise correlations $\langle\sigma_i\sigma_j\rangle$ and can be thought of as a generalisation of the Ising model from statistical mechanics. It takes the form:

$$p^{(2)}(\boldsymbol{\sigma}) = \frac{1}{Z}\exp\left(-\sum_i h_i\sigma_i - \sum_{i<j} J_{ij}\sigma_i\sigma_j\right).$$

In the Ising model, which models the probability of seeing different states in a set of spins, as opposed to neurons, the parameter h_i is thought of as a local magnetic field acting on spin i, and the pairwise coupling J_{ij} quantifies the interaction between spin i and spin j. When we represent binary variables as $\sigma \in \{0,1\}$, since $\sigma_i\sigma_i = \sigma_i$, we can alternatively write h_i as J_{ii} and represent this model as $p^{(2)}(\boldsymbol{\sigma}) \propto \exp(-\sum_{i\leq j} J_{ij}\sigma_i\sigma_j)$.

Unlike the independent model, it is not simple to find expressions for the parameters $\boldsymbol{h}, \boldsymbol{J}$ in terms of the averages and pairwise correlations. Instead, we have to either use approximate solutions for the model parameters, or use optimization algorithms such as gradient ascent (see Section 3.2.1). We can think of the parameter h_i as representing neuron i's inherent bias towards firing or silence and the parameter J_{ij} as the direct mutual interaction between neurons i and j that remains in their correlated activity after we have accounted for other interactions that arise through more circuitous paths (Tkačik, Schneidman, et al., 2009).

Determining the model parameters $\boldsymbol{h}, \boldsymbol{J}$ that maximise the Lagrangian for the pairwise model, or equivalently, that maximise the log-likelihood, is a concave optimisation problem with a unique solution since our constraints obey the condition in the section on the uniqueness of MaxEnt models. This implies that regardless of how we initialise the model parameters, for a given set of constraints gradient ascent will yield the same model parameters. This also implies that the energy landscape, defined by the Hamiltonian of the pairwise model, will be the same for all pairwise models fitted to reproduce a given set of constraints, and thus we will find the same energy minima in the landscape.

Though we could theoretically continue including higher-order interaction parameters to our model, it is increasingly unlikely that our estimates of all higher-order correlations, which we use to fit the higher-order interaction parameters, are accurate, especially when the number of neurons N is very large. Thus, our next move will be away from including detailed descriptions of all correlations and towards a description of the activity of the population.

3.4.4. Population count model $p^{(K)}$

In the population count model $p^{(K)}$, we only constrain the model to reproduce the probability of observing K neurons firing. This model predicts all states with the same number of neurons firing as equally likely, and thus provides only a coarse description of the neural activity. It takes the form

$$p^{(K)}(\boldsymbol{\sigma}) = \exp\left(\ln p(K(\boldsymbol{\sigma}))_D - \ln \binom{N}{K(\boldsymbol{\sigma})}\right) = \frac{p(K(\boldsymbol{\sigma}))_D}{\binom{N}{K(\boldsymbol{\sigma})}},$$

where $K(\boldsymbol{\sigma})$ returns the number of neurons that fire in state $\boldsymbol{\sigma}$. Here, $p(K(\boldsymbol{\sigma}))_D$ is the population count distribution measured in the data. Similarly to the independent model, the population count model can be written explicitly in terms of its constraints $p(K(\boldsymbol{\sigma}))_D$. Though this model does not provide a detailed description of the probability of different states, it has been used to explore the hypothesis of criticality in a population of retinal ganglion cells (Tkačik, Olivier Marre, Mora, et al., 2013).

3.4.5. K-pairwise model $p^{(2,K)}$

If we take the pairwise model and additionally constrain it to reproduce the population count distribution $p(K)$, we obtain the K-pairwise model which takes the form

$$p^{(2,K)}(\boldsymbol{\sigma}) = \frac{1}{Z} \exp\left(-\sum_i h_i \sigma_i - \sum_{i<j} J_{ij}\sigma_i\sigma_j - V_{K(\boldsymbol{\sigma})}\right).$$

We derive this model using the method of Lagrange multipliers in the Appendix A.1.3. Here $V_{K(\boldsymbol{\sigma})}$ is referred to as the effective potential, and is a function of the number of neurons that fire in state $\boldsymbol{\sigma}$. The effective potential can be thought of as arising from placing constraints on all of the moments of the distribution $p(K)_D$, since knowing the distribution is equivalent to knowing all of its moments (Tkačik, Olivier Marre, Mora, et al., 2013).

As with the pairwise model, finding expressions for the parameters $\boldsymbol{h}, \boldsymbol{J}, \boldsymbol{V}$ in terms of the constraints is difficult and the parameters are typically determined by using optimisation algorithms such as gradient ascent. Unlike the pairwise model which has a unique set of parameters for a given set of constraints, there are multiple sets of parameters $\{\boldsymbol{h}, \boldsymbol{J}, \boldsymbol{V}\}$ for a given set of constraints which specify mathematically identical models. This is because adding a constant to all of the parameters $\boldsymbol{h}$ adds a term linear in K to $\boldsymbol{V}$, and adding a constant to all of the parameters $\boldsymbol{J}$ adds a quadratic term to $\boldsymbol{V}$ (Tkačik, Olivier Marre, Amodei, et al., 2014). If one wanted to compare the parameters of different K-pairwise models directly, it is possible to extract all the components from $\boldsymbol{V}$ that can be equivalently parameterised as offsets to $\boldsymbol{h}$ and $\boldsymbol{J}$ such that $\boldsymbol{V}$ only constrains correlations that cannot be accounted for by $\boldsymbol{h}$ and $\boldsymbol{J}$. For instance, when going from a pairwise model to the K-pairwise model we might want to quantify how much of an effect including an effective potential in the exponential has (Tkačik, Olivier Marre, Amodei, et al., 2014).

3.4.6. Other models

Though we focus on the above models in this work, other MaxEnt models exist, as well as other approaches to modelling neural activity. We briefly go over some of these below.

We start by going over a variation on the independent and population count model. This model improves the performance of the independent model by including knowledge of how many neurons are firing in total, while remaining tractable even for large populations. The so-called *population coupling* model, which reproduces the joint probability distribution of each neuron and the population count K, takes the form

$$p(\boldsymbol{\sigma}) = \frac{1}{Z} \exp\left(\sum_i h_{i,K(\boldsymbol{\sigma})}\sigma_i\right),$$

where the parameters $h_{i,K(\boldsymbol{\sigma})}$ are inferred so that the distribution agrees with the data $p(\sigma_i, K)$ for each (i, K) pair. This model has been used to explore how population

activity affects the firings of individual neurons and finds that there is a nonlinear dependency between the population activity and the probability of individual neurons firing (Gardella, Olivier Marre, and Mora, 2016). A simpler model, which only reproduces the expectations $\langle K(\boldsymbol{\sigma})\sigma_i\rangle$, is also worth consideration (Gardella, Olivier Marre, and Mora, 2016).

As we will see in Section 3.5 as well as in our results, both the vanilla independent and pairwise models fall short in predicting aspects of the population count distribution $p(K)$. The population coupling model attempts to improve the independent model through including knowledge of the population count distribution, which is a similar to what we did when we included the effective potential in the pairwise model, giving rise to the K-pairwise model. An approach which attempts to generalise the notion of adding population-level statistics as constraints to models comes in the form of the *semi-parametric energy based* model (Humplik and Tkačik, 2017). This model takes the form

$$p(\boldsymbol{\sigma}) = \frac{1}{Z}\exp\left(-V(E(\boldsymbol{\sigma}))\right),$$

where V is an arbitrary, increasing, differentiable function referred as the "nonlinearity" which is determined nonparametrically from the data, and $E(\boldsymbol{\sigma})$ is an energy function as before which is parameterised and reflects local interactions among neurons. From this class of models, we can define the semi-parametric independent and pairwise models, which have energy functions $E(\boldsymbol{\sigma})$ as defined in Sections 3.4.2 and 3.4.2. In a population of 160 retinal cells, the semi-parametric pairwise model has been shown to be a more accurate model than both the pairwise and K-pairwise model, though in including the nonlinearity, these models are no longer interpretable as maximum entropy models consistent with low-order correlations (Humplik and Tkačik, 2017).

We now move on to a model that includes arbitrarily high-order interaction parameters. If we think back to the full log-linear model (3.4.1), it is typically unfeasible to determine all of its 2^N parameters from experimental data for large N, particularly since finite data-sets rarely include states with large numbers of neurons firing together, making our estimates of higher-order correlations likely to be inaccurate. However, this does not necessarily imply that there will be no higher-order correlations that can be well estimated from finite recordings. Whereas our approach up to now has been to focus on fitting models to all the lower-order correlations, the *reliable interaction* model proposes to include any interaction parameters $\boldsymbol{\lambda}$ that are necessary to fit the most frequent states in the data, and makes no assumption about the maximum order of the interactions. Thus, the reliable interaction model may include select higher-order interactions and does not require us to learn all of the low-order interactions (Ganmor,

Segev, and Schneidman, 2011).

The probability mass function for the reliable interaction takes the same form as the full log linear model's (Equation 3.12) though only including the interaction parameters necessary to reproduce the probability of observing the most frequent states. When fitting the reliable interaction model to activity from a population of 100 retinal ganglion cells, the authors found that the model required fewer parameters (≈ 450) in comparison to the pairwise model (≈ 5000) and was able to more accurately model the data based on the log-likelihood ratio between the model and the empirical data. The data that they used came from recording the cells' responses to stimuli that involved recordings of natural scenes. They also fitted the models to responses to white noise, where the pairwise and reliable interaction models performed similarly. This suggests that higher-order interactions are at least partially driven by the higher-order statistics in natural scenes (Ganmor, Segev, and Schneidman, 2011).

In the cascading logistic model, we aim to make the problem of fitting and evaluating the probability distribution more tractable by assuming that the joint probability distribution factors with a cascaded structure,

$$p(\boldsymbol{\sigma}) = p(\sigma_1)p(\sigma_2|\sigma_1)...p(\sigma_N|\sigma_1, ..., \sigma_{N-1}).$$

This model takes the form

$$p(\boldsymbol{\sigma}) = p(\sigma_1)\prod_{i>1} P(\sigma_i|\sigma_1...\sigma_{i-1}),$$

where

$$p(\sigma_i|\sigma_1...\sigma_{i-1}) = \frac{1}{Z}\left(1 + \exp\left(h_i + \sum_{j<i} w_{ij}\sigma_j\right)\right)^{-1}.$$

With this model the partition function is tractable to evaluate and each conditional can be fit independently with logistic regression. However, unlike the pairwise model this model is sensitive to the order of neurons. In order for the probability distribution produced by this model to be close to that of the pairwise model, we require sparsity in the pairwise interaction parameters (Park et al., 2013).

Another approach to modelling higher-order interactions is to introduce hidden or latent variables $\boldsymbol{z}$ which interact with the visible neurons. A simple model that makes use of latent variables is the Restricted Boltzmann Machine (RBM) (Smolensky, 1986). In an RBM, a set of visible units $\boldsymbol{\sigma}$ are connected to a set hidden units $\boldsymbol{z}$. Each of these sets have local fields $\boldsymbol{a}$ and $\boldsymbol{b}$, respectively, and the strength of the connections between

the visible and hidden variables is quantified by the weight matrix $\boldsymbol{W}$. However, there are no direct connections between the visible variables, nor between the hidden variables, hence the term "restricted". The probability distribution over visible and hidden variables is given by

$$p(\boldsymbol{\sigma}, \boldsymbol{z}) = \frac{1}{Z} \exp\left(-\sum_i a_i \sigma_i - \sum_i b_i z_i - \sum_{i,j} \sigma_i W_{ij} z_j \right).$$

The probability of a configuration is obtained by marginalising over the latent variables which can be computed analytically up to a normalisation constant (Torlai and Melko, 2016). Sticking with our convention of $\sigma, z \in \{0, 1\}$, the probability distribution over the visible variables is

$$\begin{aligned} p(\boldsymbol{\sigma}) =& Z^{-1} e^{-E(\boldsymbol{\sigma})} \\ E(\boldsymbol{\sigma}) =& \sum_i a_i \sigma_i - \sum_j \log\left(1 + \exp(-b_j - \sum_i \sigma_i W_{ij})\right), \end{aligned}$$

where $E(\boldsymbol{v})$ is the effective visible energy. RBMs have been shown to outperform many of the other models mentioned above (Köster et al., 2014; Humplik and Tkačik, 2016; Gardella, Olivier Marre, and Mora, 2017). However, they risk over-fitting and are not easily interpretable. Interestingly, any probability distribution over N binary variables can be approximated with arbitrary precision by an RBM with sufficiently many hidden variables (Le Roux and Bengio, 2008).

Finally, we introduce the *dichotomized Gaussian* distribution (Amari et al., 2003; Yu et al., 2011; Macke, Opper, and Bethge, 2011), which models the activity of neurons as being driven by Gaussian latent variables $\boldsymbol{u} \sim \mathcal{N}(\boldsymbol{\gamma}, \boldsymbol{\Lambda})$ of dimension N where neuron i fires if u_i is positive, and is otherwise silent:

$$\sigma_i = \begin{cases} 1, & u_i > 0 \\ 0, & u_i \leq 0 \end{cases}.$$

The thresholding operation, letting neuron i fire only if its latent input is positive, changes the moments, so the distribution over $\boldsymbol{\sigma}$ will not, in general, have the same moments as $\boldsymbol{u}$ (Macke, Berens, et al., 2009).

We have introduced the prominent MaxEnt models used to model the activity of neural data. We started by introducing the full log-linear model $p^{(N)}$, and then introduced the independent $p^{(1)}$, pairwise $p^{(2)}$, population count $p^{(K)}$ and K-pairwise $p^{(2,K)}$ models as special cases of the full log-linear model. The independent and population count

models are simple models and we can derive analytic expressions for their parameters in terms of their constraints. On the other hand, the pairwise and K-pairwise models incorporate more detailed interactions, but it is difficult to express their parameters in terms of their constraints. Instead, we typically use a numerical optimisation algorithm, and for large N, we typically have to turn Monte-Carlo methods to compute the expectations required in these optimisation algorithms.

We then introduced a number of variations on and alternatives to these models. To make these MaxEnt models more expressive, we saw the inclusion of additional constraints (population coupling and reliable interaction model), the introduction of a non-linear function in the exponential (semi-parametric energy-based model), and the introduction of latent variables (RBM). Another consideration is how easily we can fit these models to large populations, and here we saw some authors sticking to either relatively simple models (population count and population coupling model) and others making simplifying assumptions such as that the joint probability distribution can be factorised into a cascading structure (cascading logistic model).
We have introduced a number of ways that we might model the activity of neurons. Next, we have to ask how well these models work.

3.5. Assessing goodness of fit

Given the early promise of pairwise results, a lot of effort has been put into fitting pairwise models to increasingly large populations (a more accurate title of this Section would be assessing the goodness of fit of largely pairwise models). Pairwise models were for a long time considered to be excellent approximations of the activity of populations ~ 40 RGCs of all functional types (Berry II and Tkačik, 2020; Schneidman, Still, et al., 2003; Tkačik, Schneidman, et al., 2006). However, for larger populations of ~ 100 RGCs, higher-order interactions become more pronounced and pairwise models cease to accurately reproduce important features of the true distribution. For instance, if we compare the population count distribution $p(K)$ predicted by the pairwise model to the empirically observed distribution, the pairwise model predicts significantly heavier tails than what is observed (Tkačik, Olivier Marre, Amodei, et al., 2014). So far, we have not made the distinction of which type of RGCs we model, and this could be a distinction worth making. Shlens et al., 2006 modelled the activity of ~ 100 RGCs of the same functional type and found that the pairwise model remained a good approximation. Restricting ourselves to modelling the activity of just a particular functional type is not something that we are able to do given then experimental data we have access

to, but it remains an important consideration. In order to model larger populations of RGCs without focusing on particular the functional type, additional constraints such as the population count distribution $p(K)$ have been included in the model, and these models yet again appear to be a good fit based on the metrics such as the closeness of the produced third-order correlations to the actual ones (Ganmor, Segev, and Schneidman, 2011; Tkačik, Olivier Marre, Amodei, et al., 2014). This brief synopsis of the successes and shortcomings of the pairwise model raises general questions about the goodness of fit of MaxEnt models. Pertinently, why do certain models become worse when we move to larger populations?

Determining whether a particular MaxEnt model is a good approximation to the true distribution of the neural activity $p(\boldsymbol{\sigma})$ under the assumption of temporal independence is challenging since we can only experimentally sample a fraction of the full distribution. If we had full knowledge of the true distribution, we could use a measure such as the Kullback–Leibler divergence to quantify how close our model is to the true distribution. However, for a population of 100 neurons, there are approximately $2^{100} \sim 10^{30}$ possible states while the experimental data that we look at includes only 283041 samples, many of which are repeated samples (Tkačik, Olivier Marre, Amodei, et al., 2014). Thus, assessing the fit of MaxEnt models to experimental data is often done heuristically, for instance by looking at observables that can be reliably measured from the data but that are not included in our MaxEnt models as constraints. For instance, we could assess whether the independent model might be a good approximation by comparing its predicted pairwise correlations $\langle \sigma_i \sigma_j \rangle = \langle \sigma_i \rangle \langle \sigma_j \rangle$ to the experimentally measured pairwise correlations.

However, these heuristic measures of goodness of fit, where we ask whether our model succeeds in capturing an aspect of the true distribution that can be well-sampled from finite data, may be misleading since our model might very well reproduce this particular aspect, yet it could still fail to reproduce other aspects that we do not consider, or are unable to estimate from finite samples. It is a lot easier to say that a particular model does a bad job of modelling the activity when it fails to capture certain aspects of observed activity. There is also the question of whether a model that does a good job of modelling the activity of a small population of neurons will remain a good model for larger populations, as in the case of the pairwise model which, although a good approximation for small populations of neurons, becomes a worse approximation for larger populations (Tkačik, Olivier Marre, Amodei, et al., 2014). This question of extrapolating performance has been answered in part in the case of the pairwise model whose good performance in modelling small populations of neurons does not imply

good performance for larger populations (Roudi, Nirenberg, and P. E. Latham, 2009). We also need keep the cautionary remarks from Section 2.2.2 in mind, which arise from modelling discretised neural data under the assumption of temporal independence. In this section, we address which metrics suggest that MaxEnt models are good approximations, and where we might be skeptical about the generality of these results.

3.5.1. Proportion of multi-information captured

One way of determining whether a MaxEnt model provides an effective description of experimental data is by looking at whether the reduction in entropy from including certain constraints in a MaxEnt model accounts for the difference in entropy between the independent model and the data distribution (Schneidman, Berry, et al., 2006). Looking back at the log-linear form introduced in Section 3.4.1, by varying the order $M = 1, 2, ..., N$ of the interaction parameters that we include in our model we can define a hierarchy over the MaxEnt models. For each model, we can compute its information theoretic entropy:

$$S_M = -\sum_{\boldsymbol{\sigma}} p^{(M)}(\boldsymbol{\sigma}) \ln p^{(M)}(\boldsymbol{\sigma}).$$

As we consider models with increasingly higher order correlations as constraints, the corresponding entropy decreases monotonically, $S_0 \geq S_1 \geq ... \geq S_N = S_D$, starting from an unconstrained model, which predicts each state as equally likely which will have the maximum entropy S_0, and ending with the model which includes arbitrarily high-order interactions and is an exact description of the data which will have entropy equal to the data distribution $S_N = S_D$. Adding additional constraints to the model can only decrease the entropy, since it gives us fewer and fewer directions in which we can move to maximise the entropy (refer back to the original formulation in terms of Lagrange multipliers and the proof of concavity in Section 3.1.2). We can quantify the reduction in entropy caused by including M^{th} order correlations as constraints as $I_M = S_1 - S_M$. The difference in entropy between the independent model and the data distribution $I_N = S_1 - S_N$ can be thought of as a measure of the total amount of correlation in the data, which we call the *multi-information*. By looking at the ratio I_M/I_N, we can then quantify the *proportion of the multi-information* that is captured by including M^{th}-order correlations in our model:

$$\textit{Proportion of the multi-information captured by } p^{(M)} = \frac{I_M}{I_N} = \frac{S_1 - S_M}{S_1 - S_N}.$$

Though there is not normally an obvious relationship between entropy and correlations, in the case of MaxEnt models, which reproduce up to M^{th}-order correlations,

we can interpret this relative difference in entropy as relating to un-modelled correlations that would still further constrain the optimisation problem and thus reduce the entropy.

The proportion of the multi-information captured by pairwise models was measured in a number of different sub-populations of 10 RGCs, where it is still computationally feasible to fit pairwise models without sampling, and was found to be around $I_2/I_N \approx 0.9$. These calculations were repeated with models trained on data recorded from a number of different settings, including data from both salamander and guinea pig RGCs, and from cultured cortical neurons, and similar proportions were observed, though in all of these settings the pairwise models were only fitted to sub-populations of 10 cells (Schneidman, Berry, et al., 2006). Results like these helped fuel the early enthusiasm around using pairwise models for modelling neural data. Estimating entropy from MaxEnt models has to be done with caution as our entropy estimates can be significantly different from the entropy of the true distribution, especially when the true distribution does not come from the same distribution as our specified model, which happens when higher-order correlations in the true distribution cannot be captured by our model (Macke, Murray, and P. Latham, 2011). Although this decomposition of the entropy appears in the early literature (Schneidman, Still, et al., 2003), such calculations rarely appear in the more recent literature. Instead, unconstrained observables are often used to assess goodness of fit, which we discuss in the next section.

3.5.2. Unconstrained observables

As we try and model larger populations of neurons, the number of states grows exponentially and our sample statistics become increasingly unreliable. While it may be possible to estimate the full probability distribution from experimental data for very small populations of neurons, we are no longer able to do this reliably for populations of $100+$ neurons. Thus, we often have to turn to heuristics in order to assess goodness of fit. For MaxEnt models, one such such heuristic is how well they predict observables that the models are not constrained to reproduce. We introduce some of the common observables that have been used to assess MaxEnt models, and then summarise where different MaxEnt models succeed or fail to reproduce these observables.

The first heuristic we identify is the population count distribution $p(K)$, where K is the number of neurons that fire within a time bin. As we will see, this heuristic is useful in assessing the performance of the independent and pairwise model. As $p(K)$ is also used as a constraint in MaxEnt models, such as the K-pairwise model, we need to

consider alternative heuristics for goodness of fit. One such heuristic is the correlation between triplets of neurons $\langle\sigma_i\sigma_j\sigma_k\rangle$, or $\langle(\sigma_i - \langle\sigma_i\rangle)(\sigma_j - \langle\sigma_j\rangle)(\sigma_i - \langle\sigma_k\rangle)\rangle$ depending on whether we are interested in fluctuations relative to the means. Finally, more recent work also uses MaxEnt models to try predict the probability of neuron N firing given the states of the other $N-1$ neurons, $p(\sigma_N|\sigma_1, ..., \sigma_{N-1})$. This conditional distribution is not used in assessing goodness of fit as much as it is used in trying to support hypotheses of a robust neural code (Tkačik, Olivier Marre, Amodei, et al., 2014), which we discuss further in Section 3.6.2. Based on these heuristics, let us now turn to look at the performance of different MaxEnt models.

Early work that focused on subsets of 10 cells from a population of 40 RGCs, found that the majority of pairwise correlations $\langle\sigma_i\sigma_j\rangle$ are weak and very close to the product of the expectations of the individual cells $\langle\sigma_i\rangle\langle\sigma_j\rangle$ (Schneidman, Berry, et al., 2006). Though this suggests using the independent model, this model falls short in predicting the population count distribution $p(K)$. Specifically, the probability $p^{(1)}(K{=}10)$ predicted by the independent model was $10^3\times$ smaller than what was observed in the data. Generally, if N independent binary variables fire with probabilities $p_1, ..., p_N$, then we expect $p^{(1)}(K)$ to follow a *Poisson binomial distribution* with probability mass function,

$$p^{(1)}(K) = \sum_{A\in F_K}\prod_{i\in A} p_i \prod_{j\in A^{\mathsf{c}}}(1-p_j),$$

where F_K is the set of all subsets of $\{1, ..., N\}$ of size K, $A^{\mathsf{c}} = \{1, ..., N\} \setminus A$. However, Schneidman et al. noted that the observed distribution $p^{(D)}(K)$ instead resembled an exponential distribution.

As this early work only considered a relatively small population of 10 cells, the authors were able to compare the predicted and observed frequencies of different states. Again, the shortcomings of the independent model became apparent as it failed to accurately predict the frequency of common states. For instance, the state (1011001010) which occurred once per minute in the recording was predicted by the independent model to occur only once per three years (Schneidman, Berry, et al., 2006).

Later works which consider larger populations of neurons re-iterate the shortcomings of the independent approximation. For both populations of $N = 40$ and $N = 100$ cells, we observe a big difference between the observed population count distribution $p^{(D)}(K)$ and the Poisson binomial distribution predicted by the independent model, especially at large K (Tkačik, Schneidman, et al., 2009; Tkačik, Olivier Marre, Amodei, et al., 2014). In 40 RGCs, approximately a third of the correlations between triplets of

cells $\langle(\sigma_i - \langle\sigma_i\rangle)(\sigma_j - \langle\sigma_j\rangle)(\sigma_i - \langle\sigma_k\rangle)\rangle_D$ were observed to be significantly different from zero, whereas an independent approximation would predict these correlations to be identically zero (Tkačik, Schneidman, et al., 2009).

Can the observed population count distribution $p^{(D)}(K)$ and the triplet correlations be explained by including pairwise interactions? Initially, when pairwise models were fitted to small populations of 10 cells, the pairwise model appeared to be a good approximation based on its accuracy in predicting the frequency of common states, the proportion of the multi-information that it captures, and the *Jensen-Shannon* divergence D_{JS} (Schneidman, Berry, et al., 2006), where the Jensen-Shannon divergence is a symmetrised version of the Kullback-Leibler divergence D_{KL}:

$$D_{JS}(p\|q) \doteq \frac{1}{2}\left(D_{KL}(p\|q) + D_{KL}(q\|p)\right), \quad D_{KL}(p\|q) \doteq \sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}) \log \frac{p(\boldsymbol{\sigma})}{q(\boldsymbol{\sigma})}.$$

However, from $N = 40$, we begin to see significant differences between the pairwise and data distribution, though these differences are not as extreme as with the independent model.

At $N = 40$ cells, we already see deviations between the predicted and observed population count distribution $p(K)$ (Tkačik, Schneidman, et al., 2006). The pairwise models predicts a distribution with noticeably heavier tails for $p(K)$ than what is observed, and it also underestimates the probability of silence as $p^{(2)}(K{=}0) = 0.502$, whereas it is measured to be $p^{(D)}(K{=}0) = 0.55$. The pairwise approximation becomes worse at $N = 100$ where its prediction for $p^{(2)}(K{=}0)$ is off by a factor of three and it continues to exhibit much heavier tails at large K than what is empirically observed (Tkačik, Olivier Marre, Amodei, et al., 2014). The silent state $(00...0)$ is a well-sampled feature even in the activity of large populations of RGCs, and the pairwise model is unable to accurately capture it.

The pairwise model also overestimates triplet correlations by 7% on average in a population of 40 cells (Tkačik, Schneidman, et al., 2009). From 40 through until 100 cells, the absolute difference between the observed and predicted triplet correlations $\langle(\sigma_i - \langle\sigma_i\rangle)(\sigma_j - \langle\sigma_j\rangle)(\sigma_i - \langle\sigma_k\rangle)\rangle$ is around 10^{-3}. Though the pairwise model represents a significant improvement to the independent model, there are still significant differences between its predictions and the experimentally observed activity of large populations of RGCs (Tkačik, Olivier Marre, Amodei, et al., 2014).

Some of these difference have been overcome by using the K-pairwise model which additionally includes the population count distribution $p(K)$ as a constraint (Tkačik,

Olivier Marre, Amodei, et al., 2014; Tkačik, Mora, et al., 2015; Berry II and Tkačik, 2020). Of course, this means we can no longer use $p(K)$ as a heuristic for goodness of fit. Instead, we start by looking at how well the K-pairwise model reproduces the triplet correlations $\langle(\sigma_i - \langle\sigma_i\rangle)(\sigma_j - \langle\sigma_j\rangle)(\sigma_i - \langle\sigma_k\rangle)\rangle$. In a population of 100 RGCs, the mean absolute difference between the predicted and observed triplet correlations was significantly below 10^{-3}, in contrast to what we saw above with the pairwise model, though this is still greater than estimates of the experimental error (Tkačik, Olivier Marre, Amodei, et al., 2014). K-pairwise models have also been shown to be fairly accurate in replicating the conditional probability of neuron N firing given the states of the other $N-1$ neurons, $p(\sigma_N|\sigma_1, ..., \sigma_{N-1})$ (Tkačik, Olivier Marre, Amodei, et al., 2014).

We have briefly summarised how well the independent, pairwise and K-pairwise models capture the activity of RGCs based on heuristics such as the population count distribution and triplet correlations. Though the K-pairwise model appears to be most accurate model, it still does not faithfully reproduce all the features of the activity of RGCs. One question that is worth asking is whether we can still learn from an approximate model, or even models that do not attempt to model the full joint distribution, such as a model of the population count distribution (Tkačik, Olivier Marre, Mora, et al., 2013). Fitting MaxEnt models often involves a compromise between the computational complexity associated with including many constrains in the model, and including enough of the well-sampled observables as constraints to obtain an accurate model. Assessing the goodness of fit is then a game of searching for significant features in the data that we have not (yet) included as constraints.

3.5.3. Overfitting

The use of MaxEnt models is motivated by building models out of statistics that we can reliably estimate from finite sets of data. However, how can we be certain that a sample statistic is sufficiently close to its true population parameter? Even if we are certain that all of our sample statistics are close to their true parameters, does a MaxEnt model built on the basis of many such statistics remain an accurate approximation to the distribution, or does the cumulative effect of many small discrepancies in our sample statistics result in a model that is very far away from the true distribution? We examine some of the techniques that allow us to assess whether the distributions specified by our models are generally good approximations to the activity of cells, or whether they are only representations of the data they are trained on.

In a data-rich context, the most straightforward way to assess how well a model gener-

alises is to reserve a set of the data from the training procedure as a test set with which we can use to evaluate the model. When we have a number of different models that may explain the data, it is common to split the data into separate training, validation and testing sets and then following training, use the validation set to select a model, typically the one that best models the validation set. Once a model has been selected, one can then evaluate its performance on the withheld test set. Thus one distinguishes between the processes of model selection and model evaluation.

In the context of MaxEnt models, we cannot blindly apply this procedure. Firstly, we need to consider that our data-set is a time series and that we made the simplifying assumption that we can take time averages of this data. Secondly, we need to consider that MaxEnt models are a product of their constraints, and it may only suffice to show that these constraints generalise across different subsets of our data in order to show that the same is true for the MaxEnt models that reproduce them. Finally, the model selection part of this procedure might not be necessary since there are cases where we may not want the most accurate MaxEnt model, for instance if we wanted to construct a null model that includes only low order moments. Rather than wanting to select the model which best describes the data, we more likely want to observe what changes as we move from one model to another. Thus our focus is not so much on model selection, but on model evaluation.

We try to tackle the above considerations in what follows. However, we first mention how this question of overfitting has been previously tackled. Since we try to maximise the log-likelihood of the data, to evaluate whether the model overfits to the training set, we could compare the log-likelihood of the training set to the log-likelihood of a withheld test set. If the log-likelihood of the training set is significantly larger than the test set, then we are likely overfitting to the training set. This has been the tactic used by Tkačik, Olivier Marre, Amodei, et al., 2014 where they randomly chose 90% of the *repeats* in the data to train the model and then compared the log-likelihood of the training data to the log-likelihood of the remaining 10% of the repeats. Each stimulus was presented to the retina multiple times producing multiple recordings of similar activity and we refer to the discretised, binned versions of these recordings as repeats. They constructed error bars around the ratio of the test and training log-likelihoods by looking at the standard deviation of models trained on different subsets of cells. Broadly, this tactic of resampling data and comparing statistics across the different samples falls under resampling methods. Before we discuss resampling further, it should be noted that when Tkačik, Olivier Marre, Amodei, et al., 2014 split their data-set they kept each repeat intact.

Bootstrapping is a resampling method which allows us to calculate measures of accuracy of statistics such as standard errors, particularly when we are not sure which distribution our data comes from. In the above case, it is not immediately clear which distribution the ratio of log-likelihoods follows, thus it makes sense to use bootstrapping to estimate error bars around our estimates. We can obtain B bootstrap samples $\{\boldsymbol{D^{*1}}, ..., \boldsymbol{D^{*B}}\}$ by randomly sampling with replacement from the data $\boldsymbol{D}$. If $\Phi(\boldsymbol{D})$ is a quantity computed from the data, then we can use our bootstrap samples to estimate aspects of the sampling distribution of $\Phi(\boldsymbol{D})$ (Davison and Hinkley, 1997). For instance, we can work out the variance in the quantity Φ as

$$\langle(\Phi - \langle\Phi\rangle_B)^2\rangle_B = \frac{1}{B-1}\sum_{b=1}^{B}(\Phi(\boldsymbol{D^{*b}}) - \langle\Phi\rangle_B)^2,$$

$$\langle\Phi\rangle_B \doteq \frac{1}{B}\sum_{b=1}^{B}\Phi(\boldsymbol{D^{*b}}).$$

We can use bootstrapping as both a means of obtaining standard errors around aspects of model performance and a means of assessing the accuracy of the statistics that we use to constrain our MaxEnt models. Whereas reserving portions of data from the training procedure can be used to assess the generality of what our model has learnt, bootstrapping allows us to quantify uncertainty of both the model's constraints and predictions.

When we assess how well our model generalises to unseen data, how we split our dataset has certain implications. Firstly, in choosing to train our model on data recorded from repeated presentations of the same stimulus to the retina and then testing it by examining its performance on unseen recordings but again involving the same stimulus, we ask whether the distribution obtained from expectations taken over the duration of the stimulus is similar across different presentations of the stimulus. In this case, if our model does appear to generalise well, it only suggests that what we have learnt is fairly consistent across different presentations of the same stimulus. Alternative ways of training and testing the generality of MaxEnt models are worth further investigation. For instance, could a MaxEnt model trained on the first half of an experiment be a good model for the activity in the second half? What could we learn if we trained a MaxEnt model with data from a variety of different stimuli? Ultimately, we are trying to build a distribution over all the states that appear in the activity of RGCs and it is not clear how close MaxEnt approximations of this distribution, estimated from finite samples of data, come to representing it fully.

3.5.4. Skepticism within the perturbative regime

In the previous few sections, we observed that the pairwise model seemed to be an accurate model for small populations of neurons but fails to account for significant features of the activity of larger populations such as the population count distribution $p(K)$ and the triplet correlations $\langle(\sigma_i - \langle\sigma_i\rangle)(\sigma_j - \langle\sigma_j\rangle)(\sigma_i - \langle\sigma_k\rangle)\rangle$. These empirical results suggest that it is unwise to extrapolate results obtained in small populations to larger populations. These observations bring the usefulness of what we learn in small populations into question. We now touch on a result which explains why we trivially observe the pairwise model being a good fit in small populations.

We start by defining the perturbative regime as the regime where the product of the mean probability δ of observing a neuron spike and the number of neurons N is very small. More accurately, δ is shorthand for the product of the mean firing rate $\langle v\rangle$ and the size of the time bin δt, $\delta \doteq \langle v\rangle\delta t$. When at most a single neuron fires within each time bin, we can identify δ as the mean probability of observing a neuron spike. Thus we can express the perturbative regime as $\delta \doteq \langle v\rangle\delta t \ll 1$. Within the perturbative regime, Roudi et al. found that relative to the Kullback-Leibler (KL) divergence between some true probability distribution $p^{(N)}$ and the independent model $p^{(1)}$, the KL divergence between the true probability distribution and the pairwise model $p^{(2)}$ is linear in $N\delta$ (Roudi, Nirenberg, and P. E. Latham, 2009). To be in line with their results, we define the KL divergence here using base 2:

$$D_{KL}(p\|q) \doteq \sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}) \log_2 \frac{p(\boldsymbol{\sigma})}{q(\boldsymbol{\sigma})}.$$

Using this definition of the KL divergence, we can write their result more formally as

$$\Delta_N \doteq \frac{D_{KL}\left(p^{(N)}\|p^{(2)}\right)}{D_{KL}\left(p^{(N)}\|p^{(1)}\right)} \propto (N-2)\delta + \mathcal{O}\left((N\delta)^2\right).$$

Note that Δ_N will be close to 0 when the pairwise model $p^{(2)}$ is close to the true distribution $p^{(N)}$, and Δ_N will be close to 1, when the pairwise model is barely better than the independent model $p^{(1)}$. As we increase the system size N while keeping δ fixed, Δ_N will scale linearly and the pairwise model becomes a worse and worse approximation regardless of the true distribution, provided that we remain in the perturbative regime. Though one seldom reads the appendix, we have gone to great lengths to re-derive and explain this result in Section A.2. We would consider the fact that have made this result more accessible as one of the significant contributions of this thesis.

The importance of this result is that the true probability distribution $p^{(N)}$ could have arbitrarily high order correlations and in fact be very different from a pairwise model.

However, by choosing to discretise our data with relatively small time bins, and by only considering small populations of neurons, the pairwise model will, almost trivially, seem like a good fit, at least based on the relative KL divergence. Roudi et al. make the even more severe statement that if we find that Δ_N scales linearly with N, then one has no information about the true distribution.

What are the dangers of choosing a very small time bin so that our model appears to be a good fit? We make the assumption of that all our states are temporally independent when we chose to model the instantaneous activity of neurons, where temporal independence is defined by $p(\boldsymbol{\sigma}_t, \boldsymbol{\sigma}_{t+1}) = p(\boldsymbol{\sigma}_t) \cdot p(\boldsymbol{\sigma}_{t+1})$. Although decreasing the width of the time bins makes the pairwise model appear to be an increasingly better model, in doing so the pairwise model becomes a worse and worse approximation for temporally correlated spike trains, and in binning our data into narrower bins we end with stronger temporal correlations (Roudi, Nirenberg, and P. E. Latham, 2009). For instance, if we had two cells which always fire together, but one lags slightly behind the other, then once our bin width is less than the lag, we end up with successive states where observing one of the cells firing in state t implies that we will see the other cell firing in state $t + 1$. This relates back to what we discussed in Section 2.2.2 about wanting to capture the combinatorial nature of the activity.

Given these cautionary remarks, how do we know when our model is actually a good fit for large populations of data? As N increases for fixed δ and we move out of the perturbative regime, if we observe that Δ_N no longer scales linearly with N and instead saturates, then what we learn may indeed be meaningful. This would require that we are able to calculate Δ_N and its associated KL divergences, which requires either access to the full true distribution, or accurate estimates of the KL divergence from finite samples of experimental data.

Though these remarks serve as a cautionary tale in the case of the pairwise model, it is worth asking whether we could generalise these results to other MaxEnt models. Going beyond pairwise models, it is common to then include the population count distribution $p(K)$ as a constraint, and it might be tempting to try to define a perturbative regime for the K-pairwise model. Though this may be possible through other means, we briefly outline the difficulty with adapting the current calculation to the K-pairwise model. As we see in the appendix, deriving results for the KL divergence between some true probability distribution and the pairwise model involved calculating multinomial expansions up to the sufficiently high order such that we observe differences between the true distribution and the pairwise model's third order correlations. What

would complicate adapting this method to the K-pairwise model is that due to its constraint $p(K)$, which reproduces aspects of the true distribution relating to correlations of all orders, there is no obvious order at which we can truncate the expansions. Thus, finding a means of defining the perturbative regime for the K-pairwise model remains an avenue for future research.

3.6. What can these models tell us?

Once we have fit a MaxEnt model to neural data and assessed its goodness of fit, what can we then learn? If our model appears to be a good approximation to the inaccessible true distribution, are the properties of our model inherently properties of the true distribution? Is there anything that we can learn from a MaxEnt model which does not do a good job of reproducing certain significant features of the data? Are there insights from statistical physics that can be brought into this discussion due to the similarity between MaxEnt models and the well-studied Boltzmann distribution? These are some of the broader questions we will attempt to answer as we elaborate on the insights we can gain from using MaxEnt models to model the activity of retinal ganglion cells. We start by explaining how we can use MaxEnt models with increasingly higher order interaction parameters as a means of identifying the statistical features of neural activity. This relates to the question of how many N-body interactions we require to model neural data, and whether we are likely to find simpler probabilistic description of the neural data. We then explore the hypothesis that retinal activity is organised into discrete clusters and how the properties of MaxEnt models support this. This prompts a discussion on the controversial topic of criticality in MaxEnt models and what this suggests about retinal activity. This discussion will attempt to bring in both the main results that arise from applying the maximum entropy principle as well as the main critiques of these results.

3.6.1. Hypothesis testing

MaxEnt models provide a systematic way of exploring the statistical features, or regularities, of neural activity. Within hypothesis tests, we can use MaxEnt models as null distributions. If we define a MaxEnt model with simple statistics as constraints, we can then compare the value of an *unconstrained quantity* produced by the model to the value of that quantity observed in the data. By unconstrained quantity we mean a quantity that the MaxEnt model has not been constrained to reproduce. When we find a significant difference between the two, this suggests that there is statistical structure within

our data that is not accounted for by the constrained statistics in our MaxEnt model (Savin and Tkačik, 2017). For instance, we may be interested in whether the pairwise correlations observed in the data can arise from a system where neurons fire independently. In this case, by using an independent model as the null distribution, we could compare the pairwise correlations produced by the distribution to the correlations observed in the data. The idea of using MaxEnt models in hypothesis testing dates backs to Martignon et al., who proposed using the full log-linear model to analytically test whether higher-order effects could be explained by lower-order moments (Martignon et al., 1995). Unfortunately, in practice fitting the full-log linear model to data requires knowledge of the full probability distribution.

An alternative approach to identifying important statistical features involves starting with a simple MaxEnt model, and then adding a further constraint to the model. If this model then better models the data, this suggest that the additional constraint may be an important feature of the data. However, one must keep in mind that refitting a MaxEnt model with additional constraints may not be trivial, for instance including all triplet correlations adds an additional $\binom{N}{3}$ parameters to the model that have to be determined, and hence the hypothesis testing approach may be preferred (Savin and Tkačik, 2017).

3.6.2. Clustering activity

We know that the activity of RGCs that we observe is noisy, in that the same stimulus will produce different states from one repeat to another, suggesting that the responses of the RGCs are inherently probabilistic (Berry II and Tkačik, 2020). We can easily illustrate this by looking at the responses of RGCs to two different repeats of the same stimulus. Though many of the same cells fire at the same time in these two different repeats, there are also cells which fire at particular times in one of the repeats but not the other. This is illustrated in Figure 3.1.

Although the exact state $\boldsymbol{\sigma}$ elicited by a stimulus may vary from one repeat to another, we might make the assumption that these different states represent the same thing, such as the detection of a particular visual feature, since the same stimulus gave rise to them. The question we have to ask ourselves is, "Are we justified in assuming that certain different states might represent the same thing, and if so, can we find a way of grouping together the states which represent the same thing?" Taking a brief step back from retinal states, we should highlight that grouping together different data points that are similar in some way is a classic problem in unsupervised leaning. This

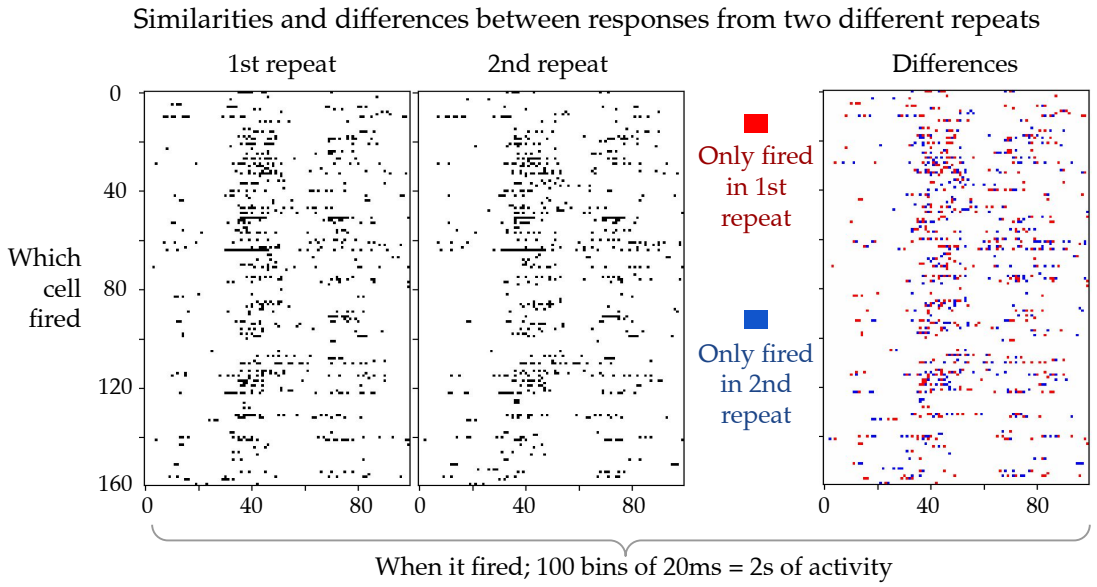
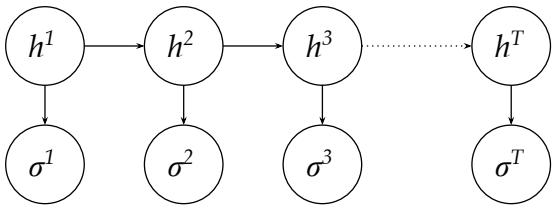


Figure 3.1: Binned recordings of the responses from 160 RGCs. The same stimulus is projected on to the retina repeatedly, and we show responses from two different repeats. The responses from these two different repeats are visually similar. However, in the third plot we highlight which cells fired at certain times in the first repeat but not in the second and vice versa. In the first repeat, we had 1 715 spikes and in the second we had 2 655. Only 245 of these spikes occurred in the same cell and within the same time bin. This highlights the noisy nature of the activity. The data for these results are introduced formally in the methodology section.

is broadly known as clustering and involves mapping each data point to a finite set of clusters. Clustering can also be thought of as a dimensionality reduction technique when we begin mapping higher-dimensional data points to lower-dimensional representations, for instance assigning binary vectors of length N to integers from 1 to M where $M < 2^N$. Later in our results, we will explore the use of autoencoders as a means of mapping the states of RGCs to lower-dimensional representations. For now, we want to explain the existing approach to clustering the activity of RGCs, as well as unpack why it might be useful to think of the activity through this lens.

The existing approach to clustering the activity of RGCs involves modelling a sequence of states in time $(\boldsymbol{\sigma}^{(1)}, ..., \boldsymbol{\sigma}^{(T)})$ using a *hidden Markov model* (HMM) (Adrianna Renee Loback, 2018). Although the individual states are the same as the ones we have been modelling, unlike the models we have been using, in the HMM we no longer ignore temporal dynamics. We view the sequence of observed states $(\boldsymbol{\sigma}^{(1)}, ..., \boldsymbol{\sigma}^{(T)})$ as arising from a hidden sequence of *latent* states $(h^{(1)}, ..., h^{(T)})$. We further make the assumption that the probability of each observed state $\boldsymbol{\sigma}^{(t)}$ only depends on the latent state $h^{(t)}$, and that each latent state $h^{(t)}$ only depends on the previous latent state $h^{(t-1)}$, which makes this process memoryless, in other words obeying the Markov property. We can graphically model a hidden Markov chain as follows:



If we know the probability of transitioning from one latent state to the next $p(h^{(t)}|h^{(t-1)})$, and the probability of observing a particular state given the latent state $p(\boldsymbol{\sigma}^{(t)}|h^{(t)})$, then we can determine the probability of observing a particular sequence of states by marginalising over all possible sequences of latent states, which we denote $\sum_{(h^{(t')})_{t'=1}^{T}}$:

$$p^{(\text{HMM})}(\boldsymbol{\sigma}^{(1)}, ..., \boldsymbol{\sigma}^{(T)}) = \sum_{(h^{(t')})_{t'=1}^{T}} \prod_{t=1}^{T} p(\boldsymbol{\sigma}^{(t)}|h^{(t)}) p(h^{(t)}|h^{(t-1)}).$$

Up until now, we have casually introduced latent states $h^{(t)}$ into our model without much explanation. Firstly, unfamiliar readers may be put off by the term "latent" which in this context is really just a fancy way of saying hidden. Though latent states can represent many different things in a model, such as unobserved effects or even a change of basis, in this application, each latent state can be thought of as a simplified description

of the observed state. In fact, we can define a mapping from each observed state $\boldsymbol{\sigma}$ at each point in time to the latent state h^* which maximises the probability of the observed state. This is that state's cluster. Importantly, we choose the number of possible latent states at each moment in time to be less than the number of binary vectors of length N. In this brief overview, we have skipped over many details, and recommend referring to Prentice et al., 2016 which first introduced this latent variable model to modelling RGCs. To recap this approach to clustering, we introduce a number of possible latent states that the population can be in at each moment in time, and instead of caring about the detailed transitions from one state $\boldsymbol{\sigma}^{(t)}$ to $\boldsymbol{\sigma}^{(t+1)}$, we assume each state can be mapped to a simplified latent state and that it suffices to model the transitions between the latent states. For emphasis,

$$\text{a state's cluster} \approx \text{a latent state of a HMM.}$$

Having outlined one mechanism of clustering retinal activity, we need to take a step back and ask why it is interesting to think of activity in terms of clusters. Berry II and Tkačik, 2020 propose three properties that we might expect from clusters:

1. Clusters exhibit *error correction* where the same stimulus maps to the same cluster despite variation in the exact responses.

2. Clusters encode qualitatively different visual features than their constituent cells.

3. Clusters can be learnt by downstream neural circuits in an unsupervised fashion.

Clearly these properties go beyond what can be inferred from HMMs and indeed, this line of research draws on results beyond HMMs. For instance, it also involves looking at the geometry of the clusters (A. Loback et al., 2017), and proposing biologically plausible means by which downstream neural circuits might recognise clusters (Adrianna R Loback and Berry, 2018). Pertinently, this hypothesis was greatly inspired by interpreting MaxEnt models through the lens of statistical physics. Although exploring all of these directions is clearly beyond the scope of this thesis, we strongly believe that it worth re-examining the role of MaxEnt models in this hypothesis.

The link between MaxEnt models and clusters is as follows. The distributions that arise from fitting MaxEnt models to the activity of RGCs are thought to have a probability landscape with many well-separated local peaks (Berry II and Tkačik, 2020). Since the energy function in MaxEnt models is proportional to the negative log probability $E(\boldsymbol{\sigma}) \propto -\ln p(\boldsymbol{\sigma})$, we also talk about the energy landscape having many well-separated local minima because the logarithm is monotonically increasing. Each local peak in the

probability landscape is then thought to relate to a distinct cluster of activity. At this stage we say "thought to", because many of these assertions have not been rigorously demonstrated, and this what we want to explore. We explain the current line of reasoning as to why this might be the case and highlight areas which require additional justification.

The assumed shape of the probability landscape of MaxEnt distributions relates to two observations. We mention these briefly now, though we will explain them in more detail below. The first observation is that in MaxEnt models trained on populations of RGCs, the pairwise interaction parameters, $\boldsymbol{J}$, seem to follow a Gaussian distribution with zero mean, and the fact that we observe both positive and negative interaction parameters gives rise to *frustration* (Schneidman, Berry, et al., 2006). The second is that if we re-scale the model parameters, so as to explore a family of MaxEnt models, the *phase* that the actual model (i.e. the model with unscaled parameters that reproduces certain aspects of the time-averaged activity) seems to be in is close to a *critical point*, characterised by a peak in the fluctuations of the energy per cell, or alternatively the specific heat.

Clearly these observations leave much to be explained, such as what frustration is, or what it means for a model to be in a particular phase, and we will have to carefully borrow concepts from statistical physics and ask whether these concepts are sensible in the context of RGCs. Although much of the promise of using MaxEnt models to model neural data is in the possibility of drawing on insights from decades of studying models such as Ising models in statistical physics, a lot of the confusion in interpreting these models arises from statistical physics concepts being used in a context where they were not developed without proper justification. To recap, we want to explore if a MaxEnt model is *frustrated* and if it is close to a critical point, then does it have many well-separated peaks in the probability landscape specified by its distribution.

We first elaborate on frustration. Frustration is said to arise when, due to the structure of our model and the values of its interaction parameters, no single combination of spiking and silence is able to minimise all the terms in the energy function and thus we have several equivalent lowest energy states (Tkačik, Olivier Marre, Amodei, et al., 2014). Typically, the presence of both positive and negative pairwise interaction parameters $\boldsymbol{J}$ in MaxEnt models has been taken as a sign that the model is frustrated – an assumption we will shortly question. A simple example of a frustrated triplets of cells is illustrated in Schneidman, Berry, et al., 2006, though we contrive our own illustration below. If we have three interacting cells $\sigma_1, \sigma_2, \sigma_3 \in \{0, 1\}$ with interaction

parameters $J_{12} = -1, J_{13} = -1, J_{23} = +1$ and without local fields $\boldsymbol{h} = \mathbf{0}$, then our pairwise model takes the form

$$p(\boldsymbol{\sigma}) = \frac{1}{Z} e^{-E(\boldsymbol{\sigma})}, E(\boldsymbol{\sigma}) = -\sigma_1\sigma_2 - \sigma_1\sigma_3 + \sigma_2\sigma_3.$$

Notice that the system has three states yielding the lowest energy, $\boldsymbol{\sigma} = (110), (111), (101)$, and thus this distribution has multiple equivalent maxima.

Though this particular toy model exhibits frustration, the existence of both positive and negative interaction parameters alone is not a guarantee of frustration. For instance, consider the above toy example with interaction parameters $J_{12}, J_{13} = 1, J_{23} = -1$, in which case there is a unique minimum at $\boldsymbol{\sigma} = (011)$. In the literature, frustration is largely assumed on the basis of observing interaction parameters with both positive and negative values (Schneidman, Berry, et al., 2006; Tkačik, Olivier Marre, Amodei, et al., 2014; Berry II and Tkačik, 2020). Importantly, the question we ask with MaxEnt models is not whether we can find a set of parameters that gives rise to frustration, but given the parameters of our model determined by the data, is the MaxEnt model frustrated, and thus has multiple local minima in its energy landscape. Currently, we are unaware of a general proof that MaxEnt models with both positive and negative interaction pairwise interaction parameters will necessarily be in a frustrated state. Unlike typical spin glass models where spins are placed on a lattice and interactions are limited to the nearest neighbours, in MaxEnt models with pairwise interactions, we typically include all possible pairwise interactions. Hence, the topology of the system is a fully connected graph.

Despite many gray areas in the argument that frustration can be inferred from the signs of the pairwise interaction parameters $\boldsymbol{J}$, there have been other, more direct attempts to show that MaxEnt models have multiple minima. Tkačik, Olivier Marre, Amodei, et al., 2014 were able to use Monte Carlo methods to descend the energy landscape of a trained MaxEnt model from different initial states and arrive at a number of distinct *locally stable* states. These locally stable states had the property that changing the value of any single cell in the state only increased the energy. This work not only provides evidence that MaxEnt models trained on retinal data can indeed have multiple local minima in the energy landscape, but also defines what it means to have a local minimum in a function over binary variables, i.e. states where changing the value of any single cell will only result in us moving to a state with higher energy. Though we speak of an energy landscape, a more appropriate metaphor for the domain would be an energy hypercube Q_N where a hypercube is a graph with 2^N vertices labeled with binary

vectors $\boldsymbol{\sigma}$ of length N and edges connecting two vertices whenever the Hamming distance $\sum_i [\sigma_i \cdot (1-\sigma_i') + (1-\sigma_i) \cdot \sigma_i']$ of their labels, $\boldsymbol{sigma}, \boldsymbol{sigma}'$ is one. The Hamming distance simply counts the number of positions where two binary vectors of length N differ. We should also point out that so far frustration has been an observation about the MaxEnt distribution learnt from data, as opposed to the actual data. Hopefully, this discussion on frustration has highlighted that it is still not exhaustively covered in the literature.

We now elaborate on how we might characterise the *phase* that a MaxEnt model is in. The mathematical form of a MaxEnt model resembles the Boltzmann distribution, and if we identify the argument of the exponential as the energy $E(\boldsymbol{\sigma})$, then the MaxEnt is isomorphic to the Boltzmann distribution at $k_B T = 1$ (See Section 3.3 for more on the Boltzmann distribution). Note, we are not assuming that we are modelling a system in thermodynamic equilibrium, but purely noting the mathematical equivalence between MaxEnt models and the Boltzmann distribution. However, due to this equivalence, it is interesting to ask whether we can perform similar analyses on MaxEnt models as we would with Boltzmann distributions. One interesting question that we can ask is which phase our model is in. To investigate phase transitions in a system modelled by Boltzmann distribution that is in the thermodynamic limit, we would adjust the temperature T and look at how this effects the heat capacity C or specific heat C/N. A divergence in these quantities indicates that our system undergoes a phase transition, and the temperature at which the divergence happens is referred to as the critical point (see Section 3.3).

MaxEnt models do not have a temperature T and, as they model finite systems of neurons, are not in the thermodynamic limit. In order to ask whether a MaxEnt model is close to a critical point, we would firstly have to introduce a fictitious temperature to the model and then as we vary this temperature while keeping all other parameters constant, look for a peak in, as opposed to the divergence of, the heat capacity. We could think of introducing a temperature into the argument of the exponential as introducing a means of rescaling our model parameters. As we move through this exponential family, we can then ask ourselves how the other models in this family compare to the model with unscaled parameters, corresponding to the actual model. We attempt to place the properties of our model in context by juxtaposing it with the other models we can obtain through rescaling the model parameters. The major difference between what we do here and what we do in models of phase transitions in statistical physics is that in statistical physics we vary the temperature of the actual system, which in turn may vary the temperature-dependent parameters in our model of the system. Here,

we change the parameters of our model of the system, while the system remains unchanged, and of course the moment we start changing the parameters of our model it is no longer an accurate description of the system.

Despite the questionable foundations of this analysis, it has been conducted in a number of works (Schneidman, Berry, et al., 2006; Tkačik, Schneidman, et al., 2006; Tkačik, Olivier Marre, Mora, et al., 2013; Tkačik, Mora, et al., 2015) where MaxEnt models have been fitted to populations ranging from $N{=}10$ to $N{=}120$. In all of these works, it has been observed that the actual MaxEnt model that fits the data, in comparison to versions of this model with re-scaled parameters, is poised close to a peak in the specific heat, or equivalently, the variance in energy per cell $\langle(\epsilon - \langle\epsilon\rangle)^2\rangle$ (both of these become functions of the re-scaling parameter T). Although a number of early works that looked at criticality in small populations of neurons speculated that larger populations will be poised at a critical point (Schneidman, Berry, et al., 2006), Berry II and Tkačik, 2020 revises this hypothesis and instead suggests that the population is in a marginally subcritical state, where the population is poised just below a critical point. Other than the fact that this finite system is not in the thermodynamic limit, a reason to doubt that we are exactly at a critical point is that criticality is associated with long-ranged correlations, however, measurements show that correlations between RGCs vanish when the cells are sufficiently far apart (Berry II and Tkačik, 2020). However, being poised close to a critical point still has certain implications for the shape of the energy landscape. Way above the critical temperature, minima become washed out, whereas way below the critical temperature, the probability distribution will be concentrated in a few peaks suggesting a low representational capacity, if we assume each local peak encodes a particular visual feature. Just below the critical temperature, we have many well-separated peaks (Berry II and Tkačik, 2020).

Interestingly, another work which looks at a K-pairwise model and re-scales the pairwise interaction parameters and the potential V_K while refitting the local field so that the model reproduces the averages $\langle\sigma_i\rangle, i \in \{1, .., N\}$ still finds that the un-scaled model is poised close to a critical point (Tkačik, Mora, et al., 2015). As opposed to the somewhat arbitrary re-scaling we saw above, in this approach we scale how much of a contribution the terms associated with higher-order interactions have in our model. This re-scaling allows us to go from a purely independent model to a model where higher order interactions completely dominate the energy function. This work provides another way of exploring the space of possible MaxEnt models and still found that the actual model was poised close to a critical point (Tkačik, Mora, et al., 2015).

While the hypothesis that we can cluster the activity of RGCs was inspired by the

suggestion that the energy landscape (or hypercube) specified by MaxEnt models has many well-seperated local minima, the current approach to learning these clusters relies on HMMs. The suggestion about the shape of the energy landscape arose through treating MaxEnt distributions as Boltzmann distributions in statistical physics, and trying to make the argument that the models we fit exhibit frustration and are close to a critical point. This could still be a valuable avenue of research, and there are promising results such as work by Tkačik, Mora, et al., 2015 which instead of introducing a fictitious temperature varied the extent to which we include higher-order interactions in our model. However, more time needs to be spent assessing whether concepts such as temperature, phases and frustration are well-defined in the context of retinal ganglion cells.

4. APPLYING THEORY

4.1. Methodology

We now move on to fitting different MaxEnt models to actual RGC data. First, we have to describe the RGC data, and we begin by visualising and summarising various aspects of it. We want to illustrate that the theory written out in the previous chapters does indeed translate into MaxEnt models that can practically be applied to modelling RGCs. To this end, we have implemented different MaxEnt models in Python which can be constrained to reproduce various observed statistics and are able to predict the probabilities of other states. These models are compared with an existing implementation written in Matlab that has been used in the literature (Maoz and Schneidman, 2017). We show agreement between the learned model weights and their predicted distributions for $N = 10$ cells.

Having shown agreement between our implementation and the existing implementation, we train independent and pairwise models on experimental data, detailed below in Section 4.1.1, and look at which features these models manage and fail to capture. We recreate the figures seen in Tkačik, Olivier Marre, Amodei, et al., 2014 showing the discrepancy between the distribution $p(K)$ predicted by the models and the empirical distribution. Although we do see differences between the predicted and empirical distributions in our results, it is not clear how these differences change as we vary the number of neurons N. To clarify this relationship, we fit models over a more finely sampled set of values for N and plot the difference between the model and empirical predictions.

In the literature, the fact the pairwise model seems to be a good approximation to RGC activity for small N and δ has been attributed to the fact that the pairwise model is in a perturbative regime where the normalised distance measure Δ_N, which tells us how far the pairwise model is from the true distribution, trivially scales relative to $N \cdot \delta$ (Roudi, Nirenberg, and P. E. Latham, 2009). Previously, Roudi, Nirenberg, and P. E. Latham, 2009 have illustrated these results by comparing a pairwise model to a third

order model with randomly initialised weights for systems of size $N \in \{4, ..., 10\}$ and for fixed values of δ. We now apply these results to our pairwise models trained on real RGC data for systems of size $N \in \{5, ..., 20\}$.

We finish by presenting some early results of another approach to simplifying the activity of RGCs. This approach takes inspiration from image compression and makes use of autoencoders, a type of artificial neural network which aimed at finding efficient representations of data. We outline each of these procedures in more detail below.

4.1.1. Data

Many of the recordings used in the papers mentioned thus far have been made publicly available. In this work, we make use of data from the paper *Searching for Collective Behavior in a Large Network of Sensory Neurons* (Olivier Marre, Tkacik, et al., 2017). This data has been made available on *research explorer*[1] (Olivier Marre, Tkacik, et al., 2017). It comprises a recording of 160 salamander retinal ganglion cells responding to 297 repeated presentations of a 19 second long so-called "natural" movie. We suspect this movie is of a fish moving since the title of the file is "...fishmovie32_100.mat" and is most likely the same movie of the fish that Gašper Tkačik shows in this *online talk*[2], which fits the description and time length.

Once the data-set has been downloaded from research explorer, we want to summarise and visualise various features of the data. Firstly, we want to show what the activity looks like. We can plot the binned retinal activity as a binary matrix where the (i, j) entry of the matrix indicates whether the i^{th} cell fired within the j^{th} time bin. We have already mentioned that the responses arise from repeated presentations of the same stimulus, and have briefly shown in Figure 3.1 that there is similarity across the responses from different repeats, but also a fair amount of noise in the responses. We supply further illustrations of how consistent activity is across different trials in our results, for instance by plotting how regularly individual cells fire. We also want to show the sample distribution of the averages $\langle \sigma_i \rangle_D$, pairwise correlations $\langle \sigma_i \sigma_j \rangle_D$, and population count distribution $p(K)_D$. We plot various summaries of these statistics in our results.

Finally, we apply Principal Component Analysis (PCA) to the activity, which we use as both a visualisation technique and a benchmark for dimensionality reduction of the activity. PCA involves computing a set of orthogonal unit vectors $\{\boldsymbol{v}_1, ..., \boldsymbol{v}_M\}, M \leq N$,

[1] https://research-explorer.app.ist.ac.at/record/5562
[2] youtu.be/m80prUNCn2g?t=1151

called the *principal components*, and then projecting the data-set onto the space spanned by the principal components. We search for the principal components such that i^{th} principal component explains as much of the variance in the data-set as possible while being orthogonal to the first $i-1$ principal components. Though there are many ways to compute the principal components, one way of obtaining them is by computing an *eigendecomposition* of the sample co-variance matrix $\boldsymbol{Q}$. Representing the state of cell i in the t^{th} sample in our data-set as $\sigma_i^{(t)}$, the entries of the sample co-variance matrix can be computed as

$$q_{ij} = \frac{1}{|D|-1}\sum_{t=1}^{|D|}(\sigma_i^{(t)} - \langle\sigma_i\rangle_D)(\sigma_j^{(t)} - \langle\sigma_j\rangle_D).$$

The eigendecomposition of the sample co-variance matrix can be written as

$$\boldsymbol{Q} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1},$$

where $\boldsymbol{V}$ is a square matrix whose columns are the eigenvectors $\boldsymbol{v}_1, ..., \boldsymbol{v}_M$ and $\boldsymbol{V}^{-1}$ is its inverse. $\boldsymbol{\Lambda}$ is a diagonal matrix whose i^{th} diagonal element is the associated eigenvalue of eigenvector $\boldsymbol{v}_i$. The eigenvectors can be identified with the principal components and the eigenvalues as relating to the proportion of variance explained by each eigenvector (Hastie et al., 2009).

Having outlined the methods used in the exploratory data analysis, we now turn our attention to how we should prepare the data-set for our models. We want to create training data-sets which we can fit models to, and test data-sets which we can use assess the generality of what we have learnt on the training sets. Where model selection and hyper-parameter tuning is necessary, we may elect to further reserve a validation data set. It is worth reflecting on how we should reserve a portion of the data, because different ways of dividing up the data-set have implications for what type of generality we asses.

The data involves recordings of the RGCs reacting to multiple presentations of the same stimulus. In reserving a portion of the data for training and validation sets, there is the question of whether we should randomly shuffle all the states and then split them into training and validations sets, whether we should keep the states associated with different presentations of the stimulus intact and randomly assign entire repeats to training and validation sets or whether we should use the first portion of every repeat as training data and the latter portion as validation data. We test whether there is any difference between these three approaches by looking at how many samples, repeats and the proportion of each repeat that we need to average over before the expectations stabilise. Although we do not expect this to be a system at equilibrium, we do expect the

responses of the cells to be similar for each presentation of the stimulus. The question then becomes how many presentations we have to average over before the estimates stabilise. The code that was used to process the Matlab data can be found *here*[3] and the code we wrote to investigate this question among many others can be found *here*[4].

4.1.2. Proof of concept

For the independent, pairwise and population count models, we compare our own implementation to the implementation in Maoz's *MaxEnt toolbox* (Maoz and Schneidman, 2017). We work with a subset of the data outlined in the previous section, looking at the activity of 10 cells. For each model, we compare the corresponding fitted model parameters from each implementation as well as the corresponding predictions for the probabilities of each of the possible states. We aim to show that the model parameters and the predicted probabilities of each state are the same for both implementations.

We can further quantify how close the corresponding models are in the two implementations by looking at the KL divergence between them:

$$\begin{aligned} D_{KL}(p^{(\text{Maoz})} \| p^{(\text{ours})}) &= \sum_{\boldsymbol{\sigma}} p^{(\text{Maoz})} \log \frac{p^{(\text{Maoz})}}{p^{(\text{ours})}} \\ &= \langle \log p^{(\text{Maoz})} - \log p^{(\text{ours})} \rangle_{p^{(\text{Maoz})}}. \end{aligned}$$

We phrase the KL divergence like this to emphasize that it is an expectation of the difference in log probabilities, and when two distributions predict identical probabilities over the state space, the KL divergence is zero. These results are presented in Figure 4.12.

4.1.3. Comparing distributions

We now move on to to fitting models to experimental data and comparing the predicted and the observed population count distributions $p(K)$. We do this for different sub-populations of size N. For each number N, we randomly select N cells from the total of 160 cells B times to give us B bootstrap sub-populations at each N. We have previously introduced bootstrap sampling in Section 3.5.3. For each sub-population of size N, we fit independent and pairwise models to the activity of those N models. Since we want to use the population count distribution as a metric for goodness of fit, which is unconstrained in the independent and pairwise models, but not in the population

[3] github.com/abramschon/maxent/blob/main/matlab/readData.m
[4] github.com/abramschon/maxent/blob/main/notebooks/EDA.ipynb

count or K-pairwise model, we restrict this analysis to the independent and pairwise models.

We make use of the MaxEnt toolbox implementation (Maoz and Schneidman, 2017) which, we have observed, manages to find the optimal model weights for the pairwise model faster than our Python implementation. The MaxEnt toolbox implements computationally intensive methods such as importance sampling in C++, which has a considerable impact on performance. Whereas the independent models can be fitted analytically for each N, we iteratively fit the pairwise model until its expectations are all within 1 standard deviation of their experimental counterparts. Our code for fitting the various MaxEnt models can be found *here*[5].

Once we have the fitted models we can then determine their predicted population count distributions. Analytically, computing $p(K = 1)$, for instance, would involve summing together the probabilities of all states where only one cell fires. As computing this distribution analytically involves predicting the probabilities of exponentially many states this becomes unfeasible for large N. It is much more convenient to determine the population count distribution using Monte Carlo methods. Hence, we use the Metropolis-Hastings algorithm to sample states from the learnt model distribution and then we compute the population count distribution from the samples, which involves computing the relative frequency that K cells fire within a state. The code for working out the expectations from the trained models can be found *here*[6].

We now can go on to comparing the predicted and observed distributions. We start by plotting the distributions predicted by the independent and pairwise model and the empirical distribution for $N = 10, 40, 100$, reproducing the result seen in Tkačik, Olivier Marre, Amodei, et al., 2014. We have fitted 10 different models at each N over this range, and thus we can get a sense of the variability of the predicted $p(K)$s. Fitting pairwise models to the activity of 100 cells takes several days, which placed practical restrictions on the number of sub-populations we could fit models to. We can plot five number summaries of the predicted $p(K)$s at each K. The five number summary of a set of values is the minimum, first quartile, median, third quartile and maximum value of the set. By looking at the distances between these summary statistics, we can get a sense of the spread of the values.

When we consider the population count distributions on a logarithmic scale (and exclude the probabilities exactly equally to zero), which makes the differences between

[5] github.com/abramschon/maxent/blob/main/matlab/trainModel.m
[6] github.com/abramschon/maxent/blob/main/matlab/compareMarginals.m

the predicted distributions more clear, it becomes apparent that the independent and pairwise models fail to capture the shape of the empirical distribution. However, it is not clear how this changes with N. Early papers claim that the pairwise model is a good approximation to the experimental data for small N, typically around $N = 10$. Can we quantify how this approximation breaks down with N?

To explore this, we plot the differences between the predicted and observed population count distributions for $N = 5, 11, ..., 20$, i.e. $p(K)_{\text{model}} - p(K)_{\text{exp}}$. Since the probability of observing many cells firing decreases as K increases, these differences trivially become small as K increases. Whereas previously, we could better visualise the differences between the tails of the empirical and model distributions by looking at the probabilities on a logarithmic scale, we are now working with a difference which can be negative and we have to consider another way of exploring these small differences. Though we could take the absolute value of the differences, this would erase information about which parts of the true distribution our model over-predicts and which parts it under-predicts. We choose to scale these differences relative to the sizes of the predicted and empirical probabilities.

The relative differences are worked out as follows. For each N, we have $B = 20$ predictions of the whole population count distribution $p(K)$ obtained from models trained on 20 different sub-populations. The absolute differences for a particular K' are $p(K')_{\text{model}} - p(K')_{\text{exp}}$, of which there will be 20. We can work out the mean difference at each K by averaging over the 20 differences associated with each sub-population $\langle p(K')_{\text{model}} - p(K')_{\text{exp}} \rangle_B$. Notice, we use the subscript B to denote an average over estimates from the $B = 20$ sub-populations. We can also work out the mean of $p(K')_{\text{model}}$ and the mean of $p(K')_{\text{exp}}$, and then take the sum of the two means, $(\langle p(K')_{\text{model}} \rangle_B + \langle p(K')_{\text{exp}} \rangle_B)$, as a scaling factor for the difference. This is by no means the definitive way of scaling this result, but this is certainly a means of looking at whether differences in the estimates of $p(K')$ are large relative to the average size of $p(K')$ for a particular K'. The resulting relative difference at a particular K' can be written as:

$$\frac{\langle p(K')_{\text{model}} - p(K')_{\text{exp}} \rangle_B}{\langle p(K')_{\text{model}} \rangle_B + \langle p(K')_{\text{exp}} \rangle_B}$$

The relative differences can be interpreted as follows. When the model *over-predicts* $p(K)$, this quantity is positive, and when the model *under-predicts* $p(K)$, this quantity is negative. When the model predicts $p(K)$ to be zero, but $p(K)_{\text{exp}}$ is observed to be non-zero, the relative difference is -1, and when the model predicts $p(K)$ to be non-zero when it is empirically observed to be zero, the relative difference is $+1$. By

over-predicts/under-predicts we mean a prediction is greater/less than the empirically observed value.

The motivation for looking at the differences relative to a sum of the means of the model and empirical $p(K)$s and not one of these alone is that as K increases, we often observe either the predicted or empirical $p(K)$ dropping to 0 faster than the other. Dividing by just one of them could then cause the relative difference to blow up since we would divide by zero even when there is a difference between the two quantities which we would be interested in seeing. Where both the empirical and model estimates of $p(K)$ are zero, we have the relative difference equal to $0/0$, and have have omitted these values from our plots. Lastly, we must emphasize that the estimates of $p(K)$ that we get for large K are likely to be increasingly inaccurate since we observe states with large numbers of cells firing increasingly rarely as K increases. Because of this, the plots of the differences, particularly towards the tails of the distributions, should be seen more as a data visualisation technique and by no means as a formal hypothesis test for whether there is a significant difference between the empirical and model estimates. The notebook for computing the differences and relative differences can be found *here*[7], though we present the main results in Section 4.4.

4.1.4. Autoencoders

So far, we have been searching for a simplified description of the activity through building MaxEnt models that predict the probabilities of all 2^N possible states but require relatively few low-order moments from the data to be trained. We now consider a different approach to finding simplified descriptions of this activity. We investigate whether we can find latent representations of the data itself through the use of *undercomplete autoencoders*, which map their input to a lower-dimensional latent representation. The use of autoencoders in this application presents many opportunities, such as dimensionality reduction, a change of basis, and even a sampling tool. We start by presenting a proof of principle that it is possible to use auto-encoders to learn mappings between our N-dimensional binary vectors to M-dimensional real-valued vectors where $M < N$. Though this continuous latent representation has its benefits, it is not clear that it reduces the dimensionality of its inputs since all binary vectors of length N can trivially be mapped to a single continuous dimension. We then alter the architecture of our autoencoder so that it can map N-dimensional binary vectors to M-dimensional binary vectors where $M < N$. In this section, we explain what auto-encoders are, and outline our training procedure. We then present our results in Section

[7] github.com/abramschon/maxent/blob/main/notebooks/correlations.ipynb

4.5, we outline future directions for this research.

Autoencoders are a type of *artificial neural network* (ANN) which takes in an input, $\boldsymbol{\sigma}$, and maps it to a so-called *latent representation*, $f(\boldsymbol{\sigma}) = \boldsymbol{h}$, before it then tries to reconstruct the input from the latent representation $g(\boldsymbol{h}) = \boldsymbol{r}$. The latent representation is effectively the image of $\boldsymbol{\sigma}$ under the transformation f. We refer to the network that maps the input to the latent representation as the encoder, $f(\boldsymbol{\sigma}) = \boldsymbol{h}$, and the network which then reconstructs the input from the latent representation as the decoder, $g(\boldsymbol{h}) = \boldsymbol{r}$. We typically impose some form of restriction on the autoencoder, such as an information bottleneck or an objective function that encourages sparsity, preventing it from simply copying its input. Our hope is that this forces it to find a useful latent representation of its input (Goodfellow, Bengio, and Courville, 2016). In the case of the undercomplete autoencoder, we require that the latent representation, $\boldsymbol{h}$, has fewer dimensions than the input, thus we can view this as a dimensionality reduction technique. The squeezing of the higher dimensional input through a lower dimensional space is called an *information bottleneck*.

We briefly explain the architecture of the auto-encoders that we use, which is represented graphically in Figure 4.1. We make use of the following notation:

L	is the number of layers in the autoencoder
N^l	is the number of units in layer l, $N^0 = N^L = N$
σ_i	is the i^{th} entry of the input
a_i^l	is the i^{th} activation of the l^{th} layer and $a^l(z)$ is the activation function
h_i	is the i^{th} entry of the latent representation
r_k	is the k^{th} entry of the reconstruction
$w_{i,j}^l$	is the weight from node i in layer $l-1$ to node j in layer l
b_i^l	is the bias term fed into the i^{th} unit of the l^{th} layer
z_j^l	is a linear component defined by $\sum_{i=1}^{N^{l-1}} a_i^{i-1} w_{i,j}^l + b_j^l$

We represent the encoder $f(\boldsymbol{\sigma})$ and decoder $g(\boldsymbol{h})$ as fully connected, feed-forward ANNs, also called multilayer perceptrons. ANNs are composed of layers of units which take the dot product between their input $\boldsymbol{a}$ and a vector of weights $\boldsymbol{w}$, add a scalar offset called the bias b, and then apply a nonlinear activation function $a(z)$. For a network with L layers, and N^l units in layer l we can represent the forward compu-

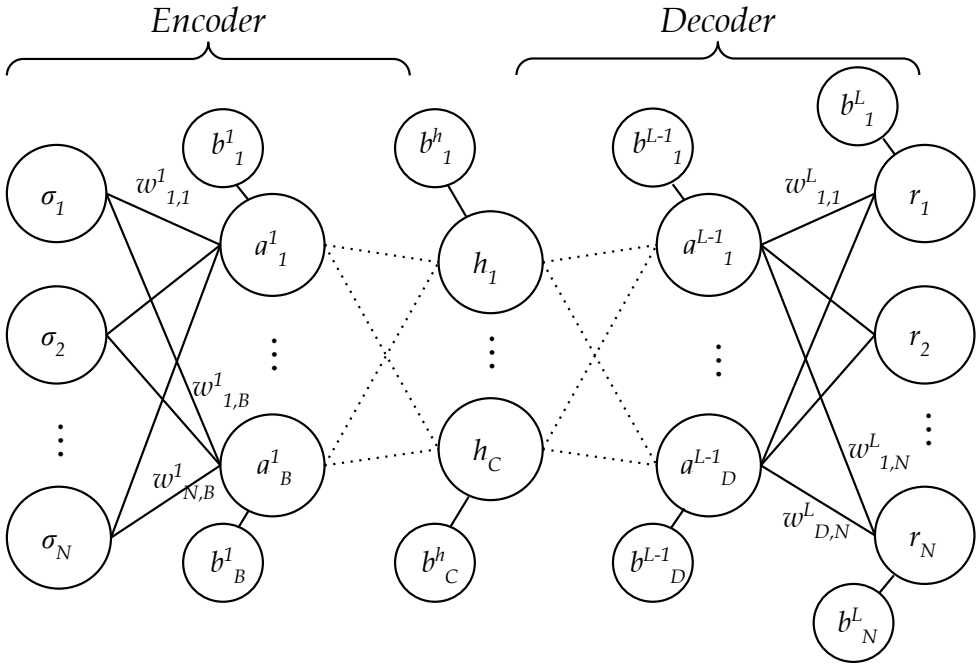


Figure 4.1: Graphical depiction of an autoencoder with L layers. There are N units in the input ($\boldsymbol{\sigma}$) and output layer ($\boldsymbol{r}$), B units in the first layer ($\boldsymbol{a}^1$), C units in the hidden layer ($\boldsymbol{h}$), and D units in the second last layer ($\boldsymbol{a}^{L-1}$). Each unit i in layer l performs a weighted sum of the outputs of the units in layer $l-1$, adds a bias b^l_i and then applies an activation function.

tation as

$$\begin{aligned} a_j^0 =& \sigma_j \\ a_j^l =& a^l \left(\sum_{i=1}^{N^{l-1}} a_i^{l-1} w_{i,j}^l + b_j^l \right) \\ a_j^L =& r_j \end{aligned} \tag{4.1}$$

where the activations of one of the intermediate layers represent the latent representation $\boldsymbol{h}$. We can also represent this using matrix multiplication as $\boldsymbol{a}^{(l)} = a(\boldsymbol{W}^{(l)}\boldsymbol{a}^{(l-1)} + \boldsymbol{b}^{(l)})$, where row i of matrix $\boldsymbol{W}^{(l)}$ represents the weights connecting unit i in layer l to the units in layer $l-1$ and the activation function is applied component-wise. In our architectures, we make use of the rectified linear unit, $\text{ReLU}(z) = \max(z, 0)$, as the activation function applied to each layer except for the latent layer.

Initially, in the *vanilla* autoencoder, we apply the identity function $a^h(z) = z$ as the activation function of the latent layer. This means that its latent representation takes on continuous real values $\boldsymbol{h} \in \mathbb{R}^{N^h}$, where N^h is the number of units in the latent layer. In the *compressive* autoencoder, we apply the sigmoid function followed by the rounding operation as the activation function of the latent layer $a^h(z) = \text{round}[1/(1+\exp(-z))]$, where the rounding operation rounds the output of the sigmoid function to either 0 or 1 depending on which is closest to the output. This means that its latent representations are binary vectors of length N^h.

The rounding operation makes gradient-based learning difficult. This is not simply because the derivative of the rounding operation is not properly defined at 0.5 – the same can be said of the rectified linear unit (its derivative is also not properly defined at 0), which has very successfully been used in deep learning. Instead, the reason is because the derivative of the rounding operation where it is properly defined is 0 which causes the backpropagated partial derivatives to vanish (see the backpropogation algorithm (Goodfellow, Bengio, and Courville, 2016)). Thus, we borrow a trick from Theis et al., 2017 and for the purposes of the partial-derivative computations in backpropagation, pretend that we have only applied the sigmoid function and not the rounding operation, and hence we pass on the partial derivatives of the sigmoid function with respect to their input. In the final layer of both the vanilla and compressive auto-encoders, we apply the sigmoid function $a^L(z) = 1/(1+\exp(-z))$, which bounds the values of the reconstructed output $\boldsymbol{r}$ between 0 and 1. We can then easily convert this to a binary vector through applying the rounding operation again.

In order to train the auto-encoder, we need to define a loss function which measures how close the model's input is to its reconstruction. We initially considered several loss functions, namely the Euclidean $(\boldsymbol{\sigma} - \boldsymbol{r})^2$, Manhattan $\sqrt{(\boldsymbol{\sigma} - \boldsymbol{r})^2}$ and Hamming distance $\sum_i [\sigma_i \cdot (1 - r_i) + (1 - \sigma_i) \cdot r_i]$, however, we were only able to train the model using the Euclidean distance as the loss function. At this stage, this is purely an empirical observation and given more time it would be interesting to find a theoretical explanation for why certain loss functions work and others do not in reconstructing binary vectors. Given a loss function, we are able to update the model weights and biases by working out the partial derivatives of the loss with respect to the model parameters and then nudging the model parameters in the direction that decreases the loss. Working out these partial derivatives efficiently can be achieved through the use of the backpropagation algorithm (Goodfellow, Bengio, and Courville, 2016).

We wanted to explore different model depths L and widths N^l, including different numbers of hidden units. These parameters which get set before the learning process begins are called *hyper-parameters*, and a good choice of hyper-parameters can have a significant impact on the model's performance. In order to make these choices, we typically fit different models with different combinations of hyper-parameters and compare their performance in a process called hyper-parameter tuning. A naïve way of hyper-parameter tuning is *grid search* where, for each hyper-parameter we define a set of values of interest and then fit models to all possible combinations of the hyper-parameters. Although this process is easily parallelisable, it is computationally intensive. For H hyper-parameters with M distinct values, the complexity of grid search is $O(M^H)$.

It has been shown empirically that *random-search* is on par with if not better than grid search in hyper-parameter tuning using a fraction of the computation time (Bergstra and Bengio, 2012). In random search, each combination of hyper-parameters are selected at random. More recent hyper-parameter tuning algorithms include versions of Bayesian optimisation (Snoek, Larochelle, and Adams, 2012), which builds a probabilistic model of the loss function and uses knowledge of previously chosen combinations to inform which combination it chooses next, or Hyperband (Li et al., 2017), which assigns a certain amount of training resources to different hyper-parameter combinations, dropping the poorly performing configurations and increasing the resources for the remaining well-performing combinations.

We carry out the following hyper-parameter tuning procedure:

- We first choose the number of hidden units N^h depending on the type of ar-

chitecture. Keeping in mind that our input is a binary vector of length 100, for the vanilla autoencoder, which allowed continuous latent states, we looked at $N^h \in \{2, 18, 34, 50\}$. For the compressive autoencoder, which restricted its latent states to binary vectors, we looked at $N^h \in \{20, 40, 60, 80\}$. The size of the latent vector defines the size of the bottleneck.

- For each each choice of bottleneck, we randomly choose $I = 20$ different architectures for the autoencoder. We do this as follows:
  - We randomly choose the depth of the network L.
  - We randomly choose $(L-2)/2$ numbers $\boldsymbol{u}, u_i \in [N^h, \lceil N \cdot 3/2 \rceil]$. The number of units in each layer of the encoder are the entries of $\boldsymbol{u}$ sorted from largest to smallest $u_1 \geq u_2 \geq ... \geq u_{(L-2)/2}$ and the number of units in each layer of the decoder are the entries from smallest to largest, mirroring the encoder.
  - We train an autoencoder with this architecture and we monitor its performance. At the end of each presentation of the training data-set, the models tries to reconstruct the validation data-set and we record the mean Euclidean distance between the reconstruction and original data. When the validation loss stops decreasing over 3 to 5 presentations of the full training set, we terminate the training procedure for this architecture and move on to the next architecture.

The outer-loop of this hyper-parameter tuning procedure varies the number of hidden units and the inner-loop explores different depths and widths of the encoder and decoder. This procedure produces a sequence of training and validation losses for each of the explored architectures. From here, we can then look at the best and worst architectures and how validation performance varies across the number of units in the hidden layer, the depth of the network, and the mean width of the network.

We make use of the same data-set described in Section 4.1.1. It is comparatively quick to fit auto-encoders to the activity of 100 RGCs, and we use the activity of the first 100 RGCs from the training set processed in Section 4.1.1 . We further randomly reserve 80% of the observations as training data and 20% as validation data. The code for the vanilla autoencoder can be found *here*[8] and the code for the compressive autoencoder can be found *here*[9]. We present these results in Section 4.5.

[8] github.com/abramschon/maxent/blob/main/notebooks/autoencoder.ipynb
[9] github.com/abramschon/maxent/blob/main/notebooks/squishautoencoder.ipynb

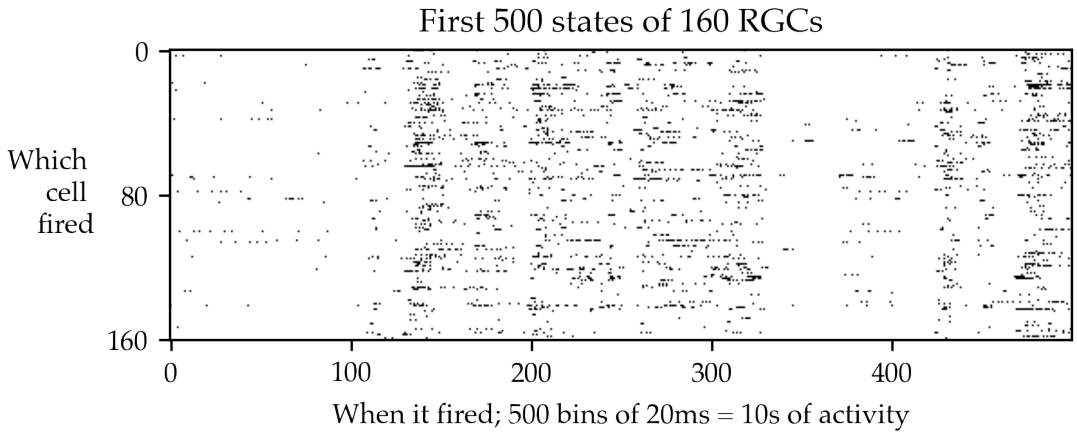


Figure 4.2: The first 500 states in the data-set of the responses of 160 RGCs to repeated presentations of a stimulus. Each vertical slice of this 160×500 matrix corresponds to a state. We shade in the entries of the matrix where at least one spike takes place within the corresponding time bin.

4.2. Exploring the data

4.2.1. What does it look like?

Having broadly outlined our methodology, we can start applying it. We present a visual introduction to the data that we model. This binned RGC activity comprises 283 041 states of 160 cells. Though we later explore shuffling the order of the states, by default the states are in sequential order through time. Within the 283 041 states, we can further identify groups of 297×953 states. Each 953 state group corresponds to the responses of the cells to one of the 297 repeated presentations of the stimulus. We informally refer to these repeated presentations as *repeats*.

We start by plotting the first 500 states in the data-set. We can think of this as a 160×500 binary matrix, which can be depicted visually by shading in the entries which are 1 and leaving the entries which are 0 unshaded. The combination of the number of the cell, i, and the time bin, t, forms the co-ordinates for each entry. If an entry is 1 it indicates that cell i fired at least once with the 20ms window encompassed by time bin t. This is a bit wordy to repeat, so often we will be informal and say that cell i spiked in time bin t. The first 500 states are shown in Figure 4.2. Though this gives us a good sense of what the raw data looks like, we can present a more a simplified description of the activity by summing together the number of cells that fire in each time bin, and plot

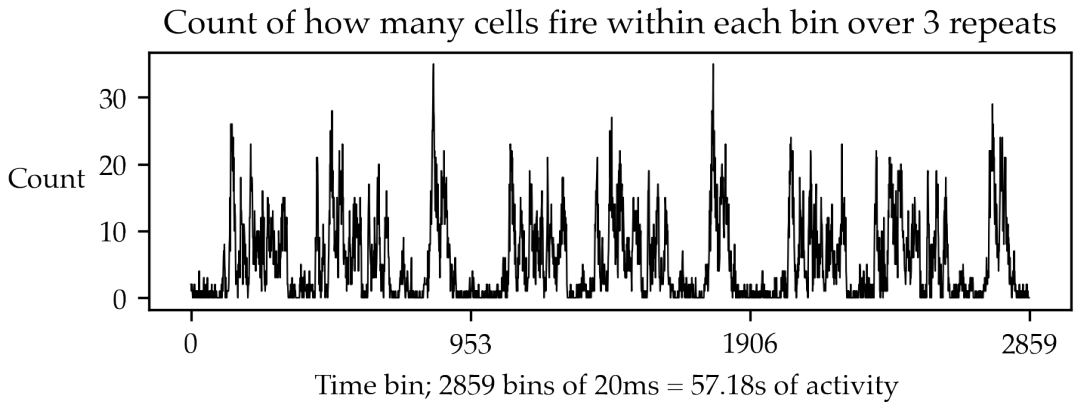


Figure 4.3: Counts of how many cells fire within each time bin over 2871 bins. As mentioned, each repeated presentation of the stimulus is associated with 953 states, and we observe similar counts of neurons firing at the same points in each repeat.

this against time. This is shown in Figure 4.3, and roughly shows us how many cells fired at different moments in time.

We want to highlight two things in this figure. The first is that although we are looking at the activity of 160 cells, we seldom see more than 30 cells fire within a state. This observation is echoed in our plot of the first 500 states which is largely blank. We will make this observation of the rareness of many cells firing at once more precise when we begin to look at the data distributions of $p(K)_D$ and $\langle \sigma_i \rangle_D$. The second is that there is some periodicity in the responses relating to the repeated presentations of the response. We also observe that the exact number of cells that fire at the same times in different repeats is not constant.

We want to explore this idea of consistency in the responses a bit further. In Figure 3.1, we highlighted the entries where a cell fired in one extract but not another. Though there was a visual similarity between the two extracts, when we overlayed the extracts we saw that there were many entries that were one in the first extract but were zero in the second, and vice versa. To put numbers to this observation, there were only 245 entries which overlapped (corresponding to the same cells firing within the same bins in different repeats), whereas there were 880 entries which did not overlap. Though we would have to perform a more rigorous analysis, this is already an illustration that the activity is noisy in some sense.

To what extent should we do expect similar responses to different presentations of the same stimulus? To explore this further, we look at the activity of individual cells across

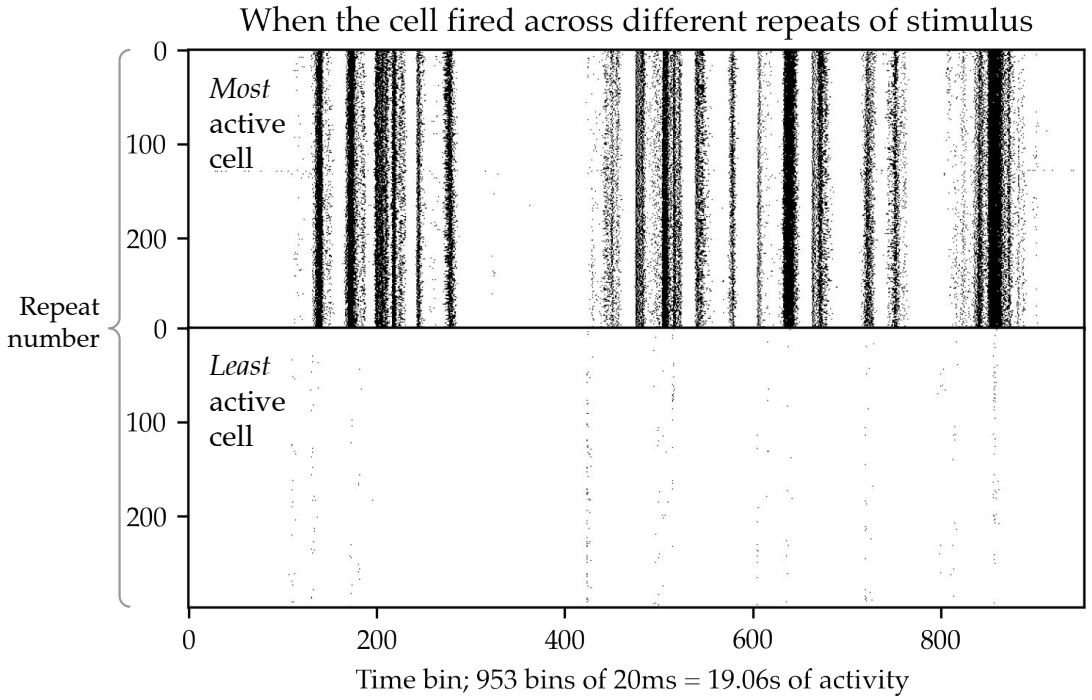


Figure 4.4: We show the times at which the most and least active cell fired over 297 repeated presentations of stimulus

the different repeats. Here, we have decided to show the activity of the most active and the least active cell, which was measured by counting the total number of bins the cells fired in. This is presented in Figure 4.4. The emergent vertical bars that we observe suggest that these individual cells often spike at similar moments in time during the stimulus. However, if we focus on the more feint vertical bars of the least active cell, we observe that it does not consistently spike across all repeats, and that there might also be a time delay to when it does spike, though this is speculation. These plots certainly highlight the need for probabilistic models which are robust to differences in the details of each state.

4.2.2. Averages, correlations and the population count distribution

Our use of MaxEnt models involves fitting the models to expectations calculated by taking averages along the temporal dimension of our data. We want to now introduce some of these expectations, such as the averages ,$\langle \sigma_i \rangle_D$, pairwise correlations, $\langle \sigma_i \sigma_j \rangle_D$, and population count distribution $p^{(D)}(K)$. We want to get a sense of how small and how spread out these values typically are. For the averages and pairwise correlations,

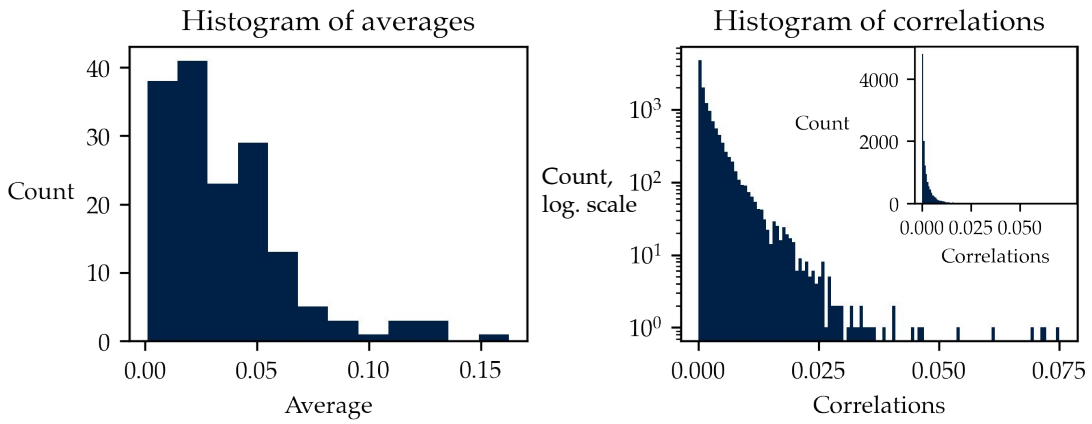


Figure 4.5: Histograms of the averages $\langle \sigma_i \rangle_D$ and the correlations $\langle \sigma_i \sigma_j \rangle_D$. We present the counts of the correlations on a logarithmic scale on the right plot, and on a linear scale in the inset.

we can do this by plotting their histograms which show how many expectations fall into different bins. We show these in Figure 4.5.

In this figure, we observe that most of the averages are very small, and the correlations are even smaller. The average for cell i involves counting in how many time bins it fires and dividing by the total number of bins. Thus we can interpret the average $\langle \sigma_i \rangle_D$ as the mean probability of observing cell i firing within a bin. Similarly, we can interpret the pairwise correlation as the mean probability of observing cell i and cell j firing within a bin. We have 160 different averages, one for each cell, of which the smallest was observed to be 0.0011 and the largest 0.1625. The mean of these averages was observed to be 0.0358, thus the average cell fires in 0.0358 bins on average. We refer to this as the *mean activity*. Of the $160 \times 159/2$ pairwise correlations, the smallest correlation was exactly 0, implying that at least one pair of cells never fired within the same bin. The largest observed pairwise correlation was 0.0751, and the mean pairwise correlation was 0.0026.

Given the small values that the averages take on, it does not seem particularly surprising that the pairwise correlations are even smaller. We might be tempted to approximate the pairwise correlation between cell i and cell j as the product of their averages, i.e. $\langle \sigma_i \sigma_j \rangle_D \approx \langle \sigma_i \rangle_D \cdot \langle \sigma_j \rangle_D$. We can show that this approximation fall shorts of describing the observed pairwise correlations by comparing the pairwise correlation, $\langle \sigma_i \sigma_j \rangle_D$, to its independent approximation $\langle \sigma_i \rangle_D \cdot \langle \sigma_j \rangle_D$, which is shown in Figure 4.6. If the actual pairwise correlations were equal to the correlations produced under the as-

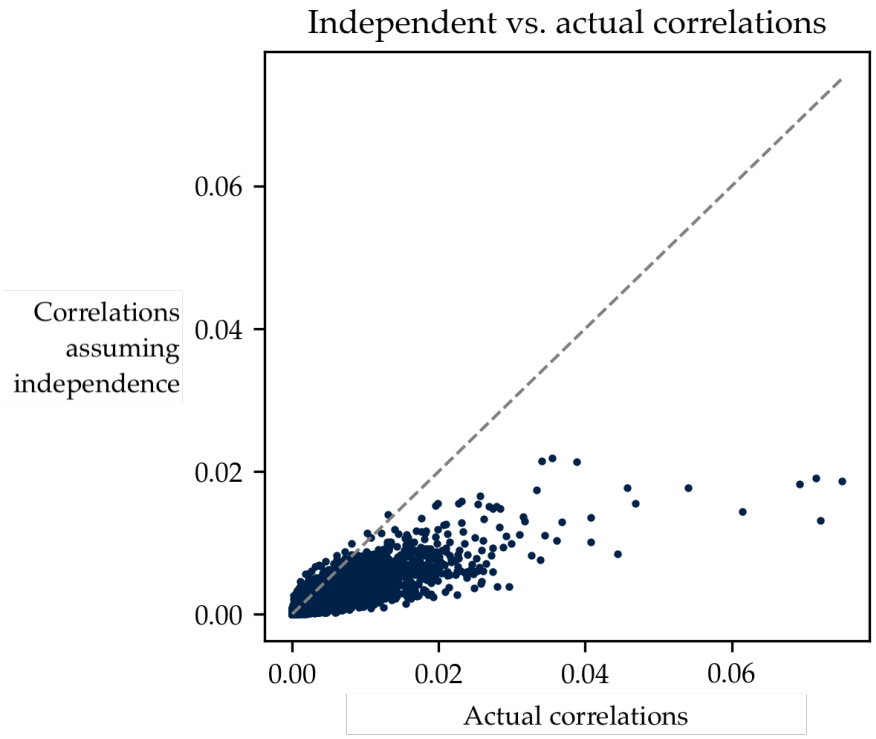


Figure 4.6: The actual pairwise correlations compared to the correlations that would be produced if we assumed that the cells fired independently. If these were the same, then the points would lie on the line $y = x$.

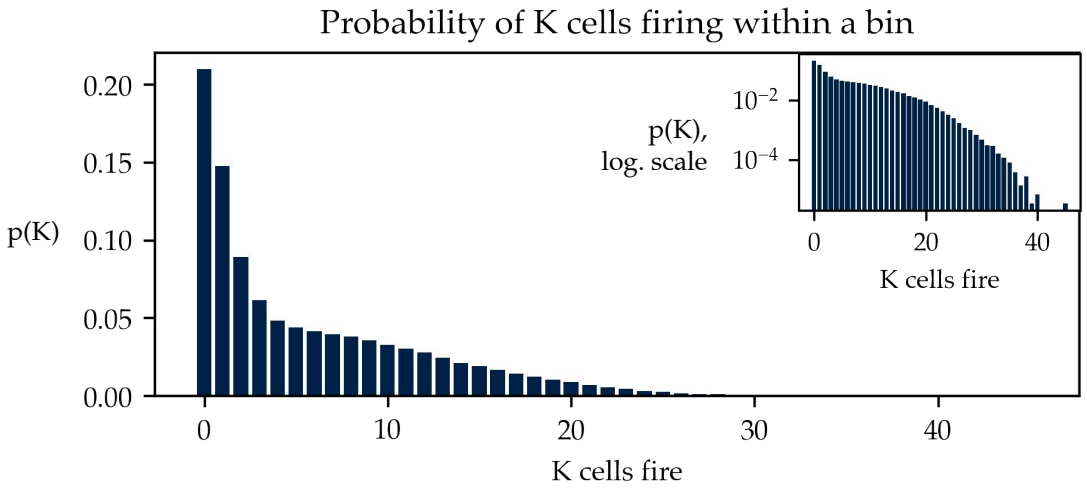


Figure 4.7: Probability of observing K cells firing within a state. In the inset, we plot the probabilities on a logarithmic scale.

sumption that the cells fire independently, then we would observe that the points lie on the line $y = x$. However, we observe that the observed correlations are largely greater than those predicted under the assumption of independence. This observation already hints at the shortcomings of the independent MaxEnt model. This suggests that the behaviour of individual cells must at least be influenced by the activity of other cells, if not the activity of the population. One snapshot of the activity of the population is the population count distribution $p^{(D)}(K)$ which looks at how often K cells fire within a state. This distribution is shown in Figure 4.7.

This plot again highlights the prevalence of silence within this data-set, since the probability of zero cells firing in a state, $p^D(K = 0) = 0.21$ has the highest probability. We observe the probability $p^{(D)}(K)$ quickly becoming small with K, and in a data-set of 283 041 states we at most observed 45 cells firing within a single state. This occurred only once, hence the probabilities of the states in the tail of the empirical population count distribution from $K = 46$ onward are estimated to be exactly zero.

4.2.3. Principal component analysis

In Figure 4.3, we provided a simplified description of the activity by summarising each state by the number of cells that fired within that state. We now consider another approach to simplification through the use of PCA, though as you will see, we we still try to convey how many cells fire within each state in our analysis. We perform an eigen-

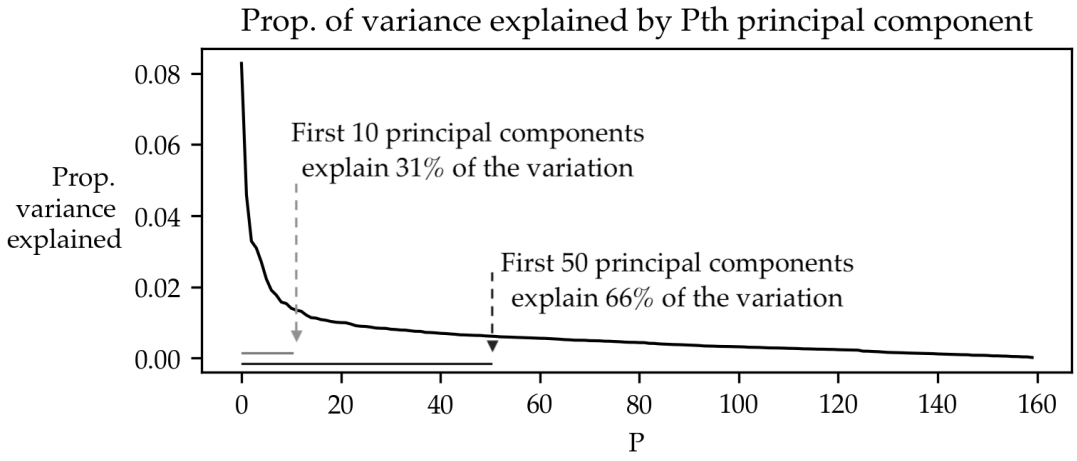


Figure 4.8: Proportion of variance explained by the P^{th} PC.

decomposition of the sample co-variance matrix and identify the eigenvectors as the Principal Components (PCs) and the eigenvectors as relating to the proportion of variance each eigenvector explains. We order the eigenvectors based on their corresponding eigenvalues such that the first PC is the eigenvector with the largest eigenvalue, the second is the eigenvector with the second largest eigenvalue, etc. We can examine the *proportion of variance* in the data-set *explained* by each of our PCs by looking at their corresponding eigenvalues, scaled by the inverse of the sum of all the eigenvalues. We show this result in Figure 4.8.

We can obtain a simplified description of our data-set by projecting it onto the first M PCs where $M < N$. In this case, our new variables would be linear combinations of the discretised activities of cells. However, based on the plot of the proportion of variance explained, we find ourselves in the situation where choosing a small number of PCs, e.g. $M = 10$, to project our data onto misses out on a lot of the variation in the data, but including more PCs explains increasingly less of the variation. The first ten PCs, which is near the elbow in the proportion of variance explained curve, only explain 31% of the variance in the data-set, and adding forty additional PCs then only explains 66% of the variance. Although there is no obvious choice of the number of PCs that we keep, which would define the granularity of our simplification, there has been interesting work which borrows ideas from the *renormalisation group* that embraces this observation and instead looks at which features emerge as we vary the amount of granularity of our simplification (Meshulam et al., 2018). PCA only allows us to perform a linear transformation to our data. Our later work with autoencoders explores a non-linear

transformation which may yield more rich representations.

We can project our data-set onto the first two PCs, which happen to explain around 13% of the variation in the data-set, so we must be wary that this is a rather drastic simplification. We colour each point, which corresponds to a state in the data-set, based on how many cells fire within that state, linking this technique back to our other summary of the activity of each state. This projection is presented in Figure 4.9.
Interestingly, the first PC, which is the direction that captures the most variation in the data-set, seems to also be the direction that largely differentiates how many cells fire within each state, which can be seen by how the emergent colour of the projection goes from dark blue on the left to yellow on the right. We also present the projection of the data onto the first three PCs in an interactive plot in a Jupyter notebook, which can be found *here*[10].

4.2.4. Sampling and generalisation

We train our MaxEnt models to reproduce certain expectations, such as the ones we have introduced above. However, we still have to quantify how representative these expectations are of the general activity of the population of RGCs. Since MaxEnt models are very much a product of the expectations they are trained to reproduce, if these expectations do not in general represent the activity very well, the MaxEnt models will not either. To investigate this more concretely, we consider different ways of collecting samples from the full data-set, and we compare how much a summary of the activity of these samples varies. The summary we use is the mean activity of the sample, which can be computed as the total the number of entries that contained spikes within the sample divided by the total number of entries. However, the emphasis is not on the mean activity, but on how this mean activity varies depending on how we collect samples, and on how large our samples are. From the full data-set of $297 \times 953 = 283041$ states, every 953 consecutive states can be associated with a repeat. We consider:

- Randomly sampling a number of the repeats. This can be thought of as shuffling the 297 repeats and taking the first R repeats.
- Randomly sampling a number of states irrespective of repeat, which can be thought of as shuffling the 283041 states and taking the first S states.
- Randomly sampling C consecutive states from all repeats. This can be thought of as randomly choosing a time bin, t', and selecting the C consecutive states starting from state $\boldsymbol{\sigma}^{(t')}$ across all repeats.

[10] github.com/abramschon/maxent/blob/main/notebooks/EDA.ipynb

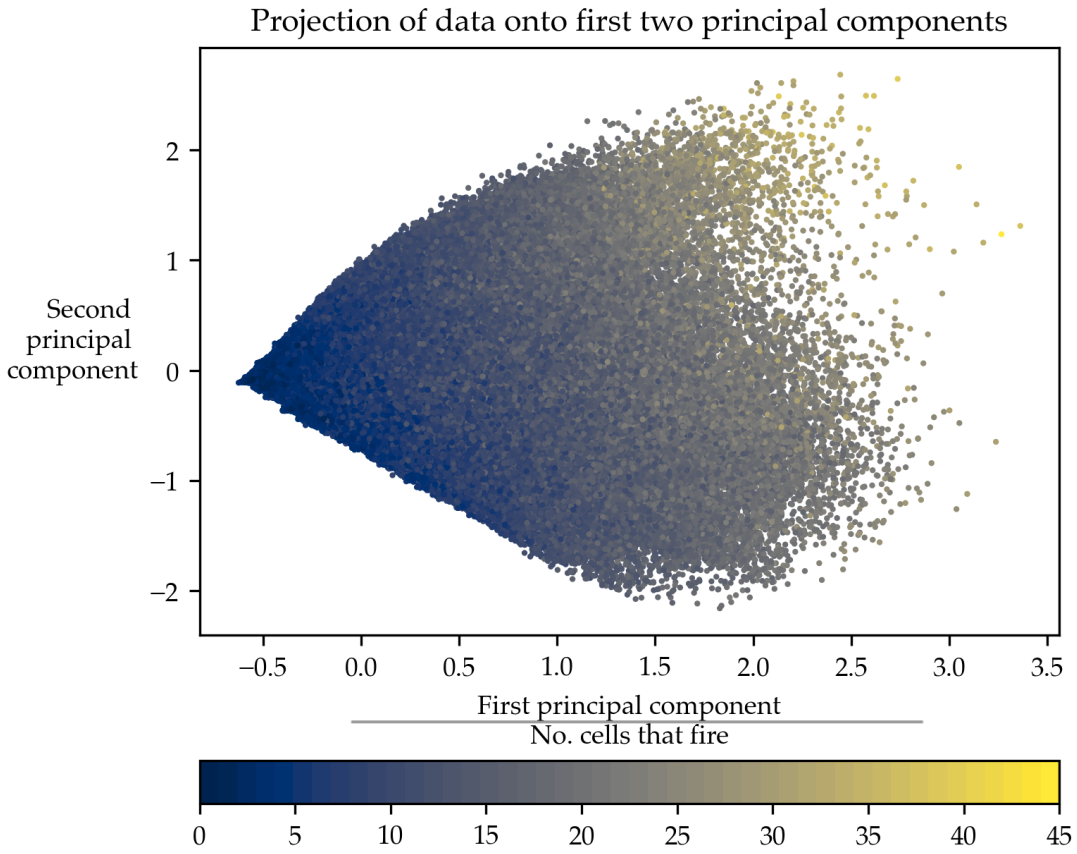


Figure 4.9: Projecting the 283 041 states onto the first two PCs. Each point is coloured based on how many cells fire within the state associated with the point. Though we have tried to separate our label for the x-axis and the title of the colourbar with a line, this still may cause some confusion. The x-axis is the first principal component and the colourbar shows the relationship between the colour of the states and the number of cells that fire in each state.

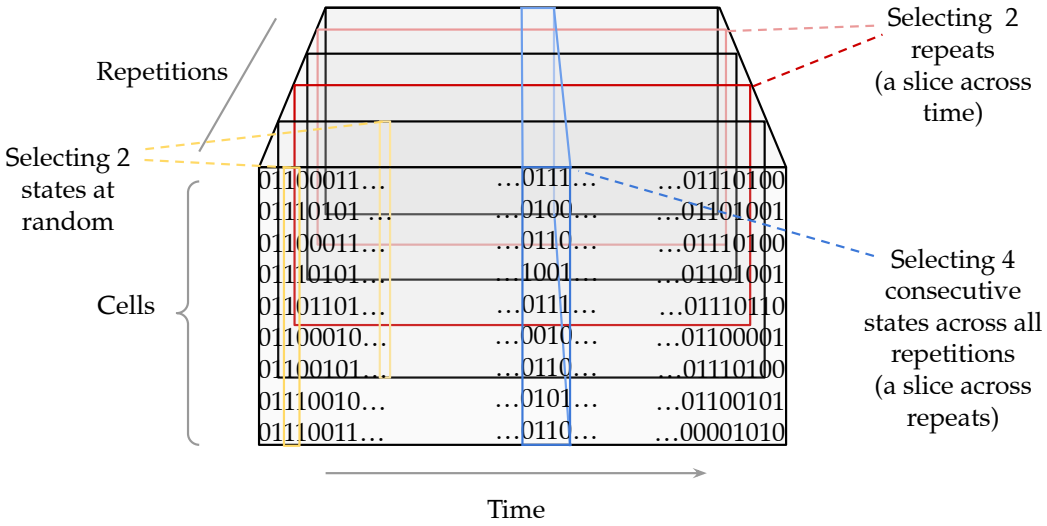


Figure 4.10: Illustrating the different ways of sampling from the data-set. Sampling a repeat can be thought of as taking random time slice from the cube, sampling consecutive states in time can be thought of as taking a random wedge across the repetitions, and sampling a state at random involves picking a random time and a random repeat.

A more elegant way of thinking about these sampling methods involves imagining the data-set as a rectangular prism, which we have illustrated in Figure 4.10. This prism has a dimension relating to the different cells, a dimension relating to the different repeats and a dimension relating to time. In all the sampling methods we are interested in population activity, hence we always select all entries across the cell dimension. However, we can elect to either take slices across time, across repeats, or randomly pick combinations of a particular repeat and a particular time.

The reason we have chosen to focus on choosing C consecutive states across all repeats and not just random states across all repeats is because we can make an analogy between the relationship that this small selection of consecutive states has to the whole recording as to the relationship that the whole recording has to a hypothetical much longer recording of the retina responding to further stimulus. We want to look at whether our brief snapshot of time might be representative of the greater picture through time. Our results from comparing the mean activity across the different sampling methods are presented in Figure 4.11.

There is quite a lot happening in these three seemingly simple graphs. Firstly, the basic unit of our plot is the five number summary of the mean activities of $B = 100$

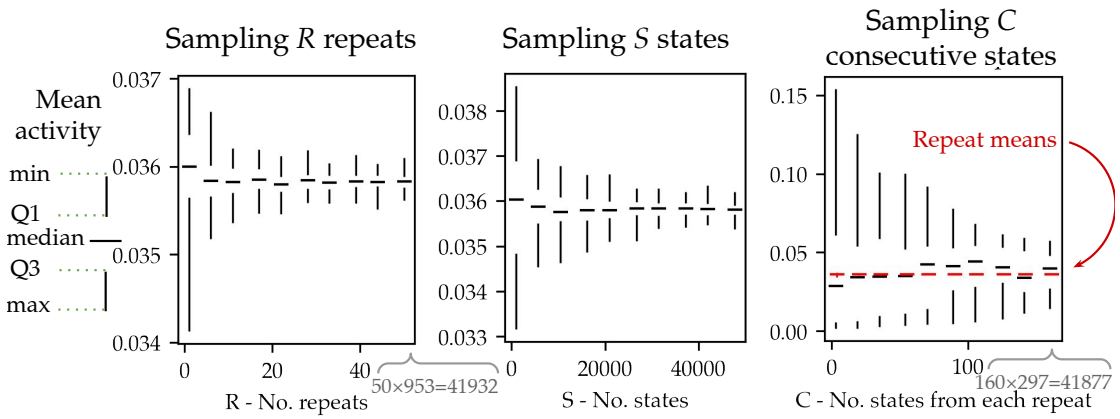


Figure 4.11: Using three different sampling approaches to obtain $B = 100$ bootstrap samples of roughly the same size, we vary the size of the samples and observe how the variability of the estimates of the mean activity decreases. Each repeat comprises 953 states, hence sampling R repeats returns $R \times 953$ states. Selecting a state at a particular time from all repeats returns 297 states, hence sampling C consecutive states returns $C \times 297$ states.

bootstrap samples of a particular size. As we have explained, there are three ways that we have obtained samples, each associated with its own plot. Across all three sampling methods, by drawing larger samples, the spread of mean activities across the bootstrap samples went down. The estimates of mean activity calculated over samples of repeats were the most stable (the left plot), and were similar to estimates calculated by sampling at random (the middle plot). Randomly taking C consecutive states across all repeats resulted in the most varied estimates of mean activity. This suggests that the activity over a small duration of time is often different to the overall activity. The median of the mean activity across all sampling techniques was roughly the same.

From repeat to repeat, the data appears to be noisy, and we may attempt to think of this as individual cells firing slightly earlier during some repeats, later at others, and sometimes not at all. We also need to take into consideration that the data we have presented has already undergone many stages of processing and simplification which will also have a significant effect on how it looks. If we do assume that similar states arise from the same stimulus, then our definition of similarity needs to be robust to differences in the exact details of the states. This idea of robustness is a recurring theme in this work. We have also highlighted that sampling from random moments in time pro-

duces expectations that are more variable than sampling from different repeats, though these methods become more similar when the expectations across repeats include more consecutive states. These exploratory data findings do a good job of setting the stage for the work that follows. That we ignore temporal information in our modelling approach is a major caveat of this work, even though this approach has still yielded interesting insights in the literature into collective behaviour of RGCs. The emphasis on not fixating on the exact description of each state helps motivate the use of probabilistic models for modelling the activity, as well as the use of compressive auto-encoders which cannot learn the exact descriptions.

4.3. Proof of concept

We present the comparison between our fitted models for the population count, independent and pairwise model in Figure 4.12. This is for $N = 10$ RGCs. For each state, we have both the probability predicted by our model and the probabilities predicted by the MaxEnt toolbox implementation. If these probabilities match, then we should observe that the points in the scatter-plot lie on the $y = x$ line, which is indeed what we observe. We observe a similar correspondence when we compare the predicted model weights. We do notice that there is a small amount of variation around the $y = x$ line with the pairwise model. This arises from our implementation being slower in converging to the optimal model parameters, especially for the pairwise interaction parameters.

Though these results would visually suggest that these models are similar, we can be more precise in quantifying this similarity. Specifically, we can look at the mean of the absolute differences between corresponding model parameters, and we can look at the KL divergence between the distributions specified by the models. For the independent models, the mean of the absolute differences in the parameters $\boldsymbol{h}$ was 6.454×10^{-3}. For reference, the mean size of these parameters was 3.871. The KL divergence between their distributions was 5.79×10^{-6}. For the population count models, the mean of the absolute differences in the parameters $V(K)$ was 8.857×10^{-5}, and for reference the mean size of $V(K)$ was 0.0909. The KL divergence between their distributions 5.0954×10^{-5}. Finally, for the pairwise models the mean of the absolute differences in the local fields $\boldsymbol{h}$ was 8.689×10^{-3}, and the mean size of these parameters was 4.092. The mean of the absolute differences in the pairwise interaction parameters, $\boldsymbol{J}$, was 6.197×10^{-2} and the mean size of these parameters was -0.361. The KL divergence between their distributions was 6.696×10^{-5}. In all of these models, the mean of the absolute differences was significantly smaller than the typical scale of the parameters,

and we observed very small KL divergences between the distributions. If we had to interpret the small KL divergences in the language of information theory, we could say we would expect almost no excess *surprise* when using our model's distribution when the actual distribution is specified by their model.

These results are just the surface of my much deeper exploration of how to code up MaxEnt models[11]. Though these results may suggest that these two implementations were developed in isolation, in reality my coding up of MaxEnt models was very much inspired by papers written by the same authors who were involved in developing this Matlab implementation, and as I later make use of their implementation to fit models to larger populations, I have had to become familiar with it. The fact these two implementations arrive at approximately the same solutions is less the message. Rather the message is that, with a bit of reading and debugging, someone with little software engineering training can code up MaxEnt models. The code that compares these two impementations can be found *here*[12].

4.4. Maximum entropy approximations

4.4.1. Independent and pairwise population count distributions

We compare the predicted and empirical population count distributions for $N = 10, 40, 100$ in Figure 4.13. This starting point reproduces the results seen in Tkačik, Olivier Marre, Amodei, et al., 2014. As we see in Figure 4.13 at $N = 100$, the independent model initially under-predicts $p(K = 0)$. It then over-predicts $p(K)$ for $K = 3, 4, 5, 6$, before going back to under-predicting the probabilities for $K \geq 7$, though we should exercise caution in predicting the values towards the tail of the distribution due to the poor sampling of this region, as mentioned above. The value of K at which the model over or under-predicts varies with N. The pattern of under-predicting silence and large groups of cells firing and over-predicting small numbers of cells firing looks as if it persists across at least these three sizes of populations.

In the case of the pairwise model at $N = 100$, the trajectory of $\log p(K)$ initially has a similar pattern to the independent model, under-predicting silence, then over-predicting small numbers of cells firing at $K = 3, 4, 5, 6$, before under-predicting over $K = 7, .., 17$. However, it ends up over-predicting the probability of large numbers of cells firing. Though we should be cautious about our estimates of $p(K)$ for large K, it does seem

[11] Excuse my temporary abandonment of the royal we.

[12] github.com/abramschon/maxent/blob/main/notebooks/compare.ipynb

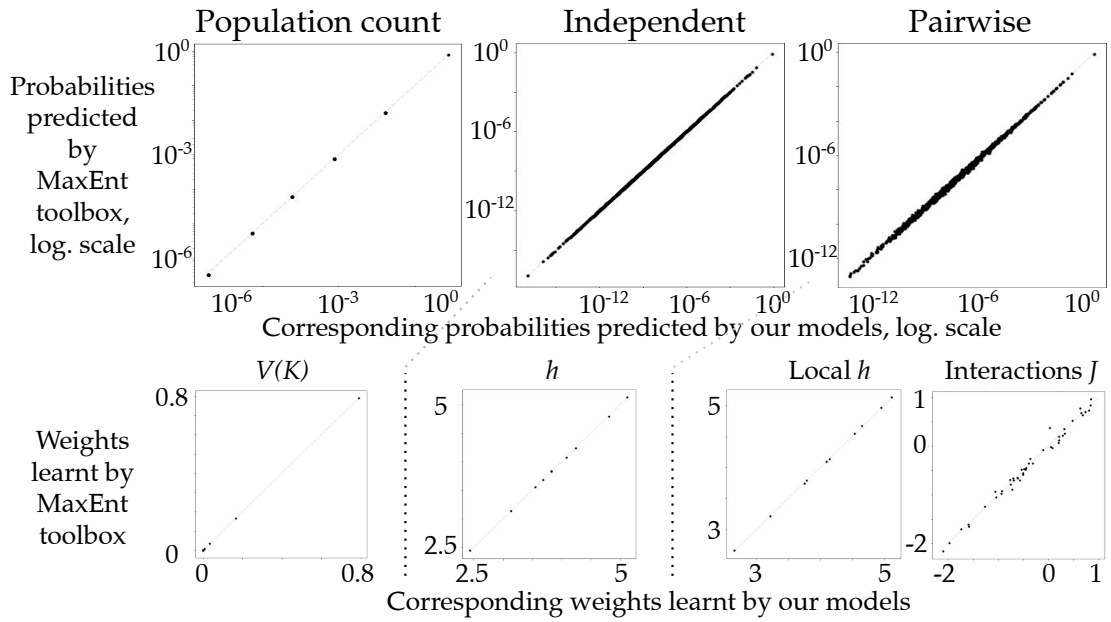


Figure 4.12: Comparing our implementation to the MaxEnt toolbox implementation for $N = 10$ cells. We matched each probability predicted by a model in the MaxEnt toolbox up with the probability predicted by our corresponding model. The same is true for the weights. We observe that both the predicted distributions (top row) and the learnt model weights (bottom row) match for the population count, independent and pairwise models. The reason only 6 distinct points for the population count model is because these models predict the probability of seeing all states where K cells fire as equally probable. For the pairwise model, we plot the correspondence between the local field parameters and pairwise interaction parameters for the pairwise models separately, and observe some variation around the $y = x$ line for the interaction parameters.

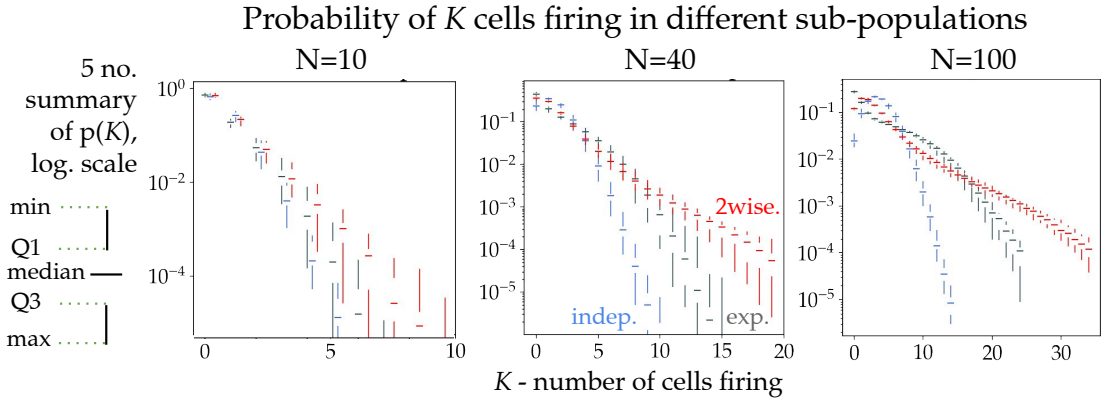


Figure 4.13: The probability of observing K cells firing in 10 sub-populations of size 10, 40 and 100 is shown above. The trajectories in grey represent the experimentally observed population count distributions and are labelled in the middle plots with exp. (the other plots follow the same pattern). The trajectories in blue represent the distributions predicted by the independent model, labelled indep., and the trajectories in red represent the distributions predicted by the pairwise model, labelled 2wise. We show the 5 number summaries of the 10 values obtained for each $p(K')$.

that the independent model predicts a population count distribution with lighter tails than what is observed, the pairwise model predicts a distribution with heavier tails, and that this pattern of over and under-predicting persists across these three different population sizes. At this stage this is purely an empirical observation. Intuitively, since the independent model does not incorporate any interactions, the probability of many cells firing is just the product of the probabilities of individual cells firing $p(\sigma_i) = \langle \sigma_i \rangle$ (since this observable selects all state where cell i fires, it is a concise way of expressing the marginal), and we know that the averages $\langle \sigma_i \rangle$ are typically very small, hence this product will become increasingly small with the number of cells that fire. We have already seen the observed pairwise correlations are greater than their independent approximation in Figure 4.6.

The five number summaries at each K give us a sense of the variability of these results across different populations. Particularly, we might like to see why the shortcomings of the pairwise model in approximating the population count distribution were not picked up in early work. For $N = 10$, we see that the range of predictions for $p(K)$ for the pairwise models largely overlaps with what is empirically observed and it is difficult to say confidently that the two distributions are significantly different for small K. Though the medians in the pairwise predictions are consistently above the medians of $p^{(D)}(K)$ from $K = 4$ onward, it is not clear that there is a significant difference

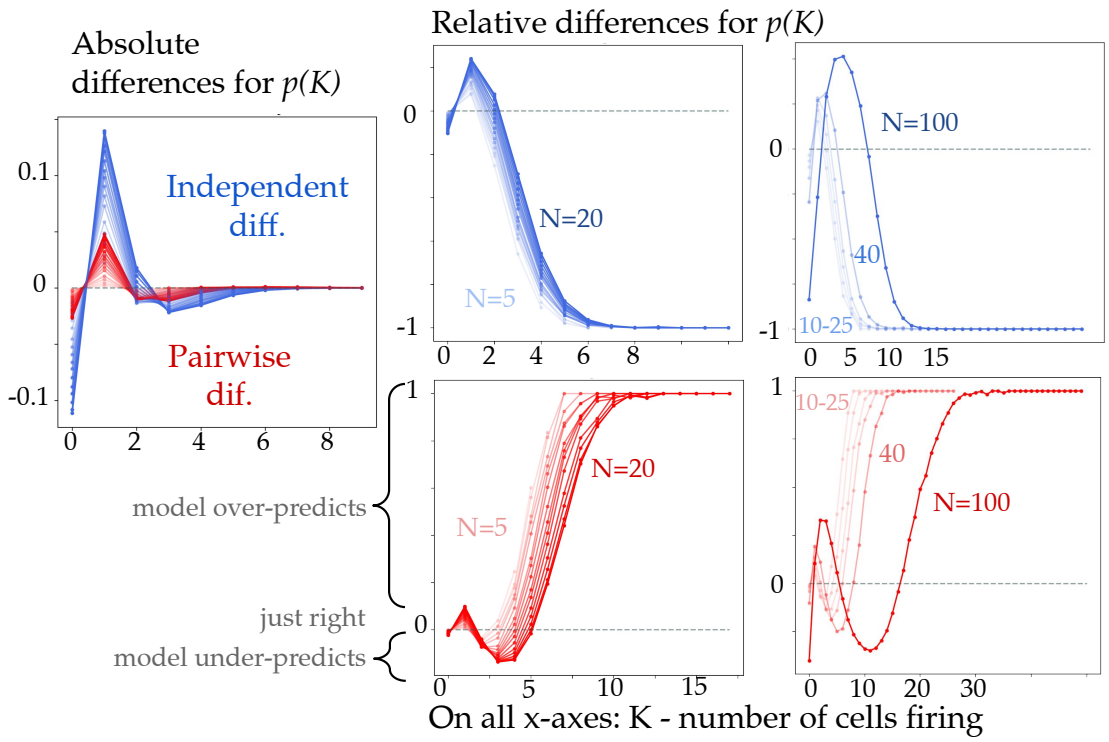


Figure 4.14: On the left, we plot the mean absolute differences between the predicted and observed $p(K)$ for the pairwise and independent models. The trajectories from lightest to darkest represent $N = 5, ..., 20$, where lighter trajectories are associated with smaller populations and darker trajectories are associated with larger ones. For the relative differences, the middle column of plots shows results for $N = 5, .., 20$ and the last column of plots zooms out and shows results for $N = 10, 15, 20, 25, 40, 100$. Note that we join the points with a line to make it easier to track all the points which are associated with a particular population size and not because we are trying to interpolate results.

between the groups of predictions and the empirical observations. It is only in the $N = 40, 100$ plots that the range of the predictions for each $p(K')$ for the pairwise model become clearly separated from the range of the observed values and there is evidence to doubt that the local fields and pairwise interactions can completely characterise the collective activity of retinal ganglion cells.

While both the independent and pairwise models have shortcomings, they are certainly not equally bad and the plot of the absolute differences illustrates the extent to which the pairwise model improves upon the independent model. The mean absolute differences plot in Figure 4.14, which illustrates the mean differences between the model and observed population count distributions in sub-populations ranging from N=5 to N=20, already reveals consistent differences between the model and observed

distributions across the range of sub-populations, both over-predicting the probability of a single cell firing and under-predicting the probability of intermediate numbers of cells firing. However, the absolute differences fail to reveal the behaviour of the models at medium and large K since the absolute differences trivially become small as K increases.

We then turn our attention to the mean relative differences plots which illustrate the mean difference between the predicted and the observed $p(K)$ relative to the sum of their means at each K. These plots clarify the pattern of the independent and pairwise model over and under-predicting and also indicate that these patterns of shortcomings are consistent at many different scales from $N = 5$ to $N = 100$. The independent relative differences tend to -1 whereas the pairwise relative differences tend to +1 which is in line with what we saw in the logarithmic plots of $p(K)$ in Figure 4.13, where we observed heavier tails for the pairwise model and lighter tails for the independent model compared to the empirical population count distribution.

These results suggest the following. A model that treats the firing of cells as independent under-predicts the probability of silence, over-predicts the probability of a few cells firing, but under-predicts the probability of large groups of cells firing. We know that the independent model reproduces the mean firing rates of cells. Thus, the fact that we observe these differences implies that the cells must be firing together more often, which is indeed what we see in the raw data in Figure 4.2. In a model which additionally takes the activity of all pairs of cells into account, we still observe that the probability of silence is under-predicted, the probability of a few cells firing is over-predicted and the probability of larger groups of cells firing is under-predicted. We refer to these larger groups as 'groups of intermediate size'.

Compared to what is predicted by the pairwise model, the actual data sees very large groups of cells firing less frequently, which suggest that when cells do fire in the actual data, they are more likely to fire together in groups of intermediate size. It is also worth highlighting that we observe these differences consistently across the range of populations sizes, N. Specifically, the local minimum in the pairwise relative differences plot shifts towards larger values of K as N increases, suggesting that the pairwise model under-predicts the probability of increasingly larger *intermediate* numbers of cells firing, and the local maximums in both the independent and pairwise plots similarly shift towards larger values of K as N increases suggesting that these models over-predict the probability of increasingly larger *small* numbers of cells firing.

Thus far, our discussion has revolved around our own results, but we also need to re-

late these results to what has been observed in previous studies. We have highlighted aspects of the population count distribution that the independent and pairwise model fall short in capturing, which begs the question whether these aspects were overlooked in early works, particularly for the pairwise model, and how we might include additional features in our models which remedy these shortcomings.

In Section 3.5, we summarised how well models in previous works captured the population count distribution, $p(K)$, and we can now compare these findings to our own results. For the independent model, it was noted that its population count distribution follows a Poisson binomial distribution with probability mass function,

$$p(K) = \sum_{A \in F_K} \prod_{i \in A} p_i \prod_{j \in A^c} (1 - p_j),$$

but that this distribution misses out on various features of the observed population count distribution. Specifically, it predicts that the population count distribution has much lighter tails than what is observed (Schneidman, Berry, et al., 2006; Tkačik, Schneidman, et al., 2009; Tkačik, Olivier Marre, Amodei, et al., 2014). This is indeed what we observe in our results, where the independent model predicts $p(K)$ to be several orders of magnitude lower than the experimentally observed values. For instance, in Figure 4.13 the independent model predicts the probability of nine cells firing in a population of forty RGCs to be around 10^{-5}, whereas it was observed to be around 10^{-3}. These results are not particularly surprising, as one would not intuitively expect a model that treats all cells as independent to be realistic. Rather, the usefulness of the independent model is as an intuitive baseline model. Later, when we examine the distances between the model and the empirical distributions, it becomes useful to scale these distances relative to the independent model's results.

Whereas the picture painted in the literature of the independent model being a poor approximation for the neural data from the start is in line with our results, at N=10, we already observe the pairwise model predicting slightly larger mean values for the population count distribution from $K \geq 5$. This seems to be at odds with previous work (Schneidman, Berry, et al., 2006) which saw it as an accurate approximation at N=10, though we can offer some explanations as to why this may be the case. Firstly, it is worth noting that, especially for small K, the range of the predicted $p(K)$s produced by our model overlaps with the observed values of $p(K)$s in Figure 4.14, thus differences between these distributions may have been dismissed as "being within experimental error". It is also worth noting that the metrics that were used in this early work, such as the frequency of common states, focused on the states that make up the

beginning of the population count distribution – states where only a few cells fire and which occur more regularly – and we do indeed seem to initially have a close correspondence between the trajectories in figure 4.13 for $K = 0, 1, 2, 3, 4$. On the other hand, the relative difference plots in figure 4.14 clarify that even in populations of 10 cells, although there are small differences between the pairwise and empirical population count distributions, these are relatively noticeable differences such as the local maximum at K=1 and minimum at K=4 which are noticeably above and below, respectively, the line representing equality.

What emerges from this discussion is that the pairwise model, which includes up to second-order correlations, does not completely capture the shape of the population count distribution, and through focusing on the relative differences, we can observe this departure in populations as small as N=10. This does beg the question of which order correlations do capture the empirical population count distribution. In theory, it would be nice to continue fitting MaxEnt models of all orders to the data, and note the order of model which does finally capture features of the data such as the population count distribution. This is in a similar vein to what we did in Figure 4.8, where we examined how much variance in our data-set was captured in varying numbers of principal components.

4.4.2. Third-order models

Though it is computationally infeasible to include third-order correlations in models of $N = 40+$ neurons with the available compute, given that looking at the relative difference seems to be a good visual tool for assessing how close models come to capturing the population count distribution, we fit third-order models to different populations RGCs for $N \in \{5, 6, ..., 20\}$. The third-order model is simply the log-linear model which includes up to third-order interaction terms (see Section 3.4.1 for more on the log-linear model). We already have results for the independent and pairwise models over this range of population sizes, which we can use as comparison. Including a third-order model in our analysis gives us a sense of how the shapes of the predicted population count distributions vary as we vary the order of the correlations in our MaxEnt model, which is very much in line with the analysis proposed by Martignon et al., 1995.

We train $N \times 20$ third-order models on $B = 20$ randomly sampled sub-populations of $N \in \{5, 6, .., 20\}$ cells until they reproduce the observed averages, $\langle \sigma_i \rangle_D$, pairwise correlations, $\langle \sigma_i \sigma_j \rangle_D$ and third-order correlations, $\langle \sigma_i \sigma_j \sigma_k \rangle_D$. Besides the computational cost associated with fitting third-order models, which we have side-stepped by focusing on

relatively small population sizes, we may be concerned that the third-order correlations that we us to fit the third-order models are not well-approximated in the finite data we have. Since we now fit our model to increasingly many potentially unstable expectations, we might expect that the cumulative effect of this is a model which does not generalise very well to modelling activity outside of the data-set it is trained on.

4.4.3. Assessing overfitting

We have to keep in mind that the variation of the mean activity calculated using *random times but across repeats* of our data-set is typically larger than the variation of the mean activity calculated using *random repeats but across time* of our data-set. Thus we take a more limited interpretation of generalisation and look at whether models trained on a random sample of repeats from our data-set will perform similarly on a different sample of repeats. We adopt the tactic used by Tkačik, Olivier Marre, Amodei, et al., 2014 where they randomly chose a proportion of the repeats in the data to train models and then compared the log-likelihood of this training set to the log-likelihood of the remaining repeats.

Though we have already introduced the log-likelihood in Equation 3.4, we recap its definition here. The likelihood refers to the joint-probability distribution $p(\boldsymbol{\sigma}^{(1)}, ..., \boldsymbol{\sigma}^{(|D|)}|\boldsymbol{\lambda})$ as a function of the parameters of the distribution, $\boldsymbol{\lambda}$, for a set of observations $\{\boldsymbol{\sigma}^{(\mu)}\}_{\mu=1}^{|D|}$. Under the assumption that the observations are independent, the log-likelihood can be expressed as

$$L_D = \frac{1}{|D|} \sum_{\mu=1}^{|D|} \ln p(\boldsymbol{\sigma}^{(\mu)}|\boldsymbol{\lambda}).$$

For each model, we compute the probabilities of the states in its training set, and the probabilities of the states in a withheld test set. By taking the empirical mean of the logarithms of these probabilities, we arrive at the log-likelihoods for the respective data-sets. We can then look at the difference between the log-likelihood of the training set and the log-likelihood of the test set. We do this for the independent, pairwise and third-order models trained on $B{=}20$ randomly sampled populations of size $N{\in}\{5, 6, ..., 20\}$. These results are shown in Figure 4.15.

The median value of the difference in log-likelihoods for the third-order model is consistently just above zero, especially for larger populations, suggesting that it may slightly over-fit, but only very slightly. The range of differences that we observe for the third-order model is largely the same as those for the independent and pairwise models and

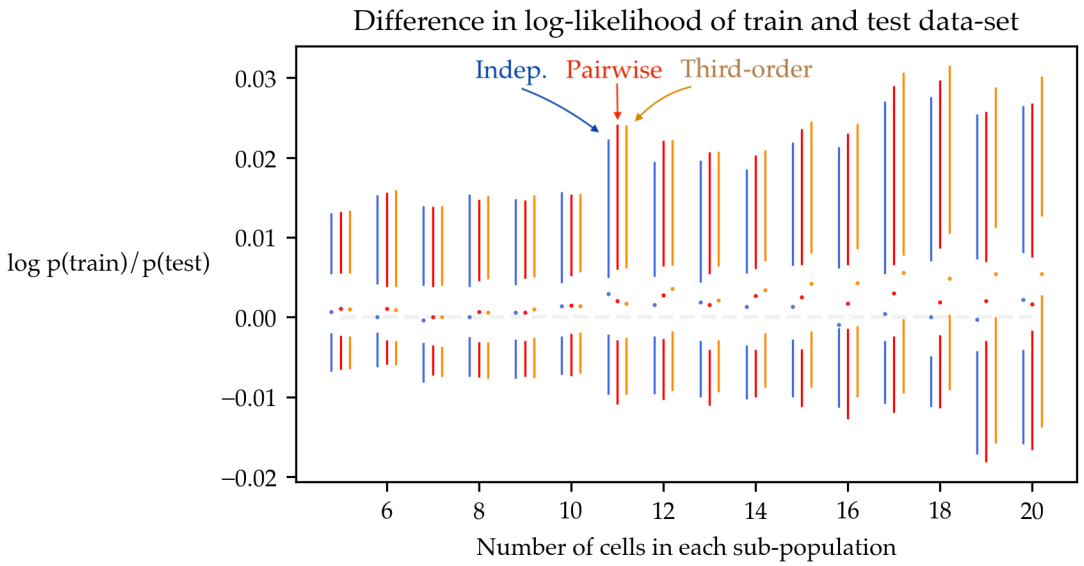


Figure 4.15: Five number summaries of the differences between the log-likelihoods of the training and test sets for the independent, pairwise and third-order models. We break from our previous convention and represent the median as a dot due to spatial constraints. For context, the absolute value of all the log-likelihoods was greater than 0.536, thus the fact that the differences that we observe are at most 0.03 suggests that the performance was very similar across the training and test sets.

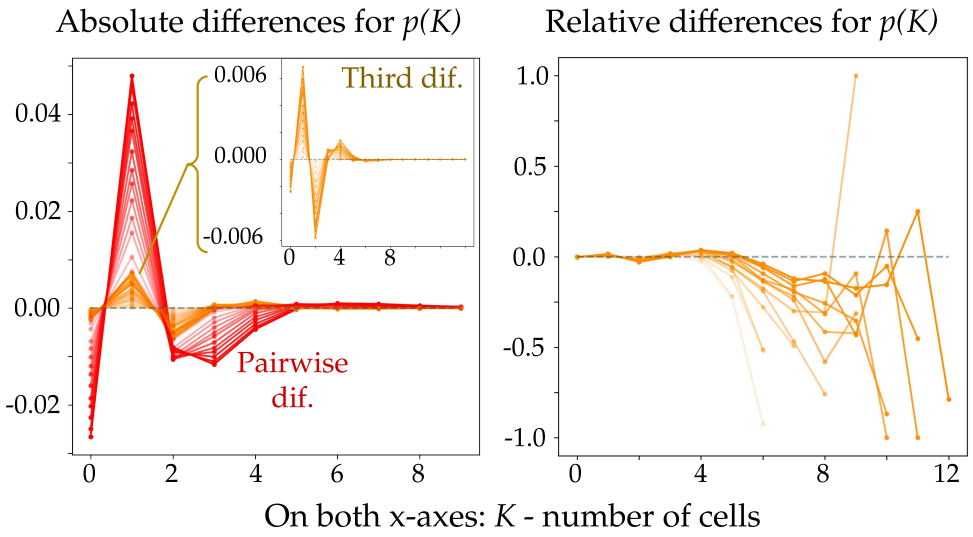


Figure 4.16: On the left, we plot the mean absolute differences between the predicted and observed $p(K)$ for the pairwise and third-order models. The trajectories from lightest to darkest represent $N = 5, 6, 7, ..., 20$. In the inset, we isolate the differences for the third-order model. The relative differences are presented on the right plot and we observe a lot of variation in the trajectories towards the right.

overall, the differences that we observe (whose absolute values are at *most* 0.03) are very small relative to the actual scale of the log-likelihoods (whose absolute values are at *least* 0.536). These results suggest that none of the models over-fit significantly.

We can now begin to interpret the extent to which the third-order model reproduces the population count distribution $p(K)$. We start by presenting the mean differences between the observed values of $p(K)$ and those predicted by the third-order model in Figure 4.16. In the plot of the relative differences on the right of Figure 4.16, we now observe that the differences no longer follow neat trajectories for $K > 6$. Whereas we previously observed either heavier or lighter tails in the pairwise and independent model, the tail of the third-order model is very close to that of the empirical distribution and the effects of the finite samples used to estimate the increasingly small quantities in the tails of the distributions become apparent. It would seem that the absolute differences plot is more informative in this case. As was the case when going from the independent model to the pairwise model, we again see a significant reduction in the mean absolute differences for the third-order model. At N=20, the differences are largely bounded between positive and negative 0.006. For comparison, the differences for the pairwise model are bounded between positive and negative 0.05. We do still

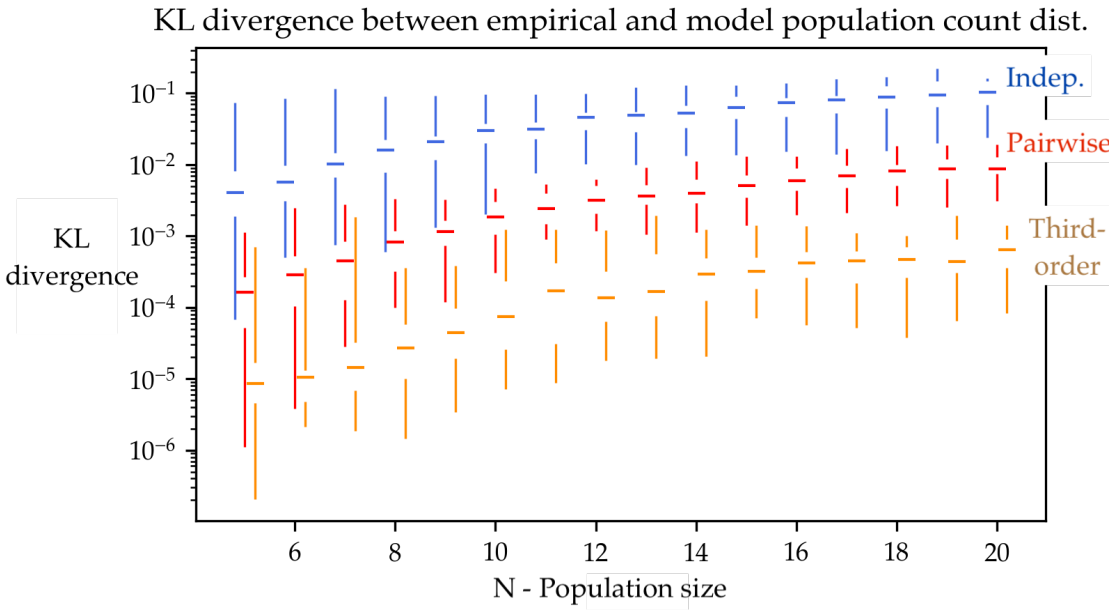


Figure 4.17: Summary of the KL divergences between the empirical population count distributions and those predicted by the independent, pairwise and third-order MaxEnt model. At each N, we randomly sample 20 populations of N cells. We compare the observed and model population count distributions for each of these, hence we have 20 KL divergences at each N. We then present the 5 number summary of these 20 divergences.

observe that the probability of observing states where a single cell fires, $p(K{=}1)$, is over-predicted and that the probability of observing states where only two cells fire, $p(K{=}2)$, is under-predicted, along with other small but consistent discrepancies across different population sizes.

Though we do observe these slight discrepancies, we have to take a step back and ask what the cumulative effect is of these small differences. We can be more precise about how close the third-order model's population count distribution is to the empirical distribution by looking at the KL divergence between these distributions, $D_{KL}(p^{(D)}(K)\|p^{(3)}(K)) \doteq \sum_K p^{(D)}(K)\log(p^{(D)}(K)/p^{(3)}(K))$. If we look at the values of the KL divergence for the third-order model, which are shown in Figure 4.17, we observe that they are all below 10^{-3}, suggesting the third-order model's population count distribution is a fairly accurate approximation to the empirical distribution. Drawing on terminology from information theory, we can say there is very little excess surprise when using the third-order model's population count distribution. We also notice that

the range of the values of the KL divergences between the different models initially overlap, but become well separated from around $N = 14$. This suggests that the behaviour of smaller populations of cells is largely captured in its first two moments and that we do not gain much information from additionally including third-order moments in our models. On the other hand, when we move towards larger populations, the pairwise and third-order model clearly do a better job than the independent and pairwise model, respectively, in explaining the empirical population count distribution.

So far we have been using the population count distribution as a way of measuring whether up to third-order MaxEnt models might be accurate approximations of the true distribution of RGCs. Of course, the population count distribution is only one of many aspects of this true distribution. Given that we are currently working with relatively small populations, a more direct way of assessing the goodness of fit would be to look directly at the KL divergence between the empirical distribution and our model distributions. Again, we estimate the KL divergences over multiple sub-populations for different system sizes, and examine how these estimates scale with system size, N. These results are presented in Figure 4.18.

In these results, we observe that the KL divergences of all three models increase with the system size. For small system sizes $N{\approx}5$, the median KL divergences of the pairwise models are around 10^{-3}, and medians for the third-order models are around 10^{-4}. These results suggest that in small systems, as we include additional low-order moments in our models, such as adding pairwise correlations to go from an independent model to a pairwise model and adding third-order correlations to go from a pairwise model to a third-order model, these models become increasingly accurate estimates of the true distribution. Of course, this is not a particularly surprising, and it leads into the final aspect of this analysis – that the fact that the pairwise and third-order models are good approximations at this scale is trivial.

4.4.4. The perturbative regime

The fact that we observe that the KL divergence between the empirical and pairwise model is small for small system sizes where cells fire infrequently has been addressed by characterising a perturbative regime. We quickly provide the necessary information in order to assess whether the pairwise model is in the perturbative regime. We should note these results are specific to the pairwise model, though we suggest how we might extend them for the third-order model. When we are in the perturbative

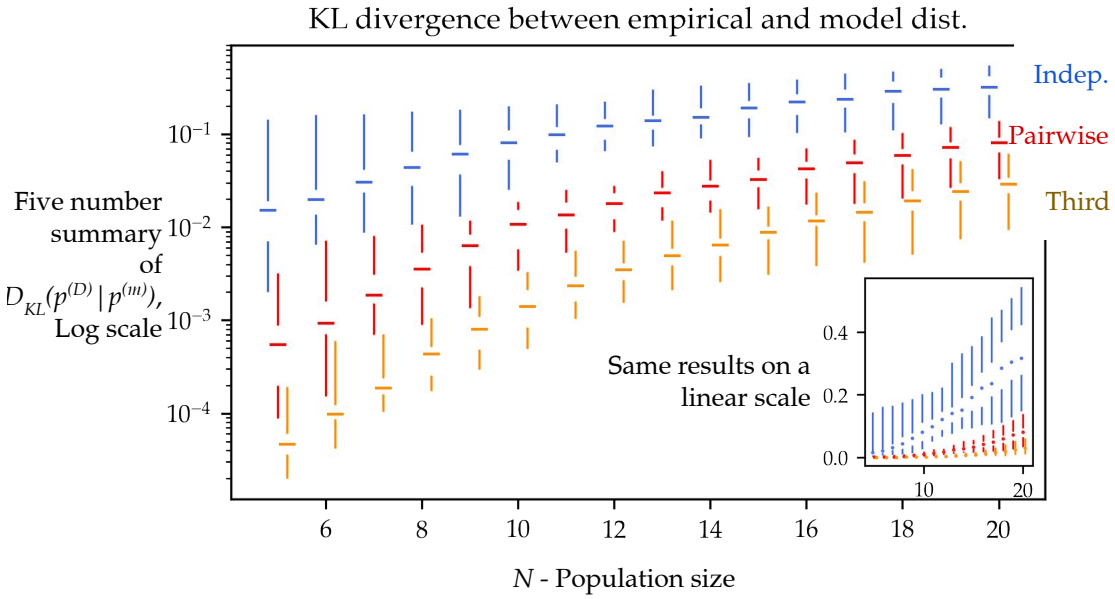


Figure 4.18: Summary of the KL divergences between the empirical distributions and the distributions of the independent, pairwise and third-order MaxEnt model. For clarity, Figure 4.17 looks at the KL divergences between the population count distributions $p(K)$ and not the full distribution $p(\boldsymbol{\sigma})$. At each N, we randomly sample 20 populations of N cells. We compare the empirical and model distributions for each of these, hence we have 20 KL divergences at each N. We then present the 5 number summary of these 20 divergences.

regime, characterised by a small probability of observing a neuron spike δ and a small number of neurons N, the pairwise maximum entropy model will appear to be a good model for any distribution (Roudi, Nirenberg, and P. E. Latham, 2009). However, we cannot extrapolate these results to larger systems and assume that the pairwise model will remain a good fit outside of the perturbative regime. The fact that we can fit pairwise models to relatively small populations of neurons should not come as a particular surprise. We can formally state this result as

$$\Delta_N \doteq \frac{D_{KL}\left(p^{(N)} \| p^{(2)}\right)}{D_{KL}\left(p^{(N)} \| p^{(1)}\right)} \propto (N-2)\delta + \mathcal{O}\left((N\delta)^2\right).$$

When the pairwise model $p^{(2)}$ is very close to the true distribution $p^{(N)}$, Δ_N is close to 0, and when the pairwise model is no better than the independent model $p^{(1)}$, Δ_N is equal to 1. This result tells us that in the perturbative regime, the *distance* between some true probability distribution with arbitrarily high-order interactions and the pairwise model relative to its distance from the independent model appears linear in $N\delta$. We use the word distance informally since the KL divergence is not symmetric $D_{KL}(p\|q) \neq D_{KL}(q\|p)$.

We go through the derivation of this result in detail in the appendix, in Section A.2, though we want to highlight some of the main steps in deriving this result so that it makes more intuitive sense, or at least give the reader an appreciation for the effort it takes to derive this seemingly simple result. The first step involves defining the Sarmanov-Lancaster expansion (Sarmanov, 1962; Lancaster, 1958; Lancaster, 1963)

$$p(\boldsymbol{\sigma}) = p^{(1)}(\boldsymbol{\sigma})\left(1 + \xi_p(\boldsymbol{\sigma})\right), \quad \xi_p(\boldsymbol{\sigma}) = \sum_{i<j} \delta\sigma_i \delta\sigma_j \mathcal{J}^p_{ij} + \sum_{i<j<k} \delta\sigma_i \delta\sigma_j \delta\sigma_k \mathcal{K}^p_{ijk} + ...$$

where $\delta\sigma_i = \sigma_i - \langle\sigma_i\rangle$, and the terms $\mathcal{J}^p_{ij}$, $\mathcal{K}^p_{ijk}$ relate to the second and third-order correlations of the distribution. This expansion relates a distribution to the independent model, plus a series of corrections relating to its higher order correlations. Using the Sarmanov-Lancaster expansions of two distributions p and q, we can express the KL divergence between these two distributions as a Taylor expansion

$$D_{KL}(p\|q) = \frac{1}{\ln 2} \sum_{m+n\geq 2} a_{mn} \langle \xi_p(\boldsymbol{\sigma})^m \xi_q(\boldsymbol{\sigma})^n \rangle_{p^{(1)}},$$

where the terms a_{mn} are used to encapsulate the partial derivatives and constants in each term of the expansion. At this point, we perform multinomial expansions of each of the terms ξ_p^m and ξ_p^n in this expansion up to order $\mathcal{O}((N\delta)^4)$. Importantly, it is the multi-nomial expansions of the terms in the Taylor expansion that we truncate up to

order $\mathcal{O}((N\delta)^4)$, and not the Taylor expansion itself. We do this because this is the order at which we can examine the leading order of behaviour not captured by pairwise models. This is also the point at which we could adapt this analysis to the third-order model by performing multi-nomial expansions of ξ_p^m and ξ_p^n up to order $\mathcal{O}((N\delta)^5)$, though we leave this for future research when we have the time to keep track of the increasingly many terms that arise in these expansions.

Once we derive a general expression of the KL divergence between two distributions, where each term in the expansion has been truncated at order $\mathcal{O}((N\delta)^4)$, we can consider the specific case where the KL divergence is between some distribution with arbitrarily high-order correlations $p^{(N)}$ and the independent model $p^{(1)}$, and when it is between $p^{(N)}$ and the pairwise model $p^{(2)}$. This leads to the following approximations of the KL divergences:

$$D_{KL}\left(p^{(N)}\|p^{(1)}\right) = \frac{1}{\ln 2}\sum_{i<j}\bar{\sigma}_i\bar{\sigma}_j f\left(\rho_{ij}^{(N)}, 0\right) + \mathcal{O}\left((N\delta)^3\right), \tag{4.2}$$

$$D_{KL}\left(p^{(N)}\|p^{(2)}\right) = \frac{1}{\ln 2}\sum_{i<j<k}\bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k f\left(\bar{\rho}_{ijk}^{(N)}, \bar{\rho}_{ijk}^{(2)}\right) + \mathcal{O}\left((N\delta)^4\right), \tag{4.3}$$

where ρ_{ij}^p is the normalised correlation coefficient defined as

$$\rho_{ij}^p \doteq \frac{\langle\sigma_i\sigma_j\rangle_p - \bar{\sigma}_i\bar{\sigma}_j}{\bar{\sigma}_i\bar{\sigma}_j}, \quad \rho_{i_1 i_2 \ldots i_k}^p \doteq \frac{\langle(\sigma_{i_1} - \bar{\sigma}_{i_1})(\sigma_{i_2} - \bar{\sigma}_{i_2})\ldots(\sigma_{i_k} - \bar{\sigma}_{i_k})\rangle_p}{\bar{\sigma}_{i_1}\bar{\sigma}_{i_2}\ldots\bar{\sigma}_{i_k}},$$

and where we use the superscript p to specify which distribution the expectations are calculated over. We have also made use of the more compact notation $\bar{\sigma}_i \doteq \langle\sigma_i\rangle_p$. $\bar{\rho}_{ijk}^p$ is defined in terms of the third-order normalised correlation coeffient as

$$\bar{\rho}_{ijk}^p \doteq \rho_{ijk}^p + \rho_{ij}^p + \rho_{ik}^p + \rho_{jk}^p = \frac{\langle\sigma_i\sigma_j\sigma_k\rangle_p - \bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k}{\bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k},$$

and the function f is defined as

$$f(x, y) \doteq (1 + x)\big[\ln(1 + x) - \ln(1 + y)\big] - (x - y).$$

Upon taking the ratio of Equation 4.2 and Equation 4.3, and identifying the summations as scaling with the system size N, and $\bar{\sigma}_i$ as scaling with δ, we return to the stated result:

$$\Delta_N \doteq \frac{D_{KL}\left(p^{(N)}\|p^{(2)}\right)}{D_{KL}\left(p^{(N)}\|p^{(1)}\right)} \propto (N-2)\delta + \mathcal{O}\left((N\delta)^2\right).$$

For fixed δ, if we observe that Δ_N scales linearly with N and $N\delta \ll 1$, then the fact that we observe a small KL divergence between the pairwise model and some true

distribution is trivial because the first two moments of the distribution will necessarily capture most of the activity. In a sense, there is not sufficient scale to observe the effects of higher-order correlations. What would be significant is if we saw Δ_N saturating when we move to sufficiently large scales of N for fixed δ.

Roudi et al. present experimental results to validate these findings where they examine the fit of the pairwise model to a third-order model at $N = 5, ..., , 10$. The parameters of the third-order model were randomly initialised, and importantly, the third-order model contained non-zero third-order interactions parameters which cannot be decomposed into pairwise interactions. Hence, the third-order model presents a distribution that the pairwise model cannot fully model. They compared their theoretical results to the actual ratio of KL divergences between the third-order and pairwise and the third-order and independent model for a range of δs and found that the theoretical predictions were in good agreement with the measured ratio of KL divergences over $N = 5, ..., 10$ (Roudi, Nirenberg, and P. E. Latham, 2009).

We examine whether what we observe is also in line with these theoretical results. From the exploratory data analysis notebook, we know that the average probability of observing a neuron fire is $\delta = 0.0356$. Thus for $N \in \{5, 6, ..., 20\}$, $\delta N \leq 0.712$. We already have estimates for how the KL divergences between the empirical distribution and the independent, pairwise and third-order model scales with N in Figure 4.18. What remains is to look at the KL divergence of the pairwise model relative to that of the independent model. We present this result in Figure 4.19.

We refer to $D_{KL}\left(p^{(D)} \| p^{(m)}\right) / D_{KL}\left(p^{(D)} \| p^{(1)}\right)$ as the relative KL divergence. For the pairwise model, the relative KL divergence is simply Δ_N. In Figure 4.19, we observe that the relative KL divergence scales linearly with N for both the pairwise model and the third-order model. If we take the mean of the relative KL divergences at each N, $\langle D_{KL}\left(p^{(D)} \| p^{(m)}\right) / D_{KL}\left(p^{(D)} \| p^{(1)}\right) \rangle_B$ and fit a simple linear regression model using N as the explanatory variable and and $\langle D_{KL}\left(p^{(D)} \| p^{(m)}\right) / D_{KL}\left(p^{(D)} \| p^{(1)}\right) \rangle_B$ as the response, we observe an almost perfect linear relationship. The R^2 value, which measures the proportion of the variation in the response that is predictable from the explanatory variable and takes a value between 0 and 1, is 0.99 for the relative divergence of the pairwise model and 0.924 for the third-order model. These results strongly suggest that the arguments about the pairwise model being in the perturbative regime for small $N\delta$ apply to this experimental data.

Though we have not derived results for the perturbative regime for the third-order model, we can also take the scaling of the relative KL divergence at face value. As

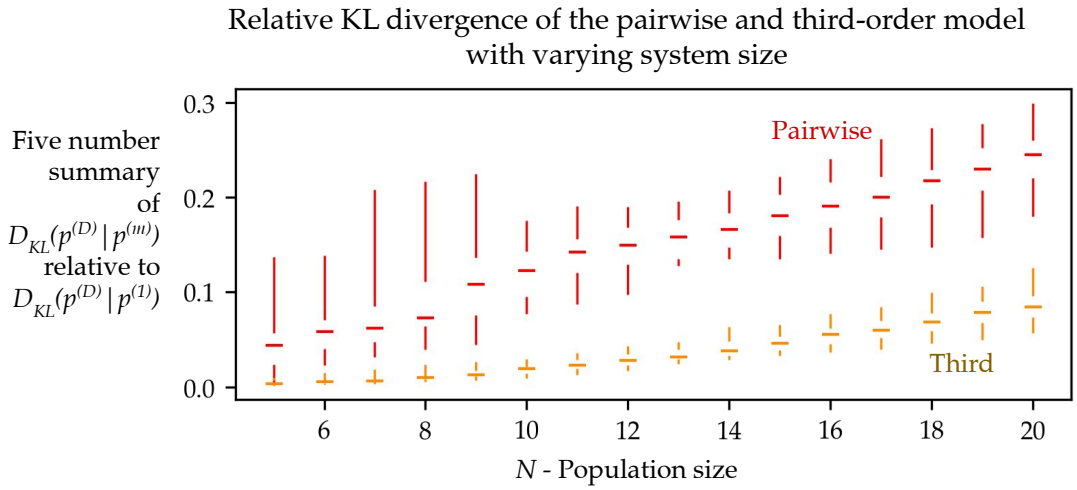


Figure 4.19: The KL divergences of the pairwise and third-order model relative to that of the independent model, $D_{KL}\left(p^{(D)}\|p^{(m)}\right)/D_{KL}\left(p^{(D)}\|p^{(1)}\right)$ where $m\in\{2,3\}$. These results were obtained from fitting models to 20 sub-populations at each N and calculating the relative KL divergences for each. Thus the spread of the results reflects how much these results vary depending on the sub-population.

we move towards larger systems sizes, the third-order model also becomes a worse approximation, and there is no hint that the relative KL divergence saturates over $N\in\{5,6,...,20\}$. We can tie this observation in with our earlier result that looked at the absolute differences in the population count distribution in Figure 4.16. Here we observed that even though the third-order model's population count distribution is close to the empirical distribution, there are small but consistent discrepancies, seen in the inset in Figure 4.16, between these distributions. Drawing on these empirical results, we postulate that although third-order models are close approximations to the empirical distribution at small N and for fixed δ, they also do not fundamentally capture the collective behaviour and the fact that they are a close approximation at this scale similarly reflects that in small populations where cells fire infrequently, the low-order moments reflect the majority of the activity. In future work, we hope to extend the perturbative regime results to third-order models as a means of theoretically validating these observations.

To summarise, in this section we started by examining how well the independent and pairwise MaxEnt models reproduce the empirical population count distribution. We observed that these models under-predicted silence and the probability of intermediate numbers of cells firing, and that these shortcomings became more pronounced as

we looked at larger populations. From here, we examined whether we could better summarise the distribution through including third-order correlations in our MaxEnt models. We first assessed whether these models might overfit to their training data-set by examining the difference in the log-likelihood of the training and test data-sets. The ranges of the differences were largely the same across the independent, pairwise and third-order models and were close to 0, especially relative to the typical size of the log-likelihoods. This came with the caveat that we assess generalisation across repeats in our data-set. Though the distribution associated with the third-order model is very close to the true distribution as measured by the KL-divergence, we observe that, similarly to the pairwise model, the relative KL-divergences Δ_N scales linearly with the population size N for fixed δ and shows no sign of saturating over $N \in \{5, 6, ..., 20\}$.

This represents the current limits of decomposing activity into interaction parameters of increasingly higher orders. The third-order models are computationally intensive to train and necessarily have to be fit to these relatively small populations. Although we can say that the activity of these small populations is largely summarised by their low-order moments, we cannot extrapolate these results to large populations. To our knowledge, this is the first work that empirically illustrates the limits of using MaxEnt models with low-order moments to model the activity of RGCs.

How can we use these insights to inform better models of RGC activity? Going beyond pairwise models, some authors have adopted K-pairwise models which additionally include the whole population count distribution as a constraint (Tkačik, Mora, et al., 2015; Berry II and Tkačik, 2020). In our description of the data-set in Section 4.2, we showed that the probability of viewing many cells firing within a state is increasingly rare, and that the probabilities of the states in the tail of the empirical population count distribution are estimated to be exactly zero because we simply do not observe these states in finite recordings of data. Instead of including the whole population count distribution, we believe a more interesting avenue of research would be to look at the effect of including just a few population count constraints, which are well-sampled, such as the probability of silence. As we noted, the silent state $00...0$ occurs frequently in experimental data and hence the probability of silence is a well-estimated feature even in the activity of large populations of RGCs. The benefit of only including this single additional constraint is that we can still use the rest of the population count distribution to assess the model. The probability of silence has been used as a constraint in a modified version of the independent model (Shimazaki et al., 2015), but we are unaware of a paper which focuses on the pairwise model with this additional constraint. We leave this direction for future work.

4.5. Compressive autoencoders

Lastly, as the beginning of an avenue for further research, we present our results from fitting autoencoders to the activity of 100 RGCs. These autoencoders map each state $\boldsymbol{\sigma}$ to a latent state $\boldsymbol{h}$ before trying to reconstruct the input. In the vanilla autoencoder we use continuous latent states $\boldsymbol{h} \in \mathbb{R}^{N^h}$ and in the compressive autoencoder we use discrete latent states $\boldsymbol{h} \in \{0,1\}^{N^h}$.

We start by presenting the results from hyper-parameter tuning where we have varied the width and depth of our network, as well as the number of units in the hidden layer of our network. We first introduce the results for the vanilla autoencoder which are represented visually in Figure 4.20. In the top left plot in Figure 4.20, we see that the validation loss, defined by the mean euclidean distance between the input and reconstruction, drops significantly when going from 2 hidden units to 18 hidden units, and then only slightly decreases from 18 through to 50 hidden units. This suggests that there are diminishing returns with having too many hidden units. This is echoed in the loss trajectories plot to the right where the loss trajectories of best performing models plateau at similar low values for both 34 and 50 hidden units.

It would seem that for a bottleneck of 2 hidden units, depth becomes more important. Similarly, wider networks seem to perform better. The models that achieved lower validation losses had 4 layers in the encoder. As we move towards higher mean layer widths, the colour of the points becomes darker suggesting wider networks have lower validation losses. On the other hand, for $N^h = 18, 34$, the models that achieved lower validation losses had only 2 layers in the encoder. There seems to be little connection between the mean width of the network and the validation loss for larger numbers of hidden units. These results suggest that for the vanilla autoencoder, depth and width are more important when we only have two hidden units, however given more hidden units, shallow networks suffice to reconstruct the output.

We now move onto the compressive autoencoder which includes the additional constraint that the latent representations have to be binary vectors. We again start by summarising the results of hyper-parameter tuning. In the plot of the bottleneck versus the validation loss at the right of Figure 4.21, we observe that the validation losses decrease as we increase the number of hidden units, which is as we would expect. Again, there seems to be diminishing returns with including more hidden units if we compare the decrease in validation loss from 20 to 40 units to the decrease from 60 to 80 units. In the plot of the loss trajectories on the left of Figure 4.21 we similarly observe architectures

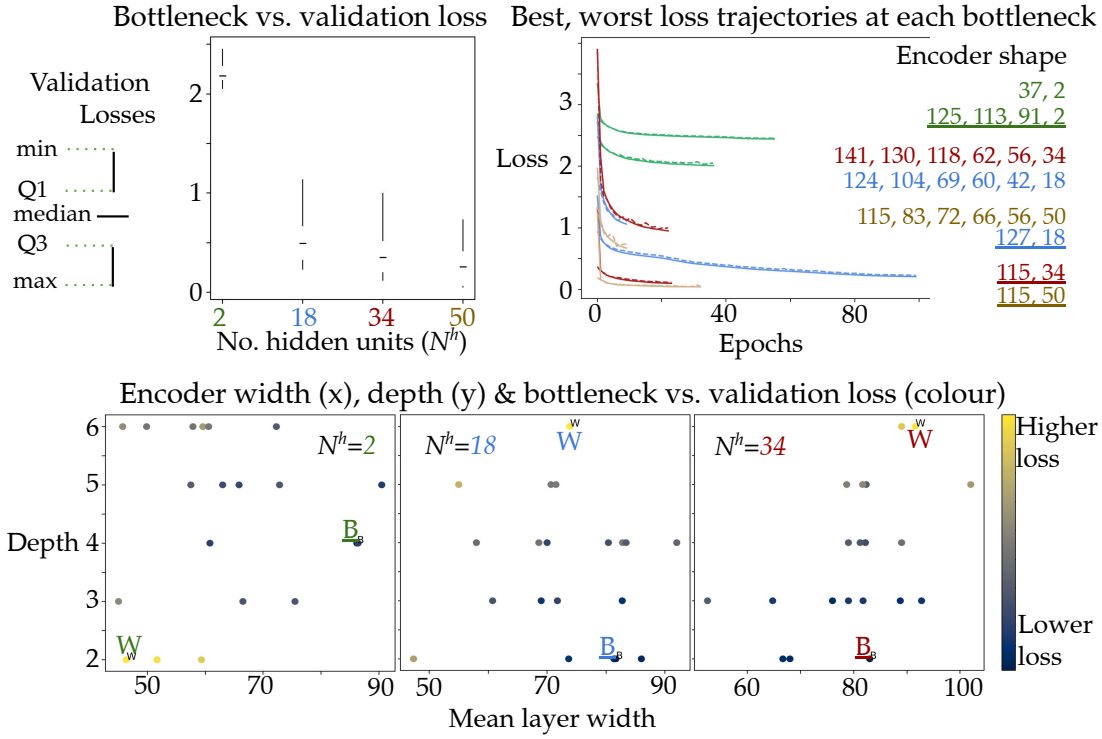


Figure 4.20: Hyper-parameter tuning results for the vanilla autoencoder, involving different sizes of bottleneck (i.e. the number of hidden units), model depths (i.e. the number of layers) and units per layer. For each number of hidden units, we randomly choose 20 different architectures. We assume that the shape of the encoder mirrors the decoder, thus it suffices to define the shape of the encoder. We randomly select the number of layers from $[2, 6]$, and the number of units per layer from $[N^h, \lceil N \cdot 3/2 \rceil]$. In the top left plot, we summarise the validation loss, the mean euclidean distance, over the 20 architectures for each number of hidden units. On the top right, we plot the trajectory of the loss over the training epochs for the best and worst architecture for each number of hidden units. The solid line represents the training loss and the dashed line represents the validation loss. To the right of this plot, we write the corresponding shape of the encoder for each trajectory. The colour of the trajectory and shape is associated with the number of hidden units, and we underline the best performing architecture for each number of hidden units. On the bottom row of plots, we plot the depth and mean layer width of each of the encoder architectures we explored. The left plot shows models with a bottleneck of 2 units, the middle plot 18 units and the right plot 34 units. The best and worst performing architectures are annotated with 'B' and 'W', and we underline annotations associated with the best performing models. The better performing models are defined by having lower validation losses and are visually associated with darker blue colours.

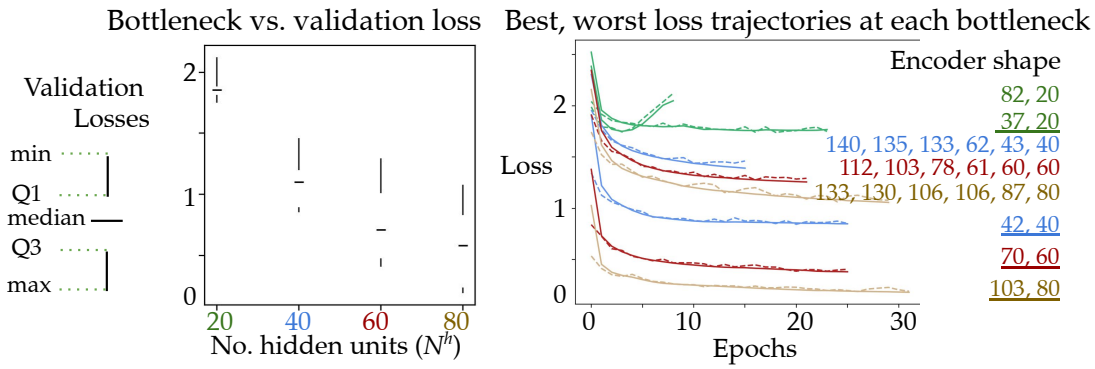


Figure 4.21: Hyper-paramter tuning results for the compressive autoencoder. Again, for each bottleneck (i.e. the number of hidden units), we randomly choose 20 different architectures. On the left, we plot the five number summary for the validation losses of the 20 randomly chosen architectures for each bottleneck. We again use the mean Euclidean distance as the validation loss. On the right, we plot the loss trajectories for the best and worst architecture for each bottleneck, and also supply annotations for the shape of the encoders, which mirrors the shape of the decoders. The solid lines represent the training loss trajectories and the dashed lines represent the validation loss trajectories.

with more hidden units achieving lower losses. Strangely, the loss trajectory of one of models with twenty hidden units starts increasing, suggesting that it begins to learn increasingly worse reconstructions of the data. This is not overfitting since both the validation loss, represented by the dashed line, and the training loss, represented by the solid line increase together.

Though we have plotted both the validation and training loss trajectories, these trajectories seem to be largely the same across all the autoencoders we have considered, perhaps suggesting it is generally hard to overfit in this problem. Aside from this one anomalous trajectory, all other trajectories are seen to plateau suggesting that they converge. Tying in with our observation about the diminishing returns of including more hidden units, we observe that the best performing model with 40 hidden units outperforms the worst performing model with 80 hidden units. This suggests a trade-off between the amount of compression we want to achieve and the fidelity with which we reproduce the inputs.

We now turn our attention to the plot of the validation loss against the depth and width of the encoder in Figure 4.22. In these plots, we consistently observe the best performing models in the lower left corner of the plots suggesting that simpler models perform better. At 40, 60 and 80 hidden units, model performance seems to vary mostly with changes in depth, with deeper models achieving higher validation losses than

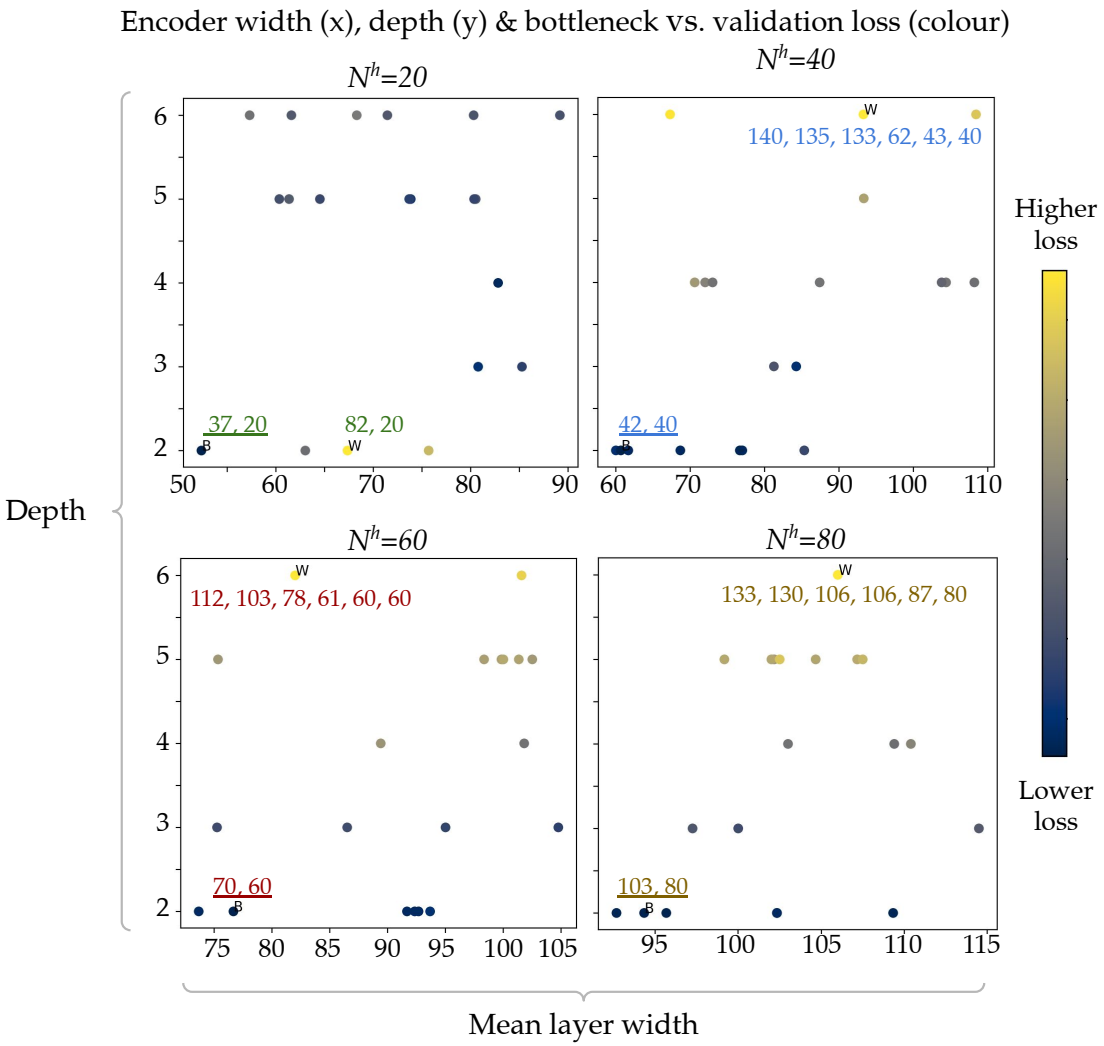


Figure 4.22: Hyper-parameter tuning results for the compressive autoencoder where we examine how depth and mean layer width of the encoder relates to the validation loss. Points correspond to different models and the colour of the points relates to the final validation loss achieved by that model, where darker blue points correspond to lower validation losses and lighter yellow points correspond to higher validation losses. We also supply annotations for the shape of the encoders of the best and worst performing models. If the annotation is underlined, and if there is a small 'B' next to the point, then it was the best performing model.

N^h	Mean Hamming distance	K that Hamming distance > 1
2	2.399	3 (3163 states where 3 cells fire)
18	0.470	9 (1556 states where 9 cells fire)
34	0.107	16 (322 states where 16 cells fire)
50	0.043	20 (73 states where 20 cells fire)

Table 4.1: Hamming distances for vanilla autoencoders with varying numbers of hidden units.

shallower models. On the other hand, at 20 hidden units, we observe both the highest and lowest validation loss from models with only two layers in the encoder.

4.5.1. Quality of the reconstruction

In order to compare the vanilla and compressive autoencoder results we started by identifying the best models for each bottleneck based on which have the lowest mean Euclidean error. We then made use of the Hamming distance to quantify how well the different types of models reconstruct their inputs. The Hamming distance is defined as $\sum_i [\sigma_i \cdot (1 - r_i) + (1 - \sigma_i) \cdot r_i]$, and counts the number of incorrectly predicted digits. For instance, the Hamming distance between (0001) and (0010), is 2.

One aspect of the data-set that we have to be wary of is that states where few cells fire are a lot more probable than states where many cells fire. A model which reconstructs every state as the all silent state $(00...0)$ achieves a mean Hamming distance of 3.87 on the validation data. This means that, out of the 45363 observations in the validation data set, if we consistently predict that all 100 cells are silent, we are on average only incorrectly predicting the states of 3.87 cells. Thus, it makes sense not to only talk about the mean Hamming distance over all states, but the mean Hamming distance over the subset of states where K cells fire. Thus, we report the value of K for which the autoencoder's reconstruction starts to be, on average, wrong by 1 digit. We tabulate these results for the vanilla autoencoder in Table 4.1, and the results for the compressive autoencoder in Table 4.2. For reference, we also include how many states in the validation set have K cells firing. In this data-set, the highest number of cells we observed firing was 26, which we only observed in two states.

We first make some observations on the Hamming distances for the vanilla autoencoder. Whilst the model with 2 hidden units does better than a model that maps everything to the all silent state, it already begins missing one digit on average in states where 3 cells fire. Comparatively, the model with 50 hidden units only starts missing

N^h	Mean Hamming distance	K that Hamming distance > 1
20	2.334	3 (3163 states where 3 cells fire)
40	1.056	6 (2156 states where 6 cells fire)
60	0.477	9 (1556 states where 16 cells fire)
80	0.240	12 (857 states where 20 cells fire)

Table 4.2: Hamming distances for compressive autoencoders.

one digit on average for states where 20 cells fire. However, even with a latent representation of 50 continuous units, we do not always perfectly reconstruct the input.

All the compressive autoencoders also do a better job than a model that maps everything to the all silent state. On one end of the spectrum, the model with 20 hidden units mispredicts, on average, 2.334 digits in each state, and on the other, the model with 20 hidden units mispredicts 0.240 states on average. The The Hamming distances for the compressive autoencoders make the benefits of having continuous latent representations more apparent. With 20 binary hidden units, we observe a mean Hamming distance of 2.334, which is on par with what we achieve with only 2 continuous hidden unit.

As a more visual way of assessing the quality of the reconstructions, we plot an extract of the activity of 100 RGCs, push it through each of our compressive autoencoders, and then plot both the reconstructions, as well as the differences between the reconstructions and the original extract. This is shown in Figure 4.23. Clearly, none of our models perfectly reconstruct all of the states, however, we have also remarked that the activity of RGCs is inherently noisy. See Figure 3.1 for instance where we plotted the differences between the responses on different repeats. So far, we have considered the model that maps all states to the silent state as an admittedly low benchmark. Perhaps we can come up with a better benchmark by looking at how consistently we observe the same state at the same point in time during different repeats. Specifically, we can compare the Hamming distance between states that occurred at the same points in time relative to when stimulus started being presented to the retina, but during different repeats. A quick implementation of this involves pairing up all of the repeats, computing the Hamming distance between pairs of repeats and then taking the average. If we follow this procedure then we arrive at a mean Hamming distance of 6.794. This implies that between pairs of repeats, states at the same moments in time differ on average in 6.794 digits. This is interesting since the mean Hamming distance of the model that mapped

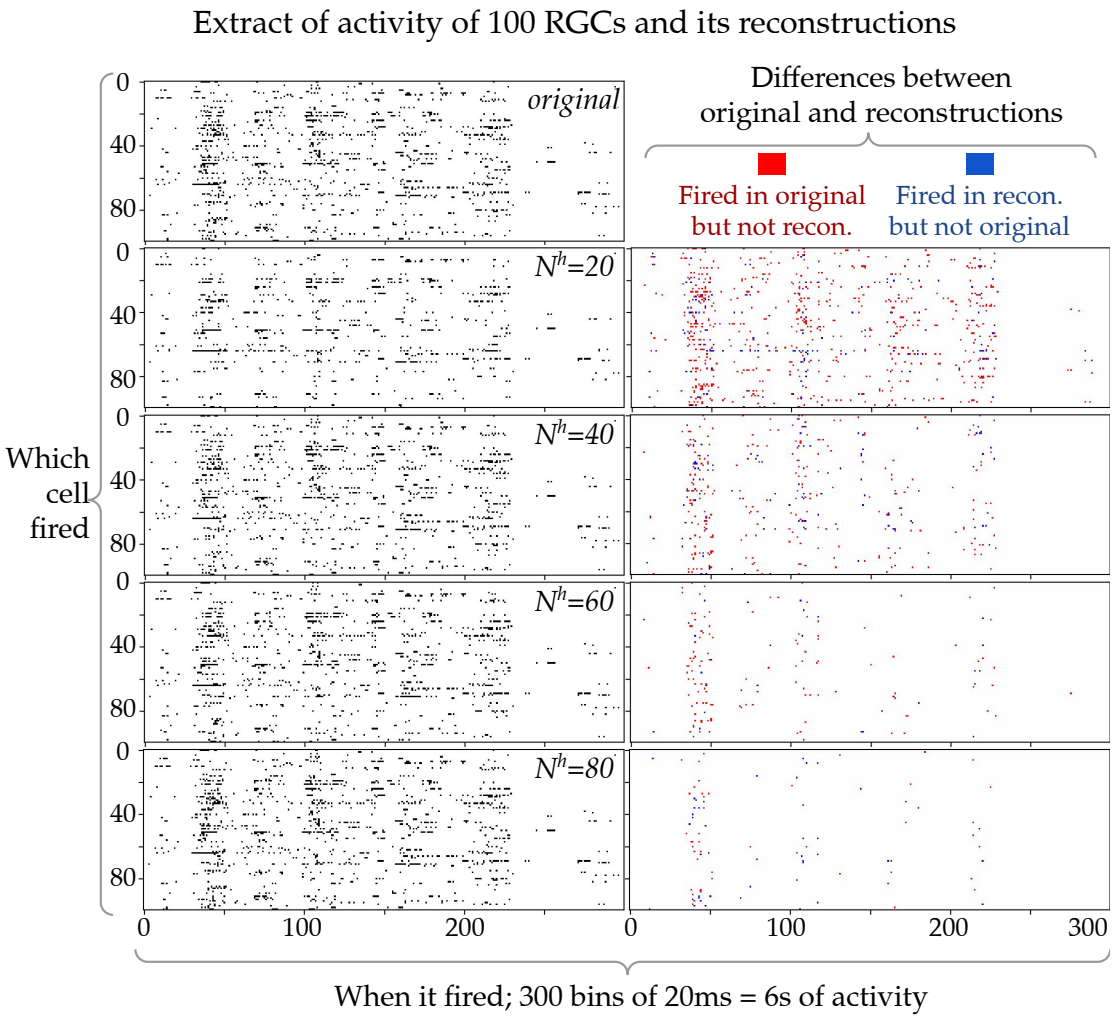


Figure 4.23: We take an extract of the activity of 100 RGCs over 300 bins and plot the reconstructions produced by compressive autoencoders with $N^h \in \{20, 40, 60, 80\}$ hidden units. On the left column, we show the actual reconstructions and on the right column, we highlight which spikes in the original extract were missing from the reconstructions, and which spikes in the reconstructions do not exist in the original extract.

all states to the silent state was a lot lower. This also ties in with our earlier observations about the noisiness of the responses in Section 4.2. Though we have emphasized the quality of the reconstruction, which the Hamming distance is a good metric for, an interesting direction for future work would be to come up with metrics that assess how consistent the latent representations are for states from different repeats of the same stimulus. We might hope that these states, which might look slightly different in their raw representations but arise from the same stimulus, get mapped to similar latent representations.

4.5.2. Future directions

So far, we have largely presented a proof of principle that it is possible to use autoencoders with varying bottlenecks to translate the states $\boldsymbol{\sigma}$ to latent representations $\boldsymbol{h}$, and we have highlighted the depth and width of autoencoders which perform well in this application. We foresee two promising extensions of this work. This first extension has to do with compression of the state space, which we have already begun to explore in our results, and the second has to do with approximating expectations over the original space by sampling from the latent space.

Compressing the state space is interesting because it involves quantifying how much information (in the information theory sense) there is in the observed states. This ties in with the idea that the high-dimensional state space might lie on a lower dimensional manifold. Using vanilla autoencoders, we mapped higher-dimensional binary vectors to lower dimensional continuous vectors, where it was not clear that this simplified the state space since we could trivially encode each binary vector of length N as a single natural number, ($(0000)\rightarrow 0$, $(0001)\rightarrow 1$, $(0010)\rightarrow 2$, etc.). What is more interesting is seeing whether we can compress the high-dimensional binary vectors as lower-dimensional binary vectors and this is the direction we took with the compressive autoencoders where we applied a sigmoid function to the continuous output of the latent layer and then rounded off the output.

This is in line with what is done in lossy image compression (Theis et al., 2017), and in general, learning lower-dimensional discrete latent representations of high-dimensional data is well explored in the image and video compression literature. The process of mapping continuous latent representations to discrete ones relates to the quantization process (Theis et al., 2017). Though we feel additional work needs to done in both developing better compression models for this domain, as well as developing better arguments for how much we should compress the inputs, our work using compressive

autoencoders demonstrates that it can be done. Once we have these simplified binary representation, $\boldsymbol{h}$, of our states, $\boldsymbol{\sigma}$, we could possibly even train MaxEnt models on these representations $\boldsymbol{h}$.

The other interesting direction relates to mapping the state space to latent probability distributions, from which we can then sample. In this direction, we can take inspiration from variational autoencoders, which canonically map the input space to a multi-dimensional Gaussian latent space. By sampling from the latent space and then decoding the samples back into our original space, we effectively have a generative model for RGC data. These generative models could then be used to compute quantities which we cannot compute from finite samples of experimental data. This is helpful if we want to explore MaxEnt models that are constrained to reproduce quantities that rarely arise in experimental data. Though it would be unwise to extrapolate results from these generative models to the actual activity of RGCs, this approach could facilitate further insights into the modelling techniques that we use to study RGCs.

4.5.3. Visualising the latent space

Finally, as food for thought, we present a visualisation of the continuous latent space learnt by an autoencoder with two hidden units. We map each state in the data-set to their latent representation, and then colour each state by its energy, which is the output of the energy function $E(\boldsymbol{\sigma}) = \sum_i h_i \sigma_i + \sum_{i<j} \sigma_i J_{ij} \sigma$ learnt by a pairwise model trained on the same data-set. While Figure 4.24 exhibits a complex geometry in the latent dimensions, it does seem to indicate that there is a measure of clustering of higher and lower energy states. This, in turn, suggests that the autoencoder has learnt a representation that captures salient features of the data.

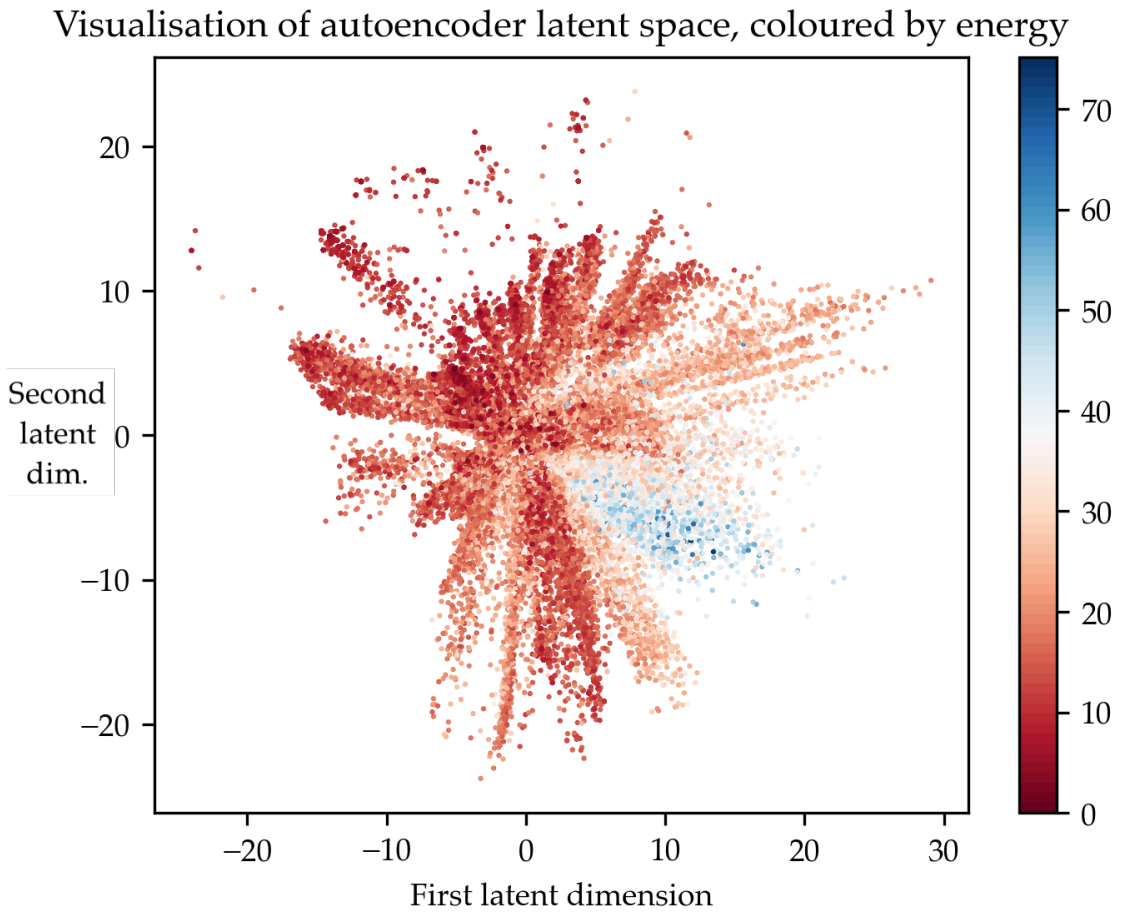


Figure 4.24: Visualisation of the two dimensional latent space learnt by an autoencoder. Each point in this space represents the encoding of a state in our data-set. The colour of each state is the energy predicted by a pairwise MaxEnt model.

5. Conclusions and future directions

In this thesis we explored how to model the activity of retinal ganglion cells using maximum entropy models that are constrained to reproduce time-averaged expectations of the activity, but otherwise make as few assumptions as possible about the distribution of states of the activity. As the signals from retinal ganglion cells feed into the brain via the optic nerve, our internal 'picture' of the visual world starts with the activity of these cells. This work asked how accurately we can model the instantaneous states of this activity using models built on the time-averaged, low-order moments of these states. These models are necessarily probabilistic because the exact states that we observe in the population differ, even we we present the same stimulus. However, these models are conceptually more than just probabilistic models that reproduce certain low-order moments of the data. Their formulation as maximum entropy models means that they are the least bias explanation of the available information (Jaynes, 1957a).

We presented maximum entropy models as arising from a constrained optimisation problem that can be framed using the method of Lagrange multipliers. We then showed that when no non-trivial linear combination of the problem constraints has vanishing fluctuations, the model that satisfies these constraints and that maximises the entropy is unique. Though for some models we can express their parameters analytically in terms of their constraints, others require their parameters to be worked out numerically. We explained how this can be done using gradient ascent, and since the update rule in gradient ascent involves working out expectations over a number of states that are exponential in N, we cover Monte-Carlo methods that can be used to approximate expectations for large N, and how we can recycle samples for multiple updates.

Having provided the mathematical intuitions for maximum entropy models and how to fit them, we then reviewed the types of models used in the literature. Specifically, we focus on introducing the independent, pairwise, population count and K-pairwise models. These models can all be viewed as special cases of the more general log-linear model. The popularity of these models can in part be attributed to the fact that they are trained on statistics that are relatively well-sampled in finite, but sufficiently large

samples of data. We review how the goodness of fit of MaxEnt models has been assessed in the literature and review how well these popular models have been reported to perform against these measures. We largely find that unconstrained variables such as the third-order correlations and the population count distribution have been used to assess goodness of fit.

In our results, we initially focused on the independent and pairwise models, viewing them as two simplifications of the activity, the former being one where cells all fire independently with a mean probability and the latter being one where cells have fixed mean probabilities of firing but are additively more or less likely to fire with other individual cells. Firstly, as a proof of principle, we showed that our implementation of these MaxEnt models produces the same results as an existing implementation used in the literature. Then, using these models and fitting them to data at different scales, ranging from N=10 to N=100, we showed the discrepancies between these simplifications and the real experimental data based on the difference between the model's and the empirically observed population count distribution $p(K)$.

By fitting models regularly to sub-populations ranging from size N=5 to $N = 20$, we were able to observe how these differences vary with scale. This analysis highlighted silence and the probability of intermediate numbers of neurons firing as prominent features of the activity that are not captured by our models. By additionally including third-order models into our analysis we were able to better capture the population count distribution, though there were still noticeable discrepancies. In order to fit third-order models, we necessarily are confined to a regime where $N\delta$ is small. We took this an an opportunity to present empirical evidence that the pairwise model is in the perturbative regime by looking at how the relative KL divergence between the empirical distribution and our models scale. We similarly observe that the relative KL divergence for the third-order model scales with system size.

Given that the third-order model falls short in predicting aspects of the population count distribution, we suggest that although third-order models are close approximations to the empirical distribution, they still do not fundamentally capture the activity. These results highlight the current limits of simplifying the activity of RGCs using low-order MaxEnt models. We could include higher-order moments in our models, but because of the difficulties in estimating higher-order moments from finite data and the computational costs associated with fitting these higher-order models, we recommend either including select aspects of the population count distribution $p(K)$ such as the probability of silence, introducing latent variables (Gardella, Olivier Marre, and

Mora, 2017; Berry II and Tkačik, 2020), or accepting that lower-order maximum entropy models are only approximate and the accuracy of this approximation scales with system size. After all, all models are wrong, though some are useful.

We finished by presenting early results that illustrate the use of autoencoders as a means of finding latent representations of the activity of RGCs. We demonstrated the use of both vanilla autoencoders, which map the activity to continuous-valued latent representations, and compressive autoencoders, which map it to binary-valued latent representations. We were able to show how well the original activity is reconstructed when it is put through bottlenecks of varying sizes, and we were able visualise the latent representations learnt by the autoencoders. This visualisation suggested that our models learn mappings that cluster higher and lower energy states together. We outlined two extensions of this work. This first builds on our work using compressive autoencoders to simplify the state space and involves quantifying the amount of information in the RGC activity. The second extension involves investigating variational autoencoders as generative models, which we can use to train and study the next generation of approaches to modelling the activity of RGCs.

Though we have tried to create a thorough overview of MaxEnt models for populations of RGCs, there are inevitably topics we did not get around to covering and directions hinted by this work that we did not (yet) get the chance to pursue. The main questions that spring out are how we might model temporal dynamics, how we might scale these models to much larger populations of neurons and to what extent our results generalise to other types of stimulus. Given the computational cost of fitting MaxEnt models to the distribution $p(\boldsymbol{\sigma})$, it would be rather unwieldy to use them as is to model $p(\boldsymbol{\sigma}_t, \boldsymbol{\sigma}_{t-1})$, and hence attempts to include temporal dynamics in MaxEnt models have largely been restricted to small populations of neurons or have looked at modelling the population count distribution $p(K)$ as opposed to the distribution over all binary words (see Section 3 of Gardella, Olivier Marre, and Mora, 2019 for a review). We were largely constrained by experimental data that has been made publicly available. If we were able to get more data, it would be interesting to see which pictures emerge from varieties of stimuli, such as other natural movies, or white noise. The data we looked at did not distinguish between the types of RGCs and it would be interesting to see whether models trained on a specific functional type of RGC are better able to capture their behaviour, which would require the identification of the types of the recorded RGCs.

The field of computational neuroscience is inherently interdisciplinary, and it defi-

nitely shows. From involved calculations, such as the perturbative results involving the Sarmanov-Lancaster expansion in Section A.2, to intriguing hypotheses involving deep dives into statistical mechanics, this thesis has demanded an exploration of a range of fields. It has certainly been a tiring but rewarding experience.

A. APPENDIX

A.1. Maximum entropy models

We want to find models of neural activity that reproduce certain observable quantities, but otherwise make as few assumptions as possible about the nature of the activity. A way of achieving this is by maximising the entropy of our model, while imposing certain constraints on it. *Entropy* is a measure of the uncertainty of a system. If our system can take on a number of discrete states with probability p_i, then the entropy is defined as:

$$S = -\sum_i p_i \ln p_i$$

and for continuous probability distributions, the definition of entropy becomes an integral as opposed to a summation over the possible states. A system that takes on all states with equal probability, i.e. follows a uniform distribution, will have maximum entropy, whereas a system that only ever takes on a single state will have an entropy of 0.

In order to maximise the entropy while satisfying certain constraints, we introduce the method of Lagrange multipliers:

A.1.1. Method of Lagrange multipliers

The method of Lagrange multipliers is a strategy for finding the extrema of a function that also has to satisfy certain equations exactly. The Lagrange multiplier theorem states:

> Let $f : \mathbb{R}^N \to \mathbb{R}$ be the objective function and $g : \mathbb{R}^N \to \mathbb{R}^C$ be the constraints function, both of which have continuous first derivatives. If we use $\boldsymbol{x}^*$ to denote an optimal solution to the following optimisation problem
>
> $$\begin{aligned} &\text{Maximise } f(\boldsymbol{x}) \\ &\text{Subject to } g(\boldsymbol{x}) = \mathbf{0}, \end{aligned}$$

where $Dg(\boldsymbol{x})$ denotes the matrix of partial derivatives $\frac{\partial g_j}{\partial x_k}$, then there exists a unique set of Lagrange multipliers $\boldsymbol{\lambda}^* \in \mathbb{R}^C$ such that $Df(\boldsymbol{x}^*) = \boldsymbol{\lambda}^{*\top} Dg(\boldsymbol{x}^*)$.

The set of points that satisfy the constraints are the points for which $g_i(\boldsymbol{x}) = 0$. Every point, $\boldsymbol{x}$ has a space of directions that we can move to from $\boldsymbol{x}$ and still satisfy the constraints, which is the space of directions perpendicular to $\nabla g_i(\boldsymbol{x})$. When we are at a maximum of $f(\boldsymbol{x})$, we should not be able to find any direction in which $f(\boldsymbol{x})$ increases. Therefore, we seek $\boldsymbol{x}$ such that moving in any direction away from $\boldsymbol{x}$ is still perpendicular to $\nabla f(\boldsymbol{x})$ – the direction in which $f(\boldsymbol{x})$ increases. Therefore, we are trying find a point $\boldsymbol{x}$ where the gradient of the objective at $\boldsymbol{x}$ is in the span of the constraints' gradients at $\boldsymbol{x}$:

$$\nabla f(\boldsymbol{x}) = \sum_{i=1}^{C} \lambda_i \nabla g_i(\boldsymbol{x}). \tag{A.1}$$

If we introduce an auxiliary function, called the Lagrangian

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \sum_{i=1}^{C} \lambda_i g_i(\boldsymbol{x}),$$

then finding points that satisfy equation A.1 as well as the C constraints is related to finding the stationary points of the Lagrangian. We do this by taking the partial derivatives of the Lagrangian with respect to x_i and λ_i and finding the points where the partial derivatives are equal to zero.

Here is an example of how this method works in 3 dimensions. Imagine we are in a room with varying temperatures around the room and we are trying to find the point in the room with the maximum temperature. However, we cannot move around the room freely. Instead, we are forced to move along a line that lies at the intersection of two planes. Now, the span of the gradients of the two planes is the plane S orthogonal to the line we are allowed to move along. The gradient of the objective function at a point can be pictured as an arrow pointing in the direction where the temperature increases the most. For instance, imagine we have a 'heat compass' that points in the immediate warmest direction. We move along our line, trying to follow the arrow pointing to where the heat increases as best as we can. The arrow can point in any direction in 3D space and thus might not point entirely forwards or backwards. In the method of Lagrange multipliers, we seek to slide the plane S along the line that satisfies the constraints until the arrow pointing to where the heat increases falls onto the plane S, meaning it is orthogonal to the line that satisfies the constraints. Thus, moving along

the line no longer brings us to a point that is immediately warmer. We would have to abandon the constraint to continue moving in the direction where the heat increases.

A.1.2. Uniform distribution from maximum entropy

As an example, we search for a discrete probability distribution $\boldsymbol{p} \in \mathbb{R}^M$ over M states that maximises the entropy $S = -\sum_{i=1}^{M} p_i \ln p_i$. We denote the probability of state i as p_i. We constrain the p_is by specifying that they must sum to 1, which can be written as a constrain as $\sum_{i=1}^{M} p_i - 1 = 0$. Hence, we have the following Lagrangian:

$$\mathcal{L}(\boldsymbol{p}, \lambda) = -\sum_{i=1}^{M} p_i \ln p_i - \lambda \left(\sum_{i=1}^{M} p_i - 1 \right).$$

Taking the partial derivative with respect to p_j and setting it equal to 0, we obtain:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p_j} &= -\ln p_j - 1 - \lambda = 0 \\ p_j &= e^{-1-\lambda}. \end{aligned}$$

Notice that this implies that all p_js are equal. Then, taking the partial derivative with respect to λ and setting it equal to 0, we recover the constraint:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \lambda} &= \sum_{i=1}^{M} p_i - 1 = 0 \\ p_i &= \frac{1}{M}. \end{aligned}$$

Thus we establish that the probability distribution that maximises the entropy is the one in which every state is equally likely, which corresponds to the uniform distribution.

A.1.3. K-pairwise model from maximum entropy

We now want to find the maximum entropy model consistent with the averages $\langle \sigma_i \rangle$ the pairwise correlations $\langle \sigma_i \sigma_j \rangle$ and the population spike distribution $p(K)$. As above,

we formulate the Lagrangian:

$$\begin{aligned}
\mathcal{L}(p(\boldsymbol{\sigma}), \boldsymbol{h}, \boldsymbol{J}) = &- \sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}) \ln p(\boldsymbol{\sigma}) \\
&- \lambda \left(\sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}) - 1 \right) && \text{Specify the distributio must sum to 1} \\
&- \sum_{i=1}^{N} h_i \left(\sum_{\boldsymbol{\sigma}} \sigma_i p(\boldsymbol{\sigma}) - \langle \sigma_i \rangle_D \right) && \text{Constraint on the averages} \\
&- \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} J_{ij} \left(\sum_{\boldsymbol{\sigma}} \sigma_i \sigma_j p(\boldsymbol{\sigma}) - \langle \sigma_i \sigma_j \rangle_D \right) && \text{Constraint on the correlations} \\
&- \sum_{K} V_K \left(\sum_{\boldsymbol{\sigma}} \delta \left[\sum_{i=1}^{N} \sigma_i - K \right] p(\boldsymbol{\sigma}) - p(K)_D \right). && \text{Constraint on } p(K)
\end{aligned}$$

We use λ, $\boldsymbol{h}$, $\boldsymbol{J}$ and $\boldsymbol{V}$ as the Lagrange multipliers. The summation $\sum_{\boldsymbol{\sigma}}$ is over all possible binary words of length N. We now need to find the stationary points of the Lagrangian, i.e. the points $(p(\boldsymbol{\sigma}), \boldsymbol{h}, \boldsymbol{J}, \boldsymbol{V})$ where the gradient of the objective function is in the span of the constraints' gradients. We take the partial derivative of the Lagrangian with respect to the probability of a particular configuration $p(\boldsymbol{\sigma}')$:

$$\frac{\partial \mathcal{L}}{\partial p(\boldsymbol{\sigma}')} = - \ln p(\boldsymbol{\sigma}') - 1 - \lambda - \sum_{i=1}^{N} h_i \sigma_i - \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} J_{ij} \sigma_i \sigma_j - V_{K(\boldsymbol{\sigma})'} = 0.$$

Here, $K(\boldsymbol{\sigma}')$ is the number of neurons that fire in state $\boldsymbol{\sigma}'$. Rearranging, and absorbing the $e(-1-\lambda)$ into a constant $1/Z$ we obtain the K-pairwise model:

$$p^{(2,K)}(\boldsymbol{\sigma}) = \frac{1}{Z} \exp \left(- \sum_{i} h_i \sigma_i - \sum_{i<j} J_{ij} \sigma_i \sigma_j - V_{K(\boldsymbol{\sigma})} \right).$$

As shorthand we use $\sum_i \doteq \sum_{i=1}^{N}$, $\sum_{i<j} \doteq \sum_{i=1}^{N-1} \sum_{j=i+1}^{N}$. The independent and pairwise models are special cases of this, obtained by ignoring the additional constraints.

A.2. The perturbative regime

Roudi et al. suggest that when we are in the perturbative regime, characterised by a small probability of observing a neuron spike and a small number of neurons, the pairwise maximum entropy model will appear to be a good model for any distribution (Roudi, Nirenberg, and P. E. Latham, 2009). However, we cannot extrapolate these results to larger systems and assume that the pairwise model will remain a good fit

outside of the perturbative regime. The fact that we can fit pairwise models to relatively small populations of neurons should not come as a particular surprise. In short:

> In the perturbative regime, the distance between some true probability distribution with arbitrarily higher order interactions and the pairwise model relative to true distribution's distance from the independent model appears linear in $N\langle v\rangle\delta t \doteq N\delta$,

where N is the number of neurons in the population, $\langle v\rangle$ is the mean firing rate of the neurons and δt is the size of the time bin. As shorthand, we define $\delta \doteq \langle v\rangle\delta t$, and when at most a single neuron fires within each time bin, we can identify δ as the mean probability of observing a neuron spike[1]. We use the Kullback-Leibler divergence $D_{KL}(p\|q) \doteq \sum_i p_i \log_2(p_i/q_i)$ in order to get a sense of how far distribution q is from distribution p. Here we use bits in line with the authors. Thus, we can write their result more formally as

$$\Delta_N \doteq \frac{D_{KL}\left(p^{(N)}\|p^{(2)}\right)}{D_{KL}\left(p^{(N)}\|p^{(1)}\right)} \propto (N-2)\delta + \mathcal{O}\left((N\delta)^2\right). \tag{A.2}$$

When the pairwise model $p^{(2)}$ is very close to the true distribution $p^{(N)}$, Δ_N is close to 0, and when the pairwise model is no better than the independent model $p^{(1)}$, Δ_N is equal to 1.

To derive this result, we have to first derive the following approximations to the KL divergences:

$$D_{KL}\left(p^{(N)}\|p^{(2)}\right) = \frac{1}{\ln 2}\sum_{i<j<k}\bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k f\left(\bar{\rho}^{(N)}_{ijk}, \bar{\rho}^{(2)}_{ijk}\right) + \mathcal{O}\left((N\delta)^4\right) \tag{A.3}$$

and

$$D_{KL}\left(p^{(N)}\|p^{(1)}\right) = \frac{1}{\ln 2}\sum_{i<j}\bar{\sigma}_i\bar{\sigma}_j f\left(\rho^{(N)}_{ij}, 0\right) + \mathcal{O}\left((N\delta)^3\right), \tag{A.4}$$

where $\bar{\sigma}_i = \langle\sigma_i\rangle$ is the expected value of σ_i and

$$f(x,y) \doteq (1+x)\big[\ln(1+x) - \ln(1+y)\big] - (x-y). \tag{A.5}$$

This equation will make more sense when we define it in the context of the KL divergence. ρ^p_{ij} is the normalised correlation coefficient defined as

$$\rho^p_{ij} \doteq \frac{\langle\sigma_i\sigma_j\rangle_p - \bar{\sigma}_i\bar{\sigma}_j}{\bar{\sigma}_i\bar{\sigma}_j}, \quad \rho^p_{i_1 i_2 \ldots i_k} \doteq \frac{\langle(\sigma_{i_1}-\bar{\sigma}_{i_1})(\sigma_{i_2}-\bar{\sigma}_{i_2})\ldots(\sigma_{i_k}-\bar{\sigma}_{i_k})\rangle_p}{\bar{\sigma}_{i_1}\bar{\sigma}_{i_2}\ldots\bar{\sigma}_{i_k}}$$

[1] $\langle v\rangle \cdot \delta t = \frac{\text{avg. no. spikes}}{\text{time}} \cdot \frac{\text{time}}{\text{no. bins}}$

and

$$\bar{\rho}^p_{ijk} \doteq \rho^p_{ijk} + \rho^p_{ij} + \rho^p_{ik} + \rho^p_{jk} = \frac{\langle \sigma_i \sigma_j \sigma_k \rangle_p - \bar{\sigma}_i \bar{\sigma}_j \bar{\sigma}_k}{\bar{\sigma}_i \bar{\sigma}_j \bar{\sigma}_k}.$$

We use the superscript p to specify which distribution the expectations are calculated over. In order to arrive at the perturbative result in equation A.2, we need to identify the sum $\sum_{i_1<i_2\ldots<i_k}$ as scaling with the system size as $N(N-1)\ldots(N-k+1)/k!$, and each $\bar{\sigma}_i$ as being equal to δ on average. The terms involving the normalised correlation coefficients, $f(\bar{\rho}^{(N)}_{ijk}, \bar{\rho}^{(2)}_{ijk})$, $f(\rho^{(N)}_{ij}, 0)$ have equal powers of $\bar{\sigma}$ in the numerator and denominator and thus behave as constants on average.

We now explain how to obtain the equations for the KL divergences (A.3 and A.4). Let us consider the KL divergence between two probability distributions $D_{KL}(p\|q)$. Our first step involves writing the Sarmanov-Lancaster expansions of p and q. The Sarmanov-Lancaster expansion (Sarmanov, 1962; Lancaster, 1958; Lancaster, 1963) of a probability distribution is written as

$$p(\boldsymbol{\sigma}) = p^{(1)}(\boldsymbol{\sigma})\left(1 + \xi_p(\boldsymbol{\sigma})\right), \quad \xi_p(\boldsymbol{\sigma}) = \sum_{i<j} \delta\sigma_i \delta\sigma_j \mathcal{J}^p_{ij} + \sum_{i<j<k} \delta\sigma_i \delta\sigma_j \delta\sigma_k \mathcal{K}^p_{ijk} + \ldots$$

where $\delta\sigma_i = \sigma_i - \bar{\sigma}_i$. This expansion relates a distribution to the independent distribution, plus a series of corrections relating to the normalised correlation coefficients of the distribution. For instance, $\rho^p_{ij} = (1-\bar{\sigma}_i)(1-\bar{\sigma}_j)\mathcal{J}^p_{ij}$, $\rho^p_{ijk} = (1-\bar{\sigma}_i)(1-\bar{\sigma}_j)(1-\bar{\sigma}_k)\mathcal{K}^p_{ijk}$ and in general,

$$\rho^p_{i_1 i_2 \ldots i_k} = (1-\bar{\sigma}_{i_1})(1-\bar{\sigma}_{i_2})\ldots(1-\bar{\sigma}_{i_k})\mathcal{C}^p_{i_1 i_2 \ldots i_k}.$$

This property, along with others such as $\langle \sigma_i \rangle_p = \langle \sigma_i \rangle_{p^{(1)}} = \bar{\sigma}_i$, and $\langle \xi_p \rangle_{p^{(1)}} = 0$, can be shown by noting that whenever we have a term linear in $\delta\sigma_i$, the expectation of that term over the independent distribution will be 0, since we can factorise out the expectation of $\delta\sigma_i$ which evaluates to 0. For example,

$$\begin{aligned}
\sum_{\boldsymbol{\sigma}} p(\boldsymbol{\sigma}) =& \sum_{\boldsymbol{\sigma}} p^{(1)}(\boldsymbol{\sigma}) \left(1 + \sum_{i<j} \delta\sigma_i \delta\sigma_j \mathcal{J}^p_{ij} + \sum_{i<j<k} \delta\sigma_i \delta\sigma_j \delta\sigma_k \mathcal{K}^p_{ijk} + \ldots\right) \\
=& 1 + \langle \delta\sigma_1 \delta\sigma_2 \rangle \mathcal{J}^p_{12} + \text{expectations that will vanish similarly} \\
=& 1 + \underbrace{\langle \sigma_1 - \bar{\sigma}_1 \rangle}_{=\bar{\sigma}_1 - \bar{\sigma}_1 = 0} \langle \sigma_2 - \bar{\sigma}_2 \rangle \mathcal{J}^p_{12} + \ldots \\
=& 1.
\end{aligned}$$

Since all expectations on the right hand side of line two are over the independent distribution, we can factorise them. Using the Sarmanov-Lancaster expansions of p and q

in the KL divergence, we have

$$\begin{aligned} D_{KL}(p\|q) =& \frac{1}{\ln 2}\sum_{\boldsymbol{\sigma}} p^{(1)}(\boldsymbol{\sigma})(1+\xi_p(\boldsymbol{\sigma}))\left[\ln(1+\xi_p(\boldsymbol{\sigma}))-\ln(1+\xi_q(\boldsymbol{\sigma}))\right] \\ =& \frac{1}{\ln 2}\langle f(\xi_p(\boldsymbol{\sigma}),\xi_q(\boldsymbol{\sigma}))\rangle_{p^{(1)}}, \end{aligned}$$

which follows from eq. A.5 and noting that $\langle \xi_p \rangle_{p^{(1)}} = 0$. To derive equations A.3 and A.4, we Taylor expand $f(\xi_p,\xi_q) \doteq (1+\xi_p)\big[\ln(1+\xi_p)-\ln(1+\xi_q)\big]-(\xi_p-\xi_q)$ around $\xi_p = \xi_1 = 0$ and truncate each term in the expansion after the order $\mathcal{O}((N\delta)^3)$ term. Notice we are not truncating the Taylor expansion, but each term in the Taylor expansion. Also, notice how the $-(\xi_p-\xi_q)$ term is included in f so that the first partial derivatives all vanish, hence we only consider higher order derivatives in the expansion. Using a_{mn} to encapsulate the partial derivatives and constants in each term of the expansion, we can write the KL divergence as

$$D_{KL}(p\|q) = \frac{1}{\ln 2}\sum_{m+n\geq 2} a_{mn}\langle \xi_p(\boldsymbol{\sigma})^m \xi_q(\boldsymbol{\sigma})^n\rangle_{p^{(1)}}.$$

At this point, we perform multi-nomial expansions of ξ_p^m and ξ_p^n, and ask ourselves which terms make up the order $\mathcal{O}(\delta^2)$ terms and which terms make up the order $\mathcal{O}(\delta^3)$ terms.

We arrive at

$$\begin{aligned} \langle \xi_p(\boldsymbol{\sigma})^m \xi_q(\boldsymbol{\sigma})^n\rangle =& \frac{1}{\ln 2}\sum_{i<j}\Bigg[\bar{\sigma}_i\bar{\sigma}_j\left(\rho_{ij}^p\right)^m\left(\rho_{ij}^q\right)^n + \bar{\sigma}_j\left(-\bar{\sigma}_i\rho_{ij}^p\right)^m\left(-\bar{\sigma}_i\rho_{ij}^q\right)^n \\ &+ \bar{\sigma}_i\left(-\bar{\sigma}_j\rho_{ij}^p\right)^m\left(-\bar{\sigma}_j\rho_{ij}^q\right)^n\Bigg] \\ &+ \frac{1}{\ln 2}\sum_{i<j<k}\bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k\left(\bar{\rho}_{ijk}^p\right)^m\left(\bar{\rho}_{ijk}^q\right)^n + \mathcal{O}\left((N\delta)^4\right). \end{aligned} \tag{A.6}$$

As mentioned, it is the multi-nomial expansions of the terms in the Taylor expansion that we truncate up to order $\mathcal{O}((N\delta)^4)$. If we plug equation A.6 into each term of the Taylor expansion of $\langle f(\xi_p(\boldsymbol{\sigma}),\xi_q(\boldsymbol{\sigma}))\rangle_{p^{(1)}} = \sum_{m+n\geq 2} a_{mn}\langle \xi_p(\boldsymbol{\sigma})^m\xi_q(\boldsymbol{\sigma})^n\rangle_{p^{(1)}}$, we can rearrange the Taylor expansion of $f(\xi_p(\boldsymbol{\sigma}),\xi_q(\boldsymbol{\sigma}))\rangle_{p^{(1)}}$ as the sum of Taylor expansions of $f(\rho_{ij}^p,\rho_{ij}^q)$, $f(-\bar{\sigma}_i\rho_{ij}^p,-\bar{\sigma}_i\rho_{ij}^q)$, $f(-\bar{\sigma}_j\rho_{ij}^p,-\bar{\sigma}_j\rho_{ij}^q)$and $f(\bar{\rho}_{ijk}^p,\bar{\rho}_{ijk}^q)$. Thus, we can define the following general expression for the KL divergence for small $N\delta$:

$$\begin{aligned} D_{KL}(p\|q) =& \frac{1}{\ln 2}\sum_{i<j}\Bigg[\bar{\sigma}_i\bar{\sigma}_j f\left(\rho_{ij}^p,\rho_{ij}^q\right) + \bar{\sigma}_j f\left(-\bar{\sigma}_i\rho_{ij}^p,-\bar{\sigma}_i\rho_{ij}^q\right) \\ &+ \bar{\sigma}_i f\left(-\bar{\sigma}_j\rho_{ij}^p,-\bar{\sigma}_j\rho_{ij}^q\right)\Bigg] \\ &+ \frac{1}{\ln 2}\sum_{i<j<k}\bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k f\left(\bar{\rho}_{ijk}^p,\bar{\rho}_{ijk}^q\right) + \mathcal{O}\left((N\delta)^4\right). \end{aligned} \tag{A.7}$$

Considering the case where the divergence is between the true distribution $p^{(N)}$ and the independent model $p^{(1)}$, the terms $\rho_{ij}^{(1)}, \bar{\rho}_{ijk}^{(1)}$ will be equal to zero, since $\langle\sigma_i\sigma_j\rangle_{p^{(1)}} = \langle\sigma_i\rangle_{p^{(1)}}\langle\sigma_j\rangle_{p^{(1)}}$. Thus, for this case, equation A.7 simplifies to

$$\begin{aligned} D_{KL}(p^{(N)}\|p^{(1)}) =& \frac{1}{\ln 2}\sum_{i<j}\Big[\bar{\sigma}_i\bar{\sigma}_j f\left(\rho_{ij}^{(N)},0\right) + \bar{\sigma}_j f\left(-\bar{\sigma}_i\rho_{ij}^{(N)},0\right) \\ &+ \bar{\sigma}_i f\left(-\bar{\sigma}_j\rho_{ij}^{(N)},0\right)\Big] \\ &+ \frac{1}{\ln 2}\sum_{i<j<k}\bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k f\left(\bar{\rho}_{ijk}^{(N)},0\right) + \mathcal{O}\left((N\delta)^4\right). \end{aligned}$$

Focusing on the function $f(x,0) = (1+x)\ln(1+x) - x$ and using the Taylor expansion $\ln(1+x) = x - x^2/2 + x^3/3 + ...$, we have that $f(x,0) = x^2/2 - x^3/6 + ...$ Thus, the terms $\sum_{i<j}\bar{\sigma}_j f(-\bar{\sigma}_i\rho_{ij}^{(N)},0)$ and $\sum_{i<j}\bar{\sigma}_i f(-\bar{\sigma}_j\rho_{ij}^{(N)},0)$ are both $\mathcal{O}(N^2\delta^3)$. This leads to the following approximation to the KL divergence between the true and independent distribution:

$$D_{KL}\left(p^{(N)}\|p^{(1)}\right) = \frac{1}{\ln 2}\sum_{i<j}\bar{\sigma}_i\bar{\sigma}_j f\left(\rho_{ij}^{(N)},0\right) + \mathcal{O}\left((N\delta)^3\right)$$

which is the result we stated in equation A.4.

For the divergence between the true distribution $p^{(N)}$ and the pairwise model $p^{(2)}$, note that since we define the pairwise model as the model which reproduces the pairwise correlations of the true distribution, the correlation coefficients $\rho_{ij}^{(2)}$ and $\rho_{ij}^{(N)}$ will be identical. Since $f(x,x) = 0$, equation A.7 simplifies to:

$$D_{KL}\left(p^{(N)}\|p^{(2)}\right) = \frac{1}{\ln 2}\sum_{i<j<k}\bar{\sigma}_i\bar{\sigma}_j\bar{\sigma}_k f\left(\bar{\rho}_{ijk}^{(N)},\bar{\rho}_{ijk}^{(2)}\right) + \mathcal{O}\left((N\delta)^4\right)$$

which is again the result we stated in equation A.4.

BIBLIOGRAPHY

Ahrens, Misha B. et al. (May 2013). "Whole-brain functional imaging at cellular resolution using light-sheet microscopy". en. In: *Nature Methods* 10.5, pp. 413–420. ISSN: 1548-7105.

Amari, Shun-ichi et al. (Jan. 2003). "Synchronous Firing and Higher-Order Interactions in Neuron Pool". In: *Neural Computation* 15.1, pp. 127–142. ISSN: 0899-7667. DOI: 10.1162/089976603321043720. URL: https://doi.org/10.1162/089976603321043720.

Barlow, HeB, RM Hill, and WR Levick (1964). "Retinal ganglion cells responding selectively to direction and speed of image motion in the rabbit". In: *The Journal of physiology* 173.3, pp. 377–407.

Bergstra, James and Yoshua Bengio (2012). "Random search for hyper-parameter optimization." In: *Journal of machine learning research* 13.2.

Berry, Michael J et al. (1999). "Anticipation of moving stimuli by the retina". In: *Nature* 398.6725, pp. 334–338.

Berry II, Michael J. and Gašper Tkačik (2020). "Clustering of Neural Activity: A Design Principle for Population Codes". In: *Frontiers in Computational Neuroscience* 14. ISSN: 1662-5188. DOI: 10.3389/fncom.2020.00020. URL: https://www.frontiersin.org/articles/10.3389/fncom.2020.00020/full.

Binder, Kurt et al. (2012). *Monte Carlo methods in statistical physics*. Vol. 7. Springer Science & Business Media.

Broderick, Tamara et al. (Dec. 2007). "Faster solutions of the inverse pairwise Ising problem". In: *arXiv:0712.2437 [cond-mat, q-bio]*. arXiv: 0712.2437. URL: http://arxiv.org/abs/0712.2437.

Davison, Anthony Christopher and David Victor Hinkley (1997). *Bootstrap methods and their application*. 1. Cambridge university press.

Deshmukh, Nikhil Rajiv (2015). "Complex computation in the retina". PhD thesis. Princeton University.

Donner, Christian, Klaus Obermayer, and Hideaki Shimazaki (Jan. 2017). "Approximate Inference for Time-Varying Interactions and Macroscopic Dynamics of Neural

Populations". en. In: *PLOS Computational Biology* 13.1. Publisher: Public Library of Science, e1005309. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1005309. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005309.

Ferrari, Ulisse (Aug. 2016). "Learning maximum entropy models from finite-size data sets: A fast data-driven algorithm allows sampling from the posterior distribution". In: *Physical Review E* 94.2. Publisher: American Physical Society, p. 023301. DOI: 10.1103/PhysRevE.94.023301. URL: https://link.aps.org/doi/10.1103/PhysRevE.94.023301.

Ferrari, Ulisse et al. (2018). "Separating intrinsic interactions from extrinsic correlations in a network of sensory neurons". In: *Physical Review E* 98.4, p. 042410.

Ferrenberg, Alan M. and Robert H. Swendsen (Dec. 1988). "New Monte Carlo technique for studying phase transitions". In: *Physical Review Letters* 61.23. Publisher: American Physical Society, pp. 2635–2638. DOI: 10.1103/PhysRevLett.61.2635. URL: https://link.aps.org/doi/10.1103/PhysRevLett.61.2635.

Franzosi, Roberto (Mar. 2018). "Microcanonical entropy for classical systems". en. In: *Physica A: Statistical Mechanics and its Applications* 494, pp. 302–307. ISSN: 0378-4371. DOI: 10.1016/j.physa.2017.12.059.

Ganmor, Elad, Ronen Segev, and Elad Schneidman (June 2011). "Sparse low-order interaction network underlies a highly correlated and learnable neural population code". en. In: *Proceedings of the National Academy of Sciences* 108.23. Publisher: National Academy of Sciences Section: Biological Sciences, pp. 9679–9684. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.1019641108. URL: https://www.pnas.org/content/108/23/9679.

Gardella, Christophe, Olivier Marre, and Thierry Mora (Aug. 2016). "A Tractable Method for Describing Complex Couplings between Neurons and Population Rate". In: *eNeuro* 3.4, ENEURO.0160–15.2016. ISSN: 2373-2822.

– (Feb. 2017). "Restricted Boltzmann Machines provide an accurate metric for retinal responses to visual stimuli". en. In: URL: https://openreview.net/forum?id=Sk1OOnNFx.

– (Feb. 2019). "Modeling the Correlated Activity of Neural Populations: A Review". In: *Neural Computation* 31.2, pp. 233–269. ISSN: 0899-7667. DOI: 10.1162/neco_a_01154. URL: https://doi.org/10.1162/neco_a_01154.

Geman, Stuart and Donald Geman (Nov. 1984). "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.6. Conference Name: IEEE Transactions on

Pattern Analysis and Machine Intelligence, pp. 721–741. ISSN: 1939-3539. DOI: 10.1109/TPAMI.1984.4767596.

Golub, Matthew D, M Yu Byron, and Steven M Chase (2015). "Internal models for interpreting neural population activity during sensorimotor control". In: *Elife* 4, e10015.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep learning*. MIT press.

Greenberg, David S, Arthur R Houweling, and Jason ND Kerr (2008). "Population imaging of ongoing neuronal activity in the visual cortex of awake rats". In: *Nature neuroscience* 11.7, pp. 749–751.

Hastie, Trevor et al. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer.

Hastings, W. K. (Apr. 1970). "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1, pp. 97–109. ISSN: 0006-3444. DOI: 10.1093/biomet/57.1.97. URL: https://doi.org/10.1093/biomet/57.1.97.

Humplik, Jan and Gašper Tkačik (May 2016). "Semiparametric energy-based probabilistic models". In: *arXiv:1605.07371 [cond-mat, q-bio, stat]*. arXiv: 1605.07371. URL: http://arxiv.org/abs/1605.07371.

– (Sept. 2017). "Probabilistic models for neural populations that naturally capture global coupling and criticality". en. In: *PLOS Computational Biology* 13.9. Publisher: Public Library of Science, e1005763. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1005763. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005763.

Ioffe, Mark L (2017). "Adaptation in Coding by Large Populations of Neurons in the Retina". PhD thesis. Princeton University.

Ioffe, Mark L. and Michael J. Berry II (Oct. 2017). "The structured 'low temperature' phase of the retinal population code". en. In: *PLOS Computational Biology* 13.10. Publisher: Public Library of Science, e1005792. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1005792. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005792.

Jaynes, E. T. (May 1957a). "Information Theory and Statistical Mechanics". In: *Physical Review* 106.4. Publisher: American Physical Society, pp. 620–630. DOI: 10.1103/PhysRev.106.620. URL: https://link.aps.org/doi/10.1103/PhysRev.106.620.

– (Oct. 1957b). "Information Theory and Statistical Mechanics. II". In: *Physical Review* 108.2. Publisher: American Physical Society, pp. 171–190. DOI: 10.1103/PhysRev.108.171. URL: https://link.aps.org/doi/10.1103/PhysRev.108.171.

Kardar, Mehran (2007). *Statistical physics of particles*. Cambridge University Press.

Köster, Urs et al. (July 2014). "Modeling Higher-Order Correlations within Cortical Microcolumns". en. In: *PLOS Computational Biology* 10.7. Publisher: Public Library of Science, e1003684. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1003684. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003684.

Lancaster, HO (1958). "The structure of bivariate distributions". In: *The Annals of Mathematical Statistics* 29.3, pp. 719–736.

– (1963). "Correlation and complete dependence of random variables". In: *The Annals of Mathematical Statistics* 34.4, pp. 1315–1321.

Le Roux, Nicolas and Yoshua Bengio (June 2008). "Representational Power of Restricted Boltzmann Machines and Deep Belief Networks". In: *Neural Computation* 20.6, pp. 1631–1649. ISSN: 0899-7667. DOI: 10.1162/neco.2008.04-07-510.

Li, Lisha et al. (2017). "Hyperband: A novel bandit-based approach to hyperparameter optimization". In: *The Journal of Machine Learning Research* 18.1, pp. 6765–6816.

Liu, Jun S. (Jan. 2008). *Monte Carlo Strategies in Scientific Computing*. en. Springer Science & Business Media. ISBN: 978-0-387-76369-9.

Loback, Adrianna et al. (Dec. 2017). "Noise-Robust Modes of the Retinal Population Code Have the Geometry of "Ridges" and Correspond to Neuronal Communities". In: *Neural Computation* 29.12, pp. 3119–3180. ISSN: 0899-7667. DOI: 10.1162/neco_a_01011. URL: https://doi.org/10.1162/neco_a_01011.

Loback, Adrianna R and Michael J Berry (2018). "A Biologically Plausible Mechanism to Learn Clusters of Neural Activity". In: *bioRxiv*, p. 389155.

Loback, Adrianna Renee (2018). "Representational and Robustness Principles of the Retinal Population Code". Accepted: 2018-02-05T16:46:50Z Publisher: Princeton, NJ : Princeton University. PhD thesis.

Macke, Jakob H., Philipp Berens, et al. (Feb. 2009). "Generating Spike Trains with Specified Correlation Coefficients". In: *Neural Computation* 21.2, pp. 397–423. ISSN: 0899-7667. DOI: 10.1162/neco.2008.02-08-713. URL: https://doi.org/10.1162/neco.2008.02-08-713.

Macke, Jakob H., Iain Murray, and Peter Latham (2011). "How biased are maximum entropy models?" en. In: *Advances in Neural Information Processing Systems* 24.

Macke, Jakob H., Manfred Opper, and Matthias Bethge (May 2011). "Common Input Explains Higher-Order Correlations and Entropy in a Simple Model of Neural Population Activity". In: *Physical Review Letters* 106.20. Publisher: American Physical Society, p. 208102. DOI: 10.1103/PhysRevLett.106.208102. URL: https://link.aps.org/doi/10.1103/PhysRevLett.106.208102.

Maoz, Ori and Elad Schneidman (2017). *maxent_toolbox: Maximum Entropy Toolbox for MATLAB, version 1.0.2*. Version 1.02. DOI: 10.5281/zenodo.191625. URL: https://orimaoz.github.io/maxent_toolbox.

Marre, O. et al. (Apr. 2009). "Prediction of Spatiotemporal Patterns of Neural Activity from Pairwise Correlations". In: *Physical Review Letters* 102.13. Publisher: American Physical Society, p. 138101. DOI: 10.1103/PhysRevLett.102.138101. URL: https://link.aps.org/doi/10.1103/PhysRevLett.102.138101.

Marre, Olivier, Dario Amodei, et al. (Oct. 2012). "Mapping a Complete Neural Population in the Retina". en. In: *Journal of Neuroscience* 32.43. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.0723-12.2012. URL: https://www.jneurosci.org/content/32/43/14859.

Marre, Olivier, Gasper Tkacik, et al. (2017). "Multi-electrode array recording from salamander retinal ganglion cells". In:

Martignon, L. et al. (June 1995). "Detecting higher-order interactions among the spiking events in a group of neurons". eng. In: *Biological Cybernetics* 73.1, pp. 69–81. ISSN: 0340-1200. DOI: 10.1007/BF00199057.

Meshulam, Leenoy et al. (2018). "Coarse-graining and hints of scaling in a population of 1000+ neurons". In: *arXiv preprint arXiv:1812.11904*.

– (2021). "Successes and failures of simplified models for a network of real neurons". In: *arXiv preprint arXiv:2112.14735*.

Mora, Thierry, Stéphane Deny, and Olivier Marre (Feb. 2015). "Dynamical Criticality in the Collective Activity of a Population of Retinal Neurons". In: *Physical Review Letters* 114.7. Publisher: American Physical Society, p. 078105. DOI: 10.1103/PhysRevLett.114.078105. URL: https://link.aps.org/doi/10.1103/PhysRevLett.114.078105.

Nakahara, Hiroyuki and Shun-ichi Amari (2002). "Information-geometric measure for neural spikes". In: *Neural computation* 14.10, pp. 2269–2316.

Newman, M and G Barkema (1999). "Monte carlo methods in statistical physics chapter 1-4". In: *New York, USA*.

Nguyen, H. Chau, Riccardo Zecchina, and Johannes Berg (July 2017). "Inverse statistical problems: from the inverse Ising problem to data science". In: *Advances in Physics* 66.3. arXiv: 1702.01522, pp. 197–261. ISSN: 0001-8732, 1460-6976. DOI: 10.1080/00018732.2017.1341604. URL: http://arxiv.org/abs/1702.01522.

Ohiorhenuan, Ifije E et al. (2010). "Sparse coding and high-order correlations in fine-scale cortical networks". In: *Nature* 466.7306, pp. 617–621.

Ölveczky, Bence P, Stephen A Baccus, and Markus Meister (2003). "Segregation of object and background motion in the retina". In: *Nature* 423.6938, pp. 401–408.

Park, Il Memming et al. (2013). "Universal models for binary spike patterns using centered Dirichlet processes". en. In: *Advances in neural information processing systems* 26, pp. 2463–2471.

Prentice, Jason S. et al. (Nov. 2016). "Error-Robust Modes of the Retinal Population Code". en. In: *PLOS Computational Biology* 12.11. Publisher: Public Library of Science, e1005148. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1005148. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1005148.

Puchalla, Jason L. et al. (May 2005). "Redundancy in the Population Code of the Retina". In: *Neuron* 46.3, pp. 493–504. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2005.03.026. URL: https://www.sciencedirect.com/science/article/pii/S0896627305003119.

Ramón y Cajal, Santiago (1888). *Estructura de los centros nerviosos de las aves*.

Rey, Hernan Gonzalo, Carlos Pedreira, and Rodrigo Quian Quiroga (2015). "Past, present and future of spike sorting techniques". In: *Brain Research Bulletin* 119, pp. 106–117. ISSN: 0361-9230. DOI: https://doi.org/10.1016/j.brainresbull.2015.04.007.

Rossant, Cyrille et al. (2016). "Spike sorting for large, dense electrode arrays". In: *Nature neuroscience* 19.4, pp. 634–641.

Roudi, Yasser, Sheila Nirenberg, and Peter E. Latham (May 2009). "Pairwise Maximum Entropy Models for Studying Large Biological Systems: When They Can Work and When They Can't". en. In: *PLOS Computational Biology* 5.5. Publisher: Public Library of Science, e1000380. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1000380. URL: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1000380.

Sanes, Joshua R and Richard H Masland (2015). "The types of retinal ganglion cells: current status and implications for neuronal classification". In: *Annual review of neuroscience* 38, pp. 221–246.

Sarmanov, O (1962). "Maximum correlation coefficient (nonsymmetric case)". In: *Selected Translations in Mathematical Statistics and Probability* 2, pp. 207–210.

Savin, Cristina and Gašper Tkačik (Oct. 2017). "Maximum entropy models as a tool for building precise neural controls". en. In: *Current Opinion in Neurobiology*. Computational Neuroscience 46, pp. 120–126. ISSN: 0959-4388. DOI: 10.1016/j.conb.2017.08.001.

Schneidman, Elad, Michael J Berry, et al. (2006). "Weak pairwise correlations imply strongly correlated network states in a neural population". In: *Nature* 440.7087, pp. 1007–1012.

Schneidman, Elad, Susanne Still, et al. (Dec. 2003). "Network Information and Connected Correlations". In: *Physical Review Letters* 91.23. Publisher: American Physical Society, p. 238701. DOI: 10.1103/PhysRevLett.91.238701. URL: https://link.aps.org/doi/10.1103/PhysRevLett.91.238701.

Schwartz, Greg et al. (2007). "Detection and prediction of periodic patterns by the retina". In: *Nature neuroscience* 10.5, pp. 552–554.

Sherrington, Charles Scott (1906). "Observations on the scratch-reflex in the spinal dog". In: *The Journal of physiology* 34.1-2, pp. 1–50.

Shimazaki, Hideaki et al. (Sept. 2015). "Simultaneous silence organizes structured higher-order interactions in neural populations". en. In: *Scientific Reports* 5.1, p. 9821. ISSN: 2045-2322. DOI: 10.1038/srep09821.

Shlens, Jonathon et al. (Aug. 2006). "The Structure of Multi-Neuron Firing Patterns in Primate Retina". en. In: *Journal of Neuroscience* 26.32, pp. 8254–8266. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.1282-06.2006.

Smolensky, Paul (1986). *Information processing in dynamical systems: Foundations of harmony theory*. Tech. rep. Colorado Univ at Boulder Dept of Computer Science.

Snoek, Jasper, Hugo Larochelle, and Ryan P Adams (2012). "Practical bayesian optimization of machine learning algorithms". In: *Advances in neural information processing systems* 25.

Tang, A. et al. (Jan. 2008). "A Maximum Entropy Model Applied to Spatial and Temporal Correlations from Cortical Networks In Vitro". en. In: *Journal of Neuroscience* 28.2, pp. 505–518. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.3359-07.2008.

Theis, Lucas et al. (2017). "Lossy image compression with compressive autoencoders". In: *arXiv preprint arXiv:1703.00395*.

Tkačik, Gašper, Olivier Marre, Dario Amodei, et al. (2014). "Searching for collective behavior in a large network of sensory neurons". In: *PLoS Comput Biol* 10.1, e1003408.

Tkačik, Gašper, Olivier Marre, Thierry Mora, et al. (2013). "The simplest maximum entropy model for collective behavior in a neural network". In: *Journal of Statistical Mechanics: Theory and Experiment* 2013.03, P03011.

Tkačik, Gašper, Thierry Mora, et al. (2015). "Thermodynamics and signatures of criticality in a network of neurons". In: *Proceedings of the National Academy of Sciences* 112.37, pp. 11508–11513.

Tkačik, Gašper, Elad Schneidman, et al. (2006). "Ising models for networks of real neurons". In: *arXiv preprint q-bio/0611072*.

– (2009). "Spin glass models for a network of real neurons". In: *arXiv preprint arXiv:0912.5409*.

Torlai, Giacomo and Roger G Melko (2016). "Learning thermodynamics with Boltzmann machines". In: *Physical Review B* 94.16, p. 165134.

Vasquez, J. C. et al. (May 2012). "Gibbs distribution analysis of temporal correlations structure in retina ganglion cells". en. In: *Journal of Physiology-Paris*. Neuronal Ensemble Recordings in Integrative Neuroscience 106.3, pp. 120–127. ISSN: 0928-4257. DOI: 10.1016/j.jphysparis.2011.11.001.

Wood, F. et al. (June 2004). "On the variability of manual spike sorting". In: *IEEE Transactions on Biomedical Engineering* 51.6. Conference Name: IEEE Transactions on Biomedical Engineering, pp. 912–918. ISSN: 1558-2531. DOI: 10.1109/TBME.2004.826677.

Yu, Shan et al. (Nov. 2011). "Higher-Order Interactions Characterized in Cortical Activity". In: *Journal of Neuroscience* 31.48, pp. 17514–17526.

Yuste, Rafael (Aug. 2015). "From the neuron doctrine to neural networks". en. In: *Nature Reviews Neuroscience* 16.8, pp. 487–497. ISSN: 1471-0048. DOI: 10.1038/nrn3962.