Competitive learning in rate coded and spiking neural networks models with applications to vision and audition

Nasir Ahmad

St Hilda's College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Hilary 2019

Abstract

Competitive learning is a common and successful approach used to train unsupervised rate-based neural network models. We apply such a technique in this thesis and produce a rate-coded neural network model of pitch processing which provides insights into the training protocols necessary to develop robust pitch representations. However, the extension of reliable unsupervised competitive learning approaches from rate-coded to spiking neural networks has proven challenging, especially when biological plausibility and detail are desired. Transitioning to spiking neural network models is made more difficult by the comparatively high computational cost and complexity of these network models compared to rate-based models.

We describe a transition from rate-based to spiking neural network models and address these outstanding issues. First, we focus on increasing simulation efficiency. We develop a state of the art graphical processing unit (GPU) based spiking neural network simulator which adopts optimisations from central processing unit (CPU) and cluster-based simulators. In benchmarks, we show that our novel simulator is capable of simulation speeds up to an order of magnitude greater than contemporary simulators. This greater simulation speed is intended to enable a higher throughput of spiking neural network simulations and thereby accelerate research. In order to ensure that present and future simulators can be efficiently and effectively compared, we also compile a repository of benchmarks which allow validation of simulator performance and simulated neural network dynamics.

Having developed efficient simulation techniques, we propose a novel excitatory plasticity rule which, when used to train spiking neural networks in conjunction with an existing inhibitory plasticity rule, produces reliable competitive learning. These rules are both unsupervised, use local only information, and depend upon spike-timing in order to compute synaptic weight updates. This learning rule framework is used to produce a model of V1 simple cell receptive field development and shows qualitatively similar behaviour to traditional sparse coding models. During the development of these models we come across a number of challenges, especially in the interactions of inhibitory and excitatory neurons. We pose solutions to these challenges and thereafter suggest some future research questions and avenues for exploration.

Acknowledgements

The research produced in this thesis was funded by the Biotechnology and Biological Sciences Research Council (BBSRC) [grant number BB/J014427/1], and by the Oxford Foundation for Theoretical Neuroscience and Artificial Intelligence (OFTNAI, Charity No. 1116075). Without this support, it would not have been possible for me to spend this time pursuing my passions and I sincerely thank my funders for enabling this incredible experience.

Equally, it is only with the support of colleagues and staff in a number of groups and laboratories, both within the University of Oxford and without, that the production of this work has been possible. I shall remember with fondness the the Interdisciplinary Bioscience Doctoral Training Partnership (DTP) and the Doctoral Training Centre (DTC) as the places I first arrived in Oxford to learn and explore with a group of fellow wide-eyed DPhil students, many of whom I can now call good friends. Gyana, the startup with which I spent a good part of a year, proved a well of unending passion and intensity as well as friendship. The Auditory Neuroscience Group (ANG) and the Walker lab (of my co-supervisor) have consistently been the most welcoming and curious collection of researchers I've come across. My only regret is that I spent less time with them all than I should have. A place where I was most grateful to have been welcomed was the Vogels' lab. Attending their meetings was a privilege, which I was given freely and openly, and my work has benefitted immeasurably from their input. Their kindness will not be forgotten, and neither will their rigour.

Last, but most definitely not least, the Stringer lab (of my primary supervisor) has been my home for some years now and it is with sadness that I consider leaving the friends that I have made there. No doubt I will see them all flourish both personally and professionally, and I wish that for each and every one. I wish the same for my friends and colleagues in the Experimental Psychology department where I was hosted.

Above all else, my family and friends have been the core of my support. They were not only present for every twist and turn of the last few years but more importantly believed in me even when I did not. To name any single or few of these people would be an injustice to the rest so I leave this crowd of supporters with my best wishes and I intend to thank each of them through my actions in the coming months rather than just with words.

Contents

1

Introduction

The activation of sensory organs produces electrical stimulation which is transmitted to the brains of complex organisms. There, these stimulations are processed to form perception, a phenomenon which enables you to read these words. In the processing stage, information relevant to the sensory environment is extracted from incoming stimulation and is encoded in neural activity. This neural activity is ultimately used to drive behaviour. The specific neural and synaptic mechanisms by which the information extraction and storage is achieved is an open topic of research.

The process of extracting relevant information is thought to be optimised through learning in neural circuits [110], and the method by which the brain thereafter represents sensory features is referred to as the neural code. These phenomena of neural plasticity and neural coding have been studied in electrophysiological and modelling studies across a range of biological scales. Modelling studies most commonly abstract away details on the sub-cellular scale and consider neural network

models with neurons represented by nodes which are connected by synapses (edges). These models are used to examine theoretical properties of neural circuits and to reproduce experimental observations of neural circuit responses. We primarily focus on such node (point) based neural network models with two levels of biological detail, rate-based and spiking neural networks.

Rate-based neural networks are by far the most widely investigated. The nodes in these networks have activity levels which are denoted by a scalar value which depends upon the input stimulation. This activity level is used to determine the output of each cell – outputs which were originally proposed as representing the rate at which a neuron is producing action potentials. These rate neurons are often connected by simple synaptic models which weight the incoming activation.

The successes of rate-based neural networks are in part due to their simplicity and the ease with which their dynamics can be described, manipulated, and simulated. Biologically detailed components and dynamics can be introduced to such node-based studies in order to test more detailed hypotheses about the roles of action potentials, ion channels, neuron morphology and more. Spiking neural networks are an example of such a biologically detailed extension of rate-based models. These networks feature neurons which have an associated membrane voltage, the dynamics of which are often modelled by a differential equation (see Gerstner et al. [68] for a review of various rate and spiking neuron dynamics). This membrane voltage can be increased through incoming excitation and cells can be driven to produce action potentials. It is only following an action potential that a neuron's outgoing synaptic connections provide input to its downstream neurons, and biologically these action potentials (spikes) are the primary events which allow both propagation of activity and learning. This level of modelling produces detail in the network dynamics while continuing to treat neurons as nodes. We can thus compromise between biological complexity and computational intensity.

Learning, through modification of the strength or 'weight' of synaptic connections,

can be achieved in rate-based and spiking neural network models by a number of principles. We focus primarily in this thesis upon a method known as competitive learning [175]. How this learning principle fits into the larger picture of principles for learning is described below. Competitive learning can broadly be described as a process in which synaptic strengths are modified differentially such that some subset of synapses (which are deemed to have won some competitive process for a given stimulation) are preferentially strengthened. The winner of the competitive process often depends upon the firing rate of individual neurons relative to the population firing rates but competition can also be induced on a synapse level (see Miller [126] for a combined experimental and theoretical description of competitive synaptic changes).

This thesis focuses on bridging the production of competitive neural network models from rate coded networks to spiking network models. Rate coded modelling studies of neural circuits are often described as approximations or simplifications of spiking neural network models, however a number of common features of successful rate coded models remain unrealised in spiking neural networks. The chapters of this thesis begin with the presentation of a rate coded neural network model of pitch processing, Chapter 2, with well established dynamics and learning rules. Following this, we identify some major challenges to the development of equivalent spiking neural networks, namely simulation speed and reliable competitive learning. The development of optimisation principles and software are described in Chapter 3 where we show the development of a state of the art spiking neural network simulator. Furthermore, a set of repositories are created which allow continued comparison of simulation speed and network dynamics across simulators.

Finally, in Chapter 4 we present an unsupervised and local learning procedure which implements spike timing-dependent plasticity in order to reliably produce competitive learning in a spiking neural network model. This model is extended in Chapter 5 to show how our proposed learning paradigm can be applied in

networks of excitatory and inhibitory neurons. This concludes our research arc from well established rate coded modelling to the development of efficient and reliable competitive spiking neural network models while maintaining biological plausibility.

1.1 The Function of Early Sensory Cortices

Sensory organs recruit relatively specialised cortical areas which process incoming sensory stimulation. These recruited cortical areas, also known as primary sensory areas, are the main arrival sites of sensory input. For example, visual cortical processing begins in the occipital lobe where, after being relayed via the lateral geniculate nucleus, activity arrives from the optic nerve. Similarly, auditory cortical processing begins in the temporal lobe where, relayed via the medial geniculate nucleus, activity arrives from the cochlea. We focus in general upon the primary auditory and visual cortices for this thesis, though many of the conclusions which follow also apply to somatosensory cortex and potentially to higher auditory and visual cortical areas.

Despite the range of sensory modalities and behavioural tasks which elicit responses in cortical areas, mammalian neocortical structure is observed to be largely regular. Neocortical areas are composed of a layered structure, commonly divided into six parts, each layer being composed of particular cell types, cell densities, and connectivities [133]. This homogeneity has been described both across cortical areas and also between species [160, 32], though the degree of similarity is hotly debated [186, 16, 85]. Cortical areas do exhibit some systematic architectural differences such that they have classically been classified into some 50 sub-areas in human and non-human primate studies [97]. Though the use of different measurements to carry out this classification can provide competing maps of cortical areas [4]. Regardless, the detected differences between cortical areas are often attributed to adaptations of these areas to specialised purposes [44].

Evidence for the similarity of the functions of different cortical areas is notably present for the primary auditory and visual cortices. The similar function of these areas is evidenced by studies which show the ability to achieve visual receptive field tunings in auditory cortical neurons by inducing projections from the retina to the auditory cortex [195, 161]. Other than receptive field formation, a re-routing of the visual pathway to auditory cortex has also allowed visual task learning by recruitment of the auditory cortical neurons [124]. Cortical areas have also been observed to flexibly adjust their function and become recruited for different sensory purposes depending upon sensory pressures. One example of this is the recruitment of classically visual areas of the occipital lobe for auditory processing in the congenitally blind, without any experimental manipulation [11]. Ultimately, these data suggest that learning across cortical areas achieves a similar function despite some differences in structure and implementational detail [55, 88].

The neural substrate of the aforementioned "cortical function" has also been a hot topic of debate. Structural similarities across cortical areas suggest the presence of a basic cortical building block. Early attempts at an architectural description suggested the cortical minicolumn as the basic structural unit [132]. As electrodes are pushed through the layers of cortex, from a radial direction, neural receptive fields are observed to be extremely similar. This similarity seems to be consistent for groups of approximately 50-100 neurons and these groups were labelled as cortical minicolumns. The radial similarity of receptive fields has been described as a general feature of cortical sensory areas [98]. These cortical columns were theorised to be the building block of processing such that a single column was carrying out a single function and therefore had such a systematically similar response characteristic across it. Furthermore, larger cortical areas, often called hypercolumns, were suggested as a macro organisational structure which groups these minicolumns [198, 54].

This columnar hypothesis provided an explanation for the radial similarity

of cortical responses and also provided an attractive hypothesis for sub-division of cortical computation. However, close investigation of cortical structures has identified instead what have been termed cortical microcircuits [49, 40]. In essence, rather than cortical columns acting as computational units, there appear to be specific connectivities between various cell types in cortical layers which ultimately form a circuit design for computation. This microcircuit arrangement was also suggested to be common across cortical areas and was thus labelled a 'canonical cortical microcircuit' [47, 48], however ultimately the microcircuits appear to differ across cortical areas.

Aside from the substrate, the general algorithm which the cortex computes is of great interest [121]. In primary sensory cortices, processing predominantly appears to allow the extraction of regular features of the environment. This was evidenced in early studies of visual cortical processing in cat cortex [90]. Modelling studies have attempted to uncover this general algorithm with some limited success. In particular they have shown the emergence of selected receptive field properties of cortical neurons [146, 153, 184] and larger scale cortical response statistics have been matched to models employing high-level algorithmic optimisations [214].

1.2 Theoretical proposals and models investigating the neural code

Alongside experimental investigations of cortical function, a number of theoretical proposals have been put forth to explain potential constraints and operating principles of neural circuits. We describe a few of these here.

The neural code is the method by which information is represented and propagated in neural activity. The process by which information (sensory or otherwise) is converted to this neural code is referred to as neural encoding. On the other

hand, the process by which information can be extracted from neural responses is referred to as decoding.

The nature of the neural encoding of information has been theoretically debated from a perspective in which neurons are considered only active or inactive, binarising their responses. The simplest form of representation by binary activity is a localist representation in which an individual neuron's activation represents a single concept or category. This was also commonly referred to as grandmother cell encoding [75]. The proposal that individual neurons represent entire concepts is an extreme perspective which is opposed by proposals of dense distributed representation [87] in which an unrestricted number of cells are simultaneously active and the specific pattern of their activity represents concepts of categories.

These two extreme proposals of neural representation in the brain are largely argued based upon theoretical considerations for capacity, and capability of such systems. For example, individual or localist representations are easily decoded and particularly compatible with symbolic processing systems but their benefits are offset by the very small capacities of such systems. For a system of N neurons, only N concepts can be represented. On the other hand, distributed encoding of information has a huge capacity with N neurons capable of producing 2^N unique patterns. However, decoding from a distributed encoding is difficult due to the lack of linear separability of the individual patterns. Such a distributed encoding would often also mean large numbers of simultaneously active neurons which would incur a high metabolic cost.

Ultimately, as these theoretical proposals were met with neurobiological observations of neural response statistics and considerations for metabolic constraints, an intermediate form of sparse distributed coding has been accepted as a likely basis of neural information encoding. Sparse distributed coding, often shortened to sparse coding, describes systems in which a restricted fraction of neurons are simultaneously active. Sparse coding provides a theoretical compromise between the

extreme benefits and drawbacks of localist and dense distributed representations [57]. Furthermore, metabolic constraints, which have been theorised to severely limit neural activity [113], can be met.

Another method for developing theories of cortical function is by the reproduction of features of cortical neurons. Sparse coding was shown to be successful in reproducing primary visual cortex simple cell-like receptive field tunings when trained under an objective function which both minimises the number of active neurons, ensuring sparseness, and simultaneously attempting to maximise the ability for the network to reproduce the input stimulus given its neural activations [146, 144]. For an overview of a range of studies employing sparse coding for primary visual cortex simple cell reproduction see the Introduction section of Chapter 4.

These sparse coding networks produce only simple cell responses and are unable to produce complex or end-stopping visual cell responses which are observed in the same primary visual cortical areas. Following this attempt to produce simple cell receptive field tunings, limited success was achieved in reproducing non-classical receptive field effects by a predictive coding approach [153]. This predictive coding approach implemented a hierarchical model in which top down connections attempt to remove predicted input stimulation through negative feedback. This study was thereby capable of reproducing end-stopping responses of neurons in early visual cortex. However, this model was only capable of producing simple cell-like receptive fields when modified for an alternative neural activation restriction, making its efficacy less convincing. Extensions of predictive coding are being explored in order to produce theoretical proposals for the global optimization or algorithm being carried out by the brain [10, 61] and for abstract hierarchical models of cortical processing [66].

Both sparse coding and predictive coding approaches described above implicitly rely upon competition between neurons. A single neuron which responds significantly to a stimulus contributes to a constraint imposed on the response magnitudes

of the other neurons in the network. A network with such a competitive force between its neuron activations is known as a competitive neural network. This is a general principle which is applied in many models of both cortical and sub-cortical processing circuits. Furthermore, such a network can be coupled with learning to produce competitive learning. We describe competitive neural networks and competitive learning below.

1.2.1 Competitive Neural Networks and Learning

1.2.1.1 Competitive Neural Networks

This term 'competitive neural network' is used to described a network in which neurons are in competition to become active and thereby the activation of one neuron results in a reduction in the activation of other neurons. Competition ranges from the extreme in which only a single neuron is active at any point in time, termed "winner-take-all" competition, to softer forms of competition such that there is a restriction on the total amount of simultaneous neural activation.

Competition is prevalent in neural network models and can be reduced to a very simple circuit process – excitatory neurons which become active and cause a cascade of activity, often resulting in feedback inhibition, which brings the response magnitudes of other nearby neurons to a lower level. In Chapter 2, we present a simple method of competition in a rate coded neural network where a sliding threshold allows only a specified, small percent, of neurons to be active above some baseline level.

This concept of competition through inhibition is also present in the Self-Organising Map (SOM) literature [108]. These architectures feature local excitatory synaptic connections and long range inhibitory synaptic connections. Such SOM architectures can produce the locally similar response characteristics, as observed cortical areas, and between neurons with a longer spatial separation the inhibition

is dominant and creates competition leading to different receptive field tunings.

Competition in the activation of neurons allows distinct responses of neurons. However, for neural responses to be selective to particular input stimulations, learning much also take place. When competition at the network level is coupled with learning rules for synaptic strength or weight change, we arrive at competitive learning. In the 1980s, the term competitive learning was coined by Rumelhart et al. [175]. Despite the coining of this term in the 1980s, modelled networks in which neurons compete to become active and thereafter 'learn' existed more than a decade before [118, 63].

In competitive learning, generally those neurons which 'win' some competitive process modify their synaptic connections in order to become further responsive to the stimulus which is driving their activation. One should note, however, that the term "competitive learning" is also often used in descriptions of models in which the individual synaptic connections compete during learning, though this is not a focus of our work. Ultimately, all of these descriptions of competitive learning raise the question of how learning might be accomplished.

1.2.1.2 Learning

Neural network models employ learning in order to modify synaptic connectivities, and thereby neuron response tunings. In competitive networks such as those described above, learning enables neural responses to become tuned to preferred stimuli. In modelling studies the synaptic connectivities are often initialised randomly and learning acts to modify these synaptic weights such that they converge upon weight structures which enable highly informative neural responses.

Learning can be accomplished in neural network models in a number of ways. These learning methods can be separated, on the highest level, into supervised and unsupervised approaches. Supervised learning approaches use a dataset of labelled stimuli, and present these stimuli to a network while simultaneously imposing a

target firing rate upon some neurons within the network based upon the label of the stimulus. This enforcement of neural responses is often accomplished through error propagation and gradient descent, namely a technique known as backpropagation of error [94, 179].

In comparison, unsupervised learning techniques are capable of make use of unlabelled datasets. These networks are often used to uncover underlying statistical structure or produce compressed representations of the chosen dataset. An auto-encoder is an example of such a network where the network attempts to re-produce the input image via layers of processing. By restricting the widths of these layers, a compressed representation of the image can be formed in the middle of such a network.

However, these unsupervised techniques nonetheless often employ backpropagation of error in order to minimize the particular objective function. Despite the successes of methods employing backpropagation of error, the plausibility of this method of training in the brain is unclear and currently under debate [119].

A sub-division of unsupervised neural network models, on which we are focussed in this thesis, do not employ error signalling and instead carry out learning at synapses based only upon the activities of the pre and post-synaptic neurons. Such unsupervised methods are assumed to be the primary learning mechanisms of neural circuits in the brain. The earliest descriptions of such learning processes proposed that synaptic connection strengths could be updated according to the similarity of activities of a synapse's pre and post-synaptic neurons. This principle is known as Hebb's rule [81] and often colloquially expressed: "neurons that fire together wire together". This form of learning is often described as local, as it is based upon biological quantities present at the synaptic connection (locally).

Though such local unsupervised learning does not have an explicit objective function, a number of studies have shown the ability for unsupervised learning to extract specific properties of input data. For example, local unsupervised learning

rules can be used to allow a model neuron to extract the principle components of an input sequence [141] or to carry out independent component analysis [36]. Thus, despite the lack of an explicit objective function and error signals, unsupervised learning can allow learning of statistical structure within an input dataset.

We describe some common examples of local unsupervised learning rules in rate coded neural network literature and subsequently in spiking neural network literature.

1.2.2 Local Unsupervised Learning in Rate Coded Neural Networks

In order to best describe the learning rules implemented in rate coded neural networks, we first define some terms. Suppose we have a population of input neurons with firing rates x, which project to an output population with firing rates y. This would be a feedforward system in which the activations of the output neurons, y, are dependent upon only the firing rates of the input neurons, x. The synaptic connection from some input neuron, indexed j, to an output neuron, indexed i, has a weight w_{ij}.

Neural dynamics determine how these weights affect the activity propagation from one group of neurons to the other. In Chapter 2, we describe a rate coded neuron model in detail and apply it. For now, let us briefly present a simple network so that we may discuss learning rules. Supposing that the post-synaptic neuron i receives input such that its activation, h_i, is a weighted sum of its input neuron firing rates:

$$h_i = \sum_j w_{ij} x_j.$$

where the weighting, w_{ij}, of the input neuron firing rate, x_j, is simply the synaptic weight. A linear neural system would require no filtering of this activation to form

the output neuron firing rate response, such that

$$y_i = h_i.$$

However, non-linear neural response functions, where $y_i = f(h_i)$, are commonly used as they extend the computational ability of neural processing. A commonly used filter is a sigmoid, see Chapter 2 for an example network of this style.

Given this description of neural dynamics, we can describe Hebb's rule as a plasticity mechanism which acts upon the weight, w_{ij}, such that the weight changes based upon the firing rates of the pre and post-synaptic neurons. A simple description of Hebbian learning gives the weight update rule:

$$w_{ij} \leftarrow w_{ij} + kx_j y_i,$$

where k is a scaling constant often referred to as the learning rate. Given that the firing rates, x, of input neurons in this network are non-zero, this simple Hebbian learning rule produces increases in synaptic weight which are greatest for those synapses which connect pairs of neurons which have highly correlated firing rates. In order to ensure that synaptic weights do not increase in an uncontrolled, explosive fashion we can introduce methods to constrain the maximum weight. An approach known as 'hard' bounding is when the magnitudes of synaptic weights are restricted by the simulator, for example ensuring that it cannot exceed a given value.

However, the above learning rule can only produce increases in synaptic weight. Therefore, such a simple Hebbian learning rule requires additional mechanisms to reduce synaptic weights or there is a danger of all synaptic connections rising to the maximum value.

Oja's rule [141], a theoretically motivated extension to the Hebbian learning rule, provides an alternative method to ensure implicit control over the synaptic weights. Rather than the weight update rule presented above, the weight change

includes an additional term, such that:

$$w_{ij} \leftarrow w_{ij} + k(x_j y_i - w_{ij} y_i^2).$$

This rule largely matches a simple Hebbian rule but with a "forgetting" term, $-w_{ij}y_i^2$. This forgetting term ensures that weight increases induced by the Hebbian component of this rule are offset when the weight becomes excessively large, and/or when the firing rate of the post-synaptic neuron becomes excessively large. Rules with such implicit bounds upon the weight values are called soft-bound. For a single neuron connected to a set of input sources which present, sequentially, a series of patterns, Oja's rule has been shown capable of extracting the first principle component of this pattern set [141, 140].

Similar bounds upon weights to that which Oja's provides can be imposed with other constraints, such as those introduced by the BCM rule [20]. This rule is particularly favourable given its ability to produce an explanation of receptive field development in visual cortex under binocular vision and monocular deprivation [41]. The BCM rule describes the change in a weight, w_{ij}, as;

$$\frac{dw_{ij}}{dt} = y_i(y_i - \theta_M)x_j - \eta w_{ij},$$

where

$$\theta_M = \mathrm{E}(y_i)^p.$$

This rule presents the instantaneous change in weight as dependent upon the output neuron firing rate relative to a thresholding variable, θ_M. This is a sliding threshold which depends upon the neuron average firing rate across input stimuli taken to some power, p. The BCM rule and accompanying theory on sliding-threshold based learning proves successful in explanations of how, under different

stimulation frequency protocols, synaptic weight potentiation or depression can be induced [6]. Furthermore, BCM and BCM-like rules have proven successful in predicting observed features of learning in visual cortex and hippocampus [2, 106].

Thus far we have described a few learning rules that have been applied successfully in neural network models, however we have not described how these can be combined with a competitive network. In a network of multiple neurons which are in a state of competition, Oja's rule allows those neurons to each learn a different principle component of the input dataset under some constraints of the input data and with an anti-Hebbian learning rule acting on lateral to inhibitory connections [58]. Similarly, a modified version of the BCM rule has been applied alongside network competition to produce a model of three-dimensional object recognition through unsupervised competitive learning [91].

Another similar approach to these is that taken by [163], where the standard Hebbian form of synaptic update was combined with a weight normalization mechanism and neural response functions which have a sliding threshold. This sliding threshold is based upon the population response distribution and is not directly linked to a neural mechanism, however such models have been successful in producing explanatory descriptions of the emergence of a range of neural response profiles in studies on ventral and dorsal visual cortex, and the hippocampus [166, 64, 206]. This approach is applied in Chapter 2 to investigate the emergence of pitch sensitive responses in primary auditory cortex.

Rate-based approaches for learning and competition have been widely used to explain neural response statistics but are only capable of doing so on the firing rate level and are abstractions of underlying neural dynamics. In order to provide mechanistic descriptions of the roles of action potentials, and to explain experimental observations of synaptic strength changes based upon spike timing, such competitive networks and learning rules require an extension to spiking neural network models.

1.2.3 Local Learning and Competition in Spiking Neural Networks

Hebbian learning, described above, was initially proposed as a theoretical principle of neural circuits without electrophysiological evidence. Furthermore, as described above it was not accompanied by a method for decreases in synaptic weight. Some decades later, evidence surfaced for the existence of a timing dependent change in synaptic weight. Electrophysiological data robustly showed that neural circuits perform learning based upon the timings of action potentials, a learning rule phenomenon which requires rather different consideration than learning based upon average firing rates [19, 120]. This timing dependence leads to synaptic potentiation and synaptic depression. Simultaneously, modelling studies also proposed learning rules which modify synaptic weights depending upon the timings of the action potentials produced by the pre and post-synaptic neurons [67]. These led to the development of spike-timing dependent plasticity (STDP) rules which are now a common feature of computational models of neural circuits.

STDP rules applied in modelling studies can be defined across a range of complexity in order to explain a number of biological observations. The simplest STDP rules are known as additive rules, in which updates to a given synaptic weight are dependent upon only the relative timings of pre and post-synaptic spikes, and simply cause an additive change to the current weight value. Early electrophysiological data [19] shows an exponentially decaying dependence between the magnitude of the timing difference between pre and post-synaptic neuron action potentials and the weight change, see Figure 1.1. The additive STDP rule shown is anti-symmetric, meaning that the order of the action potentials, pre vs post, determines whether the weight change is positive or negative.

Experimental observations of STDP have been fit by a very simple model of this data [225]. Consider a single synaptic connection, connecting some pre-synaptic

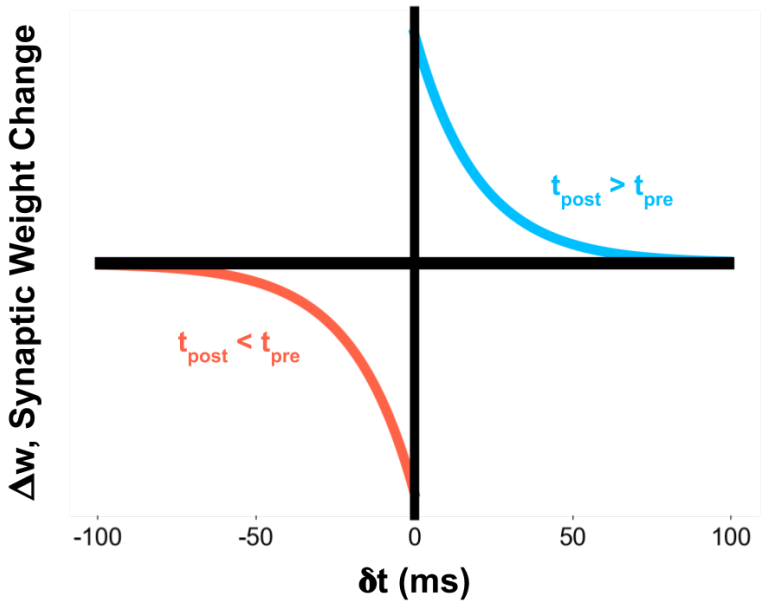

Figure 1.1: **The Effect of the Timing of Pre and Post-synaptic Spikes upon the Synaptic Strength for Additive STDP** A depiction of the dependence of synaptic weight change, Δw, upon relative timing difference between pre and post-synaptic neuron spike times (δt). This relation is shown for a case in which the STDP time constants, τ_+ and τ_-, are equal to 20ms and the magnitudes of potentiation and depression, A_+ and A_-, are equal.

neuron j and post-synaptic neuron i, with weight w_{ij}. There are a set of times at which the pre-synaptic neuron fires action potentials, and times at which the post-synaptic neuron produces action potentials. We can produce an STDP rule by considering a single pair of these action potentials where the pre-synaptic neuron spikes at a time t_{pre} and the post-synaptic neuron spikes at some time t_{post}. The simplest form of this STDP is given

$$\Delta w_{ij} = \begin{cases} A_+ \exp(-\delta t/\tau_+), & \text{if } \delta t \geq 0 \\ -A_- \exp(\delta t/\tau_-), & \text{otherwise,} \end{cases}$$

where

$$\delta t = t_{post} - t_{pre}.$$

The amount of weight change Δw_{ij} in this formulation depends directly upon the order in which the pre and post-synaptic neurons produced action potentials, δt, and upon the parameters which characterise this relationship where A_+ and A_- are the magnitudes of the positive and negative weight changes and τ_+ and τ_- are their time constants. This weight update can be computed in an online fashion by calculating the contributions to weight change upon the arrival of each pre and post-synaptic spike at a synaptic connection.

As with the simple Hebbian learning rule described earlier, such an additive STDP rule is unstable. If it is not augmented in some way by a weight limiting mechanism, the STDP rule can result unrestricted decay or increase of the synaptic weight value. These rules are often hard-bound by imposing a limit upon the maximum weight value and by not allowing the weight value to reduce below zero. Such bounding results in a bi-modal distribution of the weights. Some percent of the weights reduce completely to zero, and the rest are increased until they reach the weight maximum.

Alternative formulations of the STDP rule allow soft bounding of the weights [168]. This is achieved by making the learning rules themselves weight dependent and, as described by van Rossum et al. [168], these can reproduce observed cortical synaptic strength distributions. This rule is similar to the above, however the function used to determine weight reduction is dependent upon the current weight, such that

$$\Delta w_{ij} = \begin{cases} A_+ \exp(-\delta t/\tau_+), & \text{if } \delta t \geq 0 \\ w_{ij} \cdot A_- \exp(\delta t/\tau_-), & \text{otherwise.} \end{cases}$$

This rule is similar to the additive rule described above, however depression of synaptic weights depend upon the current weight value. This results in a system where weights never reduce to zero and instead can only asymptotically approach

zero. Similarly, the weights are unable to be potentiated indefinitely given that the extent of weight depression increases with the synaptic weight. This rule ultimately has a stable set point such that synaptic connections in networks tend to some mean value and form a distribution about this mean.

A number of extensions of these STDP rules have been developed which explain properties of observed neural circuits. For example, the pairwise interaction of a single pre-synaptic and single post-synaptic spike was found to be insufficient to describe weight change in real circuits under high pre and post-synaptic neuron firing rates. Therefore, higher-order interactions, such as weight changes based upon triplets of spikes, have been proposed [152]. Other non-linear effects of plasticity in the brain could be explained by including dependencies of the plasticity rules upon the membrane voltage of the post-synaptic cell [37]. These extensions to STDP allow greater predictive power and explanation of observed electrophysiological features of STDP [69, 35].

Simulation based investigation into STFP require the production of spiking neural network models. In order to do so, a synapse and neuron model must be defined which captures the ability for neurons to be excited and to produce action potentials. We describe a such neuron and synapse model, the leaky integrate and fire neuron with conductance based synapses, in Chapter 3. The leaky integrate and fire neuron model is widely used given that it captures the minimal dynamics necessary to produce spiking behaviour. We only discuss and simulate this neuron model in this thesis, though a number of neuron models have been developed which include more sophisticated dynamics with minimal additional computational complexity [95, 24] and could be considered for simulations in which complex neuron behaviours are desired. On the extreme end of biological realism, neurons can be modelled with individual ion channel types and complex morphologies.

Spiking neural network models of competitive systems exist in a variety of forms, many of which are trained by supervised learning. However, given our interest

in applying unsupervised, non-error based, learning rules which use STDP and biologically plausible weight updates for training, we focus on only a subset of these.

A few examples of competitive networks using STDP and local learning include those which learn based upon reservoirs of neurons [150], networks in which neurons only emit single spikes [122, 104], and networks which use a large number of interacting learning rules to produce stable dynamics [221]. These network models successfully perform specific tasks and produce biologically detailed explanations, however the number of such studies capable of producing competitive spiking neural networks while remaining unsupervised and providing a spike timing-based implementation is small. In comparison, there are a number of methods for rate-based learning in spiking neural networks or methods by which an error or reward signal is used to shape the synaptic weights [138, 151, 22, 136].

A promising method for unsupervised learning through STDP produced sparse coding models of visual processing [231, 105], however this attempt used non-spiking neurons for the input to the system and non-STDP based learning on excitatory synaptic connections. Other methods employing pure spiking [221] dynamics instead rely upon making interacting learning rules and mechanisms for network stabilization. This lack of a simple and robust method for local unsupervised competitive learning in spiking neural networks motivates the final few chapters of this thesis.

1.3 Outline of Thesis

Rate coded models have been successful in their attempts to describe a range of observations in neural activation through unsupervised training. We exhibit this ability of rate coded models in Chapter 2 and produce a model of pitch processing. However, extension of this ability to train through unsupervised and local learning to spiking neural network models has been challenging.

A number of prohibitive barriers stand in the way of developing unsupervised

spiking neural network simulations compared to rate coded models of the past. The first major prohibitive barrier is simulation speed. Where in simple rate coded models a neuron's firing rate response can often be computed in a single step, especially for single layer feed-forward systems such as that which we present in Chapter 2, spiking neural networks require a simulation of the network dynamics in an iterative manner for every neuron for tens of seconds to arrive at a reasonable estimate of the firing rate. Furthermore, the spike times of nodes in the network heavily influence learning with spike timing-dependent plasticity rules. Thus, to achieve convergence of the synaptic weights in a spiking networks, learning is generally carried out for timescales much larger than rate-based networks. Thus, an equivalent timescale of simulation of a spiking versus rate-coded network can require orders of magnitude greater computation. In order to bridge the production of models of competitive learning from rate coded to spiking neural networks, this computational barrier must be addressed. We address this challenge in Chapter 3 where we introduce a novel high speed spiking neural network simulator, Spike, and compare its performance to a range of other widely available spiking neural network simulators.

Beyond simulation speed, robust methods for producing reliable learning and competition by using local only and spike timing-dependent plasticity rules are few and far between in the spiking neural network literature. In Chapter 4, we produce a novel excitatory STDP rule which acts in a complementary fashion with an existing inhibitory plasticity rule in order to produce robust learning. These rules, when simulated in spiking neural networks, have behaviour which is well explained by a simple competitive network description. We show the effectiveness of the learning rule arrangement by producing receptive fields of primary visual cortex simple cells through training with a whitened natural image dataset. This study is followed in Chapter 5 with an extension of this principle to networks of excitatory and inhibitory neurons and we address some challenges of this transition there.

Each of these chapters has a unique focus and is therefore provided with its own

introduction to place its scientific questions in the relevant literature.

2

An Application of Rate Coded Competitive Neural Networks to Model Auditory Processing of Pitch

2.1 Introduction

2.1.1 Pitch Processing in the Brain and Modelling Approaches

2.1.1.1 Pitch Perception

Pitch, an entirely perceptual phenomenon, conveys semantic information in speech, music and animal vocalizations, and plays a key role in our ability to attend to a objects in our noisy natural environments. It can be described as the tonal ("low" or "high") quality of sound, which is most often associated with the periodicity

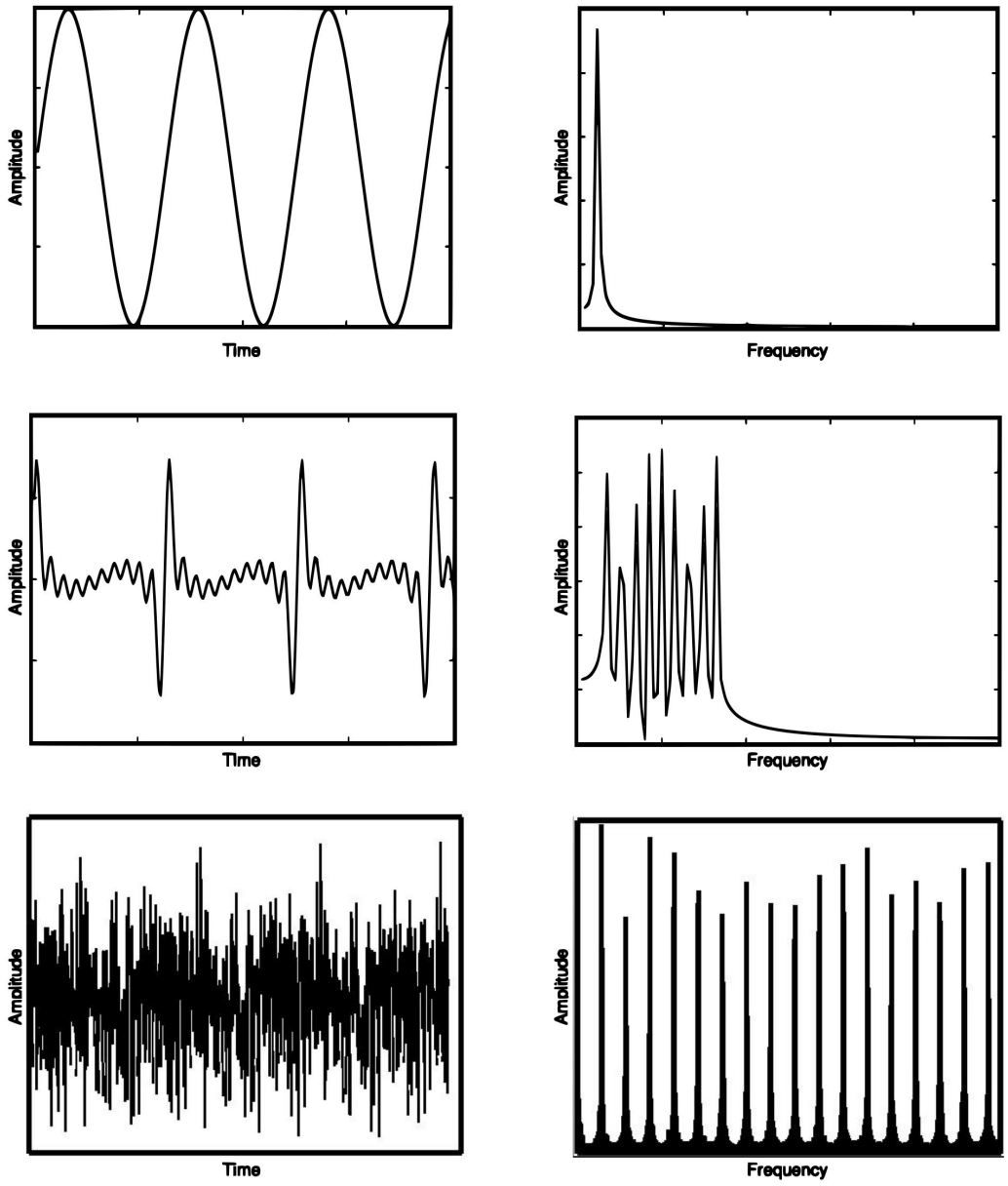

Figure 2.1: **Various auditory stimuli (waveforms left) which elicit a sensation of approximately the same pitch with the corresponding Fourier Transforms (right).** The top row shows a pure tone, the middle row shows a combination of 10 harmonics of the preceding tone without the fundamental frequency (i.e. a missing fundamental stimulus) and the bottom row shows an iterated rippled noise stimulus (30 iterations of addition).

of that sound. A perceived pitch is generally labelled by the frequency of a pure tone which elicits the same pitch sensation. Natural sounds which exhibit pitch are often composed of a particular fundamental frequency (F0) and frequency components which are integer multiples of this F0 (i.e. higher harmonics). We shall refer to such pitch stimuli as "F0-containing stimuli". However, the features which make pitch perception an interesting phenomenon are the range of instances in which a pitch sensation is elicited despite the sound having very different acoustic characteristics to the F0-containing case.

A curious case emerges when the fundamental frequency of a pitch is removed (Figure 2.1, middle row). This type of pitch sound is referred to as a missing fundamental (MF) sound. Although the resulting sound has energy only at higher harmonics, it nonetheless elicits a perception of pitch corresponding to a tone of the (missing) fundamental frequency. Another interesting sound which elicits a pitch percept is an iterated rippled noise (Figure 2.1, bottom row) [219]. Iterated rippled noises (IRNs) are sounds in which broadband noise is added to itself multiple times at a specific time delay. This time delay is the period of the pitch experienced by the listener. Thus, as the sound is made up of broadband noise, amplitude variations in the sound waveform remain largely irregular as the number of delay-and-add iterations are increased. Nonetheless, there are regular peaks in the frequency spectrum of these IRN sounds and listening to them produces a sensation of pitch.

2.1.1.2 Pitch Processing

Sound waves are first transduced into neural signals in the cochlea. In particular, incoming sounds cause vibration of a coiled structure in the inner ear called the basilar membrane. For a pure tone stimulus, the location of peak vibration along the length of the basilar membrane is approximately logarithmically dependent upon the frequency of the tone. Thus, the response from the basilar membrane is organised according to stimulation frequency, a feature called tonotopy. This vibration

stimulates the hair cells, cells embedded in the basilar membrane. Outer hair cells are believed to be related to amplification of sounds entering the cochlea. Inner hair cells transduce the vibration to then stimulate the auditory nerve. This stimulation is transmitted via the auditory nerve fibres to subcortical areas of the auditory brain, then onwards to the auditory cortex, with tonotopy persisting along this pathway. For an overview of cochlear and auditory processing, see Schnupp et al. [180].

The cochlea is thus often generally described as approximating a gammatone filter bank, with each auditory nerve fibre emitting stimulation tuned to a particular frequency. Due to the logarithmic spacing, the cochlear filters are more broad for high frequencies, such that only the lower harmonics (approximately the first 6-10 harmonics) of a pitch-evoking sound are resolved on this map [70]. Harmonics which lie above this range are generally difficult to distinguish due to the overlap and proximity of cochlea hair cell frequency receptive fields. In order to extract the pitch of a sound, the auditory processing system is therefore assumed to integrate information across cochlear frequency filters in order to compute the spacing between harmonics.

The place theory of pitch suggests that because hair cells and, in turn, auditory nerve fibres, are tonotopically mapped according to their preferred frequency, pitch is represented as the spatial pattern of excitation across this map [71, 39]. Temporal theories instead point out that as pitch-evoking sounds are periodic, their pitch can be derived from the temporal dynamics of cochlear activation. Due to the ability of neurons in the cochlea and auditory brain to produce action potentials that are phase locked to the envelope of a sound stimulus, the pitch of a sound can be determined from the dominant periodicity of spiking in responses that are pooled across the auditory nerve fibres (ANFs) [31]. The temporal theory can explain how we experience pitch even when only very high frequency harmonics (with unresolved low frequency components) are present in a sound [129, 181, 14]. However, temporal theory has more recently been argued to be insufficient to provide a complete

description of pitch [149, 183]. A combined model of both place and temporal pitch encoding is likely to be necessary to explain the full range of stimuli that evoke a perception of pitch, and these more complex models are being developed [148, 182].

2.1.1.3 Existing Models of Pitch Perception

Relatively few neural network based models have been published investigating mechanisms of pitch encoding/decoding. Among these, fewer still have been based upon unsupervised learning rules. In some cases [178, 196], a supervised network with backpropagation of error was implemented and tested. For models of pitch, this supervised training involves assigning output neurons a specific pitch 'category' across a scale and then during training stimuli with known pitch values are presented. As described in an earlier section, such a method of learning requires that specific neurons are "informed" that they must respond to a given stimulus and the neuronal weights are altered in a non-local method to ensure that this is achieved. This requirement for non-local information means that such models are often considered to be of limited plausibility. Furthermore, the fact that other primates and mammals have perceptions of pitch means that it is difficult to prove that external supervision could drive the development of such a network.

Recently an attempt was made to implement an unsupervised self-organizing map of pitch perception [220]. This model was able to form a fairly well defined representation of absolute frequency when trained with pure frequencies. After this training, the network was then tested with F0-containing harmonic tone complexes and missing fundamental stimuli and was not able to distinguish the pitches well and entirely unable to identify MF pitches. The network was then trained with harmonic tone complexes including their F0. The output network learned to classify the pitch of these harmonic tone complex, but only for higher pitched sounds. However, the model was not tested on MF stimuli or IRNs, so it is unclear if the network's pitch rules could generalize to other types of pitch-evoking sounds.

Another recent and particularly interesting attempt to represent pitch in a neural network is an implementation of a spiking neural network [111]. This paper implemented a combined place and temporal code in order to produce a representation of pitch. In this case, ANFs from all cochlear locations responding to the harmonics of a pitch were connected to a given co-incidence detector neuron by axons with finite and fixed delays based upon the expected delay between the phases of each harmonic. The fixing of axonal delays was an artificial means of producing these pitch representing co-incidence detector neurons, and it is not evidenced that such tuned and fixed axonal delays exist in the auditory system.

One particularly interesting model by Bumbacher et al. [28] attempted to describe auditory processing more generally. The network was composed of a two pathway generative model, one a fine-grained hierarchical sparse coding model and the second pathway providing coarse global spectro-temporal features by a method called gist [143]. Training was carried out with blocks of human speech and the resulting network produced non-linear features which the authors could link to pitch processing. Their model was shown to produce responses both sensitive to place and temporal features of pitch and so they argued in support of a mixture of these two principles for pitch extraction. Training with speech crucially allowed an unbiased investigation of the possible features to be extracted from such a natural dataset. However, their study did not provide a systematic analysis of the ability for the trained network to classify missing fundamental and iterated ripple noises.

2.1.2 Our Study

This investigation describes how neurons might be expected to form a representation of pitch when provided with input from a biologically realistic cochlear model. The model implemented was based upon information in the place code alone and was not extended to investigate how temporal information might contribute to the

resulting pitch representation. We propose that a simple system based upon place theory alone can produce neurons that identify the pitch of a range of complex sounds, including missing fundamental and IRN stimuli. This study and the results that follow were published as a paper in 2016 [3].

These results, and this chapter as a whole, act as an exhibition of the simplicity with which competitive learning can be implemented in a rate coded model. This provides a good starting point for our investigations which later extend unsupervised learning to spiking neural network models.

2.2 Methods

2.2.1 Cochlear Model

The Zilany et al. cochlear model [229] applied in this study has been shown to reproduce a multitude of response properties of the auditory nerve. For a review of the key processing components of the cochlea and methods by which the various biologically realistic models are constructed, see the review by Ni et al. [137]. Features which allow it to do so include: two modes of basilar membrane excitation contributing to inner hair cell (IHC) firing (via parallel filters and transduction through separate filters [228]) and power-law adaptation of IHC firing (as shown to be better suited to reproduction of physiological data when compared to alternatives e.g. exponential adaptation [227]). More recently still, tuning parameters based either on the cat cochlea or the human cochlea [229] have been made available. The implementation of the Zilany model was was provided by the online available *Cochlea* library [172]. This cochlear model was used as the first stage of our network such that it produced expected responses to various auditory stimuli which could be fed into the network neurons. For our rate coded implementation, we removed the fine structure of the neural firing patterns and defined the auditory nerve

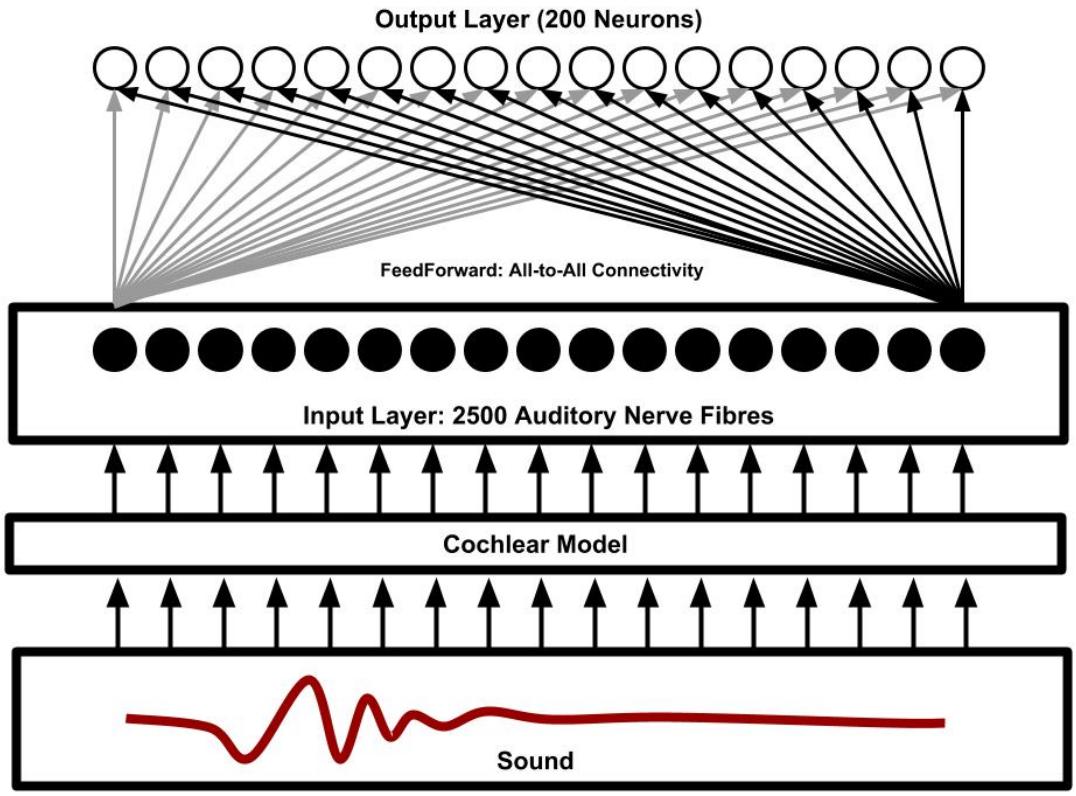

Figure 2.2: **The network structure implemented.** Auditory stimuli are processed by a cochlear model which provides the firing rates of 2500 ANFs. These 2500 ANF firing rates are used as the input layer to a network of 200 rate-based neurons.

fibre (ANF) firing rates as the average of their temporally varying firing rates over the entire time period of each stimulus.

Our simulations implemented the *Cochlea* model with 2500 ANFs in the range 125Hz to 20kHz.

2.2.2 Network Model

The rate coded competitive neural network produced is a 1-layer model (i.e. one layer of modifiable synapses) with full feedforward connectivity as shown in Figure 2.2. There is an input layer which represents the ANF firing rate output of the cochlea with associatively modifiable synaptic connections onto an output layer of neurons.

Competition is imposed between neurons in the output layer as described below.

The competitive mechanism, implemented alongside associative learning on the afferent connections to the output neurons, allows the network to self-organise pitch representations in the output layer. Such self-organisation is taken as a simple approximation of the function of early sensory cortices. The single layer architecture chosen is the minimal architecture necessary to capture these essential learning mechanisms and we show that through training it leads to the emergence of representations of pitch.

Individual output cells in this network learn to represent the pitch of the sound stimulus presented to the network through a form of coarse coding. That is, as pitch perception occurs over a continuum, neurons are expected to learn to respond over some interval of pitch with a peaked response curve.

During training and testing, a single input pattern is applied to the layer of input cells (the auditory nerve fibre layer) at a time. This input pattern is a single sound stimulus compressed into a snapshot of the average ANF firing rate response to that sound. The input pattern is applied by setting the firing rates of the input cells equivalent to the average ANF firing rates to the sound stimulus, as produced by the cochlear model. Next, activity from the input layer is propagated through the feedforward connections to activate the cells in the output layer.

The activation of the cells in the output layer are calculated according to

$$h_i = \sum_j w_{ij} r_j \tag{2.1}$$

where h_i is the activation of output neuron i, w_{ij} is the weight of the synapse from input neuron j to output neuron i, and r_j is the firing rate of input neuron j.

The activation h_i of each output neuron is then converted to its firing rate r_i using a threshold non-linear activation function. A sigmoid activation function is used, of the form

$$r_i = 1/(1 + e^{-2\beta(h_i - \alpha)}) \tag{2.2}$$

where r_i is the firing rate of the output neuron i, β determines the steepness of the sigmoid function, α is the threshold of activation of the sigmoid function, and h_i is the activation of the output neuron i.

Among neurons of the output layer, competition is implemented by shifting the threshold, α, of the non-linear sigmoid activation function. Adjustment of the threshold is used to achieve a prescribed sparseness of the representation in the output cells. This shifting threshold and resulting sparseness are a possible effect of the process by which mutual inhibition between the output cells, through inhibitory interneurons, could implements competition to ensure that there is a limited set of "winning" output neurons. If the output neurons of our network project to inhibitory interneurons with feedback inhibition, the amount of inhibition would be proportional to the output layer activity, just as our activation function threshold is proportional to the activity of our output neurons.

Sparseness is defined as the proportion [0, 1] of neurons that are active above threshold in the output layer. The sparseness was fixed for all simulations at 10%. This was achieved by determining the threshold of the activation function at which this sparseness would be achieved. As there were 200 output neurons and we desired 10% sparseness, the threshold α of the activation function in equation 2.2 was set equal to the activation of the output neuron with the 20th highest computed activation h. The threshold implemented was the same for all output neurons during a single stimulus presentation and redefined on each successive presentation.

Varying the sparseness parameter had no significant effect upon the network performance and only adjusted the number of neurons which collectively responded to a given stimulus category. The learning rate and sigmoid slope used for training, shown in Table 2.1, were identified through a grid search of parameter space and

achieved close to optimal network performance on this task after training.

During training, the synaptic weights between the active input cells and the active output cells are strengthened by associative (Hebbian) learning. The output cells self-organise such that they have preferred patterns of activity in the input layer. The associative Hebb learning rule is defined

$$\delta w_{ij} = k r_i r_j, \tag{2.3}$$

where δw_{ij} is the change in the weight of the synapse between input neuron j and output neuron i, k is the learning rate constant, r_i is the firing rate of the output neuron i and r_j is the firing rate of the input neuron j. To incorporate synaptic weight reduction and to ensure that each neuron has an equal input stimulation magnitude, the synaptic weight vectors are scaled to unit length (i.e. weight vector normalization) after the presentation of each training pattern. To implement weight vector normalization, for each output cell i we calculate the total weight vector length and divide the weight values by this length to ensure that

$$\sqrt{\sum_j (w_{ij})^2} = 1 \tag{2.4}$$

where the sum is over synapses from all input cells j to a given output neuron i. This re-normalization is carried out for each output neuron. Such a re-normalization process is evidenced in cortex by observations of a similar process in coronal slices of the amygdala [169]. Such a mechanism would require interaction between the synapses across a cell and is therefore an augmentation to our network setup.

The competitive network contained 2500 input cells, 200 output cells and was fully connected. The synaptic weights from the input to output cells were initially set to random values from a uniform distribution in the range [0,1], and the weight vectors of the individual output cells are normalized to a vector length of 1. Each simulation employed a set of pitch stimuli (see Section 2.2.3) spanning a range of

Parameter	Value
Learning Rate	0.25
Sigmoid Slope	17.45
Sparseness	10%
Number of Training Epochs	50

Table 2.1: **The parameters used for the simulations of pitch processing in a rate-coded competitive neural network described in Section 2.3 (except where stated otherwise in text)**

fundamental frequencies from 200Hz to 600Hz. These pitch stimuli were individually presented to the network after being processed through the cochlear model. For each stimulus, presentation involved calculating the average firing rates of the 2500 auditory nerve fibres (ANFs) in the cochlear model, and setting the firing rate of the input cells to the model according to the normalized average firing rate of their corresponding ANF.

The presentation of all possible stimulus training patterns (in a randomized order) corresponds to one training epoch. Unless stated otherwise in the text, the parameters shown in Table 2.1 were used for simulation and there were 50 training epochs to ensure convergence of synaptic weights.

2.2.3 Sound Stimuli

The sound stimuli presented to the network were of three main types: F0-containing harmonic tone complexes, missing fundamental harmonic tone complexes, and iterated rippled noises. All of these stimuli were produced such that the fundamental frequencies, F0, varied from 200Hz to 600Hz in 20Hz increments. The particular value of the fundamental frequency is what we refer to as the 'category' of a pitch sound. In our stimulus set, we therefore had 21 pitch categories (between 200Hz and 600Hz) with an F0-containing, missing fundamental, and iterated rippled noise stimulus in each of these categories. In particular, we only ever trained the networks with F0-containing stimuli and then tested with all three types of stimulus:

F0-containing, missing fundamental, and iterated rippled noise.

The F0-containing stimuli all contained 10 harmonics (including F0). Missing fundamental stimuli contained no F0 component but did include the 9 subsequent harmonics. Finally, Iterated rippled noises were created by producing a random broadband noise of length 30s and adding a delayed copy of the waveform to itself at delays corresponding to the period of the desired fundamental frequencies 200Hz to 600Hz. This process of delayed addition is repeated 30 times to produce the iterated rippled noise stimulus from which a 1s sample is then used as a sound stimulus. These sounds were all presented to the cochlear model scaled to 50 dB SPL with cosine onset and offset ramps. The cochlear response model produced a time-varying firing rate response for each ANF and these were averaged over time to produce a static firing rate pattern for each stimulus which was presented to the network during training and testing.

The harmonic amplitude profiles of the F0-containing stimuli and missing fundamental stimuli were the only properties varied. Rather than assuming a fixed harmonic amplitude profile, we tested the network with a set of variable profiles (Figure 2.3). These harmonic amplitude profiles were chosen to test how well the network could recall the pitch of a sound when trained on sounds with varied relative power in their harmonics. During each training epoch, the network was presented with a single set of the 21 F0-containing stimuli (all with the same harmonic amplitude decay profile) with F0 values in the range 200Hz and 600Hz. These stimuli were presented in a randomised order in every training epoch with no pitch presented twice.

The network was finally tested, after a number of training epochs, with stimuli of the same pitch (F0 values) it was trained with. In this case we first tested to see whether the neurons were able to distinguish the training set by presenting F0-containing stimuli and observing the output. Finally, missing fundamental and iterated rippled noise stimuli were presented and the output once again observed

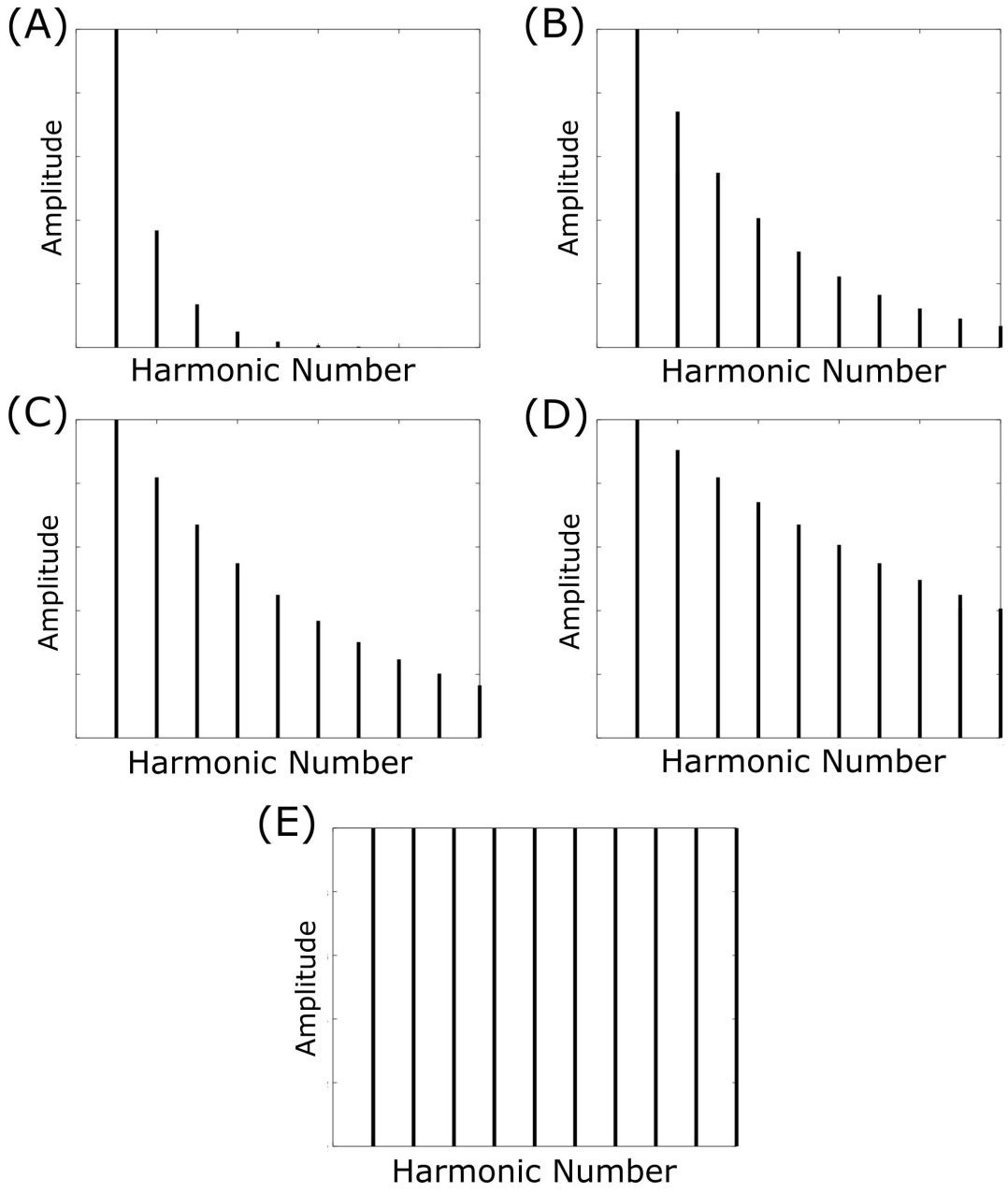

Figure 2.3: **The harmonic amplitude profiles with which the network was trained and tested.** The decays in amplitude of the harmonics are such that their amplitude is multiplied by the function $\exp(-f/\tau)$ with varying values of the decay constant τ. Decay constants for the various profiles are as follows: (A) τ=$F0$; (B) τ=$(10/3)F0$; (C) τ=$(10/2)F0$; (D) τ=$(10)F0$. Profile (E) has no decay over harmonic amplitudes and therefore has a constant amplitude over harmonics.

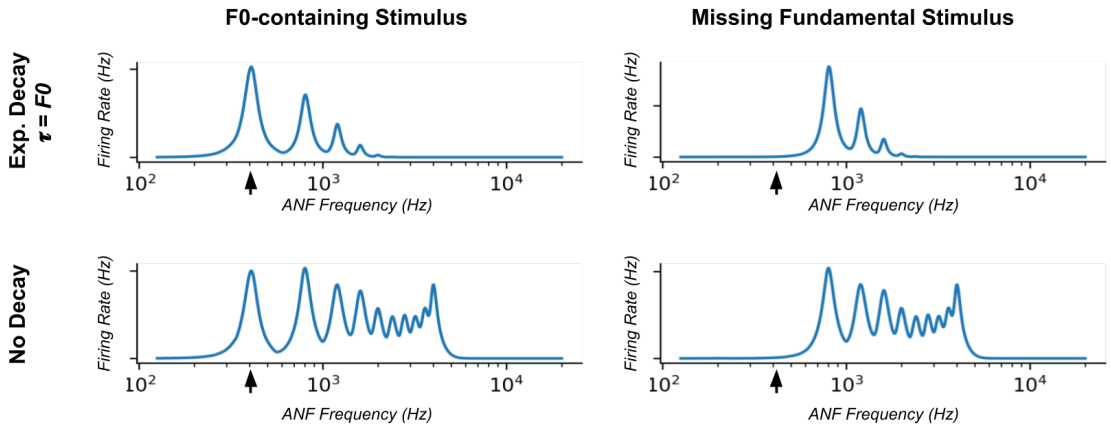

Figure 2.4: **A set of example auditory nerve fibre activation profiles for a stimulus with 400Hz fundamental.** The left columns show F0-containing stimuli and the right column shows missing fundamental stimuli. The upper row shows stimuli in which the harmonics of the sound stimuli have an exponential decay with time constant $\tau = F0$, see Figure 2.3A. The lower row shows stimuli for which there is no harmonic decay, see Figure 2.3E. A black arrow shows the location of the F0 frequency.

to check whether our network was capable of correct pitch categorization and generalization from training on only F0-containing stimuli.

Figure 2.4 shows an example of the auditory nerve fibre firing rate distributions produced for the F0-containing and missing fundamental stimuli. These are shown for two different harmonic decay conditions and illustrate the form of the inputs to our networks for this study. It is also important to note that there are no distortion products produced by this cochlear model [159]. In short, distortion products are locations of inner hair cell excitation which appear in the presence of multi-tone sounds but which are not present on any of the single-tone presentations. These distortions are a consequence of the non-linear dynamics of the cochlear apparatus and amplification processes [174]. We discuss the implications of the lack of distortion products of our chosen cochlear model in the Discussion, Section 2.4. In all results presented hereafter, training was carried out only with F0-containing stimuli.

2.2.4 Single Cell Information Analysis

A single cell information measure was applied to analyse output neuron performance [165]. This measure determines how much information an individual neuron's activation provides about the category to which a given stimulus belonged. By category, we mean the frequency of the would be F0 component of that sound. This choice of categorisation determines our calculations of information in this study. Given that we present sounds of F0 frequency between 200Hz and 600Hz (inclusive) with a 20Hz interval, there are 21 unique F0 frequencies and therefore 21 categories.

The calculation of information is based upon Shannon's information theory, describing the amount of information, $I(s, r)$, in bits, that a set of possible responses, R, give about a stimulus category, s, as

$$I(s, R) = \sum_{r \in R} P(r|s) \log_2 \frac{P(r|s)}{P(r)}, \tag{2.5}$$

where r is the response type from a discretized set of possible responses, R, of a particular neuron. Given that our rate-based neurons produce a real valued output, we must choose how to discretize their response. In our case, we binarise the responses based upon the neuron's mean firing rate across all stimuli presented (including all F0-containing and missing fundamental stimuli) – one possible response, r, is when the neuron has a higher than mean firing rate, and below mean firing rate constitutes the second possible response. This information measure is calculated based upon the probabilities of these two types of response across the presentation of a full set of stimuli of F0-containing and missing fundamental stimuli with a range of harmonic decays, and also in some cases with IRN stimuli.

The information content of a single neuron would-be greatest in this study when it is responsive to all sounds which elicit a given sensation of pitch (i.e. the F0-containing, missing fundamental, and IRN stimulus associated with the same F0) without being responsive to sounds which elicit a different sense of pitch (of a

different associated F0). This is due to our choice in making the stimulus categories equal to the expected pitch elicited, regardless of the sound type. The maximum amount of information, I_{max}, which the responses from a single cell can represent is

$$I_{max} = \log_2 N \tag{2.6}$$

where N is the total number of stimulus categories (21). Cells will be expected to increase their amount of information significantly through training though they may not reach this maximum given they are expected to form a coarse coding arrangement, discussed above.

2.3 Results

2.3.1 Training with Various Harmonic Amplitude Profiles

The key question in this study was whether the network trained on harmonic complex tones that included F0 could generalize pitch categorization to missing fundamental stimuli. We began by analysing the network response when it was trained and tested with no harmonic decay (Figure 2.3(E)). Simulation results are shown in Figure 2.5. The "block"-like structure observed in these responses (groups of neurons responding to a single stimulus as a set) is a coarse coding of the pitch stimulus which emerged through training. The size of groups depend upon the sparseness value imposed upon the output layer of the network. Figure 2.5(A) shows that, through unsupervised competitive learning, the network was able to produce output neuron responses such that each output neuron is selective to a small range of F0-containing pitch sounds.

This property of Figure 2.5 shows that the network converts the input frequency representation (which has a series of peaks dependent upon the specific pitch stimulus) into a well defined representation of pitch category such that individual

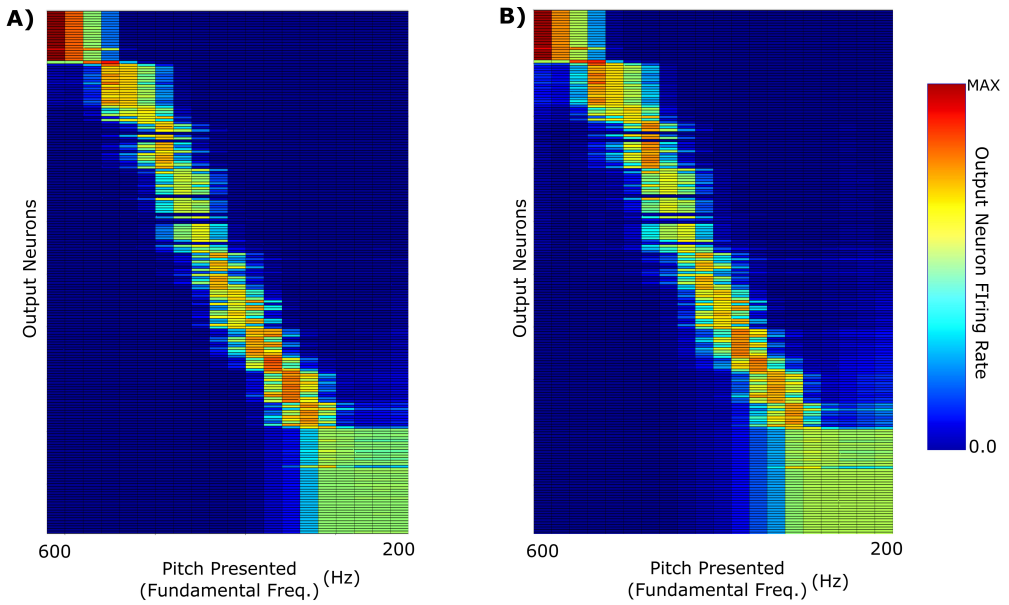

Figure 2.5: **Responses of output neurons after training the network on pitch stimuli with no decay in the harmonic profile.** The neuronal firing rate profiles were sorted along the y-axis after the simulation by the location of their response peak. This sorting is only carried out for subplot (A) and then the order is maintained for subplot (B). **(A)** The network response when it was trained and then tested with F0-containing stimuli with no decay in the harmonic profile. **(B)** Response when trained with F0-containing stimuli and then tested with missing fundamental stimuli (again of no harmonic decay).

output neurons are tuned to a small range of pitch sounds. This same network when tested with missing fundamental stimuli of no harmonic decay (Figure 2.5(B)) has a similar response style across all stimuli. Importantly, the same neurons respond to both F0-containing and MF sounds of the same pitch and therefore generalize from their training. This was facilitated by the fact that no single harmonic in the training of this network had a greater amplitude than any other. Thus, its training did not place a disproportionate amount of importance upon the (now missing) fundamental frequency.

Despite the success at the pitch sounds characterised by a high F0, one notable feature (of all the results shown in Figure 2.5) is the lack of distinct structure in neuron firing below 300Hz fundamental frequency. Due to the logarithmic distribution of frequencies across the cochlea, pitch stimuli of low fundamental

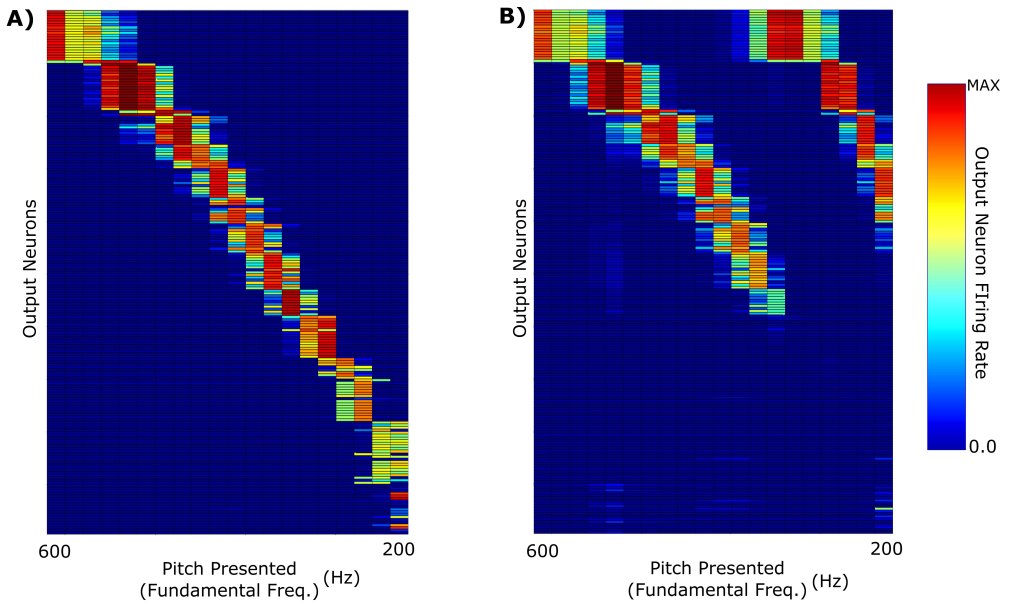

Figure 2.6: **Responses of output neurons after training the network on pitch stimuli with strong decay in the harmonic profile.** As with Figure 2.5, the neurons are sorted along the y-axis after testing by the location of their response peak. This sorting is only carried out for subplot (A) and then the order is maintained for subplot (B). **(A)** The trained network response when tested with full F0-containing pitch stimuli of harmonic profile with exponential decay ($\tau = F0$). **(B)** Response of trained network when tested with missing fundamental pitches of harmonic exponential decay ($\tau = F0$).

frequencies have highly overlapping higher harmonics. Given that this training procedure weighted all harmonics equally during training, overlap in input activity is significant for sounds characterised by low F0 and the network is unable to produce a weight structure which can distinguish these stimuli given the training set.

We continued this investigation by training and testing the network with sounds in which the fundamental frequency of the pitch was greatest in amplitude. The higher harmonics had reduced amplitudes which were modified by an exponential decay with decay constant equal to the fundamental frequency of the pitch (Figure 2.3(A)). Figure 2.6(A) shows the network's response to the F0-containing stimuli which composed the training set. The network's distinct responses to stimuli with fundamental frequency greater than 400Hz appears to be similar to the network when trained on stimuli of no harmonic decay (Figure 2.5(A)). However, at the

lower fundamental frequency range (<300Hz) this training regime produces more distinct, pitch-specific, responses. This can be attributed to the network's response being heavily biased toward the strength of the fundamental frequency. Thus, a network trained and tested on F0-containing pitch stimuli with sharply decaying higher harmonic amplitudes is better suited for low frequency pitch discrimination.

However, as can be observed in Figure 2.6(B), decaying harmonic amplitude training and testing does not allow a generalization of pitch representations from F0-containing harmonic tone complexes to MF sounds. Training the network with stimuli so heavily weighted to their fundamental frequencies results in a network almost solely dependent upon a single frequency component for discrimination. Thus, removal of the fundamental frequency (as in the MF case) means that the stimulus is discriminated based upon the next harmonic of highest intensity – a harmonic at double the fundamental frequency. This results in each pitch stimulus appearing to have a fundamental frequency that is double the true fundamental frequency and the network misidentifies the pitch of the missing fundamental stimulus as a single octave above what it should be. For example, neurons responsive to a sound with a pitch of 400Hz in F0-containing harmonic tone complexes, also respond to MF sounds of 200Hz. Thus, training a network with stimuli of sharp harmonic amplitude decay does not lead to the development of output neurons that correctly identify MF stimuli.

In this process of testing and training the networks, the question arose as to what form of harmonic decay profile would be the limit beyond which a MF stimulus would give rise to an incorrect pitch assignment. This was investigated by training and testing the network with stimuli of different harmonic decay profiles as shown in Figure 2.3(B-D). It was found that the harmonic profile for which the decay of the harmonic amplitudes followed $\exp(-f/(10 \cdot F0))$ (Figure 2.3(D)) was the point of change. A larger decay constant allows the neurons to learn a more accurate identification of the MF pitch stimulus. A smaller harmonic decay

constant leads to the network misidentifying low frequency missing fundamental stimuli (<400 Hz) as one octave too high.

In order to replicate the mechanisms by which the brain might learn pitch stimuli more realistically, the network architecture was trained with pitches of harmonic decay which were randomly varied across stimuli on every epoch of training for 100 epochs. The variation in the harmonic decay was achieved by setting the decay constant $\tau = F0 \cdot \exp(x)$ where x was a uniformly distributed real number between zero and ten which was randomly sampled for each stimulus presentation. Thus, the training stimuli varied from sharp exponential decays to almost constant amplitude stimuli. It is important to note that having the decay constant chosen randomly from an exponential distribution resulted in training with many instances of sounds with a relatively small decay constant and fewer instances of sounds with very large decay constants. This corresponds to relatively little training with stimuli of shallow harmonic decay and more training with stimuli of sharp decay. This style of training was required in order to develop the results which follow. In this regime, the stimuli were different for every single epoch with no stimulus of a given decay repeated during training.

This training resulted in a network response (Figure 2.7) with responses tuned across the F0 range for both F0-containing and MF stimuli. Crucially, exponentially decaying and constant amplitude stimuli representations are equally well distributed, though there is a degree of noise especially for MF stimuli. The network shows a greater discrimination of stimuli with low fundamental frequency (<300Hz) when tested with constant amplitude pitches compared to previous training attempts (Figure 2.7(C) compared to Figure 2.5(A)). This difference is attributed to a more appropriate weighting of higher harmonics given the diverse training. Another important observation is that the exponentially decaying stimuli have largely overcome the issue of pitch octave misidentification. This result emphasises the importance of training the auditory brain with pitches of differing harmonic

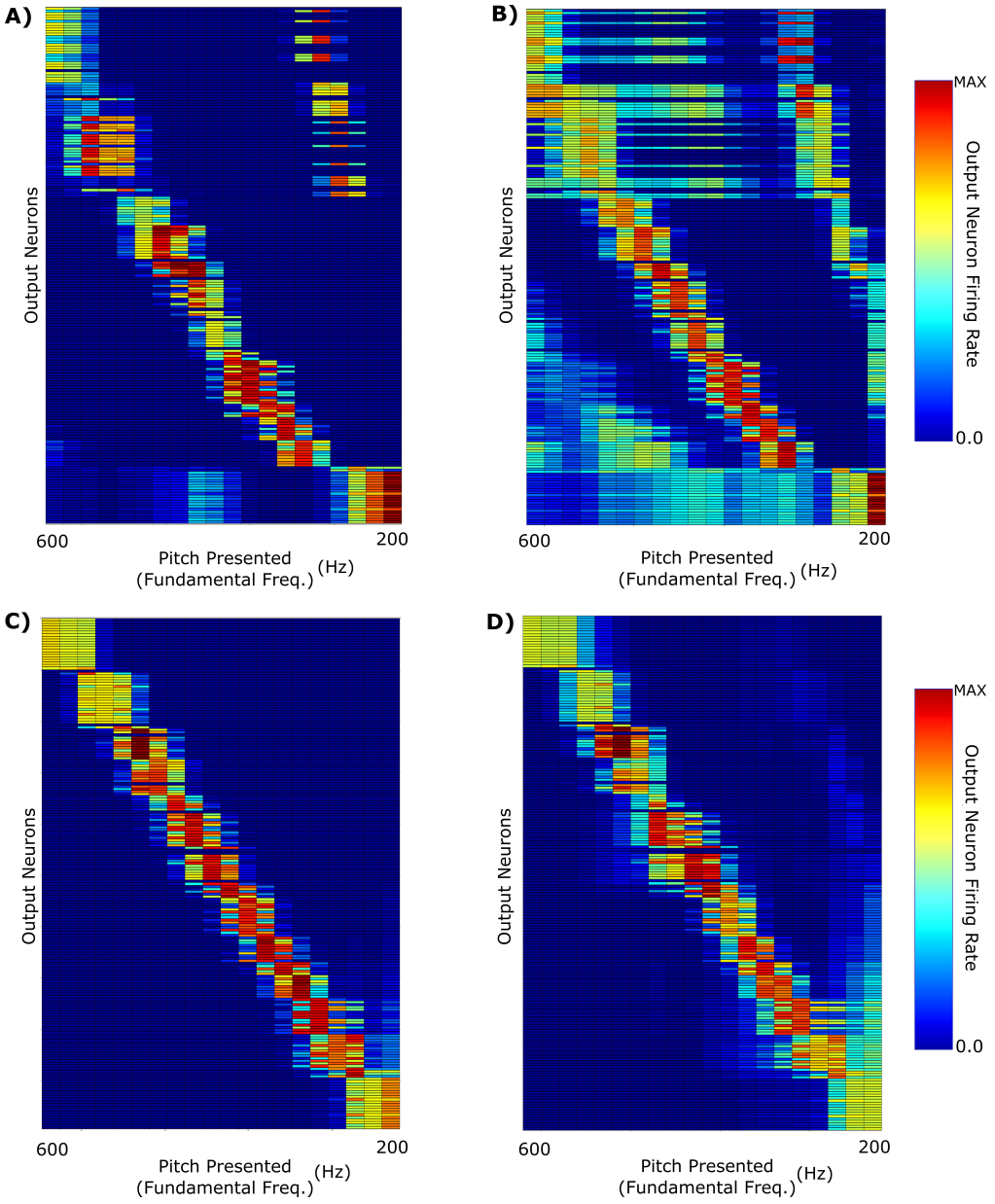

Figure 2.7: **Responses of output neurons after training the network on pitch stimuli with randomly selected harmonic decay constants.** The trained network responses when testing the network with: **(A)** F0-containing stimuli with Harmonic decay constant $\tau = F0$; **(B)** The same stimuli as (A) but with missing fundamentals; **(C)** Constant Harmonic Amplitude stimuli; **(D)** The same stimuli as (C) but with missing fundamentals.

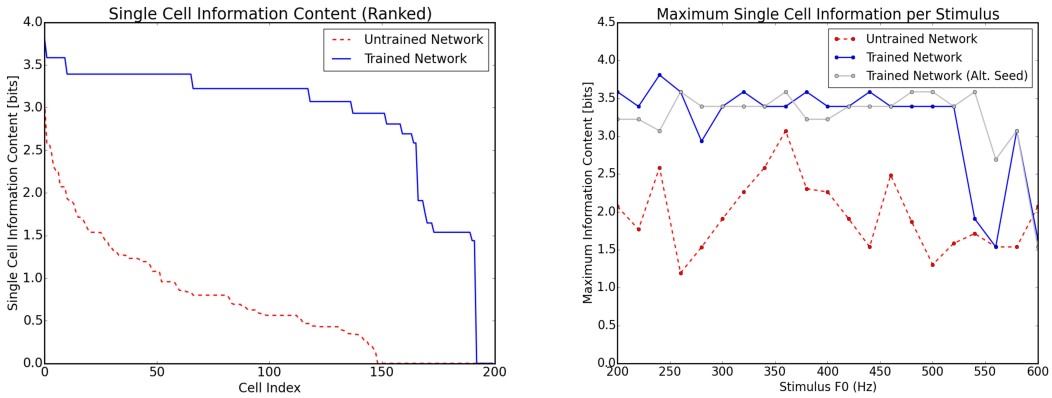

Figure 2.8: **Information theoretic analyses for the network trained on randomly selected harmonic decay profiles.** The analyses are calculated using the responses shown in Figure 2.7. Left, the ranked single cell information for the output neurons before and after training. Right, the maximum amount of information that any single cell has with respect to each stimulus in the investigated frequency range. The trained and untrained cases are shown with analysis of a network trained with a different set of initial random weights (with a different seed for the number generator). This illustrates that some of the variation in information is due to the specific random seed.

decay profiles. Failure of pitch processing under impoverished training sets could be tested in psychoacoustic experiments in order to confirm the importance of this training diversity and to replicate the failure modes.

When single cell information theoretic analyses were applied to the data shown in Figure 2.7 it showed that there was a significant increase in single cell information from the untrained to the trained case (Figure 2.8, left) as expected. This analysis was applied by binning the responses of each cell to each stimulus into one of two types. All cells with firing rate greater than 0.5 were placed into one bin and cells with lower than 0.5 firing rate into the other. The choice of binning can highly affect the amount of information extracted from neural responses. In order to ensure that we do not bias the information analysis outcome, we selected this simple binarisation. The information present in single neurons with respect to stimulus frequency was also shown to be close to maximum for the majority of the frequency range (Figure 2.8, right). This plot also shows a comparison of the network described in this

section to an identical network in which a different random seed was used such that the initial random weights and stimulus presentation order during training were different. No other components of the network were changed. As can be observed, a different random seed produces a variation in the information across stimulus frequency, F0, and therefore the variation in information can be largely attributed to this random seeding. One other contribution can be seen for frequencies above 540Hz. There is some drop in single cell information which can be traced to the response of cells to adjacent pitch stimuli above 540Hz being somewhat similar. Nonetheless, the neuron responses to these stimuli are observed to differ in the specific firing rates, see Figure 2.7. The drop in information therefore is also a reflection of our choice of simple binarisation of neuron responses, which does not capture all information present in the neural responses. Furthermore, it is important to note that by use of single cell information, we also ignore information which is decodable from the firing rates of multiple cells.

A further test carried out with this network was how well it could categorise the pitches of iterated rippled noises (IRNs) (Figure 2.9). The network producing these responses is the network trained upon randomly chosen harmonic decay profiles as described above, the same network used to produce Figure 2.7. Despite the network having never been trained with IRNs, the output shows an ability to discriminate the various pitch categories of the IRN stimuli in a similar arrangement to that seen for the more conventional harmonic pitch stimuli. The noise present in the network responses can be attributed to the fact that the network was only trained with 10 harmonics for each pitch and the IRN stimuli contain frequency components at many higher harmonics as well as non-zero firing rates across the entire frequency range. In order to assess this network's ability to identify the pitches of individual IRN stimuli, the single cell information theory was re-calculated with the IRN stimuli responses included. As can be observed in Figure 2.10, including the IRN responses produces maximal information in neural responses which is very similar

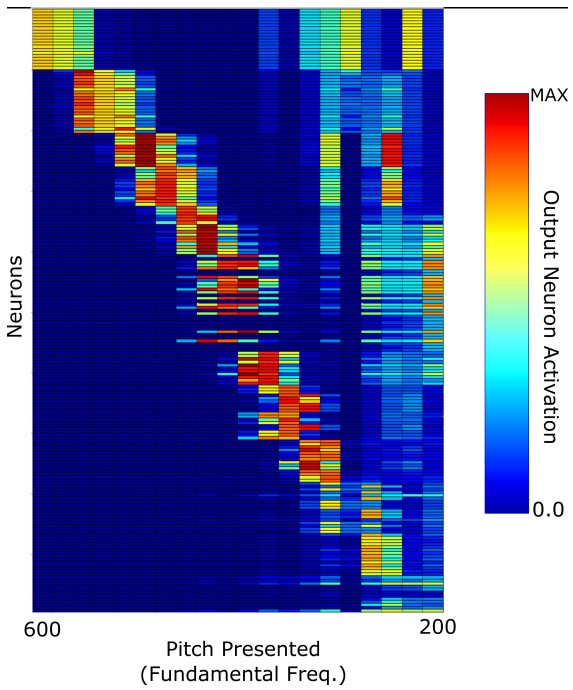

Figure 2.9: **The network response to iterated rippled noise stimuli when the network was trained with harmonic tone complexes of randomly selected harmonic decay constants.** Note that the neurons are ordered in the same order as used for Figure 2.7

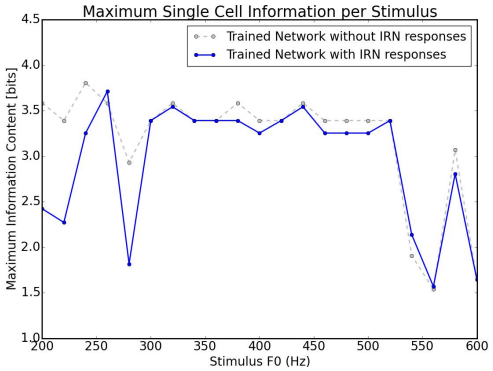

Figure 2.10: **Information theoretic analyses for the network trained on randomly selected harmonic decay profiles and tested using both harmonic stimuli and IRN stimuli.** Results are calculated using the responses shown in Figures 2.7 and 2.9. This plot shows the maximum amount of information that any single cell has with respect to each stimulus F0 in the investigated frequency range. The trained network is analysed with the responses to harmonic stimuli shown in Figure 2.7 and with the addition of the responses to IRN stimuli shown in Figure 2.9, grey and blue lines in this plot respectively. As can be observed, including IRN responses does not significantly reduce the information content across most of the frequency range, and only reduces the information content of cells informative at low frequencies.

to the information when we considered F0-containing and MF stimuli, compare Figure 2.10 and Figure 2.8 right.

Therefore, the network performance when tested upon IRNs is very similar to its response to more typical harmonic pitch associated sounds (complex tones).

Overall, the network described in this chapter shows an ability to categorise the pitches of both MF stimuli and IRNs despite never having been trained with examples of these sounds. Furthermore, since this network uses no temporal cues to categorise pitch stimuli, the results support a place theory of pitch via a very simple competitive learning mechanism for the frequency range investigated.

2.4 Discussion

In this chapter, we attempt to create the first entirely unsupervised competitive neural network structure which can learn to identify the pitch category of F0-containing stimuli, missing fundamental (MF) stimuli, and iterated rippled noises (IRNs) when presented with biologically realistic inputs from a simulated cochlea. A biologically plausible cochlear model is used as an input [227, 229] to our rate-coded competitive neural network, and we describe the outcomes of training the networks with a range of training sets. This network is found to perform the task of pitch identification well following training but its performance crucially depends upon the harmonic decay profiles of the stimuli with which it is trained. We find that training the network on F0-containing harmonic tone complexes with a range of harmonic decay profiles is key to the ability of the network to learn the robust determination of pitches when MF stimuli or IRNs are to be deciphered. In particular, this training set was biased toward harmonic decay profiles with little decay and this form of training proved most successful. Beyond F0-containing harmonic tone complexes, with which the networks were trained, single cell information analysis is used to show a high level of information in single cell responses for stimuli

distributed across the set of pitch stimuli used for training, including when the network is tested using iterated rippled noises.

The relative success of this network suggests that for pitch stimuli in this frequency range, a place theory of pitch is sufficient. The successful training of the network presented in this chapter was down to the effective network mechanisms for both learning and competition. These mechanisms are approximations of the learning rules and competitive components of real neural circuits and it is unclear how these mechanisms can be bridged to biologically realistic mechanisms in more detailed neural network models.

The explanation of pitch processing provided by investigation with this model provided some explanations of the generalisation errors and failure modes which arise when the network is trained under different harmonic decay profiles (see Figure 2.6) One potential source of defence against these failure modes are the distortion products which have been observed to be produced in the cochlea [187]. As described in the Section 2.2.3, the cochlear model which was used is biologically realistic, however distortion products are not modelled in this system. Without these distortion products, we cannot provide any indication of how these might provide robustness to the failure modes discussed. It is possible that with these distortion products, the requirement for a diverse training procedure for generalisation across missing fundamental and iterated rippled noise stimuli might be relaxed.

Beyond the limitations in the cochlear model, the training/testing dataset and model construction are limited in a number of ways. First, though the model produced neural responses which we could relate to pitch 'categories', there are many other aspects of natural sound which are both perceived and behaviourally relevant (e.g. timbre, frequency sweeps etc). Interrogating how this model might represent such perceptual qualities of sound would require a diverse training dataset of natural sound.

In order to train this model with natural sounds, a number of adaptations

would be necessary. First, the model accepts rate based inputs which we took as the average firing rate of the auditory nerve fibres over the course of a stimulus presentation. Therefore, cochlear responses to natural sounds would then need to be binned into segments, each of which could be independently presented to the network for training. However, given that this network model is not recurrent and has not temporal dynamics in firing rate, the resulting neuron receptive fields would be tuned to static patterns of the input.

In order to extend this model beyond static firing rate patterns, it would require the addition some mechanism for temporal processing of sound. This could include recurrent connections, time continuous firing rates, or as was applied by Bumbacher et al. [28], some form of coarse spectro-temporal input to the network with every static pattern. Such an extension would require significant thought and consideration but could also shed more light on the effectiveness of explanations of auditory feature encoding with simple unsupervised competitive network models.

3

Optimising Spiking Neural Network Simulation

3.1 Introduction

The description of electrical activity in neural systems as the firing rates of neurons is an abstraction. Biological neural circuit activity is not continuously propagated but instead is propagated by the emission of discrete action potentials, spikes, by individual neurons at specific times. Spiking neural network models simulate such detailed neuron behaviours and produce networks in which neurons emit action potentials with associated timings. In this chapter we explore an example of the processes by which a spiking neural network of point-neurons can be modelled and simulated. We thereafter describe the construction of a novel spiking neural network simulator which achieves state of the art performance by leveraging modern

hardware and integrating a number of optimisation mechanisms.

3.1.1 Spiking Neural Network Models

Spiking Neural Network (SNN) models generally consist of three main components: a neuron model describing changes to the neuron state variables; a synapse model describing the process by which synaptic connections affect post-synaptic neuron states; and plasticity rules describing the dynamics of synaptic weight change.

Neuron, synapse, and plasticity models can be defined with a range of biophysical detail. The most detailed of models describe the dendritic morphology and individual ion-channel dynamics. The most abstract models generally instead consider point-neuron models with a single state variable and pulse based inputs from synaptic connections.

In this thesis, we consider only point-neurons – in particular we consider leaky integrate and fire (LIF) neurons. These are implemented primarily with conductance-based synaptic inputs, though we describe both conductance-based and voltage-based synaptic connections in this chapter. Following a description of neuron, synapse, and plasticity models we describe how neural circuits are simulated.

3.1.1.1 Leaky Integrate and Fire Neurons

The leaky integrate and fire (LIF) Neuron model is one of the most common models of neural membrane voltage dynamics. This point-neuron model has a single time-varying state variable, the cell membrane voltage $V(t)$, which decays over time to a rest voltage V_{rest}. The membrane time constant, τ_{mem}, dictates the time course of this decay and is related to the cell's membrane capacitance C_{mem} and leakage conductance g_{leak} such that

$$\tau_{\text{mem}} = \frac{C_{\text{mem}}}{g_{\text{leak}}}.$$

Any incoming currents to a cell modify its membrane voltage and either hyperpolarize it, lowering its membrane voltage, or depolarize it, increasing its membrane voltage.

If we label the total incoming input currents to the cell as $I(t)$, we can represent the complete LIF neuron dynamics as

$$\tau_{\text{mem}} \frac{dV(t)}{dt} = (V_{\text{rest}} - V(t)) + \frac{1}{g_{\text{leak}}} I(t). \tag{3.1}$$

These dynamics do not describe the process which leads a neuron to emit an action potential. This is achieved by instead checking the membrane voltage against a threshold V_{thresh}. Whenever the neuron membrane voltage crosses the threshold V_{thresh} from below, the time of this threshold crossing is recorded as a spike-time and the membrane voltage reset is V_{rest}.

Following an action potential, neurons generally experience a refractory period of a few milliseconds within which the cell has become hyperpolarized and is unable to fire another action potential. This refractory period t_{ref} is another added component to these dynamics. Following a neuron reaching the action potential threshold, the membrane voltage of the cell is fixed at V_{rest} for a duration t_{ref} imposing an absolute refractory period after which the membrane voltage dynamics resume.

This neuron model, though simple, captures the minimal components necessary to produce action potentials and spiking dynamics. Thus, it can be used to interrogate spike-timing based networks.

3.1.1.2 Synaptic Inputs

The input current $I(t)$ to a neuron generally models changes in the activation of a cell's ion-channels. Such changes are often described as occurring at synaptic terminals for spiking neural networks and are dependent upon the neurotransmitters released by pre-synaptic neurons when they emit action potentials. It is important

to note that such a model of synaptic input does ignore other impacts upon a cell's membrane voltage such as the direct electrical excitation of cells which is possible through gap-junctions, or the impact of local population increases in activity and correlated changes in an individual cell's membrane voltage [134, 211]. However, considering solely the chemical neurotransmitter based interactions between cells is an abstraction of real neural circuits which enables us to model a subset of their dynamics.

In general, we can divide such synaptic effects upon an individual cell's membrane voltage into two groups: excitatory and inhibitory. Excitatory synaptic connections are those which have a depolarizing effect upon post-synaptic neurons and inhibitory synaptic connections cause hyperpolarization. Outgoing synapses from a given cell are consistently either excitatory or inhibitory depending upon the cell type. This separation of excitatory and inhibitory neurons is known as Dale's law and spiking neural networks in this thesis strictly obey Dale's law.

As with all models, the effect of activating neurotransmitter based synaptic connections upon the post-synaptic neuron's membrane voltage can be defined across a range of biophysically detailed scales. The simplest model of synaptic activation is described as causing an instantaneous but finite increase in the post-synaptic cell's membrane voltage. Such a synapse is hereafter referred to as a voltage-based synapse. Alternatively, the contribution of an activated synapse to the post-synaptic cell membrane potential can be described as dynamic with its own internal state variable. In some cases this can lead to synapses which have dynamics which are coupled to the post-synaptic neurons membrane voltage. This is more realistic when considering synapses as emitting neurotransmitters to activate ion-channels which cause a change in relative ion-concentrations in a post-synaptic cell. An example of such a model which we apply in this thesis is the conductance-based synapse model.

Voltage-Based Synapses

The simplest synaptic input method involves the instantaneous injection of current, and a corresponding instantaneous change in voltage, of the post-synaptic neuron's membrane voltage when an action potential from a pre-synaptic neuron reaches the synaptic connection. The magnitude of the change in membrane potential of the post-synaptic neuron is proportional to the synaptic weight.

This can be described such that for a given synaptic connection the current contributed to the post synaptic neuron, I_{syn}, is described

$$I_{\text{syn}}(t) = w_{\text{syn}} \cdot \sum_{f} \delta(t - t^{f}_{\text{pre}}), \tag{3.2}$$

where w_{syn} is the weight of the synaptic connection, $\delta(\cdot)$ is the Dirac delta function, and t^{f}_{pre} is the set of times at which spikes arrive at the synaptic connection from the pre-synaptic neuron, each spike having a unique time indexed by f. The total input to a post-synaptic cell from S such voltage-based synapses can be described

$$I(t) = \sum_{\text{syn}}^{S} I_{\text{syn}}(t). \tag{3.3}$$

Conductance-Based Synapses

Individual conductance-based synapses, whether excitatory or inhibitory, produce an impact upon a post-synaptic neuron's membrane voltage which depends upon a time-varying conductance value of the synapse, $g_{\text{syn}}(t)$, and the reversal potential of the specific synapse type, E_{syn}. For excitatory synaptic connections, this reversal potential is a value which is larger than the spiking threshold V_{thresh}. Conversely, inhibitory synaptic connections are characterised by reversal potentials which are lower than the rest voltage, V_{rest}, of the cell. These together produce a

time-varying input to the post-synaptic neuron such that an individual synapse's contribution to its post-synaptic cell, $I_{\text{syn}}(t)$ is described as

$$I_{\text{syn}}(t) = g_{\text{syn}}(t) \cdot (E_{\text{syn}} - V(t)). \tag{3.4}$$

Equation 3.4 shows that conductance-based synaptic connections provide an input to a cell which is proportional to the difference between the cell's membrane voltage and the reversal potential of that synaptic connection. Thus, these conductance synapses act to drive the membrane voltage to some target voltage determined by their reversal potential. Again, as in Equation 3.3, the total input current from a set of such conductance-based synapses is the sum of their individual currents.

However, the conductance value is also a dynamic value. This conductance $g_{\text{syn}}(t)$ decays to zero over time and is increased by a value equivalent to the synaptic weight, w_{syn}, upon the arrival of a pre-synaptic action potential. The decay of the conductance is governed by a synaptic time constant τ_{syn}. This behaviour can be expressed

$$\tau_{\text{syn}} \cdot \frac{dg_{\text{syn}}(t)}{dt} = -g_{\text{syn}}(t). \tag{3.5}$$

Equation 3.5 shows the ODE governing the decay process for a simple conductance-based synaptic model. Under this formulation, the conductance would decay to a value of zero and remain there. However, as described above, changes to the conductance occur upon the arrival of a pre-synaptic neuron spike at a synaptic connection such that $g_{\text{syn}} \leftarrow g_{\text{syn}} + w_{\text{syn}}$ where w_{syn} is the synaptic weight.

When modelling synaptic connectivities, we most often consider two main types of input source, excitatory and inhibitory. If all excitatory (or inhibitory) connections in a network have an equivalent reversal potential and time constant, the contributions of all excitatory (or inhibitory) synapses can be combined into

a single conductance term such that the total input to a from all of its excitatory and inhibitory conductance-based synaptic inputs can be described as

$$I(t) = g_{\text{exc}}(t) \cdot (E_{\text{exc}} - V(t)) + g_{\text{inh}}(t) \cdot (E_{\text{inh}} - V(t)) + I_{\text{bg}}. \tag{3.6}$$

In equation 3.6, g_{exc}, and g_{inh} indicate the total excitatory and inhibitory conductance values. E_{exc} and E_{inh} are the reversal potentials of the excitatory and inhibitory synaptic connections respectively. Often, these networks also include a background input in order to help maintain activity. The type of background input is often either a constant input or a stochastic 'shot' noise input. We consider only constant input in all discussion for this thesis. I_{bg} describes such a constant background input, which provides a baseline input to the given cell.

Given this combination of all excitatory and inhibitory conductance terms into a single value per neuron, these conductances can then be solved alongside the membrane voltage dynamics and need not be updated for every synaptic connections. Similar to the expressions describing the dynamics of a single synaptic connection, equation 3.5, the total excitatory and inhibitory conductances for a given cell have the dynamics described by

$$\tau_{\text{exc}} \frac{dg_{\text{exc}}(t)}{dt} = -g_{\text{exc}}(t) \qquad \text{and} \qquad \tau_{\text{inh}} \frac{dg_{\text{inh}}(t)}{dt} = -g_{\text{inh}}(t), \tag{3.7}$$

where g_{exc} (g_{inh}) is the total excitatory (inhibitory) conductance of a given neuron, and τ_{exc} (τ_{inh}) is the time constant of the decay of this conductance. As before, these conductance values are increased upon the arrival of pre-synaptic spikes at the specific synaptic connections. Upon the arrival of any pre-synaptic spikes at an excitatory synaptic connection $g_{\text{exc}} \leftarrow g_{\text{exc}} + w_{\text{e}}$, where w_{e} is the weight of the synaptic connection at which the spike arrived. Similarly upon the arrival of a pre-synaptic spike at an inhibitory synaptic connection $g_{\text{inh}} \leftarrow g_{\text{inh}} + w_i$, where w_i

is the weight of the specific inhibitory synaptic connection at which a spike arrived.

Networks of interconnected neurons with either voltage-based or conductance-based synaptic connections are often externally excited by input neurons with pre-defined spike times or firing rates, from which spike times are randomly sampled. Under some initial condition and deterministic input activation, these networks are deterministic. Often the inputs to the system are however sampled in a stochastic fashion.

3.1.1.3 Synaptic Plasticity

Models of synaptic plasticity in neural networks describe changes in synaptic strength or weight. In rate coded neural network models, these changes in weight are often based upon the firing rates of the pre and post-synaptic neurons. Plasticity in spiking neural network models instead often use spike times in order to produces changes in synaptic weight. Forms of plasticity which are dependent upon the timings of the pre and post-synaptic neuron action potentials are known as spike timing-dependent plasticity (STDP) rules. Both rate based and STDP rules were described briefly in Chapter 1, and Hebbian rate-based learning was the basis of the models described in Chapter 2. In this chapter and the chapters which follow we exclusively focus upon STDP based mechanisms for synaptic weight change.

Spike Timing-Dependent Plasticity (STDP)

STDP describes learning rules in which changes to synaptic strength are determined by the timing of pre and post-synaptic neuron spikes. Electrophysiological studies have provided evidence for this form of learning in neural circuits for decades [19, 120, 225]. These learning rules were observed in circuits where a post-synaptic neuron spiking after the arrival of a pre-synaptic spike resulted in positive weight change of the synaptic connection, such long term positive changes

in synaptic weight are known as long term potentiation (LTP). The magnitude of this change was dependent upon the time difference between these pre and post-synaptic neuron spikes. Similarly, the arrival of a pre-synaptic neuron spike after the post-synaptic neuron spike resulted in a negative weight change, with such long term reductions in synaptic weight being known as long term depression (LTD). Since this early investigation, timing based plasticity rules have been experimentally and theoretically explored and a number of forms of these have been discovered across brain areas. These have led to the proposal for a range of other forms of timing based plasticity, from short-term plasticity and triplet plasticity to inhibitory plasticity [190, 230, 152, 201]. These learning rules account for a range of electrophysiological observations and provide mechanisms by which network dynamics can be effectively controlled in modelled neural circuits.

The simplest form of STDP [19, 120] describes changes to synaptic weight as having a magnitude which is exponential in the pre and post-synaptic neuron timings. The magnitude of synaptic weight change which occurs for a given pre/post-synaptic spike pair is determined by the specific difference in time between their spikes as follows

$$\Delta w = \begin{cases} \alpha_+ \cdot e^{\frac{-(t_{\text{post}} - t_{\text{pre}})}{\tau_+}}, & \text{if } t_{\text{post}} > t_{\text{pre}} \\ -\alpha_- \cdot e^{\frac{(t_{\text{post}} - t_{\text{pre}})}{\tau_-}}, & \text{otherwise.} \end{cases} \tag{3.8}$$

Upon each pre or post-synaptic neuron spike, the weight change Δw is calculated as shown in Equation 3.8. The weight is then updated such that $w \leftarrow w + \Delta w$. α_+ and α_- describe the magnitudes of the LTP and LTD components of the STDP rule respectively. Similarly, τ_+ and τ_- describe the decay constants of the LTP and LTD components of the STDP rule respectively. The times t_{pre} and t_{post} correspond to the times at which the pre and post-synaptic neuron spikes reached the synaptic terminal. For pre-synaptic neurons, t_{pre} is offset by any transmission delay associated with

the synaptic connection. For post-synaptic neurons, t_{post} simply corresponds to the post-synaptic neuron spike time, though this can also be offset by a dendritic delay.

This rule calculates the change in weight for a specific pair of pre and post-synaptic neuron action potentials. In general, STDP based rules can be applied in one of two manners; nearest neighbour or all-to-all. Nearest neighbour based learning is when weight updates are calculated only for pairs of spikes which are neighboured to one another in time. In all-to-all STDP, when weight updates are calculated they are based upon pairing with every past spike time of the corresponding pre or post-synaptic neuron. In this thesis, we exclusively use all-to-all STDP.

The above STDP rule is what would be termed unbounded. This refers to the fact that the weights can increase or decrease without a limit upon their upper or lower values. A hard bound can be implemented in this case by creating a harsh limit such that the weight, w, of a synaptic connection is not allowed to exceed some limit. This is referred to as a hard bound and is explicit.

An alternative is to modify the STDP rule to introduce an implicit or soft bound. Such a bound is not explicit in that it does not place a limit upon the weight values. However, soft bound rules commonly modulate the levels of LTP and/or LTD in a weight dependent manner in order to create an implicit bound. An example of this was described in Chapter 1 and a weight dependent plasticity rule is used for simulations later in this chapter. We also describe a similar rule which is extended and evaluated in Chapter 4.

3.1.2 Solving Neural Dynamics and Simulating Networks

3.1.2.1 Solving Dynamics

As described above, LIF neuron and conductance-based synaptic dynamics are expressed using Ordinary Differential Equations (ODEs). Solving such ODEs can be accomplished in a number of ways. In general, producing analytic solutions to

ODEs allows the state variables to be solved without any error in the calculation. However, not all systems of ODEs have analytic solutions and, even when they do, if the inputs to the system are discontinuous, these analytic solutions need to be constantly re-computed. Numerical integration methods are an alternative which approximate the ODE solution by iterating the dynamics of the system through time. Such methods are typically used in systems which do not have analytic solutions or systems which are perturbed in a dis-continuous fashion, though their solutions are only approximate and can introduce errors.

LIF neuron models have dynamics which can be solved analytically but combination with conductance-based inputs results in a set of coupled ODEs for which only approximate analytic solutions have been proposed [173]. Furthermore the discontinuous nature of action potentials and the discontinuous updates to synaptic conductances mean that any solving method used to simulate spiking networks must be halted upon every action potential in order to both alter the membrane voltages and to change efferent synaptic conductances. For these reasons, numerical integration methods for solving SNN dynamics are common in existing simulations.

Generally, numerical integration methods incrementally evolve the state variable of a system by small steps, known as the timestep. The choice of the step size affects the accuracy of the resulting approximation such that smaller step sizes are more accurate but also more computationally expensive. Errors in the approximated solution can also compound over successive steps. In the case of spiking neuron models, following the resetting procedure, neuron responses begin once more from a reliable initial state though their spike time may well have been affected. Thus errors are shifted from membrane voltage to spike times in the short term. Shifted spike times thereafter affect the future neuron membrane voltage and network dynamics. This can ultimately cause deviations in network dynamics, especially in balanced spiking networks with chaotic dynamics [204, 223].

The most common numerical approximation technique in SNNs is the Forward

Euler solver. This solving method approximates the value of an evolving dynamic variable using the gradient of that variable. Take as an example some function of time $f(t)$, with derivative $\frac{df(t)}{dt}$. The Forward Euler solver assumes that the value of the function at some small time δt in the future is approximated by:

$$f(t + \delta t) = f(t) + \delta t \cdot \frac{df(t)}{dt} \tag{3.9}$$

as long as δ_t is sufficiently small. There are many alternative and more sophisticated methods which can be used to numerically solve an ODE. However, the repeated interruption of neuron membrane voltages by discontinuous jumps in their inputs, due to incoming synaptic activity, and the discontinuous jumps which occur when neurons execute action potentials mean that computationally simple ODE solving methods are most efficient. It is for this reason that the majority of SNN simulation software makes use of Forward Euler solvers.

For the LIF neuron in particular, an application of the Forward Euler solving means that the temporal evolution of the membrane voltage of a cell is computed as

$$V(t + \delta t) = V(t) + \delta t \cdot \frac{dV(t)}{dt}$$

where $\frac{dV(t)}{dt}$ is given by equation 3.1, and the excitatory and inhibitory conductances are similarly updated such that

$$g_{\text{exc}}(t + \delta t) = g_{\text{exc}}(t) + \delta t \cdot \frac{dg_{\text{exc}}}{dt} \qquad \text{and} \qquad g_{\text{inh}}(t + \delta t) = g_{\text{inh}}(t) + \delta t \cdot \frac{dg_{\text{inh}}}{dt}$$

where $\frac{dg_{\text{exc}}}{dt}$ and $\frac{dg_{\text{inh}}}{dt}$ are calculated according to equation 3.7.

3.1.2.2 Simulating Network Dynamics

Having discussed the component parts of SNN simulations and the methods by which they can be solved, we can now describe the process involved in simulating an entire network. In short, we carry out the following stages for each numerical timestep of a simulation;

- **Neurons**: Update the neuron membrane voltages and the cell conductances. Check for action potentials (i.e. test against the voltage threshold)
- **Synaptic Inputs**: Based upon the neurons which have fired now and in the past, update the conductances of neurons to which a spike is arriving.
- **Plasticity**: Based upon any neuron spike at the current timestep OR the arrival of a pre-synaptic neuron spike at a synaptic terminal, calculate changes in the synaptic weight

These stages are repeated for each numerical timestep in the simulation. On modern systems, the individual stages described above involve a number of data structures containing individual neuron, synapse, and plasticity state variables and procedures which carry out the corresponding updates.

A traditional approach to computing network updates involves the serial processing of each of these stages. This involves first updating all neurons, then all synapses, then considering plasticity. Furthermore, within each of these stages each component is often sequentially updated, for example updating all neurons in a network one after another for the "Neurons" stage. Beyond serial processing, parallel processing approaches allow the individual stages to be computed more efficiently. For example, multiple neurons can have their states update simultaneously and once these have all been completed the next stage ("Synaptic Inputs") can be executed. Notably, despite each individual stage being parallelised, it is important that all updates for a single stage are complete before beginning the next stage. For example,

we should await all neuron updates (even if they are happening in parallel) before beginning synapse or plasticity updates, otherwise differences in the orders of these calculations introduces errors to the simulation of the system. Thus, synchronisation of the entire network simulation is a requirement between each stage.

In this chapter we discuss the use of parallelism within graphical processing unit (GPU) devices to simulate point-neuron based spiking neural network simulations. We introduce a number of optimisations which have been explored both in GPU based but also cluster based simulators in the past and we attempt to relax some simulation synchronisation conditions in order to produce a highly efficient simulator.

3.1.3 Simulators

3.1.3.1 Existing Approaches and Simulators

Publicly available SNN simulators have been available for a few decades. NEURON [86], Brian/Brian2 [73], and NEST [51] are some of the most well established and long-running simulator projects in the field. These simulators benefit from a long history of development and large communities which use and support them. This has also resulted in them having a broad range of neuron, synapse, and plasticity models.

More modern simulators such as Auryn [223] and ANNarchy [200] have since been developed in order to provide unique blends of speed and flexibility over the existing simulators. Publications over the last decade [223, 200] have shown Auryn as a state of the art simulator in speed, outperforming all others with the highest recorded speed on traditional systems, and on systems using CPU or cluster-based parallelism.

Alternative methods of parallelism have been explored in the form of graphical processing unit (GPU) based simulators. GPUs have recently been proven to be highly effective across multiple fields, from Deep Learning to crypto-currency mining, as sources of computational power. GeNN [216], and CarlSIM [17] are two GPU based simulators which were released in the past decade in order to make

efficient use of GPU based hardware for SNN simulation. The NeMo simulator [56] was one of the first published simulators to use modern GPU technology but has had no development in recent years. However, to date a GPU based SNN simulator has not been compared in like-for-like simulation comparisons to the range of modern CPU based simulators.

In order to further develop the parallel processing capacities of GPU based simulators, and to provide like-for-like comparisons between CPU and GPU simulators, we pursued the creation and benchmarking of a novel GPU based SNN simulator, Spike, against a range of existing CPU and GPU simulators.

3.2 Methods

3.2.1 GPUs, Code, and Kernels

Graphical processing units (GPUs) are so named since the earliest use case of these devices were for computer graphics [127]. In general, these devices were required to compute updates to the geometry of objects in a scene and to colour pixels, or sub-group of pixels, for every frame of video output. In order to do so in a time-efficient manner these devices employed a great deal of parallelism when compared to CPU devices. This was particularly effective since the computations required to place and colour many object on a screen can be defined to be largely independently from each other.

The parallelism within classical and modern GPU devices is achieved by the presence of many computing cores [127]. Each core operates independently and simultaneously, often with access to shared memory areas from (to) which data can be read (written). In order to reduce the complexity of these devices, it was common to have all cores on a GPU device simultaneously execute precisely the same instructions but while indexing (reading from or writing to) different areas

of memory. As an example, this can allow a transformation, such as element-wise addition of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, to be computed such that each GPU core executes the same instructions but the first core adds data from the first rows of vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, the second core on the second row and so on. This paradigm, which is also applied in the Spike simulator, is known as single instruction multiple data (SIMD).

Quite clearly, this SIMD paradigm for parallelism is useful beyond the scope of computer graphics, and therefore GPUs have commonly been used for other parallel processing purposes. Modern GPUs are, for this reason, often referred to as general-purpose graphical processing units (GPGPUs), though we simply use the term GPUs in this thesis.

On NVIDIA GPU devices, the instructions which are applied to a range of data are known as "kernels" [139]. A kernel is written in an analogous fashion to a function, with some input data and a set of operations which affect that data. These are written in a language called CUDA which is commonly written alongside C or C++, low level programming languages which can be used to handle the CPU-side computation [177]. Another function of the CPU-side computations is to "launch" kernels on the GPU device. Launching a kernel refers to the process by which a kernel which has been defined on the CPU-side is sent to the GPU device for execution on some data. We describe a "kernel launch" overhead below. This refers to the time required (overhead) for the CPU-side to transmit a kernel to the GPU device and for the GPU device to begin execute of said kernel.

This concludes our brief description of GPU computation and code. The field of parallel computation, and GPU parallelism in particular, is complex and broad. For an overview, see Brodtkorb et al. [26].

3.2.2 The Spike Simulator

The Spike simulator has been under development since 2015 with the express aim of producing a high speed GPU based SNN simulator. It is written in C++ and CUDA with a flexible hierarchical class structure for neuron, synapse, and plasticity models. The sections which follow describe optimisations which have been implemented in the Spike simulator. These optimisations were designed to achieve a number of goals but most prominently a high speed of simulation and an insensitivity to synaptic connectivity variables such as synaptic delay. In particular, we aimed to develop a state of the art simulation tool and to ensure that no overhead was brought to a simulation even if synaptic connections have heterogeneous distributions of axonal delays or multiple connections between pairs of neurons.

3.2.3 Optimisations

3.2.3.1 Synaptic Matrices versus Lists

Neural network models with synaptic connections (or weights) often represent the connectivity between nodes using a matrix of values. In such a matrix, non-zero values indicate the existence of a connection between the nodes indexed by the row and column. Such a matrix based approach is valuable for simplifying network structures and allowing linear algebra based treatment of state updates, especially in rate based neural network models. There exist many highly optimised linear algebra and tensor libraries available to manipulate and compute with such matrices.

However, neural circuits of the brain contain connection properties that are not easily aligned to such a dense matrix based representation of weights. First, synaptic connections between populations in many modelling studies are often highly sparse. Many spiking network simulators deal with these connectivity matrices by converting them to a compressed sparse representation. Such compressed sparse matrix representations are highly efficient in terms of data storage and can be stored

in what is call "row-order" such that all post-synaptic neuron indices are stored in order for each pre-synaptic neuron. Such storage mechanisms allow efficient data operations upon these matrices [25].

Beyond this treatment, however, it is possible for neurons to be connected via multiple synapses (multapses) to a post-synaptic neuron. Single axons innervating a specific dendrite on many locations have been observed to be common [101] and there are reasons to assume that their impact upon the post-synaptic neuron could function to be more than simply additive [34]. This property means that a matrix based interpretation would need multiple connectivity matrices to represent multiple connections between pairs of neurons.

Furthermore, synaptic connections can also have a range of heterogeneous axonal transmission delays. This means that each synaptic connection would have a unique associated delay which describes the time required for the pre-synaptic action potentials to propagate to the output neuron. This delay property in particular is incompatible with the notion of a simple matrix based interpretation of the connectivity. The input activity "vector" can no longer simply be multiplied in a matrix operation through the connectivity matrix to produce the impact on post-synaptic neurons. Instead, if there are a range of delays, a given input pattern at time t impacts different post synaptic cells at times which range from time t up to some time $t + \Delta t_{\max}$, where $\Delta t_{\max}$ is the maximum delay in the synaptic connectivity structure.

To move beyond the rigidities of matrix based connectivities, the Spike simulator stores synapses in a list form rather than a compressed matrix form. This requires storage of a group of lists which contain the pre-synaptic and post-synaptic neuron IDs, the synaptic weights, and the synaptic delays. These lists are organised by pre-synaptic neuron index with a lookup for each pre-synaptic neuron to locate its outgoing synaptic connections. By storing the synaptic delays in this fashion, it is possible to store any arbitrary synaptic delay distribution or arrangement of

synaptic connections. For example, multapses or autapses (self-connections) are not treated any differently to any other synapse type and are managed effortlessly whereas in a matrix based approach, these might require multiple connectivity matrices between a pair of neural populations or other techniques for augmentation.

One major benefit of compressed sparse storage methods for connectivity matrices is the ability to store all of the efferent synaptic connections for a given neuron in consecutive memory locations. This allows a simulator to ensure that when an action potential is emitted by a neuron, all of the post-synaptic neurons of its efferent synapses are stored in a contiguous list and can be updated by iterating over this list. Ordered memory accesses are particularly efficient for GPU based simulators, which are able to access contiguous locations in memory through "coalesced" memory accesses with much greater efficiency than if the memory accesses are randomly ordered (uncoalesced). This coalesced memory efficiency is particularly useful for the SIMD paradigm adopted by GPUs (described above).

Given our switch to a list based storage of synaptic contacts, it is important to retain this benefit of storing synapses in groups of their pre-synaptic neurons. Early GPU based simulators such as NeMo [56], describe a similar organisational principle for synaptic connections in which they sorted the synaptic connections in a network according to their pre-synaptic neuron index. We apply a similar technique here in which synaptic connections are organised in a list, regardless of the order in how they were created, and are ordered according to their pre-synaptic neuron index. This allows all efferent synapses from a given neuron to be updated in a single pass over the contiguous memory locations which store these efferent synaptic connection details.

3.2.3.2 Timestep Grouping

On a GPU device, the execution of a kernel causes a small computational overhead, often referred to as the kernel launch overhead. This delay is compounded over

(A) Regular Timestep Based Network Updates

$N \leftarrow \frac{\text{simulation run time (seconds)}}{\text{timestep (seconds)}}$

for $n \leftarrow 1$ to N **do**
    Asynchronously Launch Kernels
        Update Network State
    Return to Host and Synchronise
end for

(B) Timestep Grouping for Network Updates

$N \leftarrow \frac{\text{simulation run time (seconds)}}{\text{timestep (seconds)}}$
$D \leftarrow \frac{\text{smallest network delay (seconds)}}{\text{timestep (seconds)}}$

for $n \leftarrow 1$ to $\frac{N}{D}$ **do**
    Asynchronously Launch Kernels
        for $d \leftarrow 1$ to D **do**
            Update Network State
        end for
    Return to Host and Synchronise
end for

Figure 3.1: **Network Update Optimisation (A)** Looping structured used for regular timestep based network updates. **(B)** Looping structure used for timestep grouping. The "Asynchronous Kernel Launches" indicate the stage of initiating computations on the GPU and "Return to Host and Synchronise" indicates the completion of the GPU based computation and the synchronisation of the simulation which happens when the GPU returns to host.

the course of a SNN simulation by the sheer number of kernel launches, eventually leading to a significant compounded computational cost.

In order to mitigate this issue, we use a feature of many SNN models; delayed propagation of spikes between neurons. Many SNN models define individual synapses as having some delay in the arrival time of a spike from the pre-synaptic neuron to the synaptic connection. This is often referred to as an axonal delay, though these delays can also be attributed to dendrites. We discuss only axonal transmission delays here. These delays are used here to implement timestep grouping (as first

described by Morrison et al. [131]).

Timestep grouping requires the determination of the smallest axonal delay in the network. Thereafter, the network's smallest axonal delay is used to update the network in grouped computations. Kernels are launched, updating the network dynamics at the numerical timestep by way of loops as described in Figure 3.1. Figure 3.1 A shows the regular timestep based update in which each network state is updated via a kernel launch upon each timestep. Timestep grouping allows a single kernel to update the network multiple times (up to the timestep grouping value D) as described in Figure 3.1 B. This timestep grouping is determined by calculating the size of the smallest axonal delay in the network as measured in the number of timesteps.

Any optimisations must ensure that the modelled SNN dynamics are unaffected. For timestep grouping we use the fact that for any time period of the simulation which is smaller than the minimum axonal delay, only spikes which were emitted before that time period can affect neuron dynamics during that time period (given the nature of delays). Therefore, if the timestep grouping is kept smaller than or equal to the minimum axonal delay it avoids affecting the network dynamics and is capable of integrating activity which was recorded before the beginning of the timestep grouping window. Furthermore, any spiking activity within this period only affects other neurons after the minimum axonal delay has elapsed and can therefore be collected at the end of the timestep grouping for use in subsequent timestep groupings.

This technique of iterating through calculations up to the minimum delay before synchronisation has been described in the past in the context of multi-node cluster based spiking neural network simulators. Morrison et al. [131] described this approach when they presented software for spiking neural network using distributed computing in clusters. This approach has been maintained in modern implementations of simulators such as NEST [51] and in extensions of this software

which allow a combination of rate coded and spiking neural network models [78]. However, in the context of GPU based computation on a single system, this idea of grouping all computations which occur on a timescale shorter than the smallest network delay have not been applied. Furthermore, its ability to reduce the impact of kernel launch overheads has remained unrealised. In the original paper proposing this technique for distributed computing [131], this technique was referred to as "index buffering", however given that we are describing this technique in a new context and given that "index buffering" is not an informative title for this operation, we refer to the same technique applied to GPU based computation as timestep grouping (TG).

A further benefit of the timestep grouping is a greater amount of work per kernel call. Since the kernels which compute network updates require synchronization after each update, the time required for a single update to the network is limited by the slowest computational thread. Taking neurons as an example, the slowest computational thread when updating neurons in parallel would be the computational thread which is calculating an update for a neuron which fires an action potential. Action potentials require the membrane voltage to be reset, for the action potential to be recorded, for efferent synapses to be updated, and for a refractory period to be initiated. This single computational thread, with its excess computation, would cause all other threads to await its completion. Spiking events for a single neuron are sparse on the timescale of a timestep meaning that the majority of neurons must await the cost of computing these excess updates for the few neurons which have spiked. In contrast, timestep grouping results in the grouping of computations for a number of timesteps. A greater proportion of neurons execute an action potential over this larger time period, and these can all be processed in a single kernel execution reducing the percent of neurons which are idle. This produces a secondary source of speed-up through timestep grouping, an optimisation particular to GPU simulation.

The reduced kernel launch overhead and greater work per kernel execution ultimately reduces simulation time, as explored in the Results section below.

3.2.3.3 Neuronal-Synaptic Parallelism

The number of synapses in SNN models often exceed the number of neurons by orders of magnitude. However, if the synapses act to modify variables on on the neuron scale (such as for any voltage-based synapses and particular conductance-based synapse implementations where all excitatory and inhibitory synaptic effects are grouped, see Equation 3.6) synapses only require processing when a spike from a pre-synaptic neuron reaches them, after any axonal delay.

Checking all of the synaptic connections on every timestep is the most algorithmically simple method for determining when spikes due to recent firing of pre-synaptic neurons should be propagated, see Figure 3.2 (A). Such a naive updating scheme simply checks every synaptic connection and results in a large number synapses being checked that require no action due to the inactivity of the pre-synaptic neuron. These unused threads contribute to a significant inefficiency in computation. This method was compared to a range of alternatives methods for updating network dynamics in simulation studies [25, 135, 100]. An alternative, referred to as "Neuronal-Synaptic (N-S) Parallelism" [135], which is also qualitatively similar to other approaches such as "Vectorisation over spikes, synaptic index, and neuron index (VSSN)" [25], has been shown to be optimal across a range of firing rates and network sizes. We implement Neuronal-Synaptic (N-S) parallelism in this paper and refer to it as such from here onward. In this approach, a neuron kernel is launched which checks (in parallel) all of the neurons to update their state and checks if they have emitted a spike. If a neuron has spiked, its efferent synaptic connections are flagged as active. In reality, this operation simply means identifying which memory areas refer to synaptic connections which must be activated during this stage of computation. Since our synaptic connections are stored in order according to their pre-synaptic neuron indices, this means that when a neuron spikes, we identify where its efferent synaptic connections begin in memory and we can then iterate over these.

(A) Naive Synapse Updating

$S \leftarrow$ Total Number of Synapses

Asynchronously Launch Kernels
  for $s \leftarrow 1$ to S **do**
    if $preSynapticNeuron(s)$ spiked **then**
      Update upcoming synaptic inputs
        to $postSynapticNeuron[s]$
    end if
  end for
Return to Host and Synchronise

(B) Neuronal-Synaptic Parallelism

$N \leftarrow$ Total Number of Neurons
$S \leftarrow$ Total Number of Synapses

$ActiveSynapses \leftarrow \{\}$

Asynchronously Launch Kernels
  for $n \leftarrow 1$ to N **do**
    if $Neuron(n)$ spiked **then**
      Add $SynapsesFromNeuron(n)$
      to $ActiveSynapses$
    end if
  end for
Return to Host and Synchronise

Asynchronously Launch Kernels
  for $s_{active} \in ActiveSynapticConnections$ **do**
    Update upcoming synaptic inputs
      to $postSynapticNeuron[s_{active}]$
  end for
Return to Host and Synchronise

Figure 3.2: **Synaptic Update Optimisations (A)** The computationally most simple method for detecting synaptic activations. **(B)** A description of the two stage process required for Neuron-Synapse Parallelism (N-S).

After the neuron kernel is complete, a second kernel is launched in order to update only the flagged synaptic connections. This process is outlined in Figure 3.2 (B). Thus, only active synaptic connections are considered for computation and these are all updated together in a large homogeneous group rather than using individual kernels for the efferent synapses of individual neurons. Axonal transmission delays would correspond in this approach to an adjustment during the updating of active synaptic connections. We implement a buffer based approach which is outlined in Section 3.2.3.4. Ultimately, synapses can be dealt with efficiently and computation can be kept to a minimum.

The effect of N-S parallelism on simulation time is explored in the results section below, where it is compared to a basic synapse update algorithm and a recently proposed GPU specific optimisation, referred to here as Dynamic Synapse Parallelism (DSP) [100]. Dynamic Synapse Parallelism makes use of a modern features of GPU based programming languages such that individual kernels which are computing neuron updates on the GPU device can themselves launch a new set of kernels on the device (in a dynamic fashion) whenever a neuron spikes. This paper claimed DSP as a superior method for computing network updates, though we find it lacking when compared to N-S parallelism.

3.2.3.4 Delay Insensitive

A simulator whose performance is unaffected by axonal delays is particularly desirable given the range of research avenues which use delays within spiking neural networks. SNN models investigating a range of phenomena including sound localization, reservoir computing, auditory processing and visual feature binding [72, 150, 53, 50] are but a few of the many studies which implement multiple axonal delay values within a single study for modelling purposes.

In order to meet the aim of a delay insensitive simulator, every neuron in a simulation is assigned a set of input buffers (one for each synapse type, e.g. excitatory

and inhibitory). The neurons iterate over these buffers to collect incoming synaptic updates. Upon each timestep, a single neuron shifts forward on its buffers by one location and reads the synaptic inputs for its current timestep. These buffers are circular, meaning that upon reaching the end of a buffer calculations return to the beginning again, and therefore the buffer can be treated as a ring.

If we define the length of this buffer as equal to the maximum possible synaptic delay, synaptic inputs can be placed in this buffer at a distance (corresponding to the delay of that synapse) from where the neuron is currently querying for inputs. This distance would correspond to the synaptic delay in timesteps and thereby ensures that the neuron only reaches this input after the corresponding delay. The circular nature of the buffer ensures that memory is conserved and since no incoming spike affects the neuron at any time delay longer than the maximum delay, it is sized efficiently.

Since no excess calculations are necessary, the simulation speed is unaffected by the inclusion of any delay structure in the synaptic transmission. The only consideration to be made is that of memory (these buffers introduce a memory overhead per neuron) though the Spike simulator is focused upon speed rather than memory efficiency. This technique has been applied in a number of simulators but was described first in a paper by Morrison et al. [131] where these buffers were described as "event buffers".

3.2.4 Benchmarks

3.2.4.1 Network Models

Simulators are compared in this chapter using two published benchmarks hereafter referred to as the Vogels-Abbott and Brunel benchmarks [200, 223]. The Vogels-Abbott benchmark is based upon a reduced scale version of the network presented in [202]. This network consists of 3200 excitatory and 800 inhibitory Leaky Integrate

and Fire (LIF) neurons with a 2% random synaptic connectivity between all populations of neurons. Synaptic connections follow a conductance-based input model and the network, driven by background stimulation, maintains a firing rate close to 17Hz. The specifics of the network dynamics and parameters are detailed in Appendix A.0.1.

The second benchmark network is one designed to test a network with a higher degree of connectivity and with input neurons. The Brunel benchmark is based upon a network adapted by [223], published in its original form by [27]. It consists of 10,000 excitatory input neurons which project with 10% connection probabilities to 8,000 excitatory and 2,000 inhibitory LIF neurons. These LIF neurons also have a 10% random connectivity between them with voltage-based synaptic connections. The excitatory input neurons spike randomly at time sampled from a Poisson process which produces an average firing rate of 20Hz. Appendix A.0.2 details the dynamics of this network.

The Brunel benchmark network has two modes; with and without plasticity. The plastic mode consists of a weight-dependent STDP rule operating on the excitatory to excitatory LIF Neuron connections. This STDP rule is expected to lead to a normal weight distribution with a mean of 0.1mV. The details of the STDP rule are explained in Appendix A.0.2.1.

The benchmark network parameters are equivalent to those implemented by Zenke at al. [223] and Vitay et al. [200]. For all benchmarks, only the time required to execute the simulation was recorded and any setup time disregarded. The simulations were run to produce 100s of network dynamics. Depending upon the efficiency of the simulator, this corresponded to some amount of real "processing" time required to run the simulation. We refer to this as the simulation time from now on, and it provides the amount of time which the simulator uses to compute the dynamics of the network. This simulation time was then divided by 100 to determine an average time per second of simulation, hereafter referred to as the normalized

simulation time. This term, "normalized simulation time", is one adopted from existing studies which outlined comparisons between simulators [223, 200]. It is equivalent to the inverse of simulation speed. The variation in runtime for these simulations was negligible on a system where all other non-essential processes were inactive, which makes sense given these are deterministic simulations. The code used to produce the comparisons against other simulators (below) has been collected in repository available at the URL:

https://tinyurl.com/y7gltmsw

3.2.4.2 Simulator Versions

The simulators compared in this study include four CPU based simulators – ANNarchy, Auryn, Brian2, and NEST [200, 223, 192, 114] – and a single GPU based simulator – GeNN [216]. These simulators were chosen as they remain actively developed and available for use. Some simulators which are no longer in active development were excluded, for example NeMo [56], though all comparison and simulation code has been made openly available and it is therefore fairly straightforward to extend the analyses which follow.

Simulators were collected for comparison and added to the repository (detailed below) in March 2019 from each of their git repository master branches. The specific versions of the simulators compared can be viewed at the repository listed above, or can be determined using the following git commit IDs for each simulator:

ANNarchy @ d27f8a4

Auryn @ 537f293

Brian2 @ c429811

GeNN @ 4419d8b

NEST @ a16b67f

Spike @ 68c5767

3.2.4.3 Test System

All single thread benchmarks described in this chapter were produced in a system with the following specifications;

- CPU: Intel i7-4770K
- GPU: NVIDIA GTX 1070 founders edition

The system used to test the multithreaded performance has specification;

- CPU: 2x Intel(R) Xeon(R) CPU E5-2623 v4 @ 2.60GHz

3.3 Results

3.3.1 Vogels-Abbott Benchmark

This benchmark, as described in Section 3.2.4.1 and Appendix A.0.1 , consists of 4000 leaky-integrate and fire (LIF) neurons (3200 excitatory and 800 inhibitory) with a 2% random connectivity with conductance-based synapses. This random connectivity was produced once using the Auryn simulator and was thereafter loaded for all results and simulators shown below, ensuring that all simulators are computing the same network dynamics. The neurons in the network are excited by a background 200pA input current, producing chaotic dynamics under the network connectivity. This benchmark is used to describe the relative impacts of the various optimisations described above and to compare the speed of the Spike simulator to other available simulators.

3.3.1.1 The Spike Simulator and Optimisations

The efficacies of the optimisations described in the Methods sections 3.2.3.2 and 3.2.3.3 are hereafter tested in the context of the Vogels-Abbott benchmark.

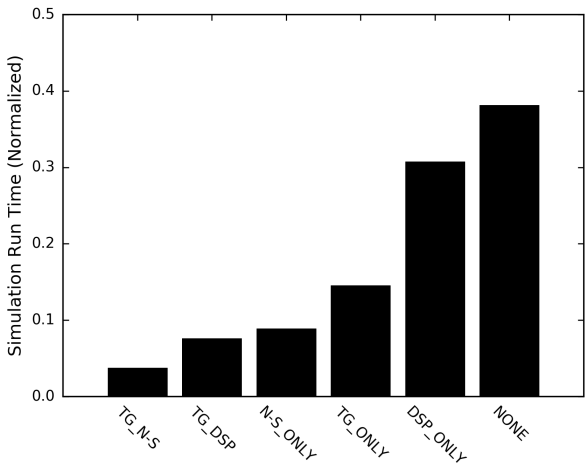

Figure 3.3: **The effects of various optimisations upon GPU SNN performance** Comparing the speed of simulations in the Spike simulator under a range of optimisation conditions. NONE; No optimisations, N-S; Neuron-Synapse Parallelism, DSP; Dynamic Synapse Parallelism, and TG; Timestep Grouping. These optimisations are also shown in combinations. The optimisations are presented fastest first (shorter is faster).

Figure 3.3 compares the simulation time required for the Vogels-Abbott benchmark in the Spike simulator across a range of conditions. It compares no optimisation (NONE) in which the synaptic updates are carried out individually in a naive fashion to a range of optimised conditions. In the NONE case, on each timestep every synapse tests whether its pre-synaptic neuron has fired and thereafter carries out any synaptic updates. This condition is compared to two other synapse relevant optimisations; Neuron-Synapse Parallelism (N-S), as described in the methods section, and Dynamic Synapse Parallelism (DSP). DSP refers to an approach which makes use of a fairly recent addition to the capability of NVIDIA GPUs in which individual threads on the GPU device can, themselves, launch more kernels (recruiting parallel threads). This is called Dynamic Parallelism. Dynamic Parallelism was identified by [100] as a potential speedup when compared to a limited set of other synaptic update methods.

Other than synaptic optimisations, we also compare the simulator speed with and without the inclusion of timestep grouping (TG). TG is further combined with

either N-S or DSP in order to show the speedup of combining these optimisations. Note that in a system running minimal background processes, these simulations run in a reliable time and therefore have no error bars.

Figure 3.3 shows a clear speedup under all optimisation conditions compared to the non-optimised case (NONE). The optimisation comparison also shows that N-S provides a significantly greater speedup than DSP. This is attributed to the fact that even though DSP ensures that non-active synapses are ignored, it nonetheless requires device-side kernel launching proportional to the number of spikes emitted during the simulation. Thus when compared to N-S (which only requires a single kernel launch per timestep and deals exclusively with active synaptic connections) DSP is markedly slower. This result clearly indicates that the comparison carried out by Kasap et al. [100] when introducing DSP was incomplete. In particular, the comparisons they provided only compared DSP to naive synaptic parallelism and to naive neuron based parallelism without combining them.

TG shows a significant speed increase both in the presence of synaptic optimisations and without. Since the network computation is identical under the NONE and TG cases, the TG benefit is attributed to the reduction of the kernel launch overhead and the increased work per kernel launch (see Methods 3.2.3.2). In the Vogels-Abbott benchmark, the minimum axonal network delay is eight times larger than the network timestep and therefore the number of kernel launches is reduced eightfold and the number of spikes processed per kernel launch eight times higher on average.

3.3.1.2 Comparing Simulators (Single Threaded)

Having established the best combination of optimisations in the previous section, we now compare the Spike simulator (with the TG and N-S optimisations) to a set of competing GPU and CPU simulators. Note that the CPU based simulators can simulate networks with multiple CPU threads, however we first consider all simulators with a single CPU thread.

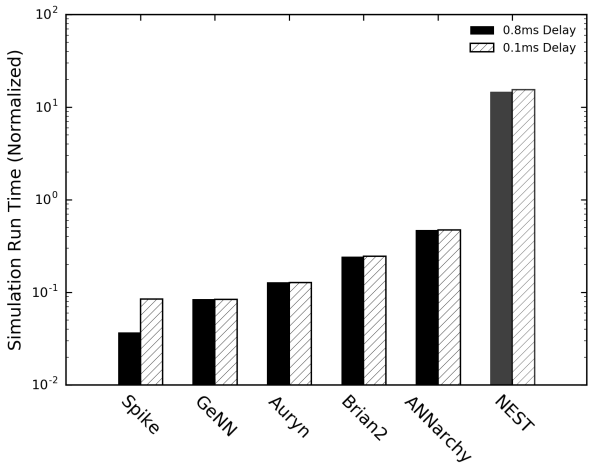

Figure 3.4: **The Vogels-Abbott benchmark comparison across simulators. All single threaded or single GPU.** Comparing the speed of a range of available simulators on the Vogels-Abbott benchmark. Shorter bars are faster. Note the log scale on the y-axis. These are ordered in speed, fastest first. The NEST simulator is shown in gray as it is the only simulator which does not implement a Forward Euler based update to the neuron dynamics. Instead it applies a higher order solver and so the comparison is less reasonable. The solid bars show the network simulated with a 0.8ms delay as described in the original benchmark studies [223]. Hatched bars shows the performance of these simulators when the axonal delay is equal to the timestep of the simulation (0.1ms).

Figure 3.4 compares the normalized simulation time required to simulate the Vogels-Abbott benchmark for a set of CPU and GPU based SNN simulators. "Normalized simulation time", as described above, is a term adopted from existing literature on spiking simulator comparison [223, 200], and is equivalent to the inverse of simulation speed. Therefore shorter bars here indicate a faster speed (shorter is better) and note that the y-axis is logarithmically scaled. The simulators are ordered by speed.

Figure 3.4 shows the benefits of GPU based simulators generally but also the particular simulation efficiency of the Spike simulator. Both GeNN and Spike, the only GPU based simulators in this list, show a significant speedup compared to competing simulators. These simulators are closely followed by Auryn which has been reported as the fastest CPU based simulator since its release [223, 200]. This

benchmark shows a like-for-like comparison of these simulators and their relative speeds. NEST is an exception on this list as it is the only simulator shown here which does not update the neuron dynamics with a Forward Euler solver and instead makes use of a higher-order Runge Kutta solver. Previous studies [223] have shown that when the NEST is modified to simulate LIF neuron dynamics with a Forward Euler solver, it gains significant ground in simulation speed. However, not only are the patches used for such a modification to the NEST out of date, the same paper also showed that nevertheless, the Auryn simulator remains the fastest CPU based SNN simulator. Therefore, we use the performance of Auryn as a bound upon existing CPU based simulator speed.

It is also noteworthy that Spike is the only simulator which significantly benefits in simulation speed from the introduction of delays in network structure. Though NEST has incorporated an optimisation based upon delays, this is ineffective when running simulations on a single machine and only produces benefits for cluster-based simulations. It is also important to note that a delay of 0.8ms is the standard for this benchmark as described by Zenke et al. [223].

3.3.1.3 Multithreaded CPU Performance

Comparison of a single GPU versus a single threaded CPU based simulation ignores the performance of multithreaded CPU based simulators. Given the multithreaded benchmark results presented by [223] and [200], Auryn remains the leading CPU based simulator both in the single and multithreaded use case. We therefore use Auryn as a comparison simulator and we simulate the standard Vogels-Abbott benchmark (with 0.8ms axonal delays).

Figure 3.5 compares the speed of Auryn in multithreaded execution mode to the Spike simulator on a single GPU. To date, no GPU based simulators, including Spike, support multi-GPU computation and therefore the speed referenced here for Spike is identical to that shown in Figure 3.4.

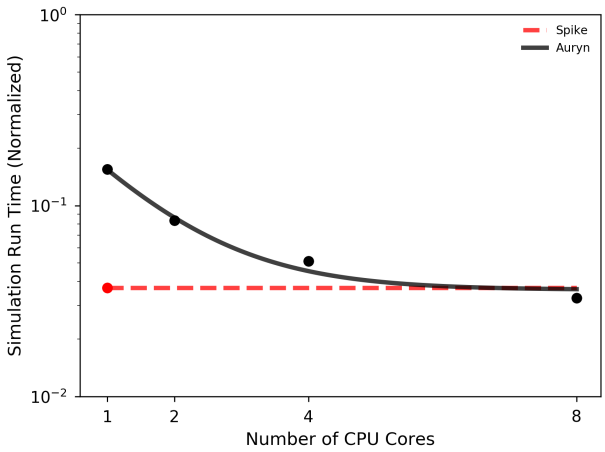

Figure 3.5: **Comparing the multithreaded CPU performance of Auryn against Spike on a single GPU.** This plot shows Spike in a single GPU mode only (therefore the dotted line is not a prediction of it's performance over more CPU cores but instead just a baseline). Auryn was benchmarked with 1, 2, 4 and 8 CPU cores. An exponential curve was then fitted to these points to produce the plot shown.

As can be observed, only under multithreading with eight threads does the Auryn simulator exceed the speed of Spike on a single GPU. Though Auryn does exceed the speed of Spike with eight threads for this network size, we show below how the speed of Auryn suffers for larger network sizes.

3.3.1.4 The Scaling Behaviour of the Spike Simulator

We presented above a comparison of the simulation speeds of a range of simulators for a fixed network size with the Vogels-Abbott benchmark. This provides a snapshot of the performance of these simulators without giving an idea of how simulation speed scales with network parameters. We vary here the network size by multiplication of the number of neurons and number of synapses by integer values. We compare the scaling performance of the Spike and Auryn simulators in order to give some idea of the scaling performance of a GPU simulator vs a CPU simulator.

Figure 3.6 shows the relative scaling of simulator performance as network sizes are increased. We multiply the network size up by directly multiplying the

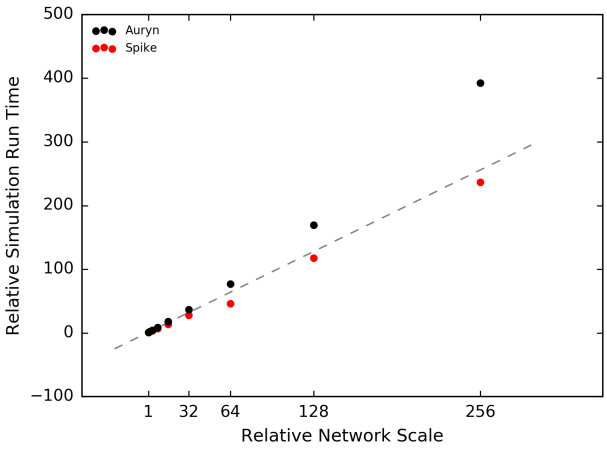

Figure 3.6: **Comparing the simulation times of the Spike and Auryn simulators as the Vogels-Abbott Benchmark's network size is scaled.** The x-axis presents the network scale relative to the original benchmark network. On this scale, a value of 1 indicates a network of 4000 neurons, and a value of 100 indicates a network of 400,000 neurons. The y-axis presents the relative time taken to simulate these networks. This time is relative to the time taken by the simulators to simulate a network of scale 1 (a network of 4000 neurons). Thus, both the Spike and Auryn simulators have a relative simulation speed of 1 at a network scale of 1. A value of 10 for the relative simulation time corresponds to ten times as much time taken relative to a network of scale 1. This normalization procedure was carried out to allow easy examination of whether simulation time in the simulators scales linearly, sub-linearly, or super-linearly with network scale. The gray dashed line indicates linear scaling.

number of neurons and number of synapses. The smallest network is the same size as our above benchmark: 4000 neurons and 320,000 synaptic connections. The largest network, at a scaling factor of 256, is a network of 1,024,000 neurons with 81,920,000 synaptic connections.

As can be seen in Figure 3.6, the Auryn simulator exhibits close to linear scaling for small network sizes but this quickly increases into a super-linear increase with larger network size. By comparison, the Spike simulator shows sub-linear scaling of performance as the network size is increased across a large range of network sizes. Larger network sizes were excluded as they require prohibitively long simulation time and eventually reach a scale at which GPU memory is exceeded. We assume that this scaling behaviour would continue given the network size does

not exceed the GPU memory capacity.

3.3.2 Brunel Plasticity Benchmark

A second SNN benchmark used to compare the simulators above is the Brunel benchmark consisting of 10,000 Poisson firing input neurons, and 10,000 LIF neurons (8,000 excitatory and 2,000 inhibitory) as detailed in Section 3.2.4.1 and Appendix A.0.2. This network is more densely connected than the Vogels-Abbott benchmark with a 10% connectivity between the LIF neurons and a 10% connectivity from the Poisson neurons to the LIF population. Furthermore, a weight dependent STDP rule is used to update the weights of excitatory to excitatory synaptic connections as described in Appendix A.0.2.1. This allows a second comparison between the SNN simulators to test their efficiencies when computing weight updates through synaptic plasticity.

For the Brian2, NEST, and Auryn simulators, rather than creating a population of input neurons which fire with times sampled from a Poisson distribution, they instead implement an optimised methods for Poisson process sampled inputs. These methods allow the simulators to produce Poisson sampled input spikes to a neuron without explicitly simulating the input neurons with their synaptic connections. In such a framework, creating 10,000 input Poisson spiking neurons and connecting 10% of them to each cell with a firing rate of 20Hz is equivalent to a Poisson sampled processes injecting 20,000Hz of input to each cell.

The Spike, GeNN and ANNarchy simulators do not feature this optimisation and instead explicitly simulate populations of input neurons (10,000) with spike times which are sampled from a Poisson process. All synaptic connections in this network model produce post-synaptic neuron inputs in the form of voltage-based synaptic connections.

Figure 3.7 shows (as above for the Vogels-Abbott Benchmark) a comparison

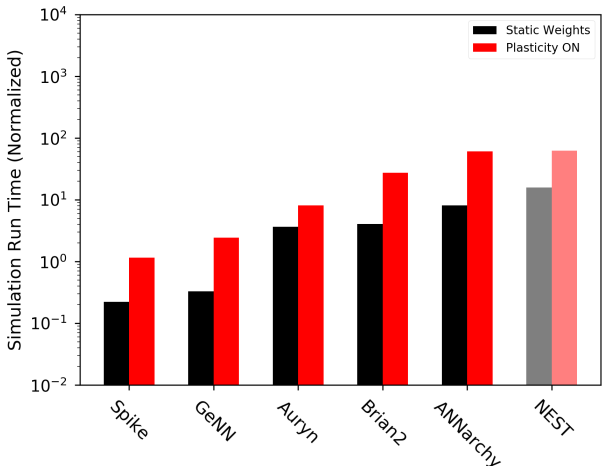

Figure 3.7: **The Brunel benchmark comparison across simulators. All single threaded or single GPU.** This benchmark was carried out with and without plasticity as shown by red and black bars respectively. Shorter bars are faster. Note the log scale on the y-axis. These are ordered in speed, fastest first. The NEST simulator result is shown as 50% transparent as it is the only simulator which does not implement a Forward Euler based update to the neuron dynamics. Instead it applies a higher order solver.

of the normalized simulation time for the range of simulators with the Brunel benchmark. The red and black bars show normalized simulation time for the same network with and without plasticity running respectively. As with the Vogels-Abbott benchmark, these results are based upon all simulators loading the same connectivity structure before the simulation begins.

The results of simulating a large network (with and without plasticity) are not qualitatively different to those seen in the prior section (compare Figure 3.7 to Figure 3.4). The order of the simulators in terms of speed is the same. However, the gap between CPU and GPU simulators begins to grow significantly for this larger network. Both Spike and GeNN have a much faster relative simulation time in both the plastic and non-plastic cases versus CPU based simulators. The gap in between the fastest CPU simulator and the Spike and GeNN simulators has grown in simulations of this scale to an order of magnitude.

In the Vogels-Abbott benchmark, the Spike simulator was approximately four

times faster than the Auryn simulator, however in the Brunel benchmark, this rises to approximately seven times faster in the plastic case and approximately 15 times faster in the non-plastic case. The speed of the GeNN simulator scales very similarly to Spike but remains on the order of two times slower for both the plastic and non-plastic cases.

3.3.2.1 Multithreaded CPU Performance

As for the Vogels-Abbott benchmark above, we also compare the performance of the Spike simulator against a multithreaded execution of the Auryn simulator. We show this result for the non-plastic case of the Brunel benchmark.

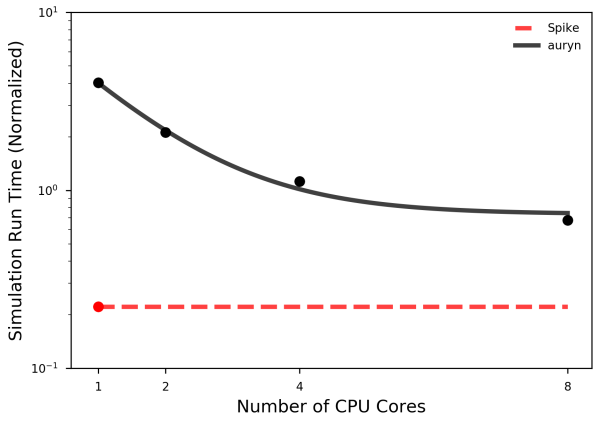

Figure 3.8: **Comparing multithreaded CPU performance with Auryn against Spike with the Brunel non-plastic benchmark.** This plot shows Spike in a single GPU mode only (therefore the dotted line is not a prediction of it's performance over more CPU cores but instead just a baseline). Auryn was benchmarked with 1, 2, 4 and 8 CPU cores. An exponential decay curve was then fitted to these points to produce the plot above.

Figure 3.8 compares the speed of Auryn in multithreaded execution mode to the Spike simulator on a single GPU. As before, the performance of Spike is only given for a single GPU and the dotted line is therefore simply a baseline and not a prediction of its performance. Compared to the multithreading comparison with the Vogels-Abbott benchmark, Figure 3.5, we see that even with eight threads, the

speed of the Auryn simulator remains significantly slower than that of the Spike simulator for a more densely connected network. Thus, GPU based simulators have the ability to outperform even multi-threaded CPU based simulations for relatively large network sizes.

3.3.2.2 Comparing Expensive Plasticity Implementations

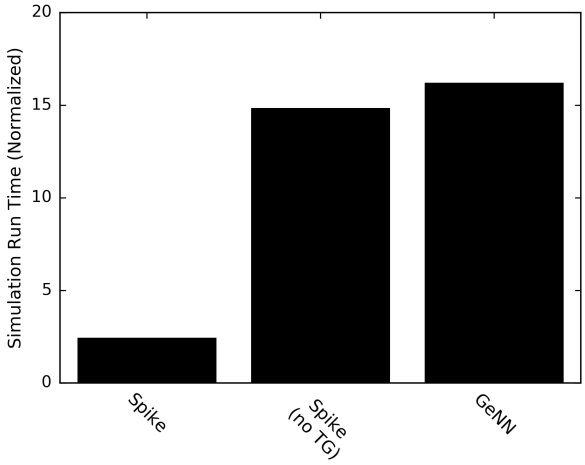

Figure 3.9: **The Brunel benchmark with plasticity variables updated every timestep**. This benchmark was carried out with plasticity active such that every plastic synapse is updated on every timestep. The default Spike simulator is compared to; a case in which timestep grouping (TG) is off, and to the default GeNN simulator. Shorter bars are faster. These are ordered in speed, fastest first.

The plasticity comparison shown previously (Figure 3.7) presented a two fold difference in speed between the Spike and GeNN simulators. However, plasticity can be implemented in a number of ways with such simulators and it remains to be seen how the speeds of Spike and GeNN compare for more computationally expensive plasticity implementations.

In short, STDP can be implemented in an online fashion where for every synaptic connection "trace" variables are stored which keep track of the contributions of all past pre and post-synaptic neuron spikes for future weight updates. Without such a trace based tracking of past activity, carrying out all-to-all STDP is extremely

expensive and requires calculations based upon every past spike in the network.

These traces used for plasticity can be updated every timestep by decaying the trace every timestep and incrementing it whenever pre or post-synaptic spikes arrive at the synaptic connection. In order to speed up synaptic plasticity for the Brunel Benchmark, the GeNN and Spike simulators avoided iteratively updating the synaptic traces (see Appendix A.0.2.1) on every timestep. Instead, upon the arrival of a spike at the synaptic connection all synaptic traces are decayed based upon the time since the last arriving synaptic spike and then used to calculate any synaptic weight changes. This is possible as the decay to the synaptic trace can be analytically defined. However, this process of avoiding synaptic updates on every timestep is not necessarily possible for any arbitrary plasticity rule. Furthermore, implementing such an optimisation requires intimate knowledge of the process by which a simulator carries out synaptic updates.

To explore the effects of Spike's optimisations, and in particular TG, for plasticity we consider updating the traces used for plasticity upon every timestep. This means that upon every timestep, traces used for plasticity are decayed and any weight changes caused by spiking activity are calculated. This can effectively make individual synaptic dynamics almost as computationally detailed as the individual neuron membrane voltage dynamics, thereby significantly increasing the computational complexity. Rather than utilising a more complex synaptic plasticity rule, we update the synaptic traces used for the plasticity rule in the Brunel benchmark on a timestep basis to emulate the process which would be required for a more complex rule. See Appendix A.0.2.1 for details on the calculation and updating of the variables used for plasticity. By updating every timestep, we no longer use the analytic solution to the STDP traces and instead treat their dynamics through numerical integration.

Figure 3.9 compares the Spike and GeNN simulators with a plastic version of the Brunel benchmark in which updates to variables used for plasticity are

calculated every timestep for every synapse. As shown, the speed of the Spike simulator approaches an order of magnitude speed increase over the GeNN simulator when considering plasticity computations which must be carried out every timestep. Noteably, this difference in speed is almost entirely accounted for by the timestep grouping (TG) optimisation. Without the inclusion of the TG optimisation, the speed of Spike would be close to that of GeNN. This provides another example of a case in which TG optimisation can provide a significant speedup. This speedup is not only due to the removal of the computational overhead associated with launching kernels, but also is due to an increased amount of work per kernel launch. Within timestep grouping, the vast majority of kernel launches consist of calculations for synapses which have no spikes. The few synapses which do have an arriving spike carry out excess calculations and slow down all other computations by forcing the other threads to await their completion. By timestep grouping, many more synapses have an arriving spike on each kernel launch (since there is a greater amount of simulation time being processed) and thus, fewer kernel launches are idly awaiting others to complete computations. Thus, more work per kernel launch is accomplished. This is a particularly significant source of speedup when applied to the dynamics of synapses given the sheer number of synaptic connections.

3.3.3 A Repository of Benchmarks and Comparisons

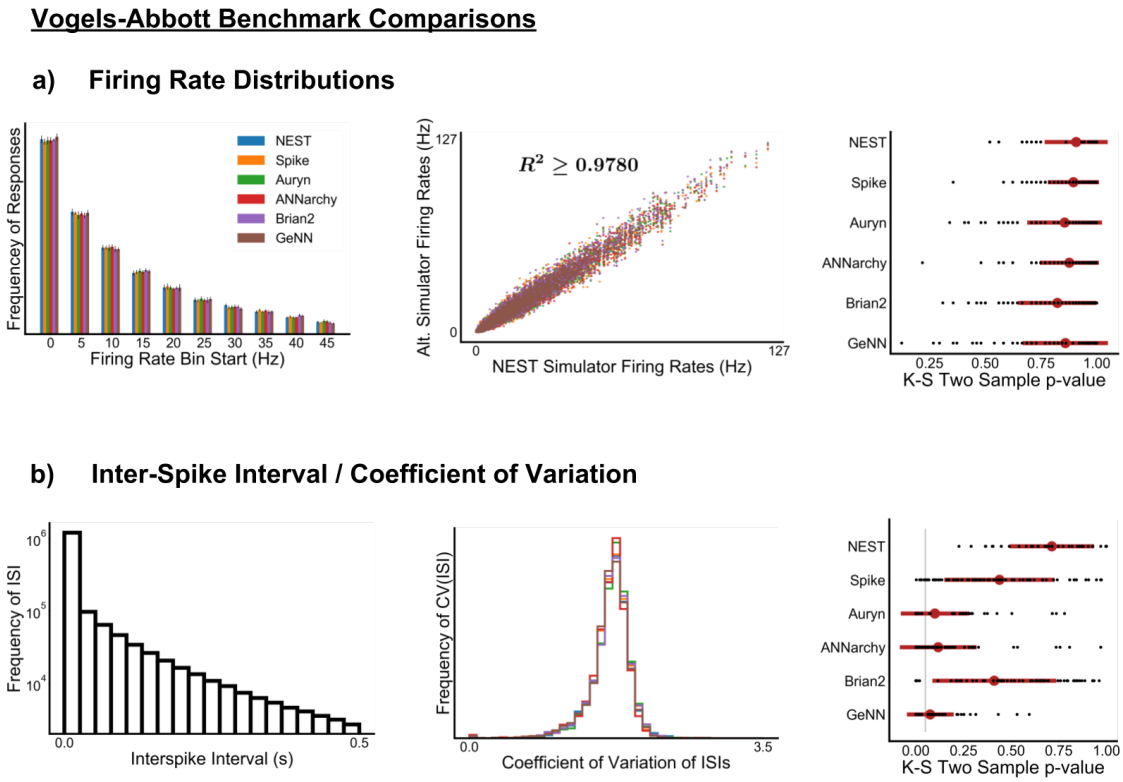

Figure 3.10: **Network properties in the Vogels-Abbott and Brunel benchmarks (a)** Firing rate data and statistics for the Vogels-Abbott benchmark. Networks were simulated to produce 100s of network dynamics and corresponding spike trains. Ten randomly randomly sampled 10s intervals (with overlap allowed) from within these spike trains were collected from each simulator for analysis. These 10s intervals were used to calculate the firing rates of each neuron for that sampled time region. The decoded firing rates could then be used to test for consistency of network behaviour. *Left*, the firing rates are binned into 5Hz bins between 0Hz and 50Hz (with x-axis labels denoting the starts of these bins), showing that the distribution of firing rates is qualitatively similar across all simulators across sampled time bins. *Middle*, we take the NEST simulator as a simulator of comparison and for a single sampled 10s interval of the spike trains across all simulators we check the firing rates of neurons. In a perfect reproduction of the NEST simulator result, these points would lie on the diagonal of this plot indicating that all corresponding neurons have an equivalent firing rate in the time window selected. The points across simulators are close to this diagonal and calculating the Pearson Correlation of individual simulator firing rates to the NEST simulator gives a minimum value of 0.9780 indicating a high level of correlation between all simulators and the NEST simulator firing rate values per neuron. *Right*, the firing rates sampled from the simulators are compared against the firing rate samples produced by the

NEST simulator using a two-sample Kolmogorov-Smirnov test which produces an estimation of whether these samples are from a different distribution. The x-axis here shows the p-value result from this two sample test. Since we sampled ten regions of time from these simulators, we compare every possible pair of samples (including testing the firing rate distributions between different samples from the NEST simulator). As can be seen, the p-values are fairly large (» 0.05) indicating that the firing rate distributions cannot be assumed to come from a different distribution. **(b)** Data and statistics of the Inter-Spike Intervals (ISIs) of neuron spike trains from ten randomly sampled 30s regions (with overlap allowed) of the spike trains produced by simulators simulating the Vogels-Abbott benchmark. *Left*, the ISI distribution for NEST is shown for a single randomly selected 30s interval. This distribution is indistinguishable across simulators when plot at this scale. *Middle*, the Coefficient of Variation (standard deviation divided by the mean) of the ISI data can be calculated for each neuron. A histogram of this statistic across all neurons is shown for a single 30s sampled time window for every simulator. *Right*, in order to quantify the degree of difference in the CV(ISI) distribution (middle) across simulators, we again compute a two-sample KS test by taking CVs calculated for the sampled 30s periods and carry out a pair-wise comparison (as in (a), right). A gray line shows p = 0.05.

Producing a comparison of the performance of SNN simulators required the collection and production of code which would execute a set of benchmarks. Despite these benchmarks having been established in a number of publications, code to reproduce these benchmarks was challenging to produce. Furthermore, beyond simply creating these benchmarks, the network dynamics produced through simulation also needed to be validated through cross-simulator testing.

In order to address these concerns, we constructed a public repository in which code used to produce the figures throughout this chapter is available. This repository includes direct references to the individual versions of the simulators (and their codebases) and contains a set of analysis tools which plot a range of comparisons of network behaviour. See Section 3.2.4.1 for a link to this repository.

Figure 3.10 shows a range of data and statistics which were collected in order to compare simulation behaviours produced by the range of simulators investigated. For the Vogels-Abbott benchmark, some key statistics identified in the past include the distributions of firing rates of neurons and the Coefficient of Variation (CV)

of Inter-Spike Intervals (ISIs) of the neuron spike trains. Figure 3.10a shows plots regarding firing rate statistics. The range of simulators tested which we tested all exhibited qualitatively similar firing rate distributions. The NEST simulator was taken as a point of comparison to test this similarity, given the more sophisticated numerical integration scheme which they used. Figure 3.10a, middle shows that when comparing individual neuron firing rates from all other simulators against the NEST simulator, the correlation of these is large ($r = 0.9780$ is the smallest correlation of all simulators). Precisely equivalent simulations would have correlation of 1.0, however, as has been described in earlier modelling studies [223], the chaotic nature of these benchmark simulations and the effects of any small shifts in spike times results in deviations across all compared simulators to the NEST simulator.

A statistical test employed by Zenke et al. [223] when comparing simulation outcomes by the Auryn simulator against the NEST simulator was the Kolmogorov-Smirnov (K-S) two sample test. This test allows two sets of sampled data to be compared to determine whether they are sampled from different distributions. The null hypothesis is that they are from the same distribution, and thus a p value smaller than 0.05 would usually be taken as an indication that two samples are from different distributions. For the results presented here, given that we would expect an equivalent simulator to produce statistics drawn from the same distributions (whether those be distributions of firing rate or ISIs etc), a p-value greater than 0.05 is desirable to indicate that it cannot be shown that the distributions are significantly different.

Figure 3.10a, right, presents the result of repeated K-S two sample tests between firing rate distributions measured from the spike train outputs by our set of tested simulators against sampled data from the NEST simulator. Ten random bins were selected from the spike times of all simulators, where these bins were of length 10 seconds. Following this, every pair of possible comparisons between all ten bins of every simulator and all ten bins from the NEST simulator were carried

out in order to produce the data shown in Figure 3.10a, right. As can be seen, all pairwise comparisons carried out with the K-S two sample comparison show p-values much greater than 0.05. This indicates that we cannot assume that the firing rate distributions between simulators are significantly different.

Figure 3.10b presents statistics regarding the ISI and CV distributions for the same simulations which were analysed in Figure 3.10a. Particularly interesting is the greater deviation here across simulators. Figure 3.10b, left, shows the ISI distribution for the NEST simulator for a single time period of length 30s. As has been described in past publications, for the Vogels-Abbott benchmark this ISI distribution is approximately an exponential distribution. Figure 3.10b, middle, shows the distributions of the Coefficients of Variation of the ISIs (CV(ISI)) of each neuron. To again quantify differences between these distributions, we sample ten 30s intervals from the spike trains produced by our range of simulators, compute the CV(ISI) distributions of these samples and then carry out K-S two sample tests across all pairs of samples between each simulator and the NEST simulator. Figure 3.10b, right, shows this statistic where, as in Figure 3.10a right, ten samples from each simulator are compared pairwise against ten samples from the NEST simulator in order to produce this data. This plot shows that simulators produce quantitatively similar CV(ISI) distributions to the NEST simulator, though not for all pairs of sampled spike trains. This result, in which there are observed cases in which other simulators compared against the NEST simulator produce significantly different CV(ISI) distributions, was explored by Zenke et al. [223] and they find that the selection of a smaller timestep increases the similarity of the CV(ISI) distributions and vice versa. The chosen timestep of 0.1ms was determined as a tradeoff between slow simulation speed and a reasonable likelihood of CV(ISI) similarity.

Aside from firing rate and spike time analyses, we can analyse differences in the weight structured produced through plasticity using the Brunel plasticity benchmark. Figure 3.11, presents data and analyses with regards to the final weight

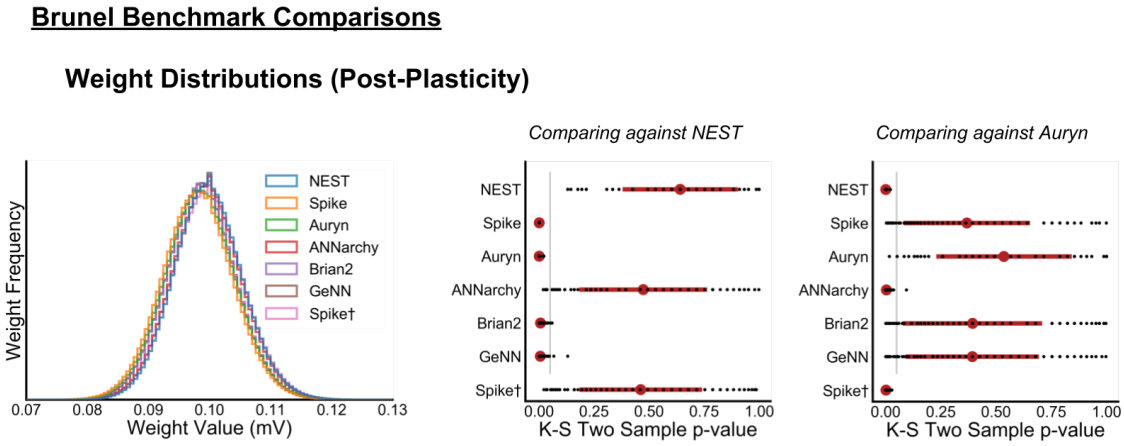

Figure 3.11: **Comparisons of the distribution of weight values (for the plastic synapses) in the Brunel Benchmark across Simulators after 20s of training.** *Left*, a histogram shows the weight distribution of the plastic weights for all simulators compared. *Middle*, 1000 weights are randomly sampled from the trained networks produced by each simulator. This random sampling is carried out ten times to produce ten samples per simulator. These samples are used in a two-sample KS test (as in Figure 3.10) in order to quantify the similarity of distributions. Sampled weights from every simulator's trained networks are compared in pair-wise tests to samples from the NEST simulator. A gray bar shows where p=0.05. *Right*, same as (c) middle, however instead of comparing samples against the samples from the NEST simulator, all comparisons are carried out against the samples from the network trained by the Auryn simulator.

distributions produced by the set of compared simulators when they execute the Brunel benchmark (with plasticity) for 20s of training. Figure 3.11, left, shows the weight distributions produced by the set of simulators. As can be observed, these weight distributions qualitatively have the same form and are highly overlapping. Upon further inspection, it can be observed that the weight distributions produced fall into two distinct groups, one with a slightly higher mean than the other.

To check this, we again test the similarities between these distributions by carrying out K-S two sample testing between ten samples of 1000 randomly selected weights from each simulator compared to ten samples from the NEST simulator (Figure 3.11, middle) and the Auryn simulator (Figure 3.11, right). Comparisons of the weight distributions shows that the simulators fall into two categories. In one

are present the NEST and ANNarchy simulators which, when compared sample to sample, do not have distinguishable weight distributions across pairwise samples, whereas the remaining simulators have significantly different weight distributions when tested. Similarly, the Spike, Auryn, Brian2, and GeNN simulators when sampled and compared to samples from the Auryn simulator, cannot be distinguished in their distributions whereas the NEST and ANNarchy simulators have highly distinguishable weight distributions.

In the paper which presented the Auryn simulator [223], they noted this difference and attributed it to differences in synaptic delays as being treated as axonal or dendritic, though they did not test this hypothesis directly. We carry out a direct comparison here and show that a modification to the plasticity rule in the Spike simulator (labelled "Spike†") can be shown to produce an indistinguishable weight distribution when compared to the NEST and ANNarchy simulators. Figure 3.11 middle and right, shows results for simulations carried out with the Spike simulator in two modes. In the normal mode, labelled "Spike", synaptic weight updates are carried out assuming that synaptic delays are axonal in nature. This means that the timings used for STDP based updates are the time of the post-synaptic neuron action potentials, and the times at which spikes from the pre-synaptic neurons "reach" the synaptic connection. The times at which a pre-synaptic neuron spike reaches a synaptic connection corresponds to the time of the pre-synaptic spike plus any axonal transmission delay time. A second set of results are shown with the Spike simulator, labelled "Spike†", in which the synaptic delay is completely ignored for synaptic updates and these are instead based directly upon only the pre and post-synaptic neuron spike times. This approach produces a distribution of weights which matches those of the NEST and ANNarchy simulators across the set of comparisons.

3.4 Discussion

The results shown above convincingly place GPU based simulators in the lead for SNN simulation on a single system. This is shown in both single and multi-threaded cases, in networks of a range of sizes, and in networks which include spike timing-dependent plasticity. Furthermore, GPU based simulators are shown to be equivalent to contemporary CPU based simulators in producing statistically similar network dynamics.

As the field moves forward, it is important that such simulators are made easy to use and flexible. Spike has attempted to make its C++/CUDA form as intuitive as possible. User guides are available and the creation of new models (for neurons, synapses or plasticity rules) is fairly intuitive to someone with a background in C++/CUDA. Nonetheless, writing models in C++ brings with it some complexity given the low-level nature of the code. Furthermore, extensive documentation for Spike has not yet been produced. Given all of this, Spike is placed to act as a simulator with some significant speed benefits and features (in areas such as axonal delay accommodation) over competing simulators, though undoubtedly with some knowledge barriers and inflexibility in defining new model components.

Though GeNN in its pure C++/CUDA form is a simulator which is fairly difficult to use (more so than Spike), the Brian2GeNN project has brought GeNN as a backend to Brian2 [191]. The Brian2 frontend provides a much easier user experience and model definition process. Beyond this, the ability to define models in Python brings major benefits in terms of usability (it being a higher level language).

GeNN's use of code generation gives it a significant advantage over simulators such as Spike since it is capable of producing tailor-made kernels for the specific simulations being produced. This allows many parameters to be set as constants within kernels and thereby saves time on memory accesses and reduces memory usage. In comparison, Spike's kernels incur a memory overhead in order to collect the

specific parameters for individual simulations. On balance, GeNN's code generation requires a multi-stage compilation process which is not required by Spike. In the most recent version of Brian2GeNN this results in a few seconds of waiting as models are compiled before they are run (though this may well change in the future). The timestep grouping (TG) optimisation implemented in Spike also gives it a significant computational advantage over GeNN.

As the field progresses, we expect GPU based simulations to become a common tool for computational neuroscientists. Just as a range of simulators have and continue to exist in the CPU space, we expect the same for GPU based simulators. Code generation approaches, such as those used by GeNN, could prove extremely valuable in producing flexible simulators, whilst more rigid simulations which require high speed and delay insensitivity could turn to Spike.

Recent comparisons of GPU based simulation, GeNN in particular, to neuromorphic hardware and CPU based simulators has highlighted the potential extreme power efficiency of GPU accelerated simulations even when compared to bespoke neuromorphic hardware [107]. We anticipate further optimisations in the future as GPU based architectures are upgraded and as algorithmic upgrades are proposed.

3.5 Future Work

In order to identify the major drawbacks and future directions for the Spike simulator, we applied to and were awarded consultation through an open call by the Software Sustainability Institute (UK). The major highlighted issues were the lack of documentation and "getting started" pages. Tools and technologies have been suggested for this purpose. These are intended to be produced by interaction within the development team and users in the future.

Future directions and developments of the simulator include considering methods to distribute processing further over multiple GPUs in a single system. This

requires efficient distribution of neurons across multiple such devices and efficient communication between these neuron units. Such a development requires substantial planning and expertise in multi-GPU memory management and computation efficiency. However, it could also lead to a significant increase in capacity of the simulator to run at a higher speed and to scale to networks beyond the memory limitations of a single GPU.

Furthermore, the Spike simulator is currently primarily written in C++ and CUDA. These languages are low-level compiled languages and require a relatively high degree of experience to use. Comparatively, simulators such as Brian2, ANNarchy, and NEST are Python based (at least in the frontend) and therefore allow users with a range of experience to make efficient use of these simulators. In order to draw a larger community of users, a higher level language frontend is hoped to be developed in future. This would bring the capabilities of the simulator to a larger user base.

Finally, rather than rigid descriptions of neuron and synapse types, competing simulators such as GeNN, Brian2 and ANNarchy allow a user to define the neuron and synaptic dynamics in text form. The text provided by the user is then processed by a symbolic processing package to produce the required device code to solve these systems. Similar integration of Spike with a high-level front-end would allow the networks to be defined and run in a much more flexible manner.

4

Leveraging Excitatory-Inhibitory Balance for Competitive Learning in a Spiking Neural Network

4.1 Introduction

Competitive learning is a paradigm in which neurons compete to become active and receptive field formation is guided by this competition. This concept relates to that of sparse coding, in which only a small proportion of neurons are simultaneously active and convey information regarding a sensory stimulus – though the training methods used are often quite different. These approaches are reported to provide significant benefits for the efficient encoding of information in both biological and

modelled neural networks [147, 57].

In order to produce competition and sparseness in a biologically realistic model, excitatory and inhibitory contributions to a neuron must be tuned. Simultaneously, balanced excitation and inhibition is widely evidenced to exist in cortical circuits. These features of balance, competition, and sparse coding have not yet been explicitly made complementary in modelled networks of spiking neurons. In particular, how local and unsupervised spike timing-dependent plasticity (STDP) rules could give rise to this range of phenomena is unclear.

In this chapter we present a mechanistic model which integrates these phenomena and shows their coherence in a single learning framework with an emphasis on biological plausibility.

4.1.1 Sparse Coding

Information in neural circuits is transmitted by the propagation of patterns of neural activity. The specific method by which this information is encoded in these activities is known as the coding scheme or neural code. Sparse coding refers to a method of information encoding such that a small fraction (often as small as possible while meeting some criteria) of neurons within a circuit are in a highly active state in response to a stimulus. In order to implement a coding strategy such as sparse coding, competition is required between neurons and any learning processes must maintain or produce the desired sparseness. Thus, competitive learning and sparse coding are highly related propositions.

Coding hypotheses of neural circuits are often debated based upon the theoretical merit of the coding scheme in terms of energy efficiency, and how efficiently information is encoded in the neural responses. Sparse coding in particular is favoured as it avoids the high metabolic cost of neural firing by ensuring a small percent of neurons are active in response to stimuli [112, 113]. This combined

with the consistency of sparse coding with dimensionality reduction, by which a high dimensional sensory input space can be compressed into a set of latent variables or features, which the active neurons represent, expresses it as an 'efficient coding' hypothesis [65, 18, 9, 8].

On one extreme, sparse coding becomes local coding when only a single neuron is in a highly active state in response to a single stimulus feature while all other neurons are inactive. Such local coding, also commonly known as grandmother cell encoding, has no redundancy and little capacity; the number of stimulus features capable of being represented is equal to the number of neurons N. However, local coding allows for the presentation of multiple stimuli such that their corresponding neurons can be co-active; thereby faithfully dealing with multiple stimuli/features. Furthermore, neural responses are linearly separable and therefore amenable to decoding by a small number of downstream neurons with a simple learning rule.

Dense distributed coding is the other extreme in which the number of highly active neurons is allowed to vary up to the total number of neurons, N. Dense distributed coding provides a significant capacity – for a population of N neurons, up to 2^N distinct patterns can be supported if we assume binary neural activities in every possible combination. This also, however, poses a difficulty when the system is presented with multiple stimuli. Activating multiple network patterns results in significant overlap between them and this makes it difficulty to distinguish which specific stimuli were present. Furthermore, a dense distributed coding scheme, with its high overlap between distinct patterns, is decodable only through non-linear decision boundaries appropriately placed in the high dimensional neural response space. Learning such boundaries requires more neurons than compared to learning linearly separable encodings and requires more sophisticated learning rules. Given the drawbacks of both the local and dense distributed coding schemes an intermediate sparse coding approach, in which relatively few (but more than one) neurons are active for a single stimulus, provides a compromise in both encoding

capacity and stimulus overlap [59].

Two distinct measures of sparseness can be discussed: population sparseness and lifetime sparseness. Population sparseness is measured across an entire population of neurons and describes the fraction of all neurons which are active in response to a stimulus. In contrast, lifetime sparseness is measured for individual neurons and describes the fraction of all stimuli to which a single neuron is active – the fraction of a neuron's 'lifetime' for which it is in an active state. Calculating sparseness values in a network of binary neurons (neurons which are only ever either "active" or "inactive") is trivial; population sparseness is the number of neurons which are active as a fraction of the total population size, and lifetime sparsity for a single neuron is the number of times it is active as a fraction of the total number of samples. However, in real neural systems, activities are not binarized and instead neurons exhibit a distribution of firing rate responses to natural stimulation. A number of population sparseness measures have emerged for distributions of firing rate responses which were compared by Willmore et al. [209]. These measures generally produce a measure of the length of the tail of the firing rate distribution. In this chapter we provide a natural method for binarisation of continuous valued firing rates and thus produce a simple measure of sparseness, avoiding the difficulty described above.

Experimental studies provide us with some indications of the sparseness of real neural systems. The specific experimental technique used to collect neural data affects the strength of this evidence. Electrophysiological studies often involve blind insertion of electrodes, a process by which neurons are searched for by moving said electrode until a signal is identified. This method, due to the searching process, ignores mostly silent neurons and therefore does not provide a representative sample of the responses of the neural population. Alternative methods employ activity independent approaches for neuron selection. For example, glass electrodes can be specifically guided to visually identified neurons via microscope. Alternatively, two-photon calcium imaging can be applied across a plane to view the post-spike

calcium transients of a field of neurons. These activity independent processes provide a less biased estimate of population statistics.

An early experimental study provided data on the population sparseness of primate temporal visual cortical neurons through blind electrode insertion [164], and showed that the sparseness of responses depended highly upon the type of stimulus presented – showing more distributed representations (less sparse) for stimuli with faces versus inanimate objects. Another blind electrode insertion study employing natural images showed that V1, V2 and V4 primate visual cortical neurons have a similar neuron lifetime sparseness across layers [210]. However, this study did not make claims regarding population sparseness due to its selection of neurons by blind electrode insertion. More recent studies of visual and auditory cortical responses indicate a close to 5% population sparseness measure [218, 89]. These studies employ glass electrode based [89] selection of neurons, and two-photon calcium imaging [218] in order to provide a measure of population sparseness while minimising the sampling bias. These data provide evidence for the hypothesis that a sparse distributed representation appears to be the encoding method for these cortical areas.

4.1.1.1 Models of Sparse Coding

Other than the theoretical explanations of the efficiency and relative computational benefits of sparse coding, modelling studies have also provided support for a sparse coding principle through reproduction of neuron receptive field tunings. An early modelling study by Olshausen and Fields [146] presented a single layer neural network model. This model produces V1 simple cell like receptive fields when trained to minimise a two factor cost function through gradient descent. These factors are the sparseness of network response (measured as the sum of all output neuron activations), and the ability for the network to reconstruct the input images given the network's response. The second component of this loss function is an information preserving pressure ensuring that the network responses maintain information

regarding the input image, essentially producing a compressed representation of the input. The model descends the cost function through training and produces receptive field tunings reminiscent of V1 simple cells. Whitened and demeaned natural images were used for training of the network. Randomly selected 16x16 pixel patches were extracted from the dataset and presented to the network. This process of random patch extraction and presentation to a network during training is a common process used in many of the sparse coding models described here.

During training, the activations of neurons were calculated in responses to patches and then these activations were adjusted to minimise the cost function (the sum of the reconstruction error and the total network activation). Fixing these activations, the network weights were modified to make a step down down the gradient of the cost function with respect to the network weights. This process was repeated, iterating between finding the activations, and adjusting the weights, in order to minimise the cost function. This process of descending the cost function landscape was carried out in a batched fashion (updates based upon the errors from batches of images) until the weights reached a stable state.

Similar to the approach of Olshausen and Fields [146], decorrelated independent component analysis applied to natural images [12] was shown to produce similar V1 simple cell-like receptive fields. These studies were early successes of sparse coding through the reproduction of early visual cortex receptive field tunings. However, these models employ rate based neuron models with learning which use non-local information. Non-local refers to the requirement during learning for information at synaptic connections about the activities of neurons which are not pre or post-synaptic – i.e. not local. The plausibility of such non-local learning rules is unclear and has been the subject of some debate [77]. Beyond the non-locality of learning, Dale's law (the fact that an individual neuron cannot send both excitatory and inhibitory projections to other neurons), spiking dynamics, an account for the function of spike timing-based learning rules observed in the

brain were not addressed in either the study by Olshausen and Fields [146] or the study by Bell and Sejnowski [12].

Attempts have been made to bridge these early sparse autoencoder models to more biologically realistic networks with local learning rules. There are a number of approaches in the rate-based network modelling literature [21], though fewer in spiking network models. One example is the modelling work by Zylberberg et al. [231] where they present a spiking neural network model trained with local learning rules which produces sparse coding and effectively carries out gradient descent upon the same loss function described earlier, though through local only learning rules. This study was followed up with a further study of a similar network but with Dalean output neurons which were either excitatory or inhibitory. This updated study emphasised in particular the decorrelative properties of inhibitory neurons [105].

Nonetheless, the model presented by King et al. [105] had a number of drawbacks. Despite the Dalean output layer neurons, the excitatory inputs to these models were current injections which were positive or negative depending upon the input stimulus. This meant that the input neurons switched from being excitatory to inhibitory on a stimulus by stimulus basis, thereby breaking Dale's law within the inputs to the system. Furthermore, the learning rules used for these modelling studies were not spike timing-dependent. Instead, stimuli were presented to the network for some time and spikes were analysed post-presentation. Weight updates were then based upon decoded firing rates, as averaged over the presentation period.

All of the sparse coding models described above also suffered from a lack of explicit control over the population sparsity. The sparsification in these models was controlled by a regularization term whose weighting affects the optimisation landscape. This optimisation landscape could be biased towards sparsification by increasing the regularization term weighting, however the effect of this bias is indirect upon the sparseness of neural activities. Producing neural network models which have an explicitly modifiable population sparseness was achieved in

the past with moving thresholds placed upon the activation functions (described in Chapter 2). Though the biological mechanisms for this are unclear, such a threshold ensured a controlled sparseness.

The importance of the strictness of implementations of sparsity is emphasised in a recent modelling study by Rehn et al. [154] which points out a lack of diversity in the receptive field tunings of the sparse coding model of Olshausen and Fields, described above [146]. Rehn et al. [154] produce a sparse coding model which employs a 'hard' sparseness in order to produce a greater diversity of tuning widths and receptive field sizes. In particular, for neural network systems in which neurons have an output which is continuous valued, the distinction between an active and an inactive neuron is unclear. The sparse coding models above all include output neurons with real valued activities and the implementations of sparseness placed a penalty upon the neural activity as a whole. Hard sparseness refers to systems in which the number of simultaneously active neurons are limited, rather than limiting or placing a cost upon the amount of activity in the network. This approach of hard sparseness reproduced a range of neural receptive field tunings with greater diversity in the sizes of receptive fields. This better matches experimentally observed in V1 simple cell receptive fields [158]. However, this modelling study implemented rate coded neurons and non-local learning, thus providing little detail in the mechanistic explanation of the emergence of such a system.

Despite the range of evidence supporting sparse coding, from theoretical efficiency to receptive field reproduction, a biologically detailed and mechanistic description of sparseness and how it can be robustly achieved with spike timing-dependent plasticity rules and Dalean spiking input neurons remains to be explained.

4.1.2 Balance

The term "balance" refers to systems in which neurons have matched excitatory and inhibitory inputs. Balanced networks can be described across a continuum from global balance, in which non-specific inhibition balances incoming excitation on the timescale of seconds, to tight balance in which inhibitory inputs to neurons efficiently cancel the impact of incoming excitatory currents on a spike by spike basis. Global balance is often described as providing a consistent and indiscriminate level of inhibition. As such, global balance is the most trivial form of balance. Tight balance is exhibited in systems with highly tuned inhibition which tracks and cancels excitation and which can give rise to precision in the timing of spikes emitted by neurons.

Detailed balance is an intermediate state of compromise between global and tight balance. Excitation and inhibition are balanced on a stimulus by stimulus basis in detailed balance such that neurons receive inhibition which correlates with their level of excitation but its arrival is not 'precise' in terms of the specific arrival times of spikes. See Hennequin et al. [83] for a review of the various types of modelled and observed balance.

4.1.2.1 Experimental Evidence for Balance

Experimental studies provide evidence for both detailed and tight balance in a range of cortical areas and across animal models. These experimental approaches are augmented by theoretical and modelling studies which provide explanations of the benefits of balanced systems.

Early modelling and theoretical studies established that the observed asynchronous and irregular firing of cortical neurons with high variance in inter-spike interval is inconsistent with randomly arriving excitatory action potentials [188]. In both leaky integrate and fire neurons and a morphologically detailed multi-

compartment pyramidal neuron model, a large number of randomly arriving excitatory inputs cause a post-synaptic neuron to fire very regularly. While such regularity encodes neural firing rates well, it is however unlike observed cortical activity. To reproduce the variability of cortical spike times, strong inhibition is required and contributes to a conversion of neuron behaviour from acting as integrators, which are regular in their firing, to sensitive co-incidence detectors, which fire with a great deal less regularity [188]. This principle was later explored in modelling studies in it was shown that balanced excitatory and inhibitory activity reproduces the variability of cortical spike times whilst maintaining the ability to propagate signals [204, 203].

Detailed balance of excitatory and inhibitory contributions to a cell is evidenced by correlated excitation and inhibition and a co-tuning of excitatory and inhibitory neurons. Correlated excitatory and inhibitory inputs to cortical cells have been observed both during and outside of sensory stimulation [142, 213]. Similarly, after manipulation of the excitatory receptive field tunings of primary auditory cortical neurons *in vivo*, a re-tuning of incoming inhibition to match the novel receptive fields has been observed [62]. In auditory cortical studies, within days of hearing onset excitatory and inhibitory receptive fields become co-tuned [194]. The degree of selectivity and relative tuning sizes of excitatory and inhibitory receptive fields during and soon after development vary across studies, however the process of refinement of excitatory receptive fields through inhibitory receptive field tuning is consistent [194, 46]. Similar observation of co-tuned excitation and inhibition has been observed in somatosensory cortex [142] and in the visual cortex [5], giving evidence for detailed balance across cortical regions.

Tight balance similarly has a range of evidence in cortical recordings which exhibit highly correlated excitatory and inhibitory currents with inhibition being slightly delayed in its arrival (1-4ms). This delayed inhibition allows rapid excitatory activity to produce spikes and subsequently this inhibition quenches the activity of

cells, producing highly precise spike timing responses in both auditory cortex [207] and in rodent barrel cortex [208]. The particular function of this delayed inhibition is evidenced in the barrel cortex literature [208] where the direction of deflection of a whisker can determine the delay of the inhibitory input to barrel cortex cells. Deflection in one direction is accompanied by delayed inhibition such that the excitatory activity produces a single excitatory spike before the inhibition arrives. Deflection in the other direction is accompanied by inhibition which quenches excitation before any spikes are emitted. Similar delays in inhibition have been observed to precede transitions between Up and Down cortical states [79].

Aside from the evidence for balance, modelling studies have addressed both computational benefits of various states of balance as well as learning mechanisms which can produce a balanced state in a network model. Models range from those making use of detailed balance to extremely precise and tight balance in both feedforward and recurrent networks. A description of some of these modelling studies and their supporting literature follows.

4.1.2.2 Models Leveraging Balance

Modelling studies provide a range of systems in which detailed and tight balance are key to computational efficacy. These focus on the effect of tuned inhibition upon network activity and how these inhibitory structures can be produced. Inhibitory plasticity has been proposed as a mechanism to produce balance and has been shown as useful for explaining a range of neural circuit observations including homeostasis, decorrelation, stable learning, and more [189, 83]. Ultimately, the shaping of cortical activity and excitatory receptive fields by inhibition is well established [92] and the studies which follow provide insights into the potential benefits and mechanisms of inhibitory tuning.

Tuned inhibition in the tight balance regime has provided models which allows inhibition to precisely 'explain away' excitatory inputs to other neurons in a network

[23, 45]. The term 'explaining away' here refers to incident inhibition acting to precisely remove the excitatory effect of a particular input feature from the membrane voltage of neurons in the network. Although the term "explaining away" is typically used in the context of statistical inference, this is a non-typical use of the term which has been adopted for the specific case of tuned inhibition in spiking network models [130, 217].

Such models can be used to produce recurrent networks of spiking neurons with local error-feedback that moulds the excitatory and inhibitory connectivity to learn the dynamics of any arbitrary linear dynamical system [22]. This tightly balanced approach produces densely connected networks in which individual spikes carry a great deal of information, allowing metabolically efficient computation and precisely balanced processing.

Work in dynamical systems has been carried out, producing models of transient motor cortical neuron responses [84]. These methods, rather than providing a learning mechanism for the inhibitory balancing feedback, instead balance inhibition by optimisation of the inhibitory connectivity matrix in order to produce a negative feedback effect for all self-amplifying states of the network. This avoids explosions of activity within the recurrent system and results in the initial state of the system leading to transient neural firing rate trajectories with multiple phases which decay to baseline, as observed in motor cortex responses.

Decorrelation of neural activities through inhibition is evidenced by neurobiological observation [185, 60]. This decorrelative property of inhibition is investigated in a number of modelling studies [105, 197, 13]. The study by King et al. [105] is an example of a spiking network model attempt, most similar to that presented here. In particular, their modelling study showed that the use of a correlative Hebbian inhibitory plasticity rule on inhibitory synaptic connections produces decorrelated neural responses. Furthermore, the strengths of inhibitory connections develop such that; the more similar the responses of the inhibitory pre-synaptic neuron

and post-synaptic neuron, the greater the weight of the synaptic connection. The result of this inhibitory structure is a decorrelation of the post-synaptic neuron activity from its afferent inhibitory neuron activity and leads to a sparse coding model. Furthermore, in the case of detailed balance a correlation between inhibitory synapse strength and the tuning of the pre-synaptic (inhibitory) and post-synaptic cell receptive field tunings is to be expected.

The local inhibitory spike timing-dependent plasticity (STDP) rule proposed in a modelling study by [201] provides not only detailed balance but also firing rate homeostasis. This STDP rule is symmetric and Hebbian, such that the change in synaptic weight is not dependent upon the order of pre and post-synaptic neuron spike times but provides the greatest weight change when pre and post-synaptic spikes are co-incident. This rule is supported by evidence in auditory cortical recordings of relatively co-incident pre and post-synaptic spikes leading to potentiation of the inhibitory synapse, regardless of which neuron spiked first [42].

This inhibitory plasticity rule, adapted to a single neuron rate coded modelling study which did not employ any spiking dynamics, was developed into a description of inhibitory decorrelation and receptive field formation [38]. This model showed the effect of a range of input tuning parameters upon the emergent excitatory receptive field. Excitatory learning was carried out using Hebbian learning combined with a weight normalization process. It was demonstrated that the tuning widths of inhibitory inputs to a single neuron affects the emergence of excitatory receptive fields.

All of the above studies provide compelling theoretical models which make use of inhibitory tuning. Ultimately these models have various drawbacks in the extent of their predictive power and in their biological plausibility. However they provide ample theoretical support for the potential computational benefits of detailed and tight forms of balance.

4.2 Hypothesis and Intuition

Despite promising results in the explorations of both sparse coding and excitatory-inhibitory balance, models are yet to faithfully bridge these fields while maintaining neurobiological details and realism. We propose a framework in which firing rate homeostasis and excitatory-inhibitory balance is achieved within a network through the local inhibitory synaptic plasticity rule of [201]. Providing there is sufficient inhibition, a given neuron's average activity tends toward a target firing rate, determined by parameters of the inhibitory synaptic plasticity rule, after training. Though a neuron's average response converges to this target firing rate, its responses to individual stimuli are distributed about this average value. Thus, a subset of stimuli elicit a higher than average firing rate and the remaining stimuli elicit proportionally lower than average firing rate responses. The specific stimuli which causes a given neuron's activity to exceed the average firing rate depends upon the tuning of incoming excitation and inhibition. Thus, mutual inhibition creates competition and receptive fields are formed based upon stimuli to which neurons are not inhibited.

We use these imperfections in inhibitory balance to allow neurons to form receptive fields. The facilitation of a receptive field to those stimuli for which the neuron has a 'higher than average' firing rate is achieved through a modified excitatory spike timing-dependent plasticity (STDP) rule based upon the weight dependent STDP rule of Van Rossum et al. [199]. Our modelling study thus employs both an excitatory and an inhibitory STDP rule which act in a complementary fashion in order to produce sparse coding.

An intuitive perspective of our proposal is to compare it to sliding-threshold-based learning rules such as the Bienenstock Cooper Munro [20] rule. In such rules changes in synaptic weight depend upon whether the firing rate of the neuron exceeds some threshold which track a measure of the neuron's past activity. In

this work, we draw an equivalence between such a sliding-threshold-based learning and learning which operates on a fixed threshold with a homeostatic mechanism. In particular, the inhibitory synaptic plasticity rule of Vogels et al. [201] "slides" the average neuron firing rate towards a fixed target value (providing homeostasis). Simultaneously, we implement a custom excitatory plasticity rule which potentiates synaptic connections weights when a neuron's response to exceed this threshold. This intuition is explored further in Section 4.3.3.2.

We describe the proposed mechanism by which sparse coding is achieved in a theoretical model before considering networks of spiking neurons. All models obey Dale's law, with separate populations of excitatory and inhibitory neurons, and inputs to the network provided by excitatory spiking neurons with spike times sampled from a Poisson distribution (see Methods Section 4.3.4). Our models are trained under two datasets: a simple artificial stimulus set, and a natural image training dataset [146].

4.3 Methods

4.3.1 Network Dynamics Intuition

In order to develop an intuition and theoretical expectation for the dynamics of the proposed network structure, we begin by considering the behaviour of a probabilistic system of inhibitory neurons with binary activities.

For the results in this Chapter we exclusively consider networks of neurons which are excited by input sources and laterally inhibit one another. This network structure is depicted in Figure 4.1(a).

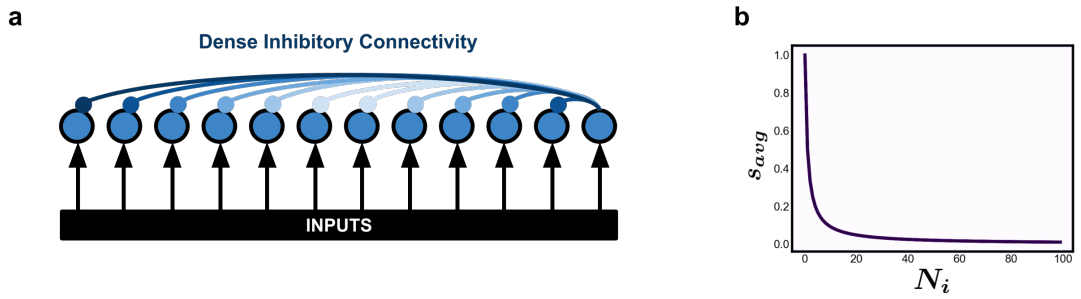

Figure 4.1: **Analysis of a densely connected network of inhibitory neurons.** *(a)* A depiction of the dense inhibitory network structure. Blue connections among the layer of inhibitory neurons indicate the lateral inhibitory connections. Black arrows indicate excitatory stimulation from some input source. The number of inhibitory neurons in the circuit is N_{i} and all inhibitory neurons form synapses targeting all other inhibitory neurons without self-inhibition. *(b)* A plot of the predicted average probability of inhibitory neuron activation, s_{avg}, (see Equation 4.1) for a range of different inhibitory population sizes, where the number of inhibitory cells in the network is N_{i}.

4.3.1.1 A Densely Connected Inhibitory Circuit

Consider a network of inhibitory neurons, each of which probabilistically occupies one of two possible states; 1 or 0 – "active" and "inactive" respectively during some arbitrary input stimulation. This network is densely connected with every inhibitory neuron connected to every other inhibitory neuron but without any self connections. For an individual neuron, its probability of being active to a stimulus has a maximum value of 1.0 which is reduced based upon the probability of any inhibitory afferent connection being active. Given the dense inhibitory connectivity, the probability of being active is thus also dependent upon the probability that any other neuron in the network is active. The network is thereby in a state of competition. We give no guarantee on the convergence or equilibrium of this system but we discuss it from here onwards to produce some theoretical expectations to which we later map spiking neural network behaviour.

The network structure, Figure 4.1a, shows a group of inhibitory neurons excited by some input source with dense lateral inhibitory synaptic connections. This

structure represents the theoretical network we have described and the number of competing inhibitory neurons is denoted N_{i}, in Chapter 5 we shall also consider excitatory neurons of number N_{e} such that the total number of neurons in a network is $N = N_{\mathrm{i}} + N_{\mathrm{e}}$. We shall primarily consider networks of dense inter-inhibitory neuron connectivity with no self-connections, however similar analysis can be applied to networks with diluted connectivity.

Let us consider this network of inhibitory neurons as being in some activity state at each moment in time depending upon the particular input stimulation. If this input state was to vary over time and we recorded the network activity for a very long time, we could measure the probability with which neurons are active across all input states. This allows us to measure the average probability with which any neuron is active as s_{avg}. Let us further assume that this network is completely homogeneous, such that every neuron's average activation probability is equal to s_{avg}.

We now wish to describe how s_{avg} depends upon the number of inhibitory neurons in this competing network of N_{i} neurons. Consider sampling a single inhibitory neuron from this network. The sampled neuron is inhibited by $N_{\mathrm{i}} - 1$ neurons, each of which are active on average with probability equivalent to the average neuron activation probability, s_{avg}. The sampled neuron is, given that we assumed a homogeneous system, also active with probability equal to s_{avg}.

In an extreme case of competition, only a single neuron in a network is active at a time. This corresponds to a case in which we have a localist form of coding. Under the assumption that all N_{i} neurons sufficiently inhibit all other such that they have anti-correlated activations, we would occupy this strong state of competition. Given this state of competition, we can describe the average activation probability of a neuron as s_{avg} such that

$$s_{\mathrm{avg}} = \frac{1.0}{N_{\mathrm{i}}}. \tag{4.1}$$

Another equivalent method by which we can describe this strong inhibition is to assume that neurons being active is a mutually exclusive phenomenon. If we also assume that all neurons have the same activation probability, s_{avg}, then we can describe this probability as

$$s_{\text{avg}} = 1.0 - s_{\text{avg}}(N_{\text{i}} - 1), \tag{4.2}$$

where the subtractive term describes the exclusive nature of each neurons activation relative to all other $(N_{\text{i}} - 1)$ neurons. This method of description proves useful when we extend our analysis in Chapter 5.

An individual neuron's activity is by this description directly dependent upon the network size, N_{i}. We can calculate the converged average activation probability of neurons for a range of network sizes and this relationship is shown in Figure 4.1b. This is the expected relationship between competing inhibitory population size and the average probability of activation in our idealized and strongly competing network setup.

Ultimately, our analysis here has described a simple system in which only a single neuron is ever active at a time. Given the assumption that all neurons having the same probability of being active, as the network size grows individual neurons are active with a smaller probability. This describes a system with a localist coding scheme, an extreme end of the sparse coding proposal.

4.3.2 Implementation in a Spiking Neural Network Model

In order to bridge the above theoretical model to spiking neural network models, we must define consistent neuron and synapse dynamics. First, we propose binarising the responses of our spiking neurons into an 'active' and 'inactive' state. This is accomplished by determining whether the firing rate of a given neuron is above or below, respectively, a threshold. Given our intention to make use of

an inhibitory synaptic plasticity rule which brings post-synaptic neurons to a target, or homeostatic, firing rate, we use this target firing rate as the binarisation threshold. This binarisation allows us to measure the sparseness of our network responses.

Aside from the inhibitory connectivity and learning, neurons must form receptive fields for those stimuli to which their firing rate rises above the firing rate threshold. In order to produce such learning, we introduce a local excitatory synaptic plasticity rule which has a dependence upon the post-synaptic neuron firing rate. The network details and plasticity rules are presented below.

4.3.3 Spiking Neural Network Details

4.3.3.1 Neuron and Synapse Models

The spiking neural network models presented in this chapter consist of interconnected inhibitory neurons. The neurons are modelled with traditional leaky integrate and fire (LIF) dynamics such that every neuron has an associated membrane voltage, $V(t)$, which decays over time and is modulated by conductance based synaptic inputs.

$$C_{\text{mem}}\frac{dV(t)}{dt} = g_{\text{leak}}(V_{\text{rest}} - V(t)) + g_{\text{exc}}(E_{\text{exc}} - V(t)) + g_{\text{inh}}(E_{\text{inh}} - V) + I_{\text{bg}} \quad (4.3)$$

Equation 4.3 shows the membrane voltage dynamics of a single cell where C_{mem} is the membrane capacitance of the cell, g_{leak} is the leakage conductance of the cell, and V_{rest} is the rest voltage to which the neuron voltage decays under zero input. The cell time constant τ_{mem} can be calculated $\tau_{\text{mem}} = \frac{C_{\text{mem}}}{g_{\text{leak}}}$. Synaptic inputs are modelled by g_{exc} and g_{inh} which are the excitatory and inhibitory synaptic conductances respectively, while E_{exc} and E_{inh} are the excitatory and inhibitory reversal potentials, respectively. Finally, I_{bg} is a constant background input current.

Note that in all simulations for this chapter, the background input current is positive and ensures excitation of neurons even under cases where their afferent excitatory synaptic connections are depressed to zero.

Neurons emit an action potential when the membrane voltage, $V(t)$, crosses a voltage threshold V_{thresh}. Upon such an action potential, the neuron's membrane voltage is hyperpolarized and fixed at the reset voltage, V_{reset} for a period of time corresponding to the absolute refractory period, τ_{ref}. Action potentials also cause the conductances of post-synaptic neurons to be increased after awaiting the axonal transmission delay – the time required for a spike to reach the synaptic connection from the soma of the pre-synaptic neuron. For an excitatory synaptic connection, upon a pre-synaptic neuron spike arrival at the synaptic connection the post-synaptic neuron's excitatory conductance is updated such that $g_{\text{exc}} \leftarrow g_{\text{exc}} + w \cdot g_{\text{leak}}$ where w is the weight of the synaptic connection. For inhibitory synaptic connections, upon a pre-synaptic neuron spike arrival at a synaptic connection the post-synaptic neuron's inhibitory conductance is updated such that $g_{\text{inh}} \leftarrow g_{\text{inh}} + w \cdot g_{\text{leak}}$. The arrival of spikes at synaptic connections is delayed by any axonal transmission delay. In this study, the axonal transmission delays are all fixed at 1ms. The synaptic conductances otherwise follow decaying dynamics governed by time constants τ_{exc} and τ_{inh} as follows;

$$\tau_{\text{exc}} \frac{dg_{\text{exc}}}{dt} = -g_{\text{exc}} \qquad \text{and} \qquad \tau_{\text{inh}} \frac{dg_{\text{inh}}}{dt} = -g_{\text{inh}}.$$

The dynamics described above are all solved with the Forward-Euler numerical integration scheme as described in Chapter 3.

4.3.3.2 Plasticity Rules

Plasticity rules modify the synaptic weights, w, of the synaptic connections between neurons. This study implements two distinct plasticity rules: inhibitory synaptic

plasticity as proposed by [201] and a custom weight-dependent excitatory plasticity rule, a modified version of that by Van Rossum et al. [199].

The specific weight updates for spike timing-dependent plasticity rules depend upon the difference in time ($\delta t = t_{\text{post}} - t_{\text{pre}}$) between the arrival time of a pre-synaptic neuron spike at the synaptic connection and the time of a post-synaptic neuron spike (t_{pre} and t_{post} respectively). The pre and post-synaptic spike pairs can be chosen in two ways: nearest only STDP or all to all STDP. Nearest only STDP is the case in which, upon a pre or post-synaptic spike, the weight update is only calculated for the most recent spikes of the paired neurons. In contrast, all to all STDP updates are the case in which weight updates are calculated for every possible spike pair in the spiking history of the paired neuron. We hereafter exclusively make use of the all to all spike STDP paradigm such that upon every action potential emitted by either the pre or post-synaptic neuron, the weight is updated based upon all past spikes of the other (paired) neuron.

Inhibitory Synaptic Plasticity

We use the inhibitory STDP rule, referred to as iSTDP in this chapter, proposed by Vogels et al. [201]. This learning rule produces firing rate homeostasis and detailed balance through a local STDP learning rule. Inhibitory synaptic connections are modified according to this rule such that post-synaptic neuron firing rates are brought to a "target" firing rate. The specific target firing rate, λ_{target}, to which the neurons are brought depends upon the parameters of this inhibitory plasticity rule.

$$\Delta w = \begin{cases} \begin{cases} e^{(-\frac{\delta t}{\tau_i})}, & \text{if } \delta t \geq 0 \\ e^{(\frac{\delta t}{\tau_i})}, & \text{otherwise} \end{cases}, & \text{for pre/post-synaptic spike pairs} \\ -\alpha_i, & \text{for every pre-synaptic spike} \end{cases} \tag{4.4}$$

Weight updates to inhibitory synaptic connections under this rule are implemented as shown in Equation 4.4, where Δw is the change in weight of a synaptic connection between a pair of neurons. δt is the post-synaptic neuron spike time, t_{post}, minus the arrival time of the pre-synaptic neuron's spike at the synaptic connection, t_{pre}. Thus for a given pair of spikes emitted by the pre and post-synaptic neurons: $\delta t = t_{\text{post}} - t_{\text{pre}}$.

For all pairs of pre and post-synaptic spikes, there is a positive change in synaptic weight (see the exponential terms). This change in inhibitory synaptic weight is greatest for pre and post-synaptic spikes which are close in time (where δt is close to zero) and is small for large differences between pre and post-synaptic spike times (where δt has a significant magnitude, positive or negative). Therefore, the weight update is symmetrical with respect to the order of the post/pre-synaptic spikes and creates the most positive change in inhibitory weight when pre-synaptic spikes arrive at a synaptic connection close in time to any post-synaptic neuron's action potential. The weight potentiation strength decays according to the time constant τ_i.

There is one source of weight reduction, LTD, which is applied at the time of each pre-synaptic spike without considering post-synaptic spike times. There is a reduction in synaptic weight by an amount labelled α_i upon each pre-synaptic spike. Updates to individual synaptic connection weights are carried out for each pre and post-synaptic spike event such that $w \leftarrow w + \eta_i \Delta w$ for each update, where η_i scales the amount of weight change per event and is the learning rate.

Potentiation and depression are in equilibrium at a specific post-synaptic neuron firing rate as described in the original publication [201]. This publication describes the appropriate setting of the α_i parameter in order to achieve a particular firing rate. For a desired target firing rate, λ_{target} of the post-synaptic neuron population, this parameter can be set:

$$\alpha_i = 2.0 \cdot \lambda_{\text{target}} \cdot \tau_i, \tag{4.5}$$

Note that this rule described above is precisely that which was outlined by Vogels et al. [201] and is kept unmodified.

Excitatory STDP Rule

We describe a custom excitatory STDP rule, referred to in this chapter as eSTDP, which is applied to all excitatory synaptic connections in our network simulations. This eSTDP rule is a modified version of the soft-bound, weight-dependent, STDP rule described by Van Rossum et al. [199].

$$\Delta w = \begin{cases} \alpha_+ e^{(-\frac{\delta t}{\tau_+})}, & \text{if } \delta t \geq 0 \\ -w \cdot \alpha_- e^{(\frac{\delta t}{\tau_-})}, & otherwise \end{cases} \tag{4.6}$$

The original learning rule, described by Equation 4.6, produces synaptic long term potentiation (LTP) of synaptic connection weights for every spike pair in which the post-synaptic spike happens after a pre-synaptic spike arrives at the synapse. The weight change during LTP is independent of the current synaptic weight and depends solely upon the time difference between the arrival times of the pre and post-synaptic neuron spike times at the synapse. This STDP rule implements long term depression (LTD) of synaptic connection weights for every spike pair in which the pre-synaptic spike arrives at the synapse after the post-synaptic spike. This depression is also dependent upon the current synaptic weight, w, in a multiplicative fashion. The effects of LTP and LTD decay according to the time constants τ_+ and τ_- respectively. The relative magnitudes of the LTP and LTD contributions are governed by α_+ and $\alpha-$ respectively. Updates to individual synaptic connection weights are carried out for every pair of pre and post-synaptic neuron spikes such that $w \leftarrow w + \eta_e \Delta w$ for each update, where η_e is the learning rate which scales the amount of weight change per synaptic weight update.

Determining the expected output weight distribution produced by a given STDP

rule is possible by a number of analyses which are described in a number of modelling studies [171, 33, 30]. These make use of the Fokker-Planck and Langevin equations, both produced originally to describe the evolution of a particle's position under the influence of Brownian motion. When applied to STDP rules, these analyses treat an individual weight as a particle under drift and noise forces; drift being the macro-scale change of the particle's mean position over time and a noise term describing how the true position is distributed about this mean.

The mean weight value, at some time t, is hereafter labelled $\bar{w}(t)$. The drift of a synaptic weight under a given STDP rule, $A(\bar{w}(t))$, corresponds to derivative of the mean of the weight over time such that $A(\bar{w}(t)) = \frac{d\bar{w}(t)}{dt}$.

For a stable STDP rule, this drift describes how the mean of a weight migrates from its initial value to some steady state value. The noise imposed on a synaptic weight by an STDP rule determines the relative distribution of the weight about this mean and is a stochastic process. These quantities depend upon the pre and post-synaptic neuron firing patterns.

For the learning rule provided by Van Rossum et al. [199], and given in equation 4.6, the drift function, $A(\bar{w}(t))$, under an all to all STDP paradigm is described in detail by Burkitt et al. [30]. The calculation for this drift term is a summation of the average potentiation and depression contributions to the weight, namely that from LTP and LTD, under the assumption that the pre and post-synaptic neurons are firing randomly according to a Poisson distribution. The analysis produces a drift term:

$$A(\bar{w}(t)) = \frac{d\bar{w}(t)}{dt} = \lambda_{\text{out}}\lambda_i[\alpha_+\tau_+ - \bar{w}(t)\alpha_-\tau_-] \tag{4.7}$$

where λ_{out} is the firing rate of the output neuron, and λ_i is the rate of arriving pre-synaptic spikes at the synaptic connections. The fixed point of this equation, where $\frac{d\bar{w}(t)}{dt} = 0$ provides the fixed point of the mean weight, $\bar{w}_{\text{fp}}$, as;

$$\bar{w}_{\text{fp}} = \frac{\alpha_+ \tau_+}{\alpha_- \tau_-} \tag{4.8}$$

providing the expected converged value of synaptic weights. Notably, this analysis and the analysis which follows assumes random and uncorrelated pre, and post-synaptic neuron spike times. This assumption does not take into account the correlating impact of incoming excitation upon the post-synaptic neuron's responses, however it does simplify this analysis. This assumption is valid if we assume that a single pre-synaptic neuron's impact upon the activity of a post-synaptic neuron is sufficiently small so as to be negligible. In the results presented below, learning in networks are consistent with these assumptions and weights converge to the fixed-points that we describe here.

The learning rule described above has a number of desirable properties including a reliable convergence of the weight distribution and a soft-bound upon weight. This soft bound and mean weight convergence is a direct consequence of the weight dependence of LTD. In this study we modify this plasticity rule in order to produce a dependence of the converged synaptic weight upon the post-synaptic neuron's firing rate. This is intended to allow potentiation of excitatory synaptic weights when the post-synaptic neuron exceeds a fixed-threshold (discussed further below).

Upon every pre-synaptic neuron spike, the synaptic connection between the pre and post-synaptic neurons is reduced by a constant term. This additional reduction in synaptic weight is a novel introduction which we propose in this chapter.

$$\Delta w = \begin{cases} \begin{cases} \alpha_+ e^{(-\frac{\delta t}{\tau_+})}, & \text{if } \delta t \geq 0 \\ -w \cdot \alpha_- e^{(\frac{\delta t}{\tau_-})}, & \text{otherwise} \end{cases}, & \text{for pre/post-synaptic spike pairs} \\ -\alpha_e, & \text{for every pre-synaptic spike} \end{cases} \tag{4.9}$$

Equation 4.9 shows the modified excitatory STDP rule (eSTDP) with its additional depression term (compare to Equation 4.6). α_e is the additional term which we propose, a constant by which synaptic weights are reduced upon each pre-synaptic spike. In particular, changes to synaptic weight based upon pairs of pre and post-synaptic neuron spikes are identical. However, there is an additional source of LTD, synaptic weight reduction, which is applied at the time of each pre-synaptic spike (ignoring pairs). This modification of the learning rule alters the drift function described above.

$$A(\bar{w}(t)) = \frac{d\bar{w}(t)}{dt} = \lambda_{\text{out}}\lambda_i[\alpha_+\tau_+ - \bar{w}(t)\alpha_-\tau_-] - \lambda_i\alpha_e \tag{4.10}$$

Equation 4.10 shows the modified drift, which may be compared to equation 4.7. Due to the independent nature of the additional depression term α_e from the pair based modification of the original rule, the drift function is modified additively by the average additional weight change per second. Given that the pre-synaptic neuron firing rate is given λ_i, the average additional weight change per second from the introduction of our independent depression term α_e is the probability of a pre-synaptic spike multiplied by the synaptic weight change per spike: $-\lambda_i\alpha_e$. Finding the set point of this modified drift is trivial (again by setting $A(\bar{w}(t)) = 0$) such that the fixed point of the mean weight, $\bar{w}_{\text{fp}}$, can be described,

$$\bar{w}_{\text{fp}} = \frac{\alpha_+\tau_+}{\alpha_-\tau_-} - \frac{\alpha_e}{\lambda_{out}\cdot\alpha_-\tau_-}. \tag{4.11}$$

Equation 4.11 presents the mean weight-fixed point as dependent upon the output neuron firing rate, λ_{out}, such that the weight fixed point is greater for synapses which connect to neurons which have a higher firing rate. The relationship shown in Equation 4.11 can be used to allow strengthening of weights under the condition that λ_{out} is greater than a threshold. This threshold can in turn be based upon the target firing rate, λ_{target}, of the iSTDP inhibitory plasticity rule,

similar to the dependence described in equation 4.5.

We can set up α_e such that the expected mean weight, $\bar{w}(t)$, tends to zero at the target firing rate, λ_{target}, as follows. Let us choose to set α_e according to

$$\alpha_e = \lambda_{\text{target}} \cdot \alpha_+ \tau_+. \tag{4.12}$$

Notably, defining α_e according to 4.12 gives both α_+ and τ_+ as factors, both of which are smaller than 1.0. The magnitude of α_e is therefore small and scales with the target firing rate, λ_{target}. We suppose that if the target firing rate of neurons is also small, the effect of this excess LTD term could be hidden among timing based weight updates in electrophysiological studies. Biological basis aside, having redefined α_e according to equation 4.12, the relationship between mean of the weight distribution and the post-synaptic neuron firing rate, λ_{out}, becomes

$$\bar{w}_{\text{fp}} = \frac{\alpha_+ \tau_+}{\alpha_- \tau_-} \left(1.0 - \frac{\lambda_{\text{target}}}{\lambda_{\text{out}}} \right). \tag{4.13}$$

This relationship converges such that $\bar{w}_{\text{fp}} \to \frac{\alpha_+ \tau_+}{\alpha_- \tau_-}$ as $\lambda_{\text{out}} \to \infty$. Similarly the fixed point of the mean weight tends to zero, $\bar{w}_{\text{fp}} \to 0$, as the post-synaptic firing rate approaches the target firing rate, $\lambda_{\text{out}} \to \lambda_{\text{target}}$. For firing rates lower than λ_{target}, the weight $\bar{w}_{\text{fp}}$ would tend to values below zero, however we hard bound all weights at zero, not allowing any weight values lower than zero.

The relationship between the mean weight fixed point $\bar{w}_{\text{fp}}$, the output neuron firing rate, λ_{out}, and the target firing rate, λ_{target}, is shown in Figure 4.2. The blue line shows this relationship for our novel STDP rule and we can observe that the target firing rate acts as a threshold, above which the mean weight fixed point is positive and below which it is negative. Note than in our implementation, we rectify the excitatory weights and therefore the minimum possible synaptic weight is zero. We also show the mean weight fixed point for the original STDP rule which we customised, plotted in red. As can be observed, there is no dependence of the mean

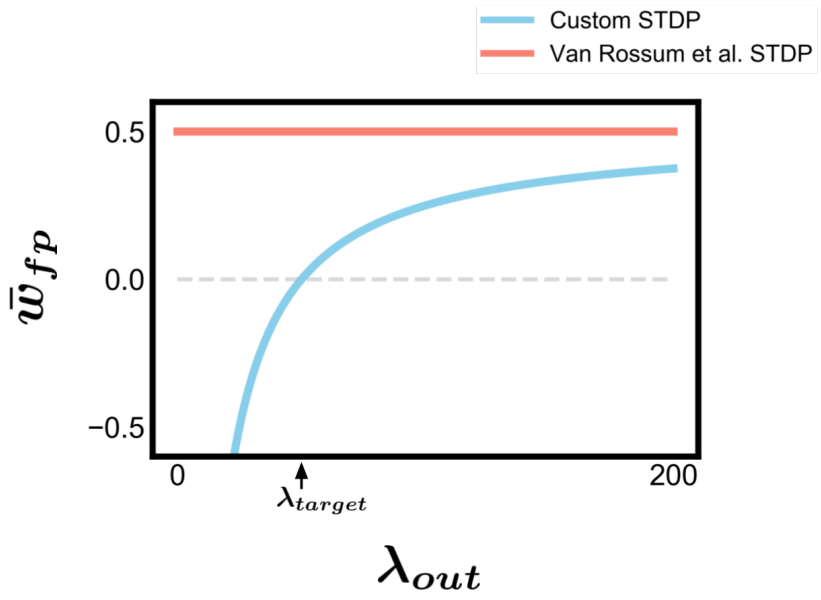

Figure 4.2: **A plot of the synaptic weight fixed points of the Van Rossum et al. [199] multiplicative STDP rule and our proposed custom STDP (eSTDP) rule against output neuron firing rate λ_{out}.** The red curve shows the relationship between the mean weight fixed point of the Van Rossum STDP rule (Equation 4.8) and the blue curve shows the same variable for the proposed custom STDP rule (Equation 4.13). These curves are plot with $\lambda_{\text{target}} = 50Hz$ and $\frac{\alpha_+\tau_+}{\alpha_-\tau_-} = 0.5$ to make clear the interaction of the fixed point with the target firing rate. The target firing rate, λ_{target} is indicated by an arrow.

weight fixed point upon the post-synaptic neuron firing rate. In the context of our earlier proposal for producing sparseness through a combination of excitatory and inhibitory learning, this excitatory learning rule formulation makes it complementary to the inhibitory learning rule described in Section 4.3.3.2. By setting the threshold or target firing rates, λ_{target}, of the excitatory and inhibitory learning rules as equivalent we produce a system in which the inhibition attempts to bring neurons to this homeostatic firing rate and for those stimuli for which it is unable to do so the excitatory learning rule potentiates weights and forms a receptive field.

We hereafter fix all parameters except α_+. α_+ acts to scale the mean weight fixed point and by changing it we modify the maximum value of the mean weight fixed point. All other STDP parameters are kept fixed for all simulations.

Excitatory and Inhibitory Plasticity Combined

Having described the complementary excitatory and inhibitory plasticity rules which we have proposed, we can provide an approximate analysis of the impact of combining them. Considering a single output neuron with firing rate λ_{out}, let us approximate the effect of the inhibitory plasticity rule described in Section 4.3.3.2 as bringing the average value of this firing rate to the target firing rate, λ_{target}, such that $\langle \lambda_{\text{out}} \rangle_t \to \lambda_{\text{target}}$. It is important to note that this is an oversimplification of the impact of inhibitory synaptic plasticity given that the rule is correlative and tends systems toward detailed balance rather than simply global balance. Without considering this correlative strengthening of inhibitory synaptic connections, we ignore the competitive element which drives receptive fields apart. Nonetheless, we can use this simplification in order to better understand the impact of the combined learning rules.

In equation 4.10, we presented the drift function which describes the change in the mean weight value $\bar{w}(t)$ as,

$$\frac{d\bar{w}(t)}{dt} = \lambda_{\text{out}}\lambda_i[\alpha_+\tau_+ - \bar{w}(t)\alpha_-\tau_-] - \lambda_i\alpha_e.$$

We later chose to adopt a specific form for the constant α_e as shown in equation 4.12. If we reframe the drift function with our chosen definition for α_e, we arrive at

$$\frac{d\bar{w}}{dt} = \lambda_{\text{out}}\lambda_i[\alpha_+\tau_+ - \bar{w}\alpha_-\tau_-] - \lambda_i\lambda_{\text{target}} \cdot \alpha_+\tau_+,$$

which can be arranged to give

$$\frac{d\bar{w}}{dt} = \lambda_i\alpha_+\tau_+[\lambda_{\text{out}} - \lambda_{\text{target}}] - \bar{w} \cdot \lambda_{\text{out}}\lambda_i \cdot \alpha_-\tau_-.$$

Given our assumption about the role of inhibitory synaptic plasticity which

tends the average output firing rate, $\langle \lambda_{\text{out}} \rangle_t$, to the target firing rate, λ_{target}, let us assume that this is achieved such that we set $\langle \lambda_{\text{out}} \rangle_t \approx \lambda_{\text{target}}$. For such an assumption to hold true, the timescale at which inhibitory plasticity occurs would need to be significantly faster than that of excitatory plasticity such that it approaches equilibrium immediately following any excitatory synaptic weight changes. In electrophysiological observations, it has instead been shown that the timescale of homeostatic influences are very slow relative to Hebbian learning [102]. Thus, our assumption here and our network setups are not consistent with the relative timescales of homeostatic and Hebbian components of network plasticity change. However studies have also shown that without a rapid compensatory process, Hebbian learning operating at a faster timescale than homeostatic plasticity mechanisms causes instabilities in learning [222, 224]. Therefore, we consider our artificial increase of the timescale of inhibition to be a mechanism by which we can achieve stability without introducing a further fast-timescale network component. In Section 4.3.3.2, we describe how we increase the learning rate of inhibitory synaptic plasticity above that of excitatory synaptic plasticity in order to achieve this.

Given the assumption that $\langle \lambda_{\text{out}} \rangle_t \approx \lambda_{\text{target}}$, we can modify our drift function, giving

$$\frac{d\bar{w}}{dt} \approx \lambda_i \alpha_+ \tau_+ [\lambda_{\text{out}} - \langle \lambda_{\text{out}} \rangle] - \bar{w} \cdot \lambda_{\text{out}} \lambda_i \cdot \alpha_- \tau_-.$$

This equation provides us with an intuition of the effect of the proposed combination of excitatory and inhibitory learning rules. In short, potentiation depends upon the neuron firing rate exceeding its average firing rate, while a background weight dependent decay helps to bound the overall weights. In this form we see a great deal of similarity to the Bienenstock-Cooper-Munro (BCM) rule [20] as described in Chapter 1 Section 1.2.2. In particular, the BCM rule can be expressed

$$\frac{dw}{dt} = \lambda_{\text{in}}\lambda_{\text{out}}(\lambda_{\text{out}} - E(\lambda_{\text{out}})^p) - \eta w$$

where $(E[\lambda_{\text{out}}])^p$ is the expected value of the output neuron firing rate taken to some power. As can be seen, though our drift function is not equivalent to that described by the BCM rule, a number of features are nonetheless very similar. Weight potentiation is proportional to both the input neuron firing rate and the difference between the current output neuron firing rate and some tracking of its mean (or a higher order power of this), and there is a weight-dependent decay in the synaptic weights. This similarity of our plasticity rule compared to the BCM rule is reassuring given how well the BCM rule was shown to fit to plastic network changes in experimental settings.

Thus, the simplifications we made here allow us to consider our learning rule as qualitatively similar to sliding-threshold-based learning rules. However, our assumptions regarding the effect of inhibitory plasticity ignore the correlative learning and detailed balance which is produced by this plasticity model. Therefore, our description does not capture the strong competitive impact of tuned inhibition. Nonetheless, our short aside and analysis of the combined effect of excitatory and inhibitory plasticity is insightful and provides some further intuition.

Parameters

The above described dynamics are implemented in a range of networks, described in the results section below, which have parameters as outlined in Table 4.1. Values of the parameters N_{i} and α_+ are modified to produce the range of results. Wherever changed, the adapted values are given in the text.

One notable feature in these parameters is that the excitatory plasticity rule learning rate is smaller than the inhibitory plasticity rule learning rate. This is maintained across all network simulations in order to limit the runaway of excitatory

Network Parameters	
Number of Inhibitory (I) Neurons, N_{i}	*varied*
Number of Excitatory (E) Neurons, N_{e}	0
Numerical Timestep Δt	0.1ms
Neuron Parameters	
C_{mem}	200pF
g_{leak}	10nS
τ_{mem}	20ms
τ_{ref}	2ms
V_{rest}	-60mV
V_{thresh}	-50mV
V_{reset}	-60mV
E_{exc}	0mV
E_{inh}	-80mV
I_{bg}	120pA
Synapse Parameters	
τ_{exc}	10ms
τ_{inh}	10ms
Initial Excitatory Weight Range	$[0.0, 1.25]nS$
Initial Inhibitory Weight Range	$0.0nS$
Axonal Delay	1ms
Plasticity Parameters	
Inhibitory STDP	
τ_i	20ms
λ_{target}	5Hz
η_i, Learning rate	0.001
Excitatory STDP	
α_+	*varied*
α_-	1.0
τ_+	20ms
τ_-	20ms
λ_{target}	5Hz
η_e, Learning rate	0.0001

Table 4.1: **Network, neuron, and synaptic parameters used for the spiking neural network models in Chapter 4.**

learning. We discuss the motivations and implications of this in Section 4.3.3.2.

4.3.3.3 Inputs

Inputs are provided to the simulated networks through excitatory input neurons which spike at times which are drawn randomly from a Poisson process. This Poisson

process is characterised by a mean which is the mean firing rate of the neuron. Each input neuron is provided a mean firing rate for the sampling of its spike times, and this mean is static for the duration of a stimulus. The stochastic nature of the sampling process means that repeated simulation of our input neuron produces a different spike train, assuming that a new random seed is provided for the random sampling. The number of input neurons and their firing rates differ across the input datasets that we use. Input neurons are always treated as excitatory and therefore increase the excitatory conductances of their post-synaptic neurons when they spike.

4.3.4 Data and Preprocessing

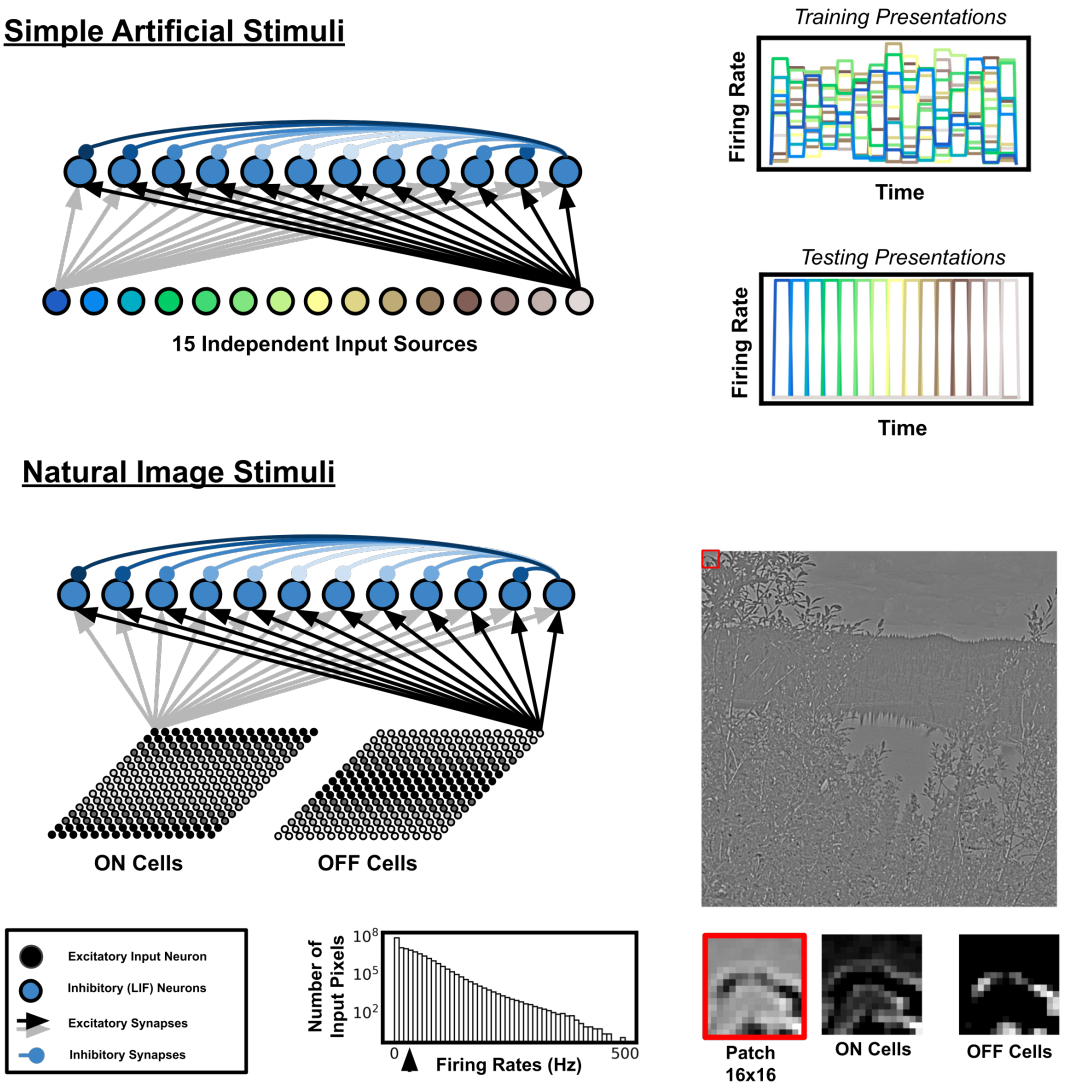

Figure 4.3: **A depiction of the two input datasets used for network training.** *Top,* a depiction of the simple artificial stimulus set and its presentation. A set of 15 independent input sources are used as the inputs to our network of spiking neurons. These input sources are activated in a random combination during training (with the summed firing rate across all input sources equal to 500Hz) but are only active one at a time during testing. Example plots of the training and testing regimes are shown right with colours corresponding to the set of input sources in the network shown left. These input sources are shown fully connected to a group of output neurons. *Bottom,* a depiction of the natural image stimulus set, and an example of patch extraction. Right, an example image from the training data set is shown. The average grey value in the image indicates a response of zero, white indicates a positive response, and black indicates a negative response. Below

is shown a single patch from this image and its decomposition into the positive and negative components (ON and OFF cells respectively). For each image patch, two groups of 16 by 16 input neurons represent the input to our networks. One group responding as ON Cells, the other as OFF Cells. These two populations of input cells to the network are depicted left with full feed-forward excitatory connectivity to a population of output LIF neurons. Bottom middle is shown the firing rate distribution of the training set for the ON and OFF cell inputs to the network. The mean firing rate of the ON and OFF cells across all possible patches is approximately 15Hz and is indicated by a black arrow on the x-axis.

4.3.4.1 Simple Artificial Dataset

A simple dataset was created in which there are 15 independent input sources, Figure 4.3, top. These input sources are each represented by a separate neuron such that they are linearly separable and non-overlapping in the neuron space. During training, individual training stimuli are constructed by randomly weighting a contribution from each input source and combining these to form a single stimulus. The random weighting applied to the input sources is first normalized, ensuring that the total summed input to the network is equivalent across training samples. For every stimulus presentation the total summed input source activity is set to 500Hz. Given the chosen magnitude, each neuron has an average firing rate of 33.3Hz across the training stimuli. The spike times of input neurons are based upon the assigned firing rates and sampled from a Poisson process. Each training stimulus is presented for 50ms followed by the next.

During testing, which is carried out post-training, all plasticity rules are off and only a single input source is active. This corresponds to the active source producing spikes at a 500Hz rate and all other producing no spikes. This is an extremely high neural response, however in Appendix B.0.1 we show equivalent results for a key figure if we instead represent each input source by multiple input neurons each of which have a more reasonable maximum firing rate. Each training sample is presented to the network for 1s and all spike times recorded. Following presentation

of the test set, the output neuron responses are then tested for selectivity and sparseness. These input sources are connected fully, through conductance based synapses, to all neurons in the networks being investigated.

4.3.4.2 Natural Image Dataset

The natural images used to train the sparse coding network of Olshausen and Fields [146] are used to train a model of V1 simple cells in this study. These images are provided demeaned and whitened. The whitening process is accomplished by a filtering procedure which is carried out in the spatial frequency space of the input images, the specifics of which are described by Olshausen and Fields [145] and code for which is provided in Appendix B.0.2. There is evidence that whitening alone does not account for the behaviour of retinal ganglion cells [74, 1], however it is taken as an approximation of their processing by Olshausen and Fields in their early sparse coding models [146]. Given the use of this whitening procedure in early sparse coding models and the continued use of this dataset for the training of models of visual processing [231], we use this same procedure for our training procedure.

The whitening and de-meaning process applied to the natural images results in pixel values which can be positive or negative. In this study, we provide all inputs to networks through excitatory input cells and therefore the negative components of these whitened images must be adjusted for presentation. Negative pixel values from the input dataset are therefore taken as positive responses by "OFF" input cells, and positive values represented as responses by "ON" input cells. These ON and OFF cells are meant as a replication of the presence of ON and OFF-centred centre-surround receptive fields which are characteristic of retinal ganglion cells.

During training, 16x16 pixel patches are randomly selected from the image dataset for presentation to the network. These image patches are pre-processed by centering (de-meaning) and variance-normalizing each patch [231]. This corresponds to subtracting the mean pixel value of the entire patch from each pixel, and dividing

each pixel by the standard deviation of the pixel values across the patch. Thereafter, the patches are split into the positive and negative contributions: for each 16x16 patch two 16x16 input sources are created, one representing the ON cell responses, and another representing OFF cell responses. All positive values within the patch drive the ON cell 16x16 input sources, all negative values within the image patch positively drive the OFF cell 16x16 input sources. Figure 4.3, bottom right, shows an example image from the dataset, an extracted patch, and the ON and OFF cell responses for this patch.

The value of the ON and OFF cell input spaces are scaled between zero and 500Hz and these values are used as the firing rates of these input cells for the presentation of this patch during training. The resulting firing rate distribution of the input cells across all input stimuli is an exponential distribution and the average cell firing rate across all stimuli is 15.25Hz. The distribution is shown in Figure 4.3 bottom middle. Note that the y-axis is logarithmically scaled and an arrow on the x-axis indicates the mean of this distribution. During training, these two sets of 16x16 input neuron groups (ON and OFF cell responses) produce action potentials at times drawn from a Poisson process. Patches are randomly selected from the input dataset and presented for 50ms each during training.

4.3.5 Sparseness Measure

In the introduction, we described two commonly used measures of sparseness: lifetime and population sparseness. For our spiking neural network analyses, we simplify these measures to a single measure that we refer to this as the "average sparseness". Given that our spiking neural network model learning rules operate based upon a target firing rate, λ_{target}, we use this firing rate as a threshold to identify active and inactive neurons. In particular, the cases in which the firing rate of the neuron is greater than the target firing rate, λ_{target} are those during which

the stimulation arrives from potentiated synapses without inhibition cancelling the excitation. Therefore, a binarisation of neural responses with a threshold based upon the target firing rate is similar in effect to determining whether the input stimulus sufficiently matches the receptive field tuning of the neuron.

Stimuli are each presented to the network for 50ms and the firing rate response over this period is calculated and subsequently binarised to determine whether an individual neuron is active/inactive. The average sparseness of a network is measured as the mean number of active neurons in this network across a set of stimulus presentations.

Let us define the firing rate of a set of neurons, indexed $n = 1, 2, 3...N$, across a range of stimuli, indexed $s = 1, 2, 3...S$, as ${r_n}^s$. We binarise these responses, such that if the firing rate is greater than or equal to a threshold equivalent to the target firing rate, λ_{target}, the binarised response, R, is equal to 1, and is otherwise zero. This can be mathematically described

$$ {R_n}^s = H({r_n}^s - \lambda_{\text{target}}), $$

where $H(x)$ is the heaviside step function.

Given this calculation of the binarised neuron responses across neuron and stimulus, the lifetime sparseness of a given neuron, indexed by n, is

$$ \frac{1}{S} \sum_{s=1}^{S} {R_n}^s, $$

and the population sparseness for the presentation of a specific stimulus, indexed s, is,

$$ \frac{1}{N} \sum_{n=1}^{N} {R_n}^s. $$

If we then attempted to calculate the average lifetime sparseness across all

neurons, or average population sparseness across all stimuli, we arrive at a single equation describing sparseness:

$$\text{Average Sparseness} = \frac{1}{N}\frac{1}{S}\sum_{n=1}^{N}\sum_{s=1}^{S} {R_n}^s$$

This is the measure of sparseness which we use for all results which follow.

4.4 Results

In this chapter we investigate a single population of neurons which we train with both simple stimuli and with patches from natural images. The stimuli used for training are described in detail in Methods Section 4.3.4 above. This circuit approximates the early non-Dalean models of sparse coding [146, 231] in which a single population of competing neurons learn to produce V1 Simple Cell like receptive fields. In our interpretation, this network is a network of densely connected inhibitory neurons in which there is feed-forward excitation and dense lateral inhibition.

Every neuron laterally inhibits all other inhibitory neurons without any self-inhibition. This network structure is depicted in Figure 4.4, top.

4.4.1 Matching Network Dynamics to Theory with Simple Stimuli

In Methods Section 4.3.1 above, we produced a theoretical expectation of the relationship between the number of competing inhibitory cells and their activation probability, or sparseness, see Figures 4.1 and 5.2. We attempt to reproduce these relationships in spiking neural network models which are trained using the simple artificial stimulus set proposed in Section 4.3.4.1. All feedforward excitatory synapses are plastic with the eSTDP rule (equation 4.9) and lateral inhibitory connections are plastic with the iSTDP rule (equation 4.4). These plasticity rules

are described in Section 4.3.3.2.

A range of network sizes are trained, up to 15 competing neurons. Briefly, during training the input sources are all co-active each with a randomly chosen strength such that the total summed firing rate across all input sources equals 500Hz. Following training, the network is tested by activation of each input source (setting its firing rate to 500Hz) for one second with all plasticity rules off. The spike train response from this testing is then used to calculate the firing rates of the output neurons.

Other than varying the network size, the α_+ parameter of the eSTDP rule is varied. In particular, this parameter scales the excitatory weight set-points and directly scales the maximum weight set-point. Otherwise, the parameters used for these simulations are described in Table 4.1. We simulate these networks five times each using a different random seed. The random seeds affect the initial excitatory weights in the network and the presentation order of the input stimuli.

In order to compare this network to our earlier expectations, see Figure 4.1, we must calculate the average sparseness values of this network after training. Average sparseness is a measure which we base upon the number of neurons which have a firing response above the target firing rate, $\lambda_{\text{target}} = 5Hz$. This calculation and the reasoning for it are described in Section 4.3.5.

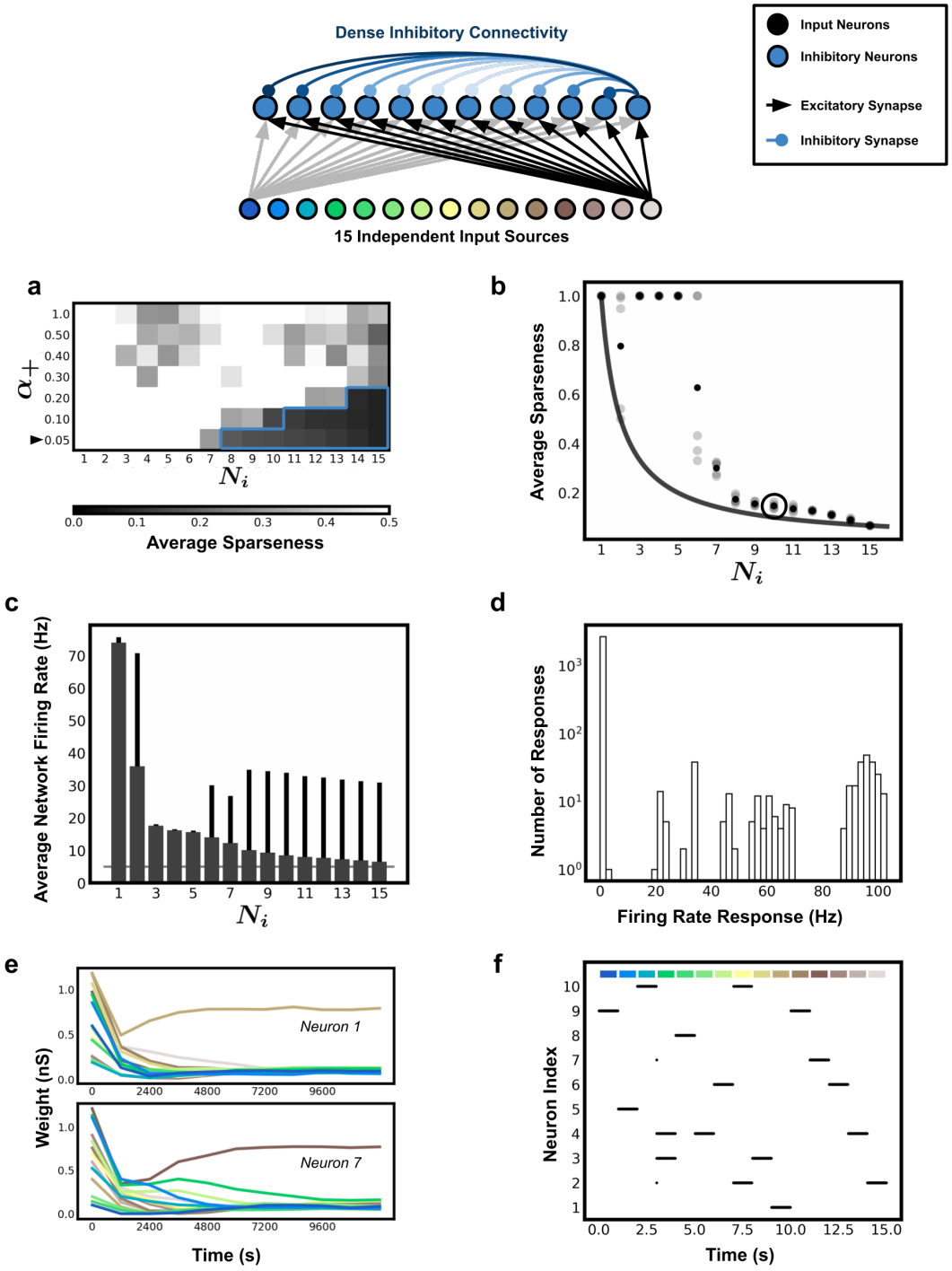

Figure 4.4: **The behaviour of a densely connected inhibitory LIF neuron circuit trained and tested with simple stimuli.** *Top,* the network structure simulated to produce the results shown. The network consists of inhibitory only neurons with dense lateral inhibitory connectivity. These cells are excited by a set of 15 independent input sources. These input sources are each represented by a single input neuron which is fully connected to all inhibitory neurons. The custom eSTDP rule is active upon all feedforward excitatory connections and the iSTDP

rule active on lateral inhibitory connections. **(a)** The average sparseness of the trained networks over a range of parameters. Network training is repeated for 5 different network random seeds. The greyscale colouring of this parameter map shows network sparseness averaged across these randomly seeded networks. The y-axis presents variation of a key parameter of the eSTDP rule (a_+), and the x-axis indicates change to the network size, N_i. A parameter range bounded by a blue line shows the set of networks within this parameter map which approximately match the theoretical expectation and produce a bimodal weight distribution through training. **(b)** Data from a row in the parameter map shown in (a), indicated by a black triangle $\alpha_+ = 0.05$, is shown plotted alongside the theoretical expectation, presented in Figure 4.1. The theoretical expectation (see Equation 4.1) is plotted as a line (black), and the simulation results are presented as grey points. A particular network size, circled in this plot, is used for the analysis in sub-plots d, e, and f. **(c)** The average firing rates of the networks in plot (b) are shown with error bars expressing the standard deviation in this firing rate across all stimuli (input sources). A grey line shows the target firing rate λ_target. **(d)** A histogram of all firing rate responses of a specific network (circled network in plot (b)). This network consists of 10 competing inhibitory neurons. **(e)** The evolution of input weights to neuron 1 (upper) and neuron 7 (lower) in the circled network of the plot (b). **(f)** A raster plot showing the spike times of all ten neurons in the selected network (circled in plot (b)). This raster plot shows the presentation of our 15 input stimulation sources, coloured bars top, which are activated independently and the neuron spike responses are shown below. For clarity, the plotted bars are coloured to correspond with the input source colouring in the network diagram top.

As is to be expected for a network of non-linear neurons, the network is capable of producing a range of behaviours under different parameter values. Our proposed probabilistic relation between the number of competing inhibitory neurons and the network average sparseness is achievable under a subset of these parameters. Figure 4.4a shows the network's average sparseness over a range of parameters. The average sparseness of the output neurons is calculated as the average number of input sources to which neurons in the network are above the target firing rate, λ_target. A significant portion of the parameter space shown in Figure 4.4a has extremely high levels of average sparseness, however a blue border encloses a subset of the parameter space in which the network dynamics match, or at least approach, the theoretical expectation. The parameter space in which network dynamics approach and match the theoretical expectation is characterised as having large network

sizes and small values of the potentiation constant α_+.

Figure 4.4b, compares a parameter slice of the parameter map (indicated by a black triangle, $\alpha_+ = 0.05$) shown in (a) to the absolute theoretical expectation developed earlier, see Figure 4.1. The solid line in this plot shows the predicted network sparseness, and grey points indicate the recorded network sparseness after training. As can be seen, the predicted and simulated network statistics are similar for relatively large network sizes, eight neurons or greater. For low network sizes, there is no discernible relationship between average sparseness and network size. This lack of consistency between network dynamics and theoretical expectation for small network sizes is proposed to be due to a lack of sufficient inhibition that would bring the network average firing rate to the target firing rate, λ_{target}.

The proposal that only networks which reach the target firing rate exhibit the expected sparseness is evidenced in Figure 4.4c where statistics of the firing rates of the networks shown in Figure 4.4b. The mean firing rates of the network from Figure 4.4b are shown plotted with error bars showing the standard deviation of these firing rates. As can be observed, those network which match the expected sparseness values also have firing rates distributions which have a mean firing rate value which is relatively close to the target firing rate. For networks which do not match the expected dynamics, their average firing rate responses far exceed the target firing rate. In particular, with a low number of inhibitory neurons these networks are not brought to the homeostatic firing rate.

Figure 4.4d shows a histogram of the output neuron firing rate responses of a particular network (circled in Figure 4.4b). The chosen network is one which is within the region of parameters which appear to match the predicted sparseness. Post-training the network exhibits responses in which "active" and "inactive" cells are distinguishable as being at zero or greater than zero firing rates respectively – a bimodal distribution. For ecological stimuli, neural responses generally vary continuously, however during testing our input sources are activated independently

and therefore produce a bi-modal neuron response. This also makes our binarisation of neural responses for sparseness calculations reasonable. Figure 4.4e shows the evolution of input weights to two neurons, indexed neuron one and seven, from the selected network. Each weight is plotted as a separate line with the colour corresponding to the particular pre-synaptic input source, see network diagram Figure 4.4 top. As can be seen, the weight values quickly stabilize and converge such that each neuron in the network has a well defined preferred input source. The selected network consists of ten neurons in total and the responses of all of these neurons to the range of input sources are shown in Figure 4.4f. The network's average sparseness is 0.1, and, as expected, these neurons are each responsive to approximately 10% of the input stimuli.

The results described above and shown in Figure 4.4 show an agreement between the trained spiking neural network behaviour and the theoretical expectations which were presented in Methods Section 4.3.1 for a subset of parameters. However, we have also proposed here that the subset of parameters which do not match our expectation have firing rates which far exceed the homeostatic firing rate and this is a potential reason for the lack of agreement. We therefore continue this investigation by attempting to bring small network sizes to the homeostatic firing rate.

4.4.1.1 Bringing the Inhibitory Neuron Networks to Homeostatic Firing Rates

The spiking neural networks described above do not reach the homeostatic firing rate, $\lambda_{\text{target}} = 5Hz$ for networks of small sizes. Instead, as can be seen in Figure 4.4c, the average firing rates of the networks consistently far exceed λ_{target} for these networks. These same small network sizes are furthermore unable to reproduce the expected network sparseness values.

The inability for the networks shown here to reach balance is primarily due to an insufficient level of inhibition. For sufficiently large network sizes and for appropriate

choices of α_+, the amount of lateral inhibition is sufficient to bring trained networks close to balance. To investigate the effect of homeostasis in the smaller network sizes, we introduce an additional source of inhibition. This additional source of inhibition must also act to bring the network to the same homeostatic state, otherwise it would be unhelpful in testing our proposal. Therefore, we assume that this background inhibition arrives from neurons which have the inhibitory plasticity rule active upon their efferent synapses with the same target firing rate. Thus, the contribution of these background inhibitory neurons is to scale the average network activity down to the homeostatic firing rate. Crucially, these neurons have no particular tuning with respect to the input sources and therefore do not directly produce competition in these networks. The biological basis of such additional inhibitory neurons would be the existence of inhibitory synaptic connections which project from neurons which do not share the same input sources and whose activity is completely uncorrelated with the neurons in our circuit.

In simulation, this background inhibition is produced by creating a population of 10 external inhibitory neurons which fire with spike times sampled from a Poisson process at a 100Hz mean firing rate. This firing rate is consistent across all stimuli and these "background" inhibitory neurons project inhibitory synaptic connections to all neurons in our network. These inhibitory connections are modified during learning by iSTDP. The purpose of these background inhibitory neurons, with iSTDP modified connection strengths, is to bring the network firing rates to homeostasis under all conditions; even for networks with very small numbers of neurons.

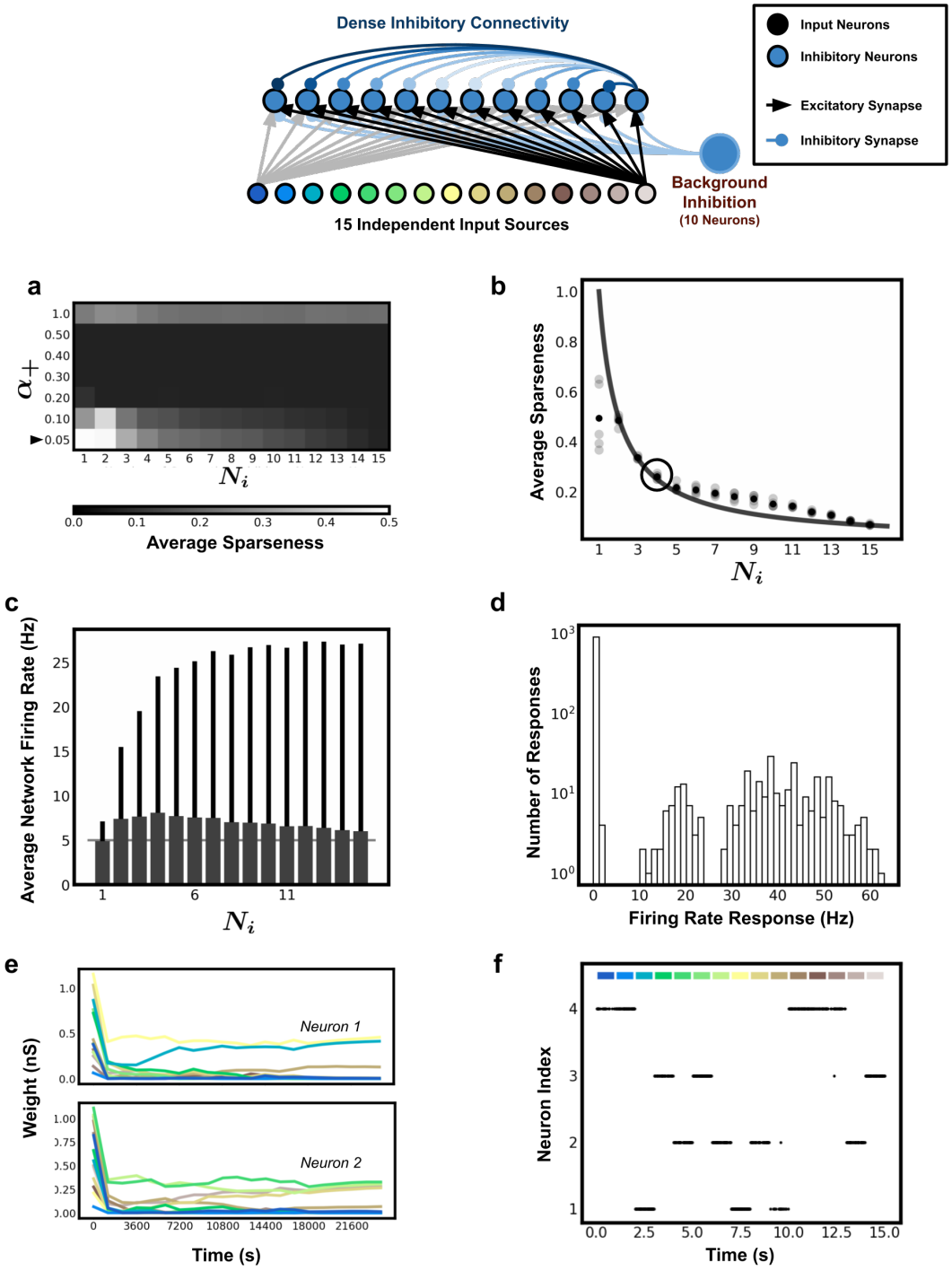

Figure 4.5: **Bringing densely connected inhibitory LIF circuits to the homeostatic firing rate through background inhibition and reproducing theoretical expectation.** *Top,* the network structure simulated to produce the results shown. The network consists of inhibitory only neurons with dense lateral inhibitory connectivity. These cells are excited by a set of 15 independent input sources. These input sources are fully connected to all inhibitory neurons. The custom eSTDP rule is active upon all feedforward excitatory connections and

the iSTDP rule active on lateral inhibitory connections. An external group of inhibitory neurons provide background inhibition. These neurons fire consistently at 100Hz firing rate and the iSTDP rule is active upon their efferent inhibitory synaptic connections. **(a)** The average sparseness of trained networks over a range of parameters. Network training is repeated for 5 different network random seeds. The greyscale colouring of this parameter map indicates average network sparseness. The y-axis presents variation of a key parameter of the eSTDP rule ,a_+, and the x-axis indicates change to the network size, N_i. **(b)** Data from a row in the parameter map – indicated by a black arrow in plot (a), $\alpha_+ = 0.05$ – is shown plotted alongside the theoretical expectation, presented earlier in Figure 4.1. The theoretical expectation (see Equation 4.1) is plot as a line (black), and the simulation results are presented as black points. **(c)** The average firing rates of the networks in plot *b* are shown with error bars showing the standard deviation in this firing rate. A grey horizontal line shows the location of the target firing rate λ_target, 5Hz. **(d)** A histogram of all firing rate responses of a specific network (circled network in plot (b)). This network consists of 4 competing inhibitory neurons. **(e)** The weight evolution of input weights to neurons 1 (upper) and neuron 4 (lower) in the circled network of plot (b). **(f)** A raster plot showing the spike times of all four neurons in the selected network (circled in plot (b)). This raster plot shows the presentation of our 15 input stimulation sources, coloured bars top, which are activated independently and the neuron responses are shown below. For clarity, the plotted bars are coloured to correspond with the input source colouring in the network diagram top.

With the addition of background inhibition, networks of all scales achieve the homeostatic firing rate ($\lambda_\text{target} = 5Hz$) for a range of parameters, as shown in Figure 4.5c.

Figure 4.5 is organised in an equivalent fashion to Figure 4.4. As mentioned, Figure 4.5c shows the network's ability to achieve firing rate homeostasis at 5Hz after the introduction of our background inhibition and excitation. Comparing Figure 4.4b to Figure 4.5b, we observe that the network can achieve a similar sparseness relation at large network sizes but also achieves close to the expected sparseness levels at networks of smaller sizes.

We also observe, see Figure 4.5a, that there appears a particular set of parameter α_+ for which the network is in the desired state. For larger potentation constants $\alpha_+ > 0.2$, the network sparseness is very low. This low sparseness is due to the stabilization of the network in a state where a single input stimulus causes excessively

large firing rate responses. In order to maintain the homeostatic average firing rate, each individual neurons responses are scaled down by inhibition to such a signficant degree that individual neurons are only active to a single stimulus and are completely inactive to every other stimulus. This results in the sparseness across the parameter space of large α_+ values being equal to 1/15, i.e. active to a single input source only. This property reveals that the system of learning rules requires tuning of this α_+ variable, though no other parameters needed adjustment in this setup.

These results were produced in networks with input sources represented by a single neuron. This allowed a simple visualization and explanation of the network dynamics. To show that this result generalizes to networks in which the firing rates are more reasonable and networks in which input sources are represented by multiple neurons, we re-produce the results described for Figure 4.5 in Appendix Section B.0.1, Figure B.1. This gives reassurances in the generality of the results shown here. Furthermore, the sections which follow present results from a stimulus set which is ecological, includes hundreds of input neurons, and in which the firing rates of input cells are reasonable during training and testing.

4.4.1.2 Reproducing V1 Simple Cell like Receptive Fields in a Dense Inhibitory-Only Neuron Network

Having trained networks upon a dataset of simple artificial stimuli, the question remains as to whether such a network is capable of learning from an ecological dataset. Existing sparse coding models have been demonstrated to be capable of producing Gabor filter-like receptive fields, which appear similar to early visual cortical receptive fields. These models often implement a single pool of competing neurons without considering separate inhibitory and excitatory neuron contributions. We model a similar circuit setup using our inhibitory neuron circuit. This network structure is equivalent to that which we investigated with simple stimuli in Section 4.4.1.1, however the input sources now represent the intensity values of individual

pixels from processed image patches extracted from a natural image dataset. The complete network structure is depicted in Figure 4.6a.

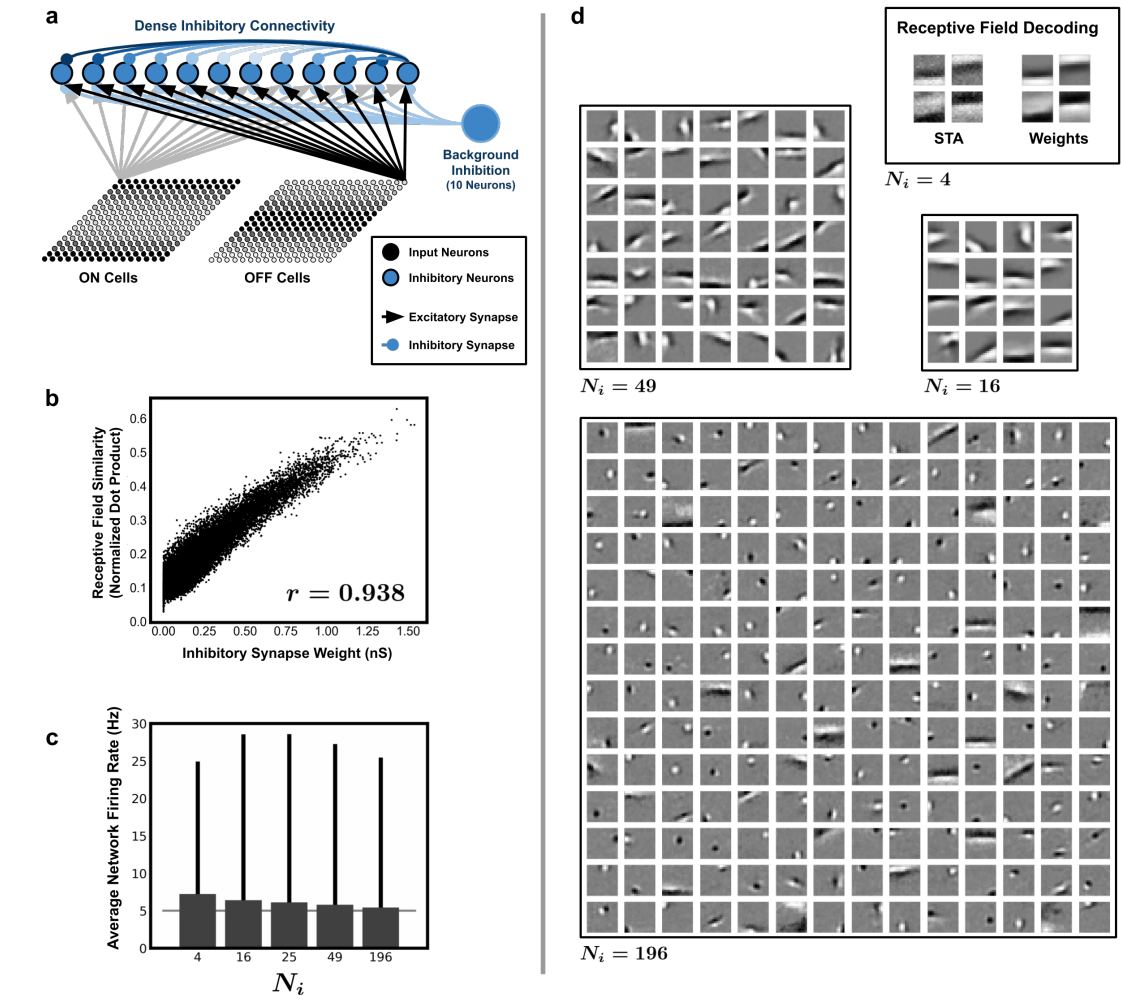

Figure 4.6: **Results of training dense competitive networks of inhibitory neurons with patches from natural images. (a)** A depiction of the network structure. Input is provided by two 16x16 input sources which represent the ON and OFF cell responses of retinal ganglion cells to a single 16x16 image patch (see Methods Section 4.3.4.2). These input populations provide excitatory input to a layer of densely connected inhibitory neurons. Training consists of the presentation of randomly selected patches from the input dataset for 50ms while the eSTDP learning rule is active on the feedforward excitatory synaptic connections. The iSTDP plasticity rule is active on lateral inhibitory connections and synaptic connections from a population of background inhibitory neurons with a constant 100Hz firing rate. Training consists of the presentation of approximately $2 \cdot 10^6$ image patches. **(b)** The trained networks have a strong positive correlation (r=0.938) between the inhibitory synaptic connection weights and the receptive field similarities

of the pre and post-synaptic neurons of these inhibitory synaptic connections. The receptive field similarity is calculated for every inhibitory synaptic connection by taking the dot product of the normalized excitatory input weight vectors of the pre and post-synaptic neurons. This value is then plot against the synaptic weight of that inhibitory synaptic connection. Given that the input excitatory weights cannot be smaller than zero, this dot product is between 0.0 and 1.0. **(c)** The mean firing rate as calculated by presentation of 1000 random patches from the training set is plotted (with error bars showing the standard deviation) for our range of network sizes. A gray horizontal line shows the target firing rate, λ_{target}, used for the learning rules. **(d)** The converged receptive field tunings of neurons in competitive networks of a range of sizes. The inset box shows the similarity of decoded receptive fields when they are decoded by their input weight vector versus use of Spike-Triggered Averaging (STA), see Appendices B.0.3 and B.0.4. Receptive field are shown using weight vector decoding in a grid with arbitrary placement. Network sizes are presented below the receptive field grids, $N_{\text{i}} = 4, 16, 49, 196$.

We train this inhibitory circuit with 16x16 pixel patches extracted from whitened natural images. Image patches, selected randomly from the input dataset, are pre-processing and then presented one at a time. The pre-processing of these image patches is described in the Methods Section 4.3.4 above, and the input to our network model is a group of ON and OFF, retinal ganglion cell-like, input neuron groups. The separate ON and OFF input neuron responses provide excitation to the network and each patch is presented for 50ms before the next randomly chosen patch. Plasticity rules modify the synaptic weight structures such that inhibitory lateral connections are modified using the iSTDP rule and excitatory feed-forward synaptic connections modified according to the eSTDP, both of which are described in Methods Section 4.3.3.2.

We train a range of network sizes in order to identify the effect of increasing competition upon the network activity. Figure 4.6d, shows the neuron receptive field tunings of trained networks of a range of sizes, from 4 to 196 competing inhibitory neurons. These receptive field tunings are shown relative to the whitened image patch pixels. We visualise the receptive fields of neurons by combining their weight vectors from the ON and OFF cell populations, subtracting the OFF cell weight

vector from the ON cell weight vector, and shading the resulting 16x16 weight vector with white indicating positive and black indicating negative values. This process abstracts away our ON and OFF cell populations and instead provides receptive fields that are relative to the original whitened image patches. These receptive field tunings are based upon the input weight structures, however carrying out spike triggered average (STA) analysis produces highly similar receptive fields, shown for a single example ($N_\mathrm{i} = 4$) in Figure 4.6 right (inset). Appendix Section B.0.3 describes this STA analysis process and Section B.0.4 provides STA decoded receptive fields for all of the neurons presented in Figure 4.6d.

Figure 4.6d, qualitatively shows a trend: increasing the number of competing neurons results in a reduction in the average neuron receptive field size; greater competition pushing neurons to smaller receptive field sizes. One other observation to be noted is that the receptive field tunings show bars oriented primarily in the horizontal direction. This betrays a bias in the input dataset for textures oriented in the horizontal direction and could be combatted by training with patches which are also randomly rotated, though for these studies we only extracted patches from the dataset of Olshausen and Fields [146] without any modification.

As described by Zylberberg et al. [231] and as expected based upon literature describing the decorrelative properties of inhibitory neurons [189], we find a positive correlation between the similarity of neuron receptive fields and the weights of the inhibitory connections between them, Figure 4.6b. Each inhibitory synaptic connection in the network is represented in Figure 4.6b by a single point. The similarity of neuron receptive fields is calculated for a given synaptic connection by taking the dot product of the normalized input weight vectors its pre and post-synaptic neurons. This is plot against the weight of the inhibitory synaptic connection. We find a strong positive correlation ($r = 0.938$) between the measure of similarity of pre and post-synaptic receptive fields and the converged inhibitory synaptic weight. This correlation reflects the iSTDP learning rule,

which has correlative inhibitory potentiation, and also reflects the degrees of receptive field overlap.

Figure 4.6c shows the average network firing rate having approached the target firing rate after training. These firing rate statistics are based upon the presentation of 1000 random patches from the training dataset for one second each. This confirms the impact of the inhibitory plasticity rule upon the network firing rate when trained upon this ecological stimulus set.

The receptive fields produced, and shown in Figure 4.6d, provide evidence of the network's ability to produce Gabor-like receptive field tunings similar to those observed in primary visual cortex. However, sparse coding models are often described as carrying out a form of information compression which should allow reconstruction of an input stimulus from the network activity. Thus far we have not discussed the ability of our network to accomplish such information compression, nor is it an explicit component of our learning paradigm. We test the ability of our network to reconstruct an input image following training in order to check for information preservation. A whitened and processed image from outside of the training set was split into 16x16 patches and presented to the 196 inhibitory neuron network, receptive field tunings shown in Figure 4.6 far right, after training. All plasticity rules were turned off for this presentation process. The reconstruction process used here is equivalent to that employed by Zylberberg et al. [231]. Input patches are reconstructed by multiplication of the neuron firing rate response by their receptive field tuning (shown in Figure 4.6d $N_\mathrm{i} = 196$), and summation of this response weighted tuning over all neurons. This produces a reconstructed patch which can be rescaled to the mean and variance of the original input patch before being stitched together for full image reconstruction. Figure 4.7a shows the original and reconstructed images. This reconstruction qualitatively matches the input image well, providing evidence of the information preserving property of this network arrangement despite this not being an explicit consideration during our production

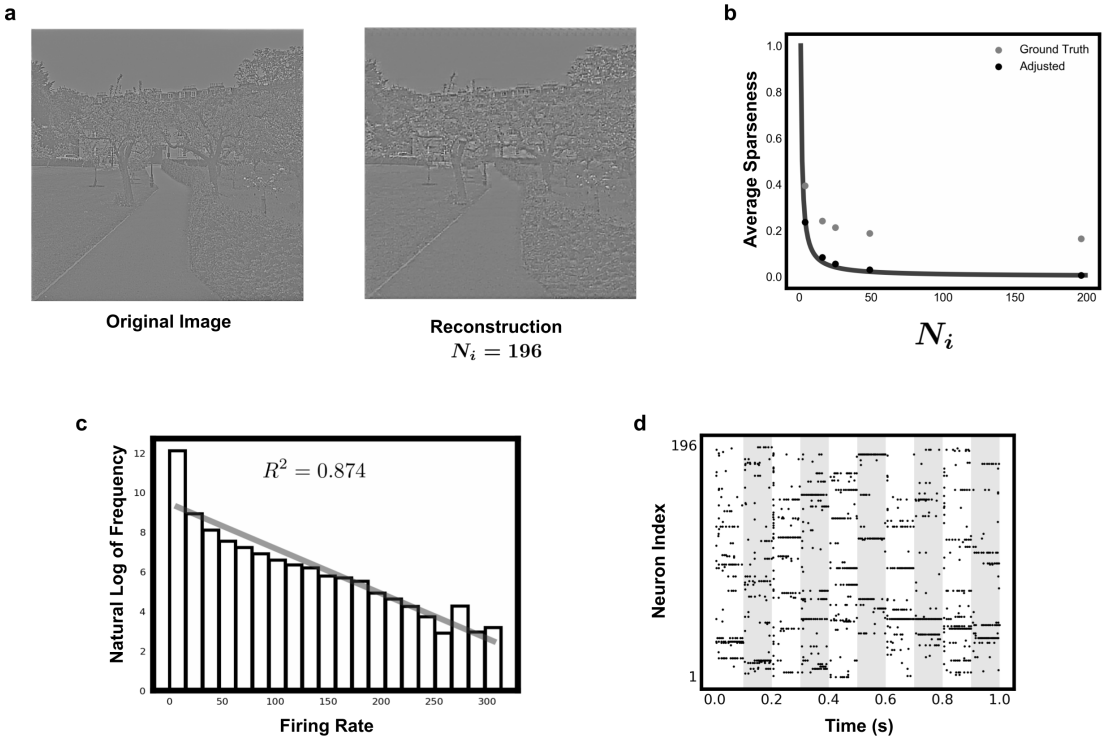

Figure 4.7: **Testing the response statistics of the trained 196 inhibitory neuron network and its ability to reconstruct an input image. (a)** A test image, left, is run through the trained 196 inhibitory neuron network (receptive fields shown in Figure 4.6d, $N_{\mathrm{i}} = 196$) and the neuron responses multiplied the receptive fields are used to re-construct each individual patch. The reconstructed patches are rescaled to the mean and variance of the original patch before being stitched together in order to form the full image reconstruction. **(b)** The network firing rate responses distribution for all patches of the original test image, shown in (a). Gray points show the network sparseness calculated for the networks whose receptive fields are shown in Figure 4.6d. An adjustment of these values is shown using the black points. To produce this adjustment, all of the gray points are shifted by removing a single value. This value is the difference between the expected and observed value of the network sparseness for the network of 196 neurons. **(c)** A log-scaled histogram of the firing rates of neurons in the 196 neuron network used to produce the reconstruction in (a). This histogram is produced by binning the firing rates into 20 bins spaced equally between the minimum and maximum firing rate response of the network. This logarithmic transform is well explained by a linear fit, showing the suitability of an exponential fit to our network response distribution. **(d)** A raster plot of the trained network spiking responses to ten randomly selected patches from the training set. Each patch is presented for 100ms before the network is reset and another patch is presented. Alternating grey and white backgrounds of width 100ms are shown on this plot to indicate the areas within which a single patch is presented. The y-axis indexes the 196 neurons of the network, and the x-axis represents time.

of the learning rules. The imperfections in the reconstruction can be at least partially attributed to the compression carried out by the network, with 196 neurons being used to reconstruct patches of size 256 pixels – a compression ratio of 0.765.

Thus far we have not commented on how well this network trained with visual stimuli meets our expected sparseness versus network size predictions from Section 4.3.1. Figure 4.7b, shows in black the theoretical expectation which we developed for an inhibitory neuron circuit. Grey points on this plot show how the average sparseness of the network varies with N_i when it is exposed to the set of patches which were presented to carry out reconstruction for Figure 4.7a. As can be seen, these average sparseness values deviate significantly from the expectation, however they appear to primarily deviate through a positive offset.

We attribute this to the fact that our visual image patch dataset contains patches with a range of mean input strengths. If our network is tuned to a particular input patch strength, then any patches which arrive with a higher strength elicit a greater response, and patches of lower strength elicit a lesser response. We would expect this to average out, however our network cannot produce a lower than zero response and we attribute this bias to that fact – there is a minimum response which the network can produce. In order to account for this, we remove a constant value from all of the sparsity values to produce the "Adjusted" points shown in black. This constant is simply the calculated sparseness value for the network of 196 neurons, minus the predicted value for 196 neurons. As can be seen, despite using a constant offset adjustment value based upon the 196 neuron network, this adjustment brings all of our network sizes into agreement with the theoretical proposal.

Another property of early visual cortex responses which we hoped to compare is that the observed firing rate responses follow an exponential distribution [210, 7]. As shown in Figure 4.7c, the network responses of our inhibitory circuit produce firing rate responses that also follow an exponential distribution. We fit this distribution through linear regression by taking the logarithm of the histogram frequencies. This

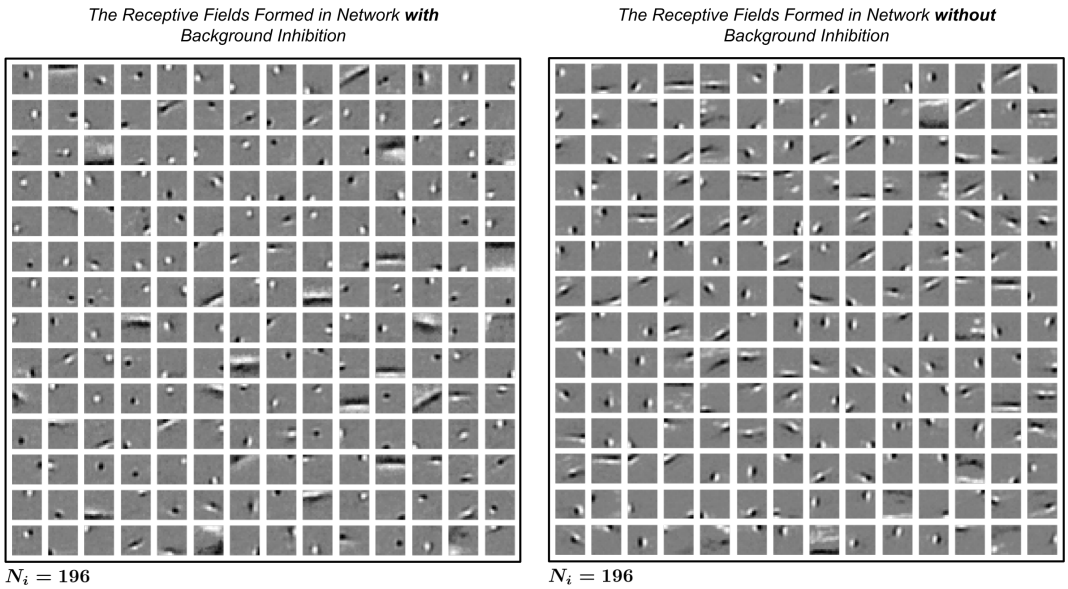

Figure 4.8: **Comparisons of receptive field tunings produced with and without background inhibition in a 196 neuron network.** A comparison of the receptive fields produced in the same networks described in Section 4.4.1.2, with and without background inhibition. Left, receptive fields formed in a network with background inhibition (equivalent to result shown in Figure 4.6d, $N_\mathrm{i} = 196$). Right, the receptive fields formed in the same network structure but without any background inhibitory neurons.

linear fit explains 87.4% of the variance of the logarithmic transform.

Thus, we have reproduced a range of the response properties of both modelled sparse networks and cortical circuits. In order to provide an example of how the network responds to individual patches, Figure 4.7d shows the spiking activity of the trained network of 196 neurons when it is presented a set of ten randomly selected patches from the training set. These patches are each presented for 100ms. As can be observed, there is some random background activity in the network and crucially individual neurons have a strong preference for some subset of the image patches.

4.4.1.3 Alternative Networks and Plasticity Rules

Having successfully trained an inhibitory neuron circuit upon a dataset of natural images, we next considered some alternative network structures and learning rules

in order to show how they compare. First, we indicated in Section 4.4.1.1 that the presence of background inhibitory neurons was primarily a mechanism which brings relatively small inhibitory neuron networks to balance. In that section we also showed that, even without background inhibition, networks of a sufficiently large size approached the desired homeostatic firing rate and exhibited the expected level of competition. For smaller network sizes, background inhibitory neurons were required to bring the system to homeostasis and to approach the expected level of competition.

Here we remove the background inhibitory neuron population from our models trained with patches from natural images. Figure 4.8 shows a comparison between the receptive fields formed in a network with and without background inhibition. Left is shows the receptive fields of neurons from a 196 neuron circuit with background inhibition after training. This is the same network presented in Figure 4.6d, for $N_{\mathrm{i}} = 196$. Right we show the receptive fields produced through training in the same network setup but without background inhibitory neurons. As can be observed, the receptive fields formed through training are oriented bar-like stimuli, which are localised – as expected for a sparse coding model of V1 Simple Cell. Furthermore, the receptive fields produced in these two cases are qualitatively very similar despite the removal of background inhibitory neurons. The ability for the network without background inhibition to produce selective receptive fields and their similarity to a network with background inhibition shows that indeed for large network sizes the background inhibition is an optional component.

Figure 4.9 shows a similar comparison for a network of intermediate size, $N_{\mathrm{i}} = 100$, in which the network is brought close to the homeostatic firing rate. However, in this case we can observe significant differences in receptive field sizes. We do not provide a quantitative analysis of this difference, however we can attribute any difference to the lack of background inhibition since all other parameters are equivalent. In this case, it can be observed that the receptive fields formed in the network without background inhibition are significantly larger in scale. We

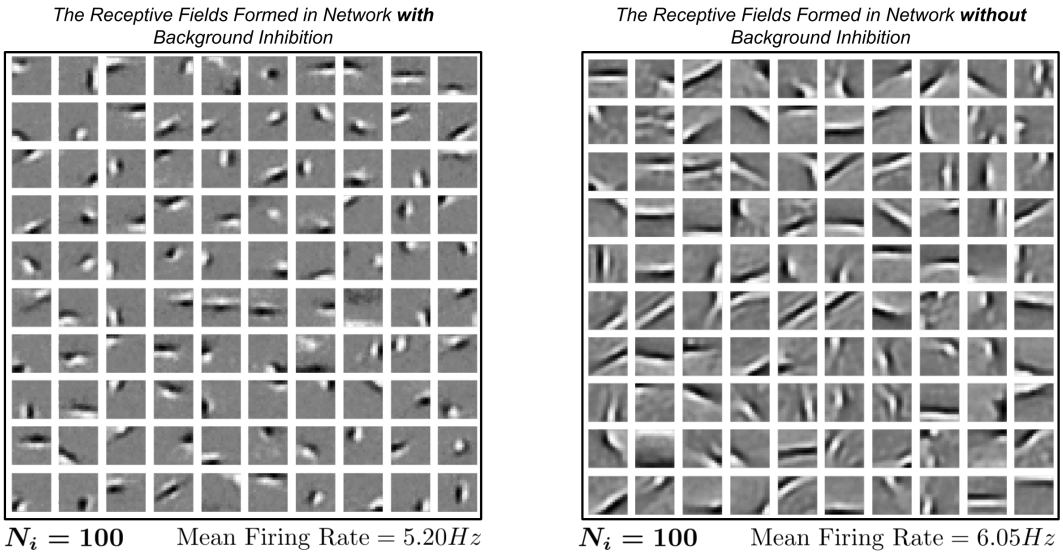

Figure 4.9: **Comparisons of receptive field tunings produced with and without background inhibition in a 100 neuron network.** A comparison of the receptive fields produced in networks trained as described in Section 4.4.1.2, with and without background inhibition. Left, receptive fields formed in a network with background inhibition. Right, the receptive fields formed in the same network structure but without any background inhibitory neurons.

attribute this to the firing rate being greater than the homeostatic firing rate on average. This indicates that the level of inhibition provided by the "within" network inhibition is insufficient to bring the network to homeostasis. As explored in Section 4.4.1.1, this can be attributed to the small number of neurons within the network.

Given the insufficient level of inhibition, we can attribute the larger receptive fields to a lack of competition between neurons. In particular, as described in Section 4.3.3.2, our learning rules are biased towards greater weight potentiation for higher post-synaptic neuron firing rates. Given that the network with higher mean firing rate has greater receptive field sizes, our learning rule property explains this phenomenon.

Finally, it is important to compare the performance of our custom excitatory plasticity rule against a tradition excitatory plasticity rule. Our custom excitatory plasticity rule, which we combine with inhibitory plasticity, was proposed as a

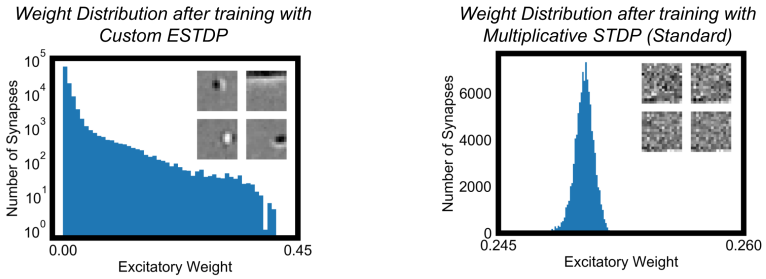

Figure 4.10: **Comparisons of the excitatory input weight distributions formed through training under the modified and unmodified excitatory plasticity rules.** Left, the excitatory input weight distribution formed after training the network whose receptive fields are shown in Figure 4.8, left. This network consists of 196 neurons with lateral inhibitory connections and with background inhibition. Right, the excitatory input weight distribution formed through training the same network described for the left plot but with replacement of the proposed custom eSTDP rule with the original multiplicative rule described by Van Rossum et al. [199] and given in equation 4.6. Four randomly selected receptive fields from each of these trained networks are shown inset.

modification to an existing multiplicative plasticity rule [199]. Figure 4.10 shows a comparison of the weight distributions produced when training with the custom excitatory plasticity rule vs the multiplicative rule which was proposed by Van Rossum et al. [199], both described in detail in Section 4.3.3.2. The networks used in these cases for training were networks of 196 inhibitory neurons with background inhibition and iSTDP active. As can be observed, our custom plasticity rule produces a weight distribution which (crucially) allows a subset of weights to reach zero weight. A peak at zero weight enables the neural selectivity and receptive fields which we observe. By comparison, the standard multiplicative rule modifies synaptic weights such that they all approach the weight set-point and produce a single weight peak at a positive weight value. This is the expected theoretical behaviour, as described by equation 4.8. Four randomly selected examples of the produced receptive fields are shown inset for each weight distribution. The weight distribution of the network trained with the standard multiplicative excitatory plasticity rule remains within a narrow distribution close to the expected weight

set point if we allow training to continue indefinitely.

The receptive fields produced are not distinct or oriented. Instead, they have a structure which approximately reflects the mean patch structure across all stimuli, with some additional noise. The similarity of these receptive fields indicates that this unmodified excitatory plasticity rule is unaffected by competition in our network setup.

4.5 Discussion

Methods for producing reliable and predictable competition in networks of spiking neurons have largely relied upon biologically unrealistic methods. We provide evidence in this chapter for the efficacy of complementary excitatory and inhibitory plasticity rules for the production of both competition and excitatory-inhibitory balance. In particular, these excitatory and inhibitory STDP rules are specially designed to produce neuron selectivity whilst regulating the average neuron firing rates which emerge in a network. The efficacy of this approach is presented in circuits of densely connected inhibitory neurons. Such circuits impose strong competition, through lateral inhibitory synaptic connections, between neuron receptive field tunings that develop during training.

A little discussed component of this learning rule arrangement is the relative speed of excitatory and inhibitory learning. In order to ensure than homeostasis is maintained despite changes to the excitatory receptive field, the inhibitory learning rate is an order of magnitude greater than that of the excitatory learning rule. The requirement of such a "fast" homeostatic mechanism has been discussed at length in neural network modelling literature [102, 222, 224]. Experimental observations are at odds with such an approach since homeostatic mechanisms are prevalent on a timescale much slower than excitatory learning. It has been proposed that homeostatic mechanisms operate on a slow timescale and separate mechanisms,

referred to as rapid compensatory processes, allow fast adaptation to compensate for the effect of Hebbian learning. In this study we have circumvented this issue by allowing our homeostatic components to adapt at a timescale much faster than the Hebbian excitatory learning. Nonetheless, future studies aught to attempt an explanation of how a rapid compensatory process could operate to stabilise learning in these networks without affecting the competitive learning mechanisms.

This approach appears very successful in producing competition between neurons and learning based upon this competitive interaction. However, one drawback is the lack of a direct determination of the implicit objective function which is being optimised by our combined learning rules. It appears from the results that the modelled networks produce strong inhibitory competition while excitatory receptive fields grow to be as large as possible without inducing significant inhibition. Thus, the excitatory receptive fields form to fill "holes" in the tuning of incident inhibition. This proposal is in some ways similar to that of some supervised sparse coding models which have been proposed, such as Local Competitive Algorithms [170].

An alternative intuitive understanding of this learning arrangement is by considering that learning occurs based upon a fixed threshold, the target firing rate λ_{target}. The inhibitory plasticity rule potentiates inhibitory synaptic connections such that neural responses "slide" toward this fixed threshold. This ultimately brings their average firing rate to this fixed threshold. Simultaneously, the excitatory plasticity rule potentiates excitatory synaptic connections when a give neuron produces a firing rate response which is above this threshold firing rate. This intuitive description is qualitatively very similar to the proposal put forth by the BCM rule [20]. We also provide a simple approximation of the combined effect of our excitatory and inhibitory learning rules in Section 4.3.3.2 and show that it exhibits a number of similarities when compared to the BCM rule. One key difference is that the BCM employs a sliding threshold based upon a running average of a neurons activity whereas this rule instead employs a fixed threshold to which

firing rates are "slid". Despite the qualitative similarity of the BCM rule and our proposed learning rule setup, we can not draw a direct theoretical equivalence between them and we have not given a rigorous mathematical description of the impact of excitatory and inhibitory learning rule interactions.

This limitation in our ability to produce a complete theoretical description of the emergent properties of our learning rules is in part due to the complexity of the system which we are considering. We have a system with spiking dynamics and a pair of learning rules which affect the development of excitatory and inhibitory synaptic weights in an indirect but interacting fashion. Ultimately, we hope that future theoretical studies might provide techniques by which we can rigorously describe the dynamics of these learning systems. Though, on balance the neural system which we described in Chapter 2 similarly does not have a rigorous mathematical description of its emergent properties and has nonetheless been applied successfully to a number of research questions on the formation of receptive fields in ventral and dorsal visual cortex [166, 64].

Aside from theoretical limitations, a number of features of this study could be extended into future work. In this study, we used homeostatic firing rates of 5Hz. This value was motivated by the use of similar values in earlier modelling literature and the fact that the timescale of learning is faster with higher firing rates allowing learning on a timescale which is feasible to simulate. However, electrophysiological evidence suggests that the average firing rate of neurons in the brain are on average below 1Hz [128]. Furthermore, electrophysiological recording studies find individual neurons each have a different homeostatic firing rate such that there exists a distribution of target firing rates [82]. In this work we abstract this distribution and all neurons have an equal homeostatic firing rate, however such a distribution could be produced by providing every neuron a unique homeostatic firing rate target. For this study we determined that adding such complexity would not qualitatively change the results but instead only introduce greater complexity.

Another feature of this network which seems unreasonable is the lack of redundancy. Individual neurons form highly orthogonal receptive fields and in reality this means a very low level of redundancy. In effect this is due to the strong competition induced by the inhibitory plasticity rule which leads to a local encoding of features. One method to introduce redundancy would be to reduce the strength of lateral inhibition. This could be achieved by modification to the inhibitory plasticity rule, or by dilution of lateral inhibitory connectivity. However, such modifications would require significant exploration to ensure stability in learning and to investigate the resulting network properties.

5

Extending Competitive Learning in Spiking Neural Networks to Populations of Excitatory and Inhibitory Neurons

5.1 Introduction

In Chapter 4, we presented a inhibitory neuron circuit which combined a novel excitatory plasticity rule with a well established inhibitory plasticity rule to produce reliable competition and learning. This competing system of neurons was shown to produce V1 simple cell-like receptive fields when trained with patches from a dataset of natural images. The combined effect of the excitatory and an inhibitory learning rules operated via the application of a threshold firing rate, could produce both

tuned neuron receptive fields (and associated selective responses) whilst maintaining a homeostatic average firing rate. In the introduction to Chapter 4 we described existing work in sparse coding and balance which provided the motivation and intuition for our learning rule setup. We do not repeat this introduction and instead direct readers to Section 4.1 of Chapter 4.

However, the use of inhibitory only neurons in the previous Chapter did not provide an explanation of the expected role and impact of the inclusion of excitatory neurons. In particular, excitatory neurons are the primary drivers of projection and propagation of activity within the brain. With the inhibitory circuit proposed in Chapter 4, projection of excitatory activity is not possible without breaking Dale's law, the separation of excitatory and inhibitory neurons. Furthermore, there are experimentally observed differences between tuning widths of excitatory and inhibitory neurons. Excitatory neurons have been observed to have smaller receptive field sizes than their neighbouring inhibitory inter-neurons in a range of cortical areas [46, 212, 162]. This broader tuning of inhibitory neurons has further been implicated in the sharpening of excitatory neuron receptive fields [212], and without identifying specific roles for excitatory and inhibitory neurons in network models, we cannot reproduce such network behaviours.

King et al. [105] produced a model of the emergence of V1 simple cell-like receptive fields in a spiking neural network structure which included excitatory and inhibitory spiking neurons. There were a number of drawbacks to this study, including the use of rate based learning of input excitatory synaptic weights and non-Dalean input synapses which switched between hyperpolarized and depolarised the cell depending upon the particular input stimulus. Nonetheless, they were able to produce networks which reproduced the relative "broadness" of inhibitory neurons when compared to excitatory neurons. Zhu et al. [226] accomplished a similar feat in rate-coded neural network models where they provided a theoretical grounding for the separation of excitatory and inhibitory neurons in a traditional

sparse coding network. Based upon this, they could then produce rate coded models in which excitatory and inhibitory neurons consisted of separate populations.

The network structures implemented in both of these studies included inhibitory neurons which do not receive excitatory input from the same excitatory input source as the excitatory neurons of the network. Instead excitatory neurons receive input from some source (sensory input) and project to inhibitory neurons which provide both feedback and lateral inhibition. Such a setup ensures that inhibitory neuron activity is driven directly by the excitatory neurons. This is the mechanism by which King et al. [105] and Zhu et al. [226] produced Dalean sparse coding models and we adopt the same arrangement. Though it should be noted that this arrangement is computationally motivated in these studies rather than being motivated by biological detail.

We can also consider non-Dalean models to have implicitly made similar assumptions about the roles of inhibitory neurons. In particular, if we consider the rate coded model described in Chapter 2, the competitive mechanism ensured that only some percent of output neurons were co-active. This implicitly assumed that the function of lateral inhibition was to ensure that competition produced between excitatory neurons was based upon the excitatory neuron activity. A similar implicit assumption is made by models in which the output neurons are non-Dalean and act to laterally inhibit their neighbours [58, 231, 151].

We produce a similar network structure in this chapter and replicate the ability for a network of interacting excitatory and inhibitory neurons to produce V1 simple cell-like receptive fields in a network of both excitatory and inhibitory neurons through all spiking dynamics and STDP based rules. This investigation is preliminary and we identify and address some challenges before discussing the open future research questions.

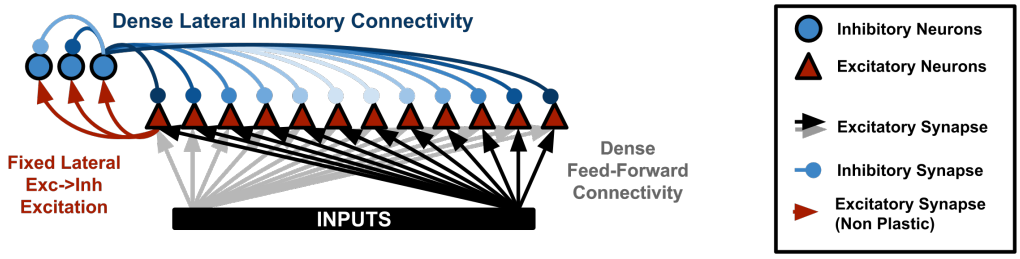

Figure 5.1: **The investigated excitatory and inhibitory neuron network structure.** Inhibitory and excitatory neurons are shown by blue circles and red triangles respectively. The number of inhibitory neurons in the circuit is N_{i}, and the number of excitatory neurons, N_{e}. Blue connections among the layer of inhibitory neurons and from inhibitory to excitatory neurons indicate the dense lateral inhibitory connectivity. All inhibitory neurons form synapses targeting all other inhibitory neurons and all excitatory neurons. Black/grey arrows show excitatory connections which arrive from an input source to excite the network's excitatory neurons. These excitatory synaptic connections from input sources are plastic. Additional excitatory synaptic connections (red) project from the excitatory to inhibitory neuron population. This connectivity is dense and is initialized in some range. However, these synaptic connections are not plastic. In all simulations the excitatory neuron population size, N_{e}, is consistently four times the number of inhibitory neurons, $N_{\mathrm{e}} = 4N_{\mathrm{i}}$.

5.2 Methods

5.2.1 Approximating Network Dynamics for an Excitatory and Inhibitory Neuron Circuit

The inhibitory neuron circuit described and developed in Chapter 4 ignored the role of excitatory neurons. In order to extend our analysis to a network of both excitatory and inhibitory neurons, we adopt a network arrangement, as depicted in Figure 5.1. In such a network, excitatory neurons are the excitation source for inhibitory neurons which, in turn, inhibit the same excitatory population which drove them, thus introducing a competitive element among excitatory neurons driving them. Furthermore, inhibitory neurons laterally inhibit one another ensuring that the

inhibition is also selective. Such an excitatory and inhibitory neuron circuit is common in neural network literature, as described in the introduction, and is also the implicit assumption for many sparsity mechanisms, such as that used in our rate-coded investigation in Chapter 2.

In order to determine an expectation for the average sparseness, as defined in Section 4.3.5, of excitatory and inhibitory neurons, we extend the description which we made to account for the inhibitory neuron circuit behaviour in Chapter 4 Section 4.3.1. The key aim of the analysis in this section is to establish some expectation for the network response statistics based upon the inhibitory and excitatory neuron population sizes, just as we produced for a network of inhibitory neurons.

Let us first assume that excitatory and inhibitory neurons are homogeneous such that all excitatory neurons have the same probability of activation s_{exc} and all inhibitory neurons have an activation probability s_{inh}.

Given our network setup, depicted Figure 5.1, excitatory neurons are activated by an external source and inhibited by the inhibitory neuron population.

We can therefore adopt a similar description of the probability of activation as we assumed in Chapter 4. In particular, we assumed that in a highly competitive state, a neuron's activation is mutually exclusive with those from which it receives inhibition. We can therefore define the activation probability of excitatory neurons, s_{exc}, based upon the probability of activation of the N_{i} inhibitory neurons, s_{inh}, such that

$$s_{\text{exc}} = 1.0 - s_{\text{inh}} N_{\text{i}}.$$

This is an equivalent expression as that constructed for inhibitory neuron activation in Section 4.3.1 and makes the same assumption about the impact of inhibition. Given that the excitatory neurons provide excitation to the inhibitory neurons, we assume that the probability of inhibitory cell activation determined by the probability of the excitatory cell activation. Given the dense lateral inhibition,

the inhibitory neurons are inhibited by $N_\mathrm{i} - 1$ neighbours and we assume this lateral inhibition acts to reduce activation probability as above. The probability of a given inhibitory neuron being active is therefore:

$$s_\mathrm{inh} = f(s_\mathrm{exc}, N_\mathrm{e}) - s_\mathrm{inh}(N_\mathrm{i} - 1),$$

where f is some function describing the effect of excitatory neuron activity upon the activity of inhibitory neurons, hereafter referred to as the transfer function. This transfer function could assume a number of forms, however in order to simplify our analysis we assume here that it is linear. This leads to a simple and illustrative description of relative excitatory and inhibitory neuron activations. Let us also assume that the transfer function has some slope C and zero offset, such that $f(s_\mathrm{exc}, N_\mathrm{e}) = C s_\mathrm{exc} N_\mathrm{e}$. We can simultaneously solve and simplify the above expressions for excitatory inhibitory activation giving

$$s_\mathrm{inh} = \frac{C N_\mathrm{e}}{N_\mathrm{i} + C N_\mathrm{i} N_\mathrm{e}} \qquad \text{and} \qquad s_\mathrm{exc} = \frac{N_\mathrm{i}}{N_\mathrm{i} + C N_\mathrm{i} N_\mathrm{e}}. \tag{5.1}$$

These expressions provide the probabilities of excitatory and inhibitory neurons activation as a function of the population sizes and of the constant C. Taking the ratio of the inhibitory to excitatory activation probabilities gives a simple relation between the relative excitatory and inhibitory activation probability:

$$\frac{s_\mathrm{inh}}{s_\mathrm{exc}} = \frac{C N_\mathrm{e}}{N_\mathrm{i}}.$$

Given this relationship, we can conclude that when $N_\mathrm{e} > N_\mathrm{i}$, and $C > \frac{N_\mathrm{i}}{N_\mathrm{e}}$ the probability of inhibitory neuron activation is greater than that of excitatory neurons. This would suggests that, inhibitory neurons would be active (above threshold) to a larger subset of stimuli and are likely to learn broader receptive field tunings. Given the evidence showing that the number of excitatory neurons far exceed the number

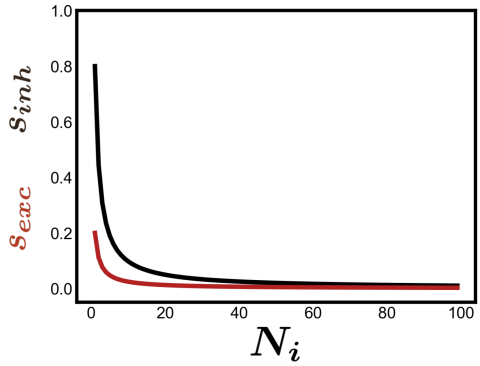

Figure 5.2: **Prediction of excitatory and inhibitory neuron activation probabilities for a range of network sizes.** A plot of the relationship between the activation probabilities of excitatory and inhibitory neurons in a network of structure such as that shown in Figure 5.1. Equation 5.1 presents these relations in the text. This plot assumes that the number of excitatory neurons in the network is consistently four times the number of inhibitory neurons, $N_\mathrm{e} = 4N_\mathrm{i}$, and that the constant C describing the slope of the excitatory to inhibitory excitation linear transfer function is 1.0. Excitatory neuron activation probability, s_exc, is plotted in red and inhibitory neuron activation probability, s_inh is shown in black.

of inhibitory neurons [125, 176] and that the inhibitory neuron receptive fields are indeed broader than those of excitatory neurons [115], this would be consistent.

For all simulations which follow, we consistently keep the number of excitatory neurons, N_e, equal to four times the number of inhibitory neurons, N_i, an experimentally observed ratio in cortex [176]. Our simple model description would therefore reproduce the greater tuning widths of inhibitory neurons given a sufficiently large value of parameter C. Recent quantitative experimental data have challenged the historical 4:1 chosen ratio of excitatory to inhibitory neurons and instead indicate that the true number of inhibitory neurons is closer to 11% [125]. The results and analyses presented in this chapter could be extended to include this smaller ratio of inhibitory to excitatory neurons, though we do not expect major qualitative differences.

Figure 5.2 shows how our probabilities of excitatory and inhibitory neuron activation, s_exc and s_inh respectively, vary for a range of network sizes where

consistently $N_e = 4 \cdot N_i$ and where $C = 1.0$. We use these curves as a theoretical expectation against which we match our network dynamics. However, it is important to re-iterate the strong assumption which is made to form these expectations: that the transfer function from excitatory to inhibitory activation is linear and has slope of 1.0. This is assumption is quite arbitrary and unlikely to be accurate, however it provides a simple explanation of network activity and could be extended under different assumptions of the transfer function.

5.2.2 Network Model, Learning Rules, and Inputs

Figure 5.1 shows the network arrangement which is simulated and trained in the results section below. The neuron types used are leaky integrate and fire (LIF) neurons and the parameter set used for these simulations are all equivalent to those used to produce the results in Chapter 4. In particular, unless stated otherwise, the parameters of the models in this section are described in Table 4.1, with one addition: excitatory neurons are included in these models. The number of excitatory neurons, N_e, is consistently four times the number of inhibitory neurons, $N_e = 4N_i$. Excitatory and inhibitory neuron parameters are otherwise equivalent.

The learning rules which are implemented to produce the results of this chapter are described in detail in Chapter 4, Section 4.3.3.2. In particular, all inhibitory synaptic connections are plastic with the inhibitory synaptic plasticity rule of Vogels et al. [201] and all excitatory synaptic connections from the input sources are plastic with the custom excitatory STDP plasticity rule described in Chapter 4 Section 4.3.3.2. Furthermore, as explored in Chapter 4 Section 4.4.1.1 we incorporate a population of background inhibitory neurons for all simulations. There are ten background inhibitory neurons in every simulation which fire with rate 100Hz, spike times sampled from a Poisson distribution. Inhibitory synaptic connections project from this background inhibitory neuron group to all excitatory

and inhibitory neurons in the network. The synaptic weights of all inhibitory synaptic connections are initialised at zero and are modified during training by the inhibitory plasticity rule. The motivation for the inclusion of background inhibitory neurons was discussed in detail in Chapter 4 4.4.1.1. In short, they allow networks of relatively few neurons to be brought to the homeostatic firing rate despite the relatively small amount of inhibition produced in the network. Network diagrams showing the network structure with these background inhibitory connections is shown in Figure 5.3.

One additional feature of note in these network models is the excitatory connectivity from the excitatory to inhibitory neuron populations. In order to reduce the space of parameters that require tuning, we implement these synaptic connections with static weights. The particular weights are chosen from a uniform distribution between zero and some maximum weight. The weights are randomly sampled from this uniform distribution before the simulations begin and are kept fixed during both training and testing.

5.2.2.1 Training and Testing Data

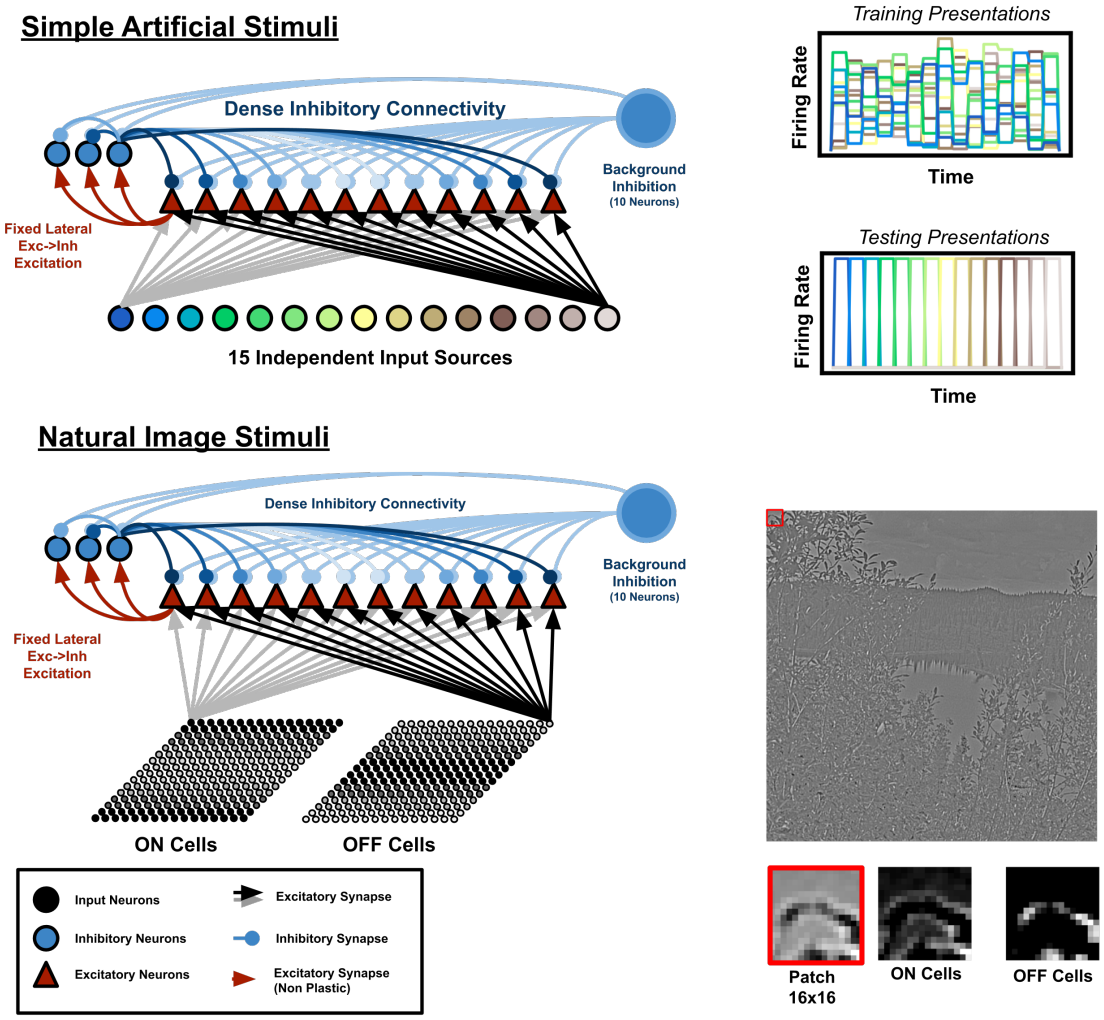

Figure 5.3: **A depiction of the input datasets training procedures.** *Top,* a depiction of the simple artificial stimulus set and its presentation. A set of 15 independent input sources are used as the inputs to our network of spiking neurons. These input sources are activated in a random combination during training (with the total summed firing rate across all input sources set to 500Hz) but are only active one at a time during testing. Example plots of the training and testing regimes are shown right with colours corresponding to the set of input sources in the network shown left. These input sources are shown fully connected to a group of excitatory output neurons. These excitatory output neurons project with fixed excitatory synaptic connections to a group of inhibitory neurons. which project feedback inhibitory synaptic connections to both the excitatory and inhibitory neuron populations. Finally, there is an additional background set of inhibitory neurons with random spike times which inhibit all excitatory and inhibitory neurons in the network. Excitatory synaptic connections (other than those which are fixed)

are plastic with the custom excitatory STDP rule described in Chapter 4, and inhibitory synaptic connections are plastic with the inhibitory synaptic plasticity rule of Vogels et al. [201]. All plasticity rules are detailed in Section 4.3.3.2. *Bottom,* a depiction of the natural image stimulus set, and an example of patch extraction. Right, an example whitened image from the training data set is shown. The average grey value in the image indicates a response of zero, white indicates a positive response, and black indicates a negative response. Below is shown a single patch from this image and its decomposition into the positive and negative components (ON and OFF cells respectively). For each image patch, two groups of 16 by 16 input neurons represent the input to our networks. One group responding as ON Cells, the other as OFF Cells. These two populations of input cells to the network are depicted left with full feed-forward excitatory connectivity to a population of excitatory and inhibitory cells in the same configuration as described for the simple artificial stimulus set, top.

The training and testing of the networks in this chapter was carried out in an identical fashion to that in Chapter 4. The input data sets are also equivalent: a simple artificial dataset and a dataset of patches extracted from natural images. The specific training procedure, and pre-processing of stimuli is described in Section 4.3.4.1. Figure 5.3 depicts both the network structures trained in this chapter and the input source arrangement.

5.3 Results

In this chapter we investigate a population of excitatory and inhibitory neurons which we train with both simple stimuli and with patches from natural images. The stimuli used for training are described briefly above and in detail in Section 4.3.4, Chapter 4. The circuit being modelled here is an extension of our inhibitory neuron competing circuit presented in Chapter 4 to a system of interacting excitatory and inhibitory neurons We first test the network dynamics when training with a simple dataset of artificial stimuli. In this attempt, we encounter issues with the convergence of the inhibitory plasticity rule. We propose a potential solution to this issue and conclude the chapter by training the circuit setup with patches

extracted from whitened natural images.

5.3.1 Matching Network Dynamics to Theory with Simple Stimuli

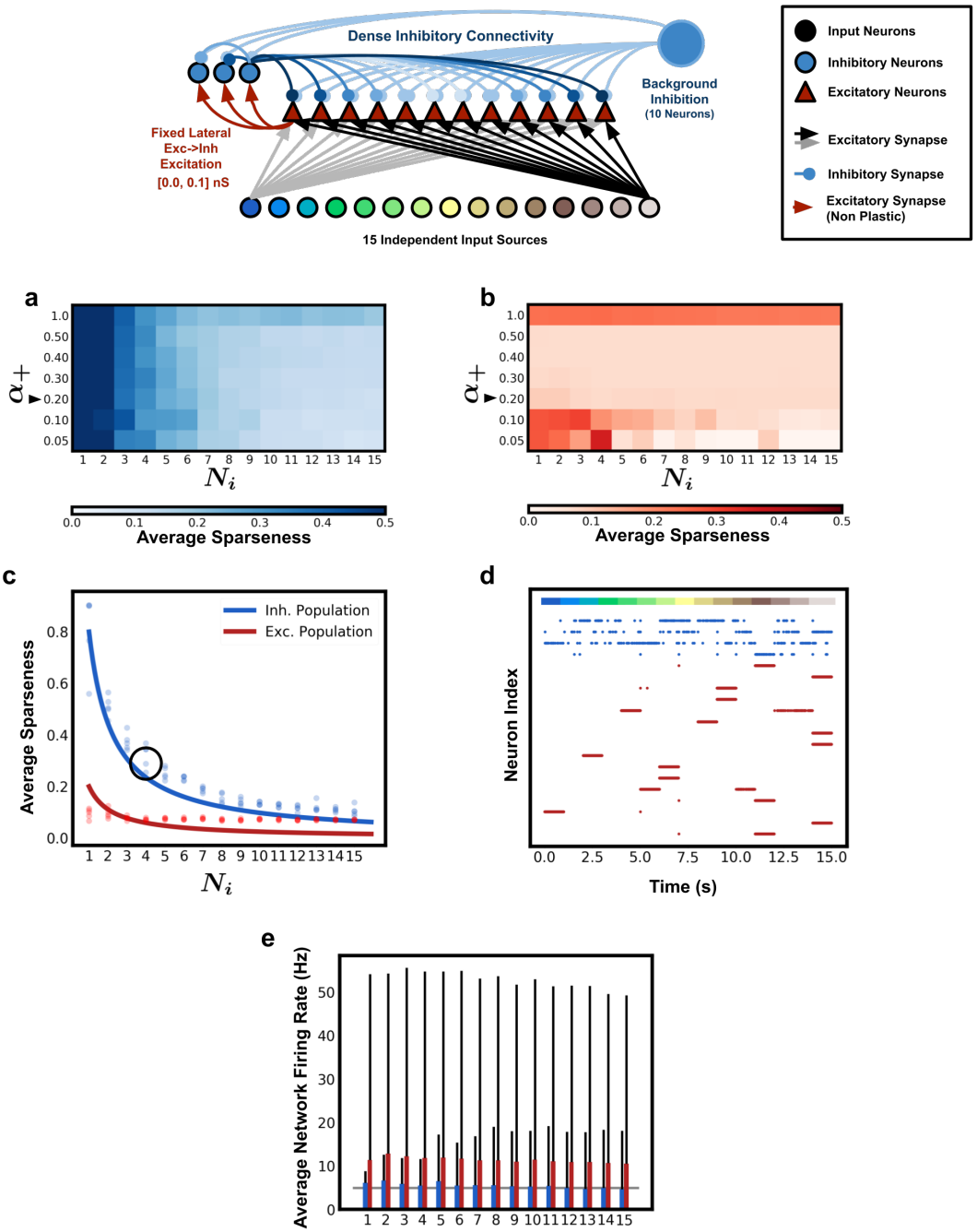

Figure 5.4: **Training a network of excitatory and inhibitory neurons for**

comparison against the predicted sparseness. Top, a depiction of the network structure trained in this section. The blue connections indicate the dense lateral inhibitory connections and the black/grey arrows indicate feed-forward excitatory synaptic connections. Input to the networks are provided by a set of 10 input sources. During training, excitatory feed-forward connections in this network are plastic with the eSTDP learning rule. All inhibitory synaptic connections are plastic with the iSTDP rule. Inhibitory neurons are excited by fixed strength (non-plastic) lateral excitatory synaptic connections from excitatory cells, coloured red. The ratio of excitatory to inhibitory cells is kept fixed at 4:1, i.e. $N_\mathrm{e} = 4N_\mathrm{i}$. **(a)** The average inhibitory neuron sparseness over a range of parameters. Average sparseness is indicated by the colouring of the parameter map. The y-axis represents variation of a key parameter α_+ of the eSTDP rule, and the x-axis represents changes to the network size N_i. **(b)** The average excitatory neuron sparseness over the same range of parameters. Colouring and axes are organised in an equivalent fashion to part (b), but instead represent the network's excitatory neuron response statistics. **(c)** The average sparseness values of the excitatory and inhibitory sub-networks are presented for a specific value of the eSTDP parameter a_+ (0.2) across a range of network sizes. Each network is trained five times under a different random seed, affecting the initial weights and stimulus presentation order. The corresponding row in the parameter maps (plots b, and c) are indicated by a black arrow. Solid lines in this plot indicate the theoretically predicted excitatory (red) and inhibitory (blue) neuron sparseness values. The semi-transparent points plotted atop these solid lines each represent a single trained network which was trained as outlined in the text. **(d)** A raster plot showing the responses of neurons, excitatory in red and inhibitory in blue, to the range of input source activations. The particular network, circled in plot (d), is composed of four inhibitory and sixteen excitatory neurons. This raster plot shows the presentation of our 15 input stimulation sources, coloured bars top, which are activated independently and the neuron responses are shown below. For clarity, the plotted bars are coloured to correspond with the input source colouring in the diagram (a). **(e)** The average firing rates of the network plotted in (c) are shown with error bars representing the standard deviation in this firing rate. The excitatory and inhibitory sub-population firing rates are shown separately, inhibitory population statistics are shown in blue and excitatory neuron population statistics in red. A gray horizontal line shows the target firing rate, λ_target, 5Hz.

Figure 5.4a and 5.4b, show the inhibitory and excitatory neuron population sparseness measures, respectively, for trained networks over a range of the model parameter α_+ and the network size, N_i. As can be observed, the inhibitory population consistently has a higher probability of activation than the excitatory population, and this relationship is robust with variation of the selected model parameters. This result is to be expected given our earlier theoretical analysis, see

Section 5.2.1, in which we described the expectation that a network with a greater proportion of excitatory vs inhibitory neurons aught to have excitatory neurons with a smaller probability of activation. This analysis was specific to the network architecture that we have employed. Figure 5.4c compares the network statistics from a particular parameter slice of the parameter maps, indicated by a black arrow in Figure 5.4a and 5.4b, to the theoretical prediction. Comparing the trained network statistics directly to the theoretical proposal presented in Figure 5.2, we see a qualitative agreement between the simulated networks and the probabilistic predictions, see Figure 5.4c. The solid lines in this plot correspond to the predicted inhibitory (blue) and excitatory (red) neuron relationships between network size and average sparseness. Semi-transparent points correspond to the average sparseness measures of inhibitory (blue) and excitatory (red) neuron populations in 5 randomly seeded networks at network sizes ranging from 1 inhibitory neuron to 15 inhibitory neurons, corresponding to between 4 and 60 excitatory neurons.

The network sparseness measurements agree in their trends with the predicted network behaviour. These data are however more varied than those of the inhibitory neuron network presented in the previous chapter. The deviation of our fit, especially for excitatory neuron sparseness, indicates that our probabilistic model is likely an oversimplification of the network dynamics. We attribute this to the complexity of the interacting recurrent spiking network and the inability for our simple probabilistic description to capture this complexity. We nonetheless accept these data as evidence for a qualitative match between our predicted and modelled networks, and leave the production of a more accurate probabilistic description of this system to future studies.

Individual excitatory and inhibitory neuron responses from a single trained network, are shown in Figure 5.4d. Each cell in the example network, circled in Figure 5.4c, has spike times as shown with excitatory neuron spikes coloured red, and inhibitory neuron spikes coloured blue. The bars at the top of this plot are

coloured according to the specific input source which is active for that time period, see network input source colours in Figure 5.4, top. As can be observed, individual neurons have strong input source preferences which are different from one another, as one would expect from a competitive system. Another key parameter that was constrained to produce this balanced network, was the range of the static excitatory connection weights from the excitatory neurons to inhibitory neurons. These weights are chosen by sampling randomly from a uniform distribution between zero and some maximum weight value at the beginning of the simulation and fixed. The weight range of these static connections was kept relatively low in order to ensure that the inhibitory lateral connections from inhibitory to excitatory neurons did not become excessively large during a reasonable training time. However, unlike the simulations presented in Chapter 4 we observe that the inhibitory synaptic connections do not converge. In particular, the inhibitory neuron to excitatory neuron synaptic connections are observed to increase linearly through training without any sign of convergence.

Figure 5.5b, shows the evolution of the mean inhibitory to excitatory synaptic connection weight in networks of 4 inhibitory and 16 excitatory neurons connected in the structure depicted in Figure 5.5a. This weight evolution is shown for several simulations with different excitatory to inhibitory connection strength initialisations. As described, excitatory to inhibitory weights are initialised by randomly sampling from a uniform distribution between 0.0 and a maximum weight value and are thereafter kept fixed.

As can be seen, during training the inhibitory weights increase at a rates somewhat proportional to the excitatory to inhibitory neuron fixed weight range and shows no indication of convergence. In simulations which were approaching convergence, we would expect the mean inhibitory weight to asymptote to a fixed value. However, for the range of connectivities presented we see the inhibitory weight range increasing in an unbounded fashion. For significantly large inhibitory

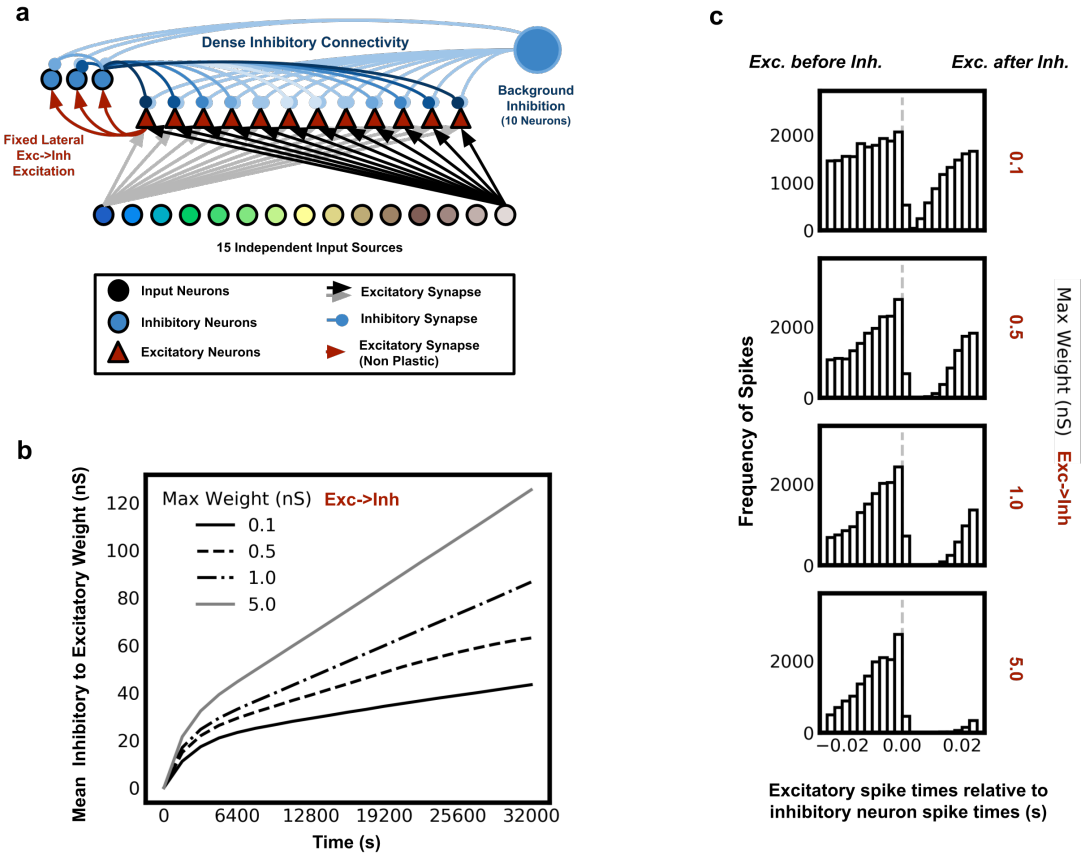

Figure 5.5: **The evolution of inhibitory to excitatory neuron synaptic weights during training for a range of lateral excitatory synaptic connection weight initialisations.** **(a)** The network structure trained to produce these results. This is equivalent to that described in Figure 5.3, top. **(b)** The evolution of the mean inhibitory to excitatory neuron synaptic weights over training for this network structure. Individual lines represent networks with a different range of weights from which the fixed excitatory to inhibitory neuron synaptic connections were sampled. The weights are sampled from a uniform distribution between 0.0nS and a maximum weight value (see legend). **(c)** The unbounded increase in the mean inhibitory feedback weight, see subplot (b), is attributed to the extreme timing relation which develops between excitatory and inhibitory neurons. These timing relations are shown here as cross-correlograms of the excitatory neuron spike times relative to inhibitory neuron spike times. Each box shows the distribution of excitatory neuron spikes about inhibitory neuron spikes. These are produced by taking the spike time of each inhibitory neuron spike, collecting the spike times of all excitatory neuron spikes within 50ms of this inhibitory neuron spike and then forming a distributions of the excitatory spike times relative to those of inhibitory neurons.

synaptic connection weights, the networks become unstable. The Forward-Euler approximation used to solve the network dynamics (as we discussed in Chapter 3) is only appropriate if the dynamics of the system are sufficiently slow compared to the numerical timestep. As the inhibitory weight values increase, their impact on the sub-millisecond scale become more and more significant until they become sufficiently large as to cause a breakdown of the Forward-Euler approximation and the membrane voltages of cells begin to oscillate wildly about the inhibitory reversal potential.

The reason for these excessively large values of the inhibitory weight are a run-away of the excitatory and inhibitory spike time correlations. In essence, activity in the excitatory neuron population causally produces activation of inhibitory neurons such that spike times of excitatory and inhibitory neurons are highly correlated.

Figure 5.5c shows cross-correlograms of the spike times of excitatory neurons and inhibitory neurons in these networks. A grey dashed line in these plots shows where the spike time differences are zero, and bars about this value represent the number of excitatory spikes which occur at that time delay with respect to inhibitory spikes. With an increase in the static excitatory to inhibitory neuron synaptic connection weights, the correlation of excitatory to inhibitory spike times increases. This can be observed as a taller peak with shallower tails. Inhibitory neurons become active consistently following excitatory neuron activity. This cross-correlogram shown in particular that the inhibitory neuron population activity is high consistently following excitatory neuron activity. This leads to an increase in all inhibitory weights, given the correlative iSTDP rule which modifies all inhibitory synaptic connection weights in the network. The feedback inhibition can never arrive before this excitatory input and therefore never stops the excitatory neurons from becoming active before the inhibitory neurons, but does stop them from becoming active after inhibitory neuron activation. Thus, the distribution of excitatory neuron spike times about inhibitory neuron spike times forms a peak prior to the inhibitory spike times and lead to this runaway increase in inhibitory feedback.

In particular this observation indicates a breakdown of the assumptions which were made when describing learning rules dynamics in Chapter 4 Section 4.3.3.2. It was assumed that the spike times of neurons could be described by some rate and that the specific spike times were Poisson distributed and random. It was based upon these assumptions that the expected set-points for plasticity rules were determined. However, with such highly correlated spike times, this assumption breaks down.

The issue of non-converging weights could be resolved in a number of ways. Methods for bounding the inhibitory weights could be explored, for example as a hard or soft-bound. However any attempt to modify the form of the learning rule (which is required for soft-bounding) would change the effect which it has of bringing neural responses to some homeostatic average. A hard-bound could be used but given the consistently rising inhibitory weights, all weights would reach the maximum value and the desired effect of detailed balance would be lost.

One other method to combat this extreme correlation between excitatory and inhibitory neuron activation is to ensure that inhibitory neurons fire action potentials more often than excitatory neurons. This would act to reduce the correlation by flattening the cross-correlogram and distributing inhibition over time. This also allows an equivalent amount of inhibition to be delivered per second with a lower inhibitory weight. This method has experimental support given that inhibitory inter-neurons are observed to have firing rates which are on the order of five times greater than those of excitatory neurons [93].

Figure 5.6, shows the result of modelling these same networks, however with a different target firing rate imposed upon inhibitory neurons compared to excitatory cells. Excitatory neurons, as before, have a target firing rate of 5Hz which is achieved through training with the iSTDP rule on their afferent inhibitory synaptic connections. However, inhibitory neurons now have a target firing rate of 25Hz. Comparing Figure 5.5b and c, to Figure 5.6 b and c shows that the introduction of this higher homeostatic firing rate for inhibitory neurons extends

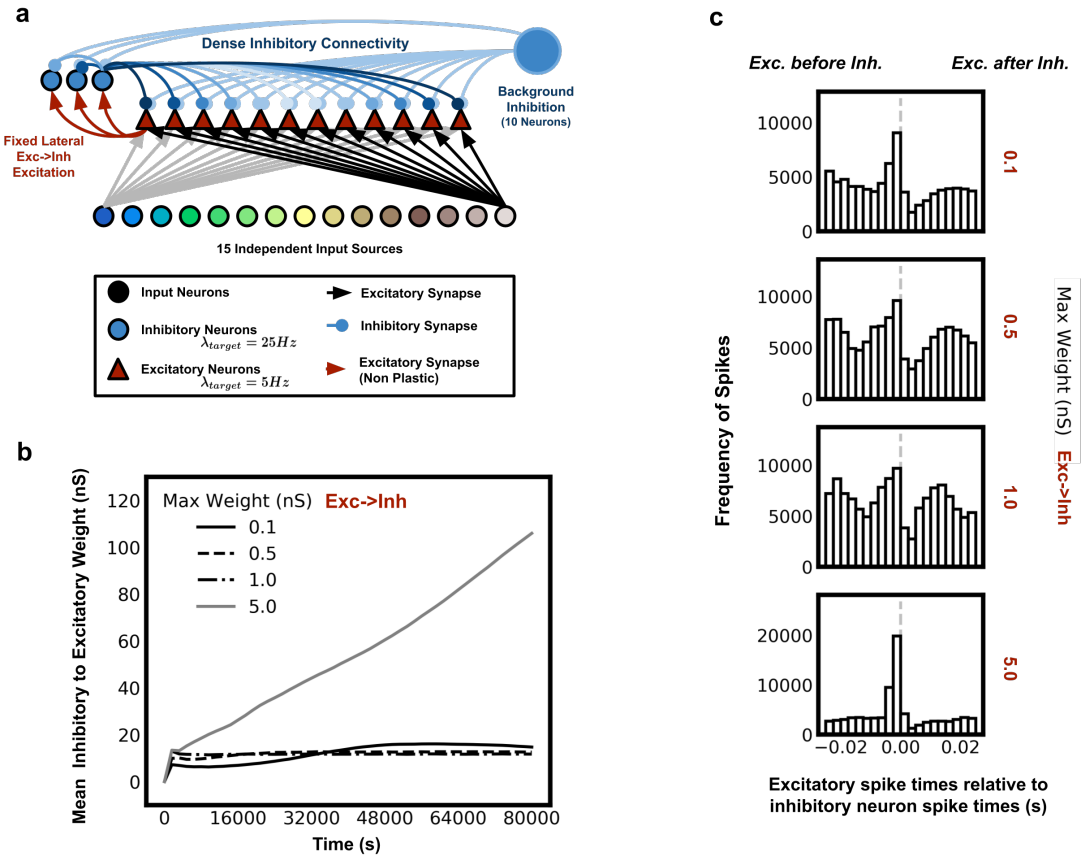

Figure 5.6: **The evolution of inhibitory to excitatory neuron synaptic weights in networks with a modified inhibitory neuron target firing rate.** **(a)** The network structure trained to produce these results. This is equivalent to that described in Figure 5.3, top, except that the inhibitory and excitatory neurons now have different target firing rates, 25Hz and 5Hz respectively. **(b)** The evolution of the mean inhibitory to excitatory synaptic connection weights for this network structure when initialised with different excitatory to inhibitory neuron synaptic weight ranges. As in Figure 5.5. Note these excitatory connections are static and therefore the initialised values remain unchanged during training. Note that there is a convergence of this mean for most parameter ranges. **(c)** In Figure 5.5, an unbounded increase in the mean inhibitory feedback weight was attributed to the extreme correlative firing which developed between excitatory and inhibitory neurons. We show here the cross-correlograms of the excitatory neuron spike times relative to inhibitory neuron spike times, as in Figure 5.5c. The cross-correlograms for this converging network setup are significantly flatter than those shown previously. We attribute the convergence of curves in subplot (b) to this flattening of the excitatory to inhibitory neuron spike time correlation.

and flattens the cross-correlogram of excitatory and inhibitory neuron spike times and allows convergence of the network inhibitory synaptic connections at relatively low inhibitory weight values. For large excitatory to inhibitory connection strength ranges, $[0.0, 5.0 \cdot nS]$, these networks still show excessively large inhibitory weight values, however this is ultimately a feature of systems in which the excitatory neurons provide significant drive to the inhibitory neurons and thereby produce highly correlated spike times. Thus, by modifying the target firing rates of our inhibitory neuron population we can achieve convergence of our inhibitory synaptic connections and resume learning in a stable regime.

Though we justify this greater firing rate of inhibitory neurons based upon experimental literature, it would be over-reaching to suggest this as the purpose of the firing rate difference. Especially given our indication that a number of other mechanism if explored sufficiently could provide methods for the bounding of inhibitory synaptic weights.

5.3.2 Reproducing V1 Simple Cell like Receptive Fields in an Excitatory and Inhibitory Neuron Network

The above analysis trained a system of excitatory and inhibitory neurons with a simple stimulus set. We now develop this model by training with patches from natural images. King et al. [105] implemented a similar network structure and described the decorrelating property of inhibitory neurons. However, their model employed non-spiking inputs, which we rectify with spiking input neurons, and they separated the network mechanisms for homeostasis and decorrelation, whereas we produce these features with an inhibitory plasticity rule. Finally, their input excitatory synaptic connections were trained with a rate-based Hebbian learning rule (Oja's rule) which we replace here with our custom STDP based plasticity rule.

In the networks simulated here, the neuron activations are brought to homeostasis

through the iSTDP rule which is active on all inhibitory synaptic connections, while the input excitatory connections are potentiated through the eSTDP rule, allowing excitatory neurons to form receptive fields. Both plasticity rules are described in Section 4.3.3.2. An external population of inhibitory neurons, firing consistently with a mean firing rate of 100Hz, provides background inhibition with inhibitory synaptic connections which project to all neurons in the network with the iSTDP plasticity rule. This component provides sufficient background inhibition to bring the network to homeostasis regardless of the network size, see Section 4.4.1.1 for a discussion.

Finally, having explored the effect of setting the target firing rate of inhibitory neurons to be higher than those of excitatory neurons in the simulations shown above, see Figure 5.6, we apply this principle here. Inhibitory neurons having a target firing rate, λ_{target}, of 25Hz while excitatory neurons have a 5Hz target firing rate. Without such an adjustment, we observed the same failure of convergence described in Figure 5.5 in which inhibitory synaptic connections grow indefinitely and cause network dynamics to become unstable.

Figure 5.7, left, depicts the network structure which we have described and which is simulated to produce the results shown right. Figure 5.7, right, shows the receptive fields of excitatory neurons in trained networks of various sizes. These networks show the same qualitative trend as was observed for the inhibitory only neuron networks trained on this same stimulus set in Chapter 4, see Figure 4.6. Larger network sizes correspond to smaller neuron receptive field sizes after training. The spatial scale of receptive fields in our combined excitatory and inhibitory neuron networks are also relatively similar to those observed for inhibitory only neuron networks. One marked difference is that these receptive fields show a greater bias towards the formation of horizontally oriented receptive fields. As discussed in Section 4.4.1.2, this may be a bias inherent to the dataset which we used. We would expect a mitigation of this bias should the training patches be randomly rotated for training, though here we use patches without modification.

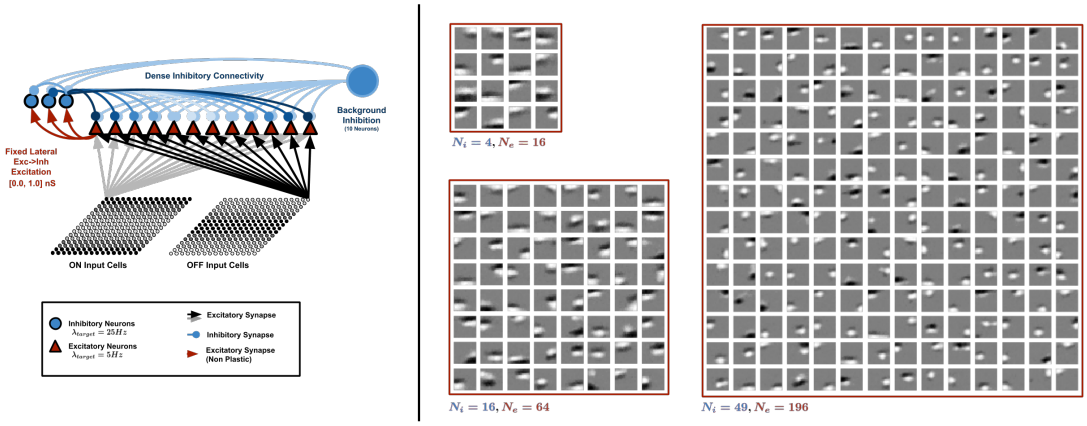

Figure 5.7: **Training a competitive network of excitatory and inhibitory neurons with patches from whitened natural images.** *Left,* a depiction of the network structure. Input is provided by two 16x16 input sources which represent the ON and OFF cell responses of retinal ganglion cells to a single 16x16 image patch, Section 4.3.4.2 provides a detailed description of this input dataset. The specifics of the network setup shown here are described in Section 5.3. Training consists of the presentation of randomly selected patches from the input dataset for 50ms while the eSTDP learning rule is active on the feedforward excitatory synaptic connections. The iSTDP plasticity rule is active on all lateral inhibitory connections. Training consist of the presentation of approximately $2 \cdot 10^6$ image patches. *Right,* the converged receptive field tunings of neurons in competing networks of a range of sizes. Network sizes are presented below the receptive field grids, $N_\mathrm{i} = 4, 16, 49$ corresponding to $N_\mathrm{e} = 16, 64, 196$. Only the excitatory receptive field tunings are shown here.

The size, or tuning widths, of the individual neuron receptive fields in these networks is dependent upon the level of competition provided by inhibitory neurons, a feature which we explored in depth in Chapter 4. In particular, a greater number of inhibitory neurons was described to lead to greater competition and thereby smaller receptive field tuning widths. In the networks described here, the number of inhibitory neurons is small relative to the number of excitatory neurons but this principle holds nonetheless. However, a key restraint in the networks described here is the lack of tuning of inhibitory neuron receptive fields. Inhibitory neuron receptive fields are instead based upon those of the excitatory neurons which project to them, through static weighted excitatory synaptic connections. It therefore seems

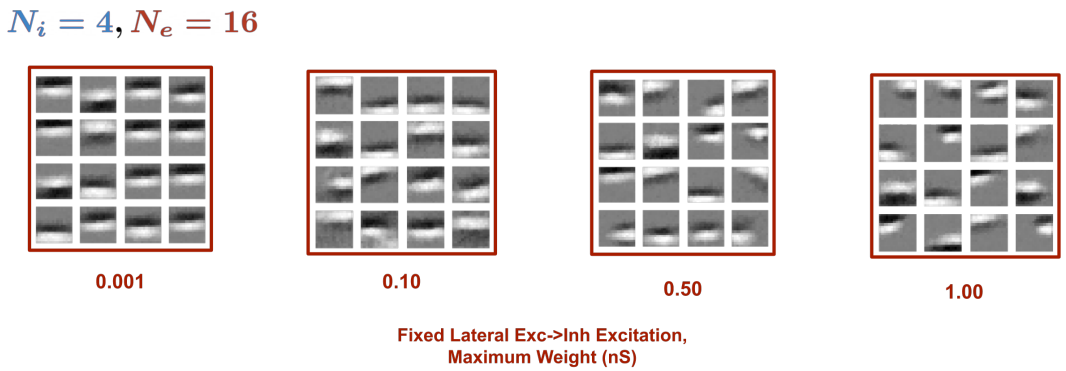

Figure 5.8: **The receptive fields produced in through training in networks with different excitatory to inhibitory neuron synaptic connection weight ranges.** We show receptive fields of excitatory neuron in network of four inhibitory and sixteen excitatory neuron under different excitatory to inhibitory synaptic connection weight initializations. These weights are static and therefore are not modified during training. These weights are sampled from a uniform distribution between 0nS and some maximum value. The maximum weight values are given below each receptive field grid. As can be observed, left to right the receptive fields of the neurons in these networks become smaller and more diverse as the weight range increases.

reasonable to expect that the degree of competition in these networks might be modulated by the strengths of these static excitatory projections.

Figure 5.8, shows how the receptive fields of excitatory neurons in a fixed network size, four inhibitory and sixteen excitatory cells, vary as the excitatory to inhibitory neuron synaptic connectivity weights are scaled. As can be seen, for low values of this lateral excitation, the receptive fields of the excitatory neurons are under little competition and are highly overlapping and relatively large in spatial scale. As this lateral excitatory synaptic weight strength is increased, the responses of inhibitory neurons produce a greater competitive effect upon excitatory neurons. The excitatory neurons in turn become more restricted in size and are less overlapping. The extent to which these receptive fields are limited seems to saturate as the static excitatory weight strengths are scaled up. This is reasonable given that the competition has an upper bound – the inhibitory neurons can at

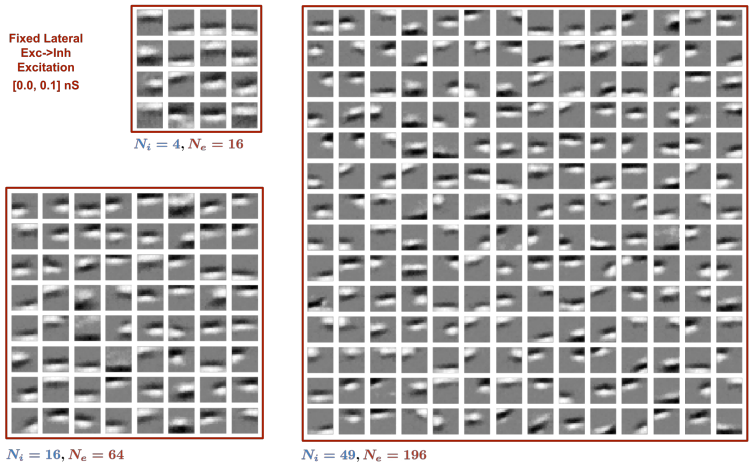

Figure 5.9: **Visualising the converged receptive fields of neurons in networks with weak excitatory synaptic connections from the excitatory to inhibitory neuron populations.** The converged receptive field tunings of neurons in competing networks of a range of sizes. The static excitatory to inhibitory neuron synaptic connection weights are sampled from a uniform distribution in the interval [0.0ns, 0.1ns]. Network sizes are presented below the receptive field grids, $N_\mathrm{i} = 4, 16, 49$ corresponding to $N_\mathrm{e} = 16, 64, 196$. Only the excitatory receptive field tunings are shown here.

maximum can create dense competition between the excitatory neurons but cannot impose greater competition than this.

Figure 5.9 shows the excitatory neuron receptive fields from a trained excitatory and inhibitory neuron network with relatively small excitatory synaptic connection strengths for projections from excitatory to inhibitory neurons. Comparing the receptive fields shown in Figure 5.9 against those produced in the networks described in Figure 5.7, we observe much broader receptive field tunings. This ability to modulate receptive field sizes, an indication of our modulation of the level of competition, by modification of synaptic weight strengths provides a mechanism for such a network to modulate receptive field overlap and neuron response redundancy. Such flexibility and redundancy is expected to be crucial in order to ensure that neural circuits are sufficiently robust.

One final question we address is that of the tuning of inhibitory neuron receptive

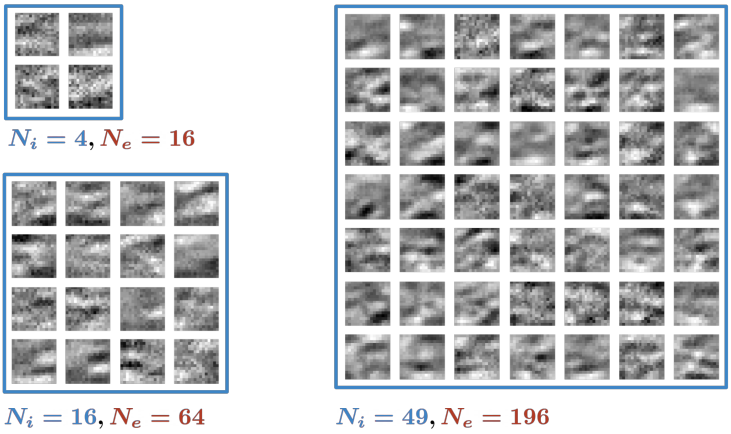

Figure 5.10: **Visualising the STA decoded receptive fields of inhibitory neurons in the networks described in Figure 5.7.** Network sizes are presented below the receptive field grids, $N_{\mathrm{i}} = 4, 16, 49$ corresponding to $N_{\mathrm{e}} = 16, 64, 196$. See Appendix B.0.3 for a description of the decoding process. These receptive fields are relatively weak in strength and could only be produced after 50,000s of simulation time, corresponding to an excess of 250,000 spikes per output neuron for this recovery process.

fields. In particular, we are no longer capable of decoding inhibitory neuron receptive fields from their excitatory input connections alone since these are not direct projections from the input layer. Instead, we make use of spike-triggered averaging (STA) in order to determine the receptive field tunings of inhibitory neurons, see Appendix B.0.3 for details.

Figure 5.10 shows the decoded receptive fields of the inhibitory neuron populations of the networks which were described in Figure 5.7. We can observe that these receptive fields, unlike those of the excitatory neurons are not tuned to oriented bars and edges. Instead, these neurons have have receptive fields which are in effect random combinations of the receptive fields of the excitatory population – a reflection of our architecture.

Such un-tuned inhibitory neuron receptive fields are uncharacteristic when compared to the inhibitory neuron receptive fields described in studies such as those by King et al [105], and Zhu et al. [226]. In these studies, which we describe in this chapter's introduction, the excitatory projections to inhibitory neurons are

learned and this results in the formation of Gabor-like receptive fields tuning of inhibitory neurons. The production of a similar extension to our model – one in which the excitatory projections to the inhibitory neuron population are learned – would require the identification of appropriate learning rules to adequately tune the excitatory to inhibitory neuron receptive fields. The form that such a learning rule would take is an open question for research.

Given our ability to produce excitatory and inhibitory networks in which excitatory neurons have qualitatively similar receptive fields when compared to the inhibitory neuron only network of the previous chapter, we conclude our analyses of these networks. In the section which follows we expand our discussion of outstanding questions and how these network models could be extended.

5.4 Discussion

This chapter explored an extension of the training procedure which we described in Chapter 4 to networks of both excitatory and inhibitory neurons. This extension gave rise to a source of instability in the inhibitory plasticity rule of Vogels et al. [201], exacerbated by the network setup we chose to investigate. In particular, when there exists dense inhibitory synaptic connectivity from inhibitory neurons to the excitatory population which provides these inhibitory neurons input activation, and there exists significant correlation between the excitatory and inhibitory neuron activation, we arrive at a system in which the inhibitory feedback synaptic weights increase in an unbounded and non-convergent fashion.

We propose one method of avoiding this: ensuring that inhibitory neurons spike with a greater average rate than their driving excitatory neuron population, thereby reducing the degree of correlation. This method proves useful and enables training the particular network architectures we are investigating. However, this is only one possible solution to the issue and there are a number of alternative open

avenues for potential extension of the inhibitory plasticity rule. Unfortunately any attempt to constrain the inhibitory weights, such as soft or hard bounds upon their maximum value, would have to be imposed alongside an adaptation of the learning rule in order to ensure that the homeostatic and detailed balancing properties of the learning rule are nonetheless maintained. Alternative approaches such as the extension of the inhibitory plasticity rule from a pair-wise spike STDP rule to rules with greater sophistication, such as a triplet-based rule [152], could allow more accurate tracking of the post-synaptic neuron firing rates and thereby greater inhibitory synaptic connection tuning accuracy. This would be an interesting avenue of research, as would be an investigations of how alternative inhibitory plasticity rules would perform in a similar network setup.

In order to reduce the complexity of the models described in this chapter, we fixed the strength of lateral excitatory to inhibitory neuron synaptic connection weights. This enforcement of static excitatory connectivity allowed the parameter space that required tuning to be reduced. However, modification of the synaptic weights of these connections through plasticity could allow the production of tuned inhibitory neuron receptive fields after training. Given the space of possible learning rules which could be applied to these connections and the increased simulation time which would be required to simulate further plastic synapses, it is likely to require extended consideration.

One criticism against our current excitatory and inhibitory network setup is that the layered structure and connectivity style – with excitatory and inhibitory neurons separated into two populations with projections between them – is an abstraction of cortical connectivity. As discussed in the Introduction, such a connectivity structure has been derived based upon the desired competitive function of inhibitory neurons rather that an observed cortical structure. This is a simple mechanism which produces competition between excitatory neurons while ensuring Dale's law is maintained. However, a number of connection types and properties

were ignored in this network model. For example, no recurrent excitatory synaptic connections existed between excitatory neurons in the output layer and the inhibitory connectivity was dense rather than sparse, as is observed in cortex. It would be fruitful to follow up these excitatory and inhibitory neuron network models with models in which excitatory and inhibitory neurons are projected to by a common input and have lateral connectivities based upon the local probabilistic connectivity observed in cortical studies [99].

Having produced a single layer sparse coding model of simple cell development in early visual processing (V1), it would also be desirable to attempt an extension of this into a multi-layer model. Such a model would be an attempt in a spiking neural network to produce a hierarchical description of visual processing, somewhat similar to the HMAX or VisNet models [156, 205]. Given the evidence that the sparseness of cortical visual areas across the ventral visual pathway are largely similar [210], it suggests that we could find some success in simply layering our network structures presented here in order to reproduce features such as increasing receptive field sizes and increasing complexity of receptive fields over this hierarchy [76, 157].

In our attempt to produce stable learning in these systems, we have adapted network dynamics to more faithfully reflect those of an equivalent rate-based network. One question which we must then consider is how much we gain from our extension of existing models to these more detailed spiking neural networks. In general its seems that we have gained little in terms of computational capacity or efficiency. However, our understanding of potential underlying mechanisms which give rise to rate-based learning become more coherent. Furthermore, the production of these networks has produced a number of questions regarding learning rule stability, inhibitory neuron receptive field formation, and more. Finally, by enabling the training of such networks we come closer to the interrogation of any potential benefits of spike timing based encodings of information over rate based.

6

Discussion

This thesis describes a research arc which began with rate-coded competitive neural network modelling and developed into an exploration of simulation tools which can be used to efficiently simulate spiking neural networks, and learning rules, which combine competitive learning and excitatory-inhibitory balance.

Rate coded models of neural processing have been demonstrated in literature as effective at producing insights into neural processing through unsupervised and local learning rules. We applied one such method in Chapter 2 and described the development of a simple neural network model of pitch processing. This model generalized to unconventional pitch eliciting stimuli such as missing fundamental and iterated rippled noise stimuli under a particular training regime. A diverse training procedure was shown to be necessary to ensure that novel stimuli, not from within the training set, could be distinguished from neural responses. Though successful in providing some insights into the training necessary for a place theory explanation

of pitch processing, this approach did not account for temporal processing and was generally limited in its scope. Furthermore, it provided no insights into the mechanistic implementations of the competitive interactions which drove learning. Since the publication of this work, a number of studies exploring pitch processing in extended models with spiking dynamics and structured delays in the propagation of action potentials between neurons have provided further accounts of both place and temporal theories of pitch processing [111, 52, 53].

In the rate coded model we developed in Chapter 2 network dynamics were highly abstract. Activities of neurons were represented with scalar rate values, competition was explicitly introduced through a threshold (dependent upon all other neuron activations), and weight vectors were normalized after every weight update. Such abstractions are common in rate coded models and enable reliable learning. However, their mechanistic counterparts are opaque and therefore it is difficult to extend these principles of competitive learning from rate-based networks to networks of spiking neurons with biologically realistic components. Furthermore, gaining insight into such mechanistic components requires simulation of models with greater complexity and corresponding computational cost.

A major benefit of rate coded models are their relative computational simplicity. Network dynamics can be treated largely as vector-vector and vector-matrix operations, which can be computed by leveraging highly optimised software libraries for linear-algebra. By comparison, spiking neural network models, which allow greater interrogation of the mechanisms of neural circuit operation, incur significantly greater computational costs. The calculation of neuron membrane voltages over time is accomplished through iterative numerical approximation of the network dynamics, as described in Chapter 3. This requires solving the individual neuron dynamics tens of thousands of times per second of simulation, an extremely computationally expensive process. Though the brain is regarded as energy efficient in its computations, attempts to model this same system in general-purpose

computer systems are highly inefficient. Relatively few processing cores are used to compute iterative dynamics of systems of large numbers of neurons with orders of magnitude more synaptic connections between them.

In Chapter 3 we presented Spike, a state of the art simulator for spiking neural network modelling, which provides at minimum a two-fold speed increase over contemporary GPU based simulators and orders of magnitude speed increases over contemporary CPU based simulators. This was accomplished by organising synaptic connections in an efficient manner and by relaxing the constraint that network updates need be made in a synchronised fashion. Instead, we use the minimum axonal delay in a network to determine the timescale on which the network dynamics must be synchronised. Such an optimisations was previously proposed in the context of cluster-based simulation of spiking networks [131], however we found a use for this optimisation to address multiple bottlenecks in GPU computation. These optimisations enable significant speed benefits and show GPU-based simulation software as highly efficient for current and future spiking neural network simulation. Increases in simulation speed enable faster parameter searches and allow a greater throughput of network training and testing.

One significant drawback of the Spike simulator is its rigidity. In order to introduce novel neuron, synapse, or plasticity types to the software, knowledge of the codebase is required. This requires familiarity not just with the current Spike software but also with the C++ and CUDA programming languages. These requirements introduce barriers for the development of novel models in this simulator.

Ultimately, Spike provides speed at the cost of flexibility. Other simulators, such as Brian2, compromise upon speed to provide flexibility. Rigid high-speed and more flexible low-speed simulators have coexisted in the field for some years, for example the Auryn, NEST, and Brian simulators [51, 223, 192]. We therefore expect a similar coexistence of high and low-speed GPU based simulators, so long as there exists a reasonable trade-off between speed and flexibility.

Projects such as Brian2GeNN [191] are another key development in this space. This project brings some of the strengths of GPU based programming, via the GeNN simulator, to the Brian2 simulator. The flexibility of a simulator such as Brian2 is due to its ability to execute arbitrary neuron dynamics. These dynamics can be specified in text before the beginning of the simulation and do not require the user to understand the simulator codebase. Furthermore, the Brian2 simulator is written for execution via the Python programming language which is significantly higher level than C++/CUDA, as are used by GeNN and Spike. The Brian2GeNN project allows users to describe the desired neural network dynamics in Brian2 and to then have these dynamics computed on a GPU through background interaction with the GeNN simulator. Though this project is still in its infancy, such a hybrid approach could provide some benefits of both speed and flexibility. Nonetheless, such an approach is still slower in computational time than the Spike simulator. The question remains as to whether rigid simulators such as Spike and Auryn might be supplanted in future by hybrid simulators which allow flexible code description and high speed simulation.

The future of spiking neural network simulation also has the potential to follow a non-traditional computing avenue. Neuromorphic computing devices are such an alternative and have been proposed to achieve high speed spiking neural network simulation. These devices use highly optimised non-traditional computing architectures in order to simulate spiking neural network models. By optimisation of the hardware and the software, rather than just the software, there is huge potential for increased efficiency and speed. Some examples of neuromorphic computers include methods which utilize tens of thousands of interconnected mobile computing cores, such as SpiNNaker [103], or methods which directly produce analogue computing components in silicon which are equivalent to neuron nodes, such as memristors or the in-silicon BrainScales project [123].

Neuromorphic approaches show a great deal of promise and are highly likely to surpass the speed of any general purpose computing based approaches to spiking

network simulation once they reach maturity. However, the timeline upon which this will happen is unclear and the availability and suitability of these devices is yet to be determined. Furthermore, though these devices are often touted as highly energy efficient – in particular the power draw for these devices is often smaller than that of general purpose computing systems – some recent studies suggest that the efficiency of a subset of neuromorphic chips, when measured in watts per synaptic event, is lower than that of GPU based simulators [107]. Given this data, the potentially long runway for some of these devices to be complete, and the difficulty of production and distribution of non-traditional systems, it seems unlikely that simulation through general purpose computing will be replaced in the near term. Furthermore, purchase of such devices in place of a general purpose computer introduces a question of the trade-off between increased speed of simulation time (with a neuromorphic computer) vs the flexibility of general purpose computers for modelling novel neural network dynamics. To mirror our above description of spiking neural network simulator co-existence, general purpose hardware and neuromorphic approaches are likely to co-exist as long as there remains an acceptable trade-off between speed and flexibility.

Having participated in developing tools to enable high speed simulation, we thereafter proposed a robust unsupervised spike timing-dependent learning rule arrangement for spiking neural networks. We develop a novel excitatory learning rule, as presented in Chapter 4, with the aim of producing competitive learning whilst maintaining firing rate homeostasis and excitatory-inhibitory balance in spiking neural networks. The combination of excitatory and inhibitory plasticity rules has been explored in the context of unsupervised competitive learning in rate coded and hybrid rate-spiking networks. Foldiak [58] produced one of the earliest such demonstrations in which Hebbian excitatory plasticity on the feed-forward input weights was combined with anti-Hebbian plasticity on lateral synaptic connections in order to produce networks in which individual rate coded neurons represented

different principle components of the dataset which was presented as inputs. Later, hybrid rate and spiking networks were produced by Zylberberg et al. [231] and King et al. [105] which provided sparse coding and the emergence of V1 simple cell-like receptive fields in networks which combine excitatory and inhibitory learning rules. Similarly, Pehlevan [151] recently presented a rate based excitatory and inhibitory learning rule arrangement (applied in spiking neural network models) which produces sparse coding, motivated by nonnegative similarity matching algorithms.

However, these examples all apply rate based plasticity on some subset if not upon all synaptic connections. Furthermore, when trained upon datasets of natural images these approaches use current based inputs through synaptic connections which switch between providing positive or negative currents to post-synaptic neurons depending upon the input stimulus chosen. Thus Dale's law, the separation of excitatory and inhibitory neurons and synapses, is broken within these inputs.

Approaches in "pure" spiking neural network modelling literature which combine multiple learning rules are less common. One example is that by Zenke et al. [221] in which four different mechanisms for plasticity were combined with neuron adaptation. The resulting system of learning rules provided multiple stability points for synaptic weights. This system could also learn a range of input patterns, though these input stimuli were all drawn from an artificial dataset and the learning procedure was not demonstrated on ecological stimuli. Furthermore, the system of learning rules and dynamics were highly complex and were required to be so in order to produce the multiple stable points for synaptic weights. Aside from the troubling complexity of the system and the impoverished training set, this study did highlight potential roles for the many interacting mechanisms of synaptic weight change.

Other attempts to produce models of visual processing with spiking neural networks have abstracted away many of the dynamics. For example, the approach used by Masquelier et al. [122] made use of neurons which spiked only once, with an immediate "winner take all" global inhibition stopping the action potentials of

other neurons, and a timing dependent plasticity rule which was qualitatively very different to the STDP curves observed in electrophysiological studies.

Our proposed novel excitatory learning rule, described in detail in Chapter 4, addresses many of these drawbacks and limitations. It was designed to complement the inhibitory synaptic plasticity rule of Vogels et al. [201] by providing competitive learning alongside the excitatory-inhibitory balance which the inhibitory learning rule produces. These learning rules are both spiking-timing dependent but it can be shown that these learning rules implicitly produce modification of synaptic weights based upon the firing rate of the post-synaptic neuron. In particular, our excitatory learning rule potentiates synaptic connections when their activation reliably elicits higher than threshold post-synaptic neuron firing. Otherwise, synaptic connections are depressed with a hard bound at zero weight. This excitatory learning rule therefore achieved multiple weight set-points, as were achieved by Zenke et al. [221], through two interacting plasticity rules and without additional mechanisms necessary for neuron spike adaptation. This potentiation of synaptic connections based upon a threshold firing rate can be compared to other methods which use a threshold, such as the Bienenstock Cooper Munro (BCM) rule [20]. In the case of the BCM rule, potentiation is based upon the neuron firing rate relative to a sliding threshold. This sliding threshold tracks the average neuron firing rate (or some power of this value) and weights are only potentiated if the neuron fires at a rate greater than this threshold, otherwise weights are depressed. In comparison, our learning rule arrangement uses a fixed threshold to which the inhibitory plasticity rule "slides" the neural activity and the excitatory plasticity rule potentiates the subset of weights which correlate with the neuron exceeding this threshold, and otherwise depresses weights. The similarity between our learning rule combination and sliding threshold based approaches is coherent given this description but is only shown under limited assumptions in Section 4.3.3.2. We leave more mathematically rigorous comparisons for future studies.

In Chapters 4 and 5, the proposed excitatory/inhibitory plasticity rule arrangement was implemented to train networks upon patches extracted from a dataset of natural images. These network developed V1 simple cell-like receptive fields through this training procedure, exhibiting the ability for our learning rule arrangement to learn features of an ecological training set. This is, to our knowledge, the first example of an all spiking neural network model (spiking inputs and outputs) uses realistic, local only, and spike timing-dependent learning rules which is capable of producing V1 simple cell-like receptive fields. In this chapter, model details were kept as close to observed neurobiology as possible, for example by employing traditional STDP curve shapes, while simplifying dynamics as much as possible. In particular, this provided an exhibition of the ability to maintain both an average homeostatic firing rate and to simultaneously produce competitive learning. Our novel excitatory learning rule was shown to be crucial to the formation of competitive learning in comparisons against traditional learning rules.

The production of competition and homeostasis was first described in an inhibitory only neuron network in Chapter 4. This work acted to illustrate the principles of the proposed learning rule arrangement and also replicated features of prior sparse coding models in a single layer spiking neural network model. Prior sparse coding models most commonly implemented a single layer of non-Dalean neurons which were optimised (sometimes implicitly) as autoencoders, developing features which can be used to robustly reproduce the input stimulation, with an additional constraint on the number of active neurons. Our inhibitory only neuron network reproduced these behaviours, with neurons producing highly sparse responses and an ability to reconstruct an input image from the network responses. This ability to perform reconstruction was an exhibition of the ability for input information to be encoded in the neural responses.

Though we were able to exhibit competitive learning, this network operates in a state of extreme competition, a single neuron on average only being highly

active at a time. Given the theoretical proposals that a sparse distributed encoding in neural circuits is an optimal balance between this local coding and a dense distributed coding [59], it is clear that there is still some work to be done in determining how the level of competition could be adjusted in such a network setup. Ultimately, our inhibitory only neuron circuit is only an approximation of the neural circuits which we intended to model.

In Chapter 5, we extended the competitive spiking neural network model explored in Chapter 4 to a circuit of both excitatory and inhibitory neurons. The inclusion of separate excitatory and inhibitory neuron populations to a sparse coding model has been explored in the past [105, 226]. We produced a similar network structure, though with spiking dynamics for all neurons, input stimulation, and for all learning rules. Interestingly, after addressing some issues of convergence of the inhibitory synaptic weights, we were able to modulate the level of competition amongst the excitatory neurons by modifying the strength of their projections to the inhibitory neuron population. The network setup produced was also capable of reproducing the electrophysiological observation that excitatory neuron receptive fields are generally smaller than those of inhibitory neurons. This investigation was preliminary and proposes some interesting issues for combining existing excitatory and inhibitory neurons in a network model. We closed this chapter by proposing investigations into novel architectures and connectivities in networks of excitatory and inhibitory neurons and investigations into alternative forms for the inhibitory plasticity rule. This is a large space of possible architectures and parameters which require an extended investigation.

This thesis has focussed on a research journey from rate-based to spiking neural network models. The efficacy of a rate-based competitive neural network was proven in Chapter 2. In order to enable the transition to spiking neural network models, two key bottlenecks were identified: simulation speed, and learning rule reliability. Chapter 3 proposed a state of the art simulator which provides significant speed-ups

over contemporary simulators. Chapters 4 and 5 then focussed upon developing and testing learning rules which could robustly learn from datasets of ecological stimuli while maintaining biological realism and plausibility as much as possible. Given the completion of this arc, the question arises as to the benefits of such a transition from rate-based to spiking neural networks provides.

Two key benefits of this transition are: insights into potential mechanistic implementations of high-level theoretical principles, and the ability to interrogate and reproduce models of phenomena on the scale of individual spikes. First, by proposing specific mechanistic implementations for procedures such as competitive learning we can provide some predictions which could be measuring in electrophysiological studies, such as the shapes of excitatory learning rules and their interaction with inhibitory learning rules. Beyond these predictions, the failure modes of our proposed learning rules also bring up new issues to be highlighted and investigated both in modelling studies and in electrophysiological studies.

Aside from predictions and mechanisms, more effective training of spiking neural networks allows an investigation into any benefits of spike-based computation over traditional rate-based computation. A commonly held perspective is that the emission of action potentials by neurons is merely an effective mechanism by which information can be propagated between nerve cells, especially across significant distances [43]. However, the use of spike timing has been theoretically proposed as beneficial for both information coding capacity but also for novel computations which are not possible in traditional rate-based networks [96, 117, 45, 50, 167]. Beyond these theoretical proposals, electrophysiological studies have measured information in the timings of emitted spikes rather than just in the rates of neural responses [80, 215, 208, 29], though currently the evidence for such temporal encoding is strongest in sub-cortical areas [109, 155, 15]. These studies on spike timing have been debated, with strong pushback on the basis of the high degree of neuron response variability and the extent to which noise affects neural

responses [116, 193]. By producing spiking neural network models which can be efficiently trained, we can enable our ability to address such questions and broaden our investigations into the functions of real neural circuits.

A

Spiking Neural Network Benchmark Models

A.0.1 Vogels-Abbott Benchmark Model

The Vogels-Abbott benchmark is based upon a reduced scale version of the network presented in the [202] publication. This reduced scale version is detailed as a benchmark in a paper by Zenke et al. [223].

The network consists of 3200 excitatory and 800 inhibitory Leaky Integrate and Fire (LIF) neurons. Conductance based synaptic connections project between these populations of excitatory and inhibitory neurons and the network is driven by constant background stimulation. Individual neuron dynamics can be described;

$$C_{\text{mem}} \cdot \frac{dV}{dt} = g_{\text{leak}}(V_{\text{rest}} - V) + g_{\text{ex}}(E_{\text{ex}} - V) + g_{\text{in}}(E_{\text{in}} - V) + I_{\text{bg}}$$

where V is the neuron membrane voltage, C_{mem} is the membrane capacitance, g_{leak} is the membrane leakage conductance and V_{rest} is the cell resting potential. Synaptic inputs are governed by the excitatory and inhibitory conductances, g_{ex} and g_{in} respectively, and the reversal potentials, E_{ex} and E_{in} respectively. Finally, cells are stimulated by a constant input I_{bg}.

If the membrane potential of a cell reaches a threshold V_{thresh}, it emits an action potential (or spike) and is thereafter brought back to the reset potential V_{reset} for a period of time equal to the refractory period of the cell τ_{ref}. After this refractory period, the cell dynamics are allowed to continue.

Synaptic connections update the excitatory and inhibitory synaptic conductances. A pre-synaptic spike on a synaptic connection causes a discontinous jump in the corresponding post-synaptic cell synaptic conductance. For excitatory synaptic connections, a pre-synaptic spike causes a jump in the post-synaptic neuron excitatory synaptic conductance, after a time dictated by the axonal delay, such that $g_{\text{ex}} \leftarrow g_{\text{ex}} + w_{\text{ex}} \cdot g_{\text{leak}}$ where w_{ex} is the weight of the excitatory synaptic connection. Similarly, inhibitory synaptic connections cause a discontinous jump in the inhibitory synaptic conductances of post-synaptic cells such that $g_{\text{in}} \leftarrow g_{\text{in}} + w_{\text{in}} \cdot g_{\text{leak}}$ where w_{in} is the weight of the synaptic connection. Finally, the cell synaptic conductances undergo dynamics;

$$\tau_{\text{ex}} \frac{dg_{\text{ex}}}{dt} = -g_{\text{ex}} \qquad \text{and} \qquad \tau_{\text{in}} \frac{dg_{\text{in}}}{dt} = -g_{\text{in}}$$

The parameters used for this model are detailed in Table A.1.

Network	-
Number of Excitatory (E) Neurons	3200
Number of Inhibitory (I) Neurons	800
Probability of Connection E->E	2%
Probability of Connection E->I	2%
Probability of Connection I->E	2%
Probability of Connection I->I	2%
Numerical Timestep δt	0.1ms
Neuron Parameters	-
τ_{mem}	20ms
τ_{ref}	5ms
V_{rest}	-60mV
V_{thresh}	-50mV
V_{reset}	-60mV
E_{ex}	0mV
E_{in}	-80mV
I_{bg}	20mV
Synapse Parameters	-
τ_{ex}	5ms
τ_{in}	10ms
w_{ex}	0.4
w_{in}	5.1
Axonal Delay	0.8ms

Table A.1: **Network, Neuron, and Synaptic Parameters used for the Vogels-Abbott Benchmark**

A.0.2 Brunel Benchmark Model

The Brunel benchmark is based upon a network adapted by Zenke et al. [223], published in its original form by Brunel et al. [27].

The network consists of 8000 excitatory and 2000 inhibitory Leaky Integrate and Fire (LIF) neurons. Synaptic connections are voltage-based and exist between the populations of excitatory and inhibitory neurons and from a population of input neurons. The network is driven by 10,000 excitatory input neurons each with a 20Hz firing rate. Individual neuron dynamics can be described;

$$\tau_{\text{mem}} \cdot \frac{dV}{dt} = (V_{\text{rest}} - V)$$

where V is the neuron membrane voltage, τ_{mem} is the membrane time constant and V_{rest} is the cell resting potential. Synaptic inputs are governed by the excitatory and inhibitory voltage injections. For excitatory or inhibitory synaptic connections, following a pre-synaptic neuron spike (and after awaiting any axonal delay), the voltage is directly modified in a discontinuous fashion such that $V \leftarrow V + w$ where w is the weight of the synaptic connection. All excitatory synaptic connections are initialised with a weight w_{ex} and all inhibitory synaptic connections are initialised with a weight w_{in}.

If the membrane potential of a cell reaches a threshold V_{thresh}, it emits an action potential (or spike) and is thereafter brought back to the reset potential V_{reset} for a period of time equal to the refractory period of the cell τ_{ref}. After this refractory period, the cell dynamics are allowed to continue.

The parameters used for this model are detailed in the Table A.2.

A.0.2.1 Plasticity

The Brunel benchmark can be run with or without a plasticity rule active. When active, the plasticity rule is a Spike-Timing Dependent Plasticity (STDP) rule which updates synaptic connections upon pre and post-synaptic neuron action potentials.

The STDP rule applied in this model is a weight dependent STDP rule. For each synapse, we have two traces – one for its pre-synaptic neuron and one for its post-synaptic neuron, z_{pre} and z_{post}. Upon a pre-synaptic spike, we wait for a duration equal to the axonal delay, following which we update the pre-synaptic trace such that; $z_{\text{pre}} \leftarrow z_{\text{pre}} + 1.0$. Upon post-synaptic spikes, the post-synaptic trace is updated such that; $z_{\text{post}} \leftarrow z_{\text{post}} + 1.0$. These traces both decay with the dynamics that follow.

$$\tau_{\text{stdp}} \frac{dz_{\text{pre}}}{dt} = -z_{\text{pre}} \qquad \text{and} \qquad \tau_{\text{stdp}} \frac{dz_{\text{post}}}{dt} = -z_{\text{post}}$$

Long Term Depression (LTD) is implemented as follows. Upon a pre-synaptic

Network	-
Number of Excitatory (E) Neurons	8000
Number of Inhibitory (I) Neurons	2000
Number of Excitatory Poisson Input Neurons (P)	10000
Probability of Connection E->E	10%
Probability of Connection E->I	10%
Probability of Connection I->E	10%
Probability of Connection I->I	10%
Probability of Connection P->E	10%
Probability of Connection P->I	10%
Numerical Timestep δt	0.1ms
Neuron Parameters	-
τ_{mem}	20ms
τ_{ref}	2ms
V_{rest}	0mV
V_{thresh}	20mV
V_{reset}	0mV
Poisson Neuron Firing Rate	20Hz
Synapse Parameters	-
w_{ex}	0.1mV
w_{in}	-0.5mV
Axonal Delay	1.5ms

Table A.2: **Network, Neuron, and Synaptic Parameters used for the Brunel Benchmarks**

spike (after awaiting the axonal delay), the weight w of the synaptic connection is also updated;

$$w \leftarrow w - \alpha\lambda w \cdot exp^{-\frac{z_{\text{post}}}{\delta t}}$$

where λ is the learning rate, and α is a scaling factor which is only applied to LTD (thus dictating the relative strength of LTD).

Long Term Potentiation (LTP) is implemented as follows. Upon a post-synaptic spike, the weight w of the synaptic connection is also updated;

$$w \leftarrow w + \lambda(1.0 - w) \cdot exp^{-\frac{z_{pre}}{\delta t}}$$

Plasticity Parameters	-
τ_{stdp}	20ms
α	2.02
λ	0.01
w_{max}	0.3mV

Table A.3: **Network, Neuron, and Synaptic Parameters used for the Brunel Benchmarks**

The plastic synaptic connections also have a hard bound upon their weight such that if the weight w increases above a maximum weight value w_{max}, it is bounded and $w \leftarrow w_{max}$. Similarly, if the weight w reduces below zero, it is bound and set to zero; $w \leftarrow 0.0$.

This STDP rule is only applied to the excitatory to excitatory ($E- > E$) synaptic connections in this model. All parameter values are presented in Table A.3.

Methods For Updating Plasticity

Two methods of carrying out STDP are implemented in Chapter 3. These methods correspond to either updating the STDP traces and synaptic weights whenever a spike occurs, event-driven, or by continual solving of the STDP traces and modification of weight values upon the arrival of spikes. These methods are described in detail here.

Event Driven As described above, synaptic traces z_{pre} and z_{post} can be calculated in order to compute STDP based updates. The dynamics which describe the decay of the synaptic traces can be analytically solved, which allows a calculation of a trace at time t after some time t_0, when the synaptic trace was equal to z_0,as;

$$z(t) = z(t_0) \cdot \exp^{-(t-t_0)/\tau_{stdp}} .$$

This allows STDP rules to be calculated efficiently by only calculating and updating the values of synaptic traces at times which correspond to pre and post-

synaptic spikes. Upon every post-synaptic spike, you can calculate the current value of the pre-synaptic trace in order to calculate the weight change required. Following that, the current value of the post-synaptic trace can be updated (due to the spike which has arrived) and the times of these updates can be stored. Similarly, every pre-synaptic spike requires a calculation of the current value of the post-synaptic trace for the weight update, followed by an update to the pre-synaptic trace for future weight changes.

Continuous Updating Continuous updating as we refer to it here is a method of calculating the synaptic trace values upon every timestep in a similar fashion to which membrane voltages of individual cells are updated every timestep. Such updating is less efficient than event driven updates however it does allow arbitrary updating rules to be used for synaptic traces and does not limit their dynamics to analytically solvable functions.

Continuous updating means that every timestep (δt), the synaptic trace is updated using a forward Euler solver;

$$z(t + \delta t) = z(t) + \delta t \cdot \frac{dz}{dt}$$

Upon pre, or post-synaptic spikes, the traces can be incremented as desired and these traces are thereby always kept up to date without requirements for future projection. This is however far more computationally expensive an approach to plasticity, bringing the dynamics of a plastic synapse close in complexity to those of a neuron. On balance, such an approach is also much more flexible allowing arbitrary dynamics in the plasticity rule.

B

Proposed Competitive Spiking Neural Network Validation and Tools

B.0.1 Equivalence of Single Neuron and Multi-Neuron Input Source Simulations

In Chapter 4, we trained networks with a simple artificial stimulus set as described in Section 4.3.4.1. This simple stimulus set consisted of a set of 15 Poisson spiking input neurons which each represented a single input sources. These input sources were randomly activated during training and presented individually for testing. For every presentation of a stimulus, whether in training or testing, the total summed firing rate across all input neurons was normalized to 500Hz. During training, this meant that the average neuron firing rate per neuron was reasonable, at 33.3Hz. However, the use of a single neuron per input source meant a high firing rate during

testing, 500Hz for the active source, which is an extreme network input activation.

Use of a single neuron per input source made the visualization and explanation of network dynamics and learning very straightforward. Furthermore, an ecological stimulus set was later used and showed agreement.

Nonetheless, we show here an equivalence of our simple artificial stimulus set in a more reasonable firing rate regime. Here, each of the 15 input sources are represented instead by five Poisson firing input neurons. During training and testing all five input neurons representing a given source all have the same exact firing rate response, though this firing rate changes depending upon how active we wish to make the input source. Thus, the neurons for each input source are always perfectly correlated in their firing rate responses.

Training involves each a random mixture of the input sources such that each input source is assigned a firing rate and all five neurons representing that input source are active at that firing rate. The total summed firing rate across all 75 input neurons is normalized to 500Hz, such that in this setup each input neuron has a firing rate average of 6.67Hz during training. During testing, each input source is activated independently (with all other input sources set to zero firing rate) and all five input neurons which represent the active input source are active with firing rate of 100Hz.

The results of this training procedure are shown in Figure B.1. Comparison of the results shown in this figure match closely (though not identically) the results presented in Figure 4.5.

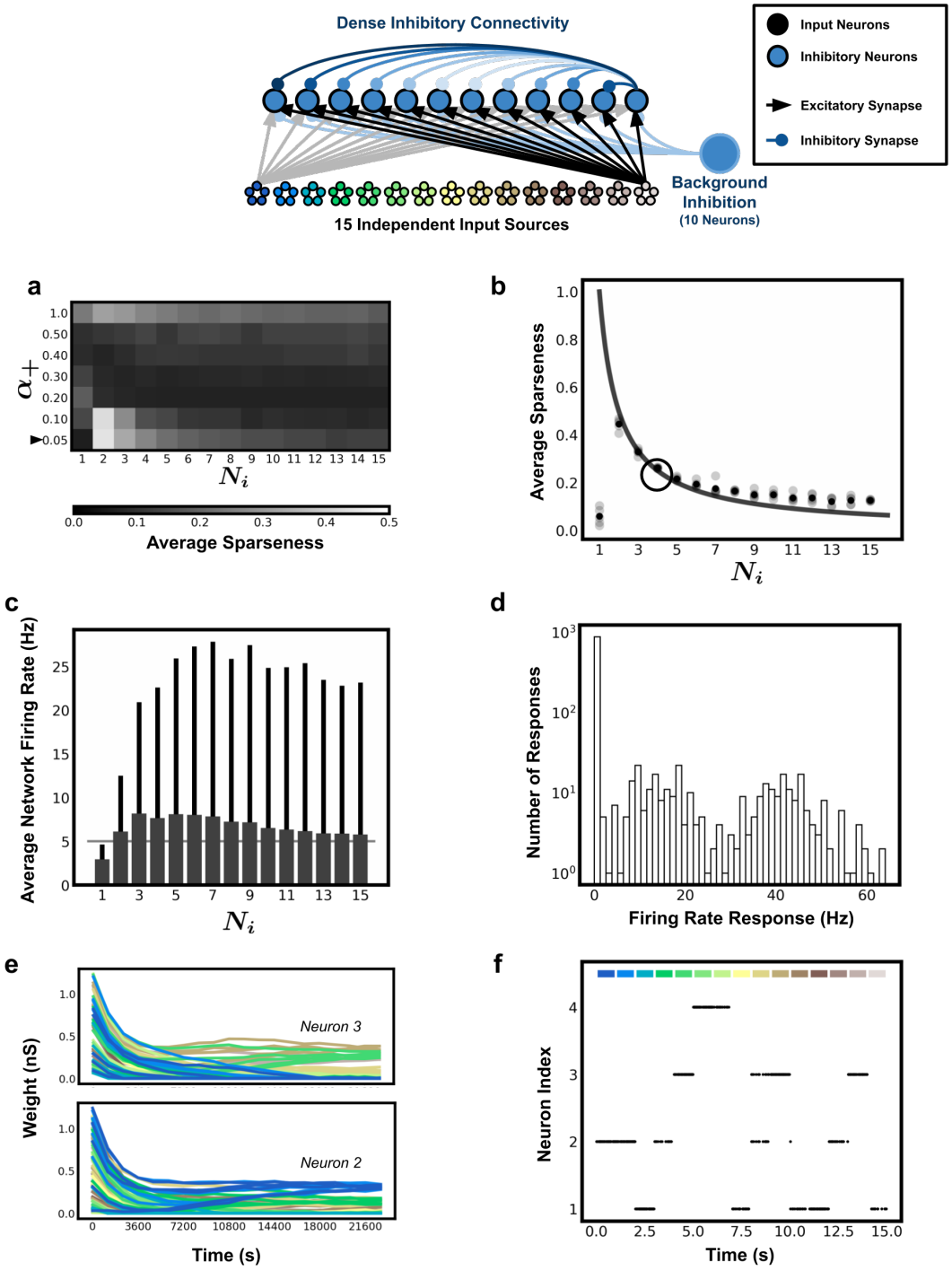

Figure B.1: **Reproducing competitive inhibitory neuron network behaviour with multiple neurons per input source. Compare to Figure 4.5** *Top,* the network structure simulated to produce the results shown. The network consists of inhibitory only neurons with dense lateral inhibitory connectivity. These cells are excited by a set of 15 independent input sources, each of which are represented by five neurons producing a total of 75 input neurons. These input neurons are fully connected to all inhibitory neurons. The custom eSTDP rule is

active upon all feedforward excitatory connections and the iSTDP rule active on lateral inhibitory connections. An external group of inhibitory neurons provide background inhibition. These neurons fire consistently at 100Hz firing rate and the iSTDP rule is active upon their inhibitory synaptic connections. **(a)** The average sparseness of trained networks over a range of parameters. Network training is repeated for 5 different network random seeds. The greyscale colouring of this parameter map indicates average network sparseness. The y-axis presents variation of a key parameter of the eSTDP rule ,a_+, and the x-axis indicates change to the network size, N_i. **(b)** Data from a row in the parameter map – indicated by a black arrow in plot (a), $\alpha_+ = 0.005$ – is shown plotted alongside the theoretical expectation, presented earlier in Figure 4.1. The theoretical expectation is plot as a line (red), and the simulation results are presented as black points. **(c)** The average firing rates of the networks in plot *b* are shown with error bars showing the standard deviation in this firing rate. A grey horizontal line shows the location of the target firing rate λ_{target}, 5Hz. **(d)** A histogram of all firing rate responses of a specific network (circled network in plot (b)). This network consists of 4 competing inhibitory neurons. **(e)** The weight evolution of input weights to neurons 1 (upper) and neuron 4 (lower) in the circled network of plot (b). **(f)** A raster plot showing the spike times of all four neurons in the selected network (circled in plot (b)). This raster plot shows the presentation of our ten input stimulation sources, coloured bars top, which are activated independently and the neuron responses are shown below. For clarity, the plotted bars are coloured to correspond with the input source colouring in the network diagram top.

B.0.2 Whitening Process Applied to Natural Images

Below is presented the Python code used to take an input image and to produce the whitened and meaned output version of that image. It is from these whitened and meaned images which 16x16 pixel patches are extracted in order to train the models of early visual cortex receptive field formation. This whitening process was produced in order to precisely mimic the MATLAB code made available by Zylberberg et al [231].

```
import numpy as np
from scipy import misc, fftpack

# Opening a 512x512 pixel image
scene = misc.imread("IMAGE.jpg")
# Converting a given colour image to grayscale
```

```
scene_gray = 0.2126*scene[:,:,0] + 0.7152*scene[:,:,1] + 0.0722*
    ↪ scene[:,:,2]

# Producing the required filter
N = 512
fx, fy = np.meshgrid(np.arange(N) - N/2,np.arange(N) - N/2)
rho=np.sqrt(fx*fx+fy*fy);
f_0=0.4*N
filt=rho*np.exp(-(rho/f_0)**4);

# Carrying out a two-dimensional spatial Fourier transform upon
    ↪ the image
If = fftpack.fft2(scene_gray)
# Filtering the Fourier transform, and carrying out an inverse
    ↪ transform to re-produce the original image
filteredim = np.real(fftpack.ifft2(If*np.fft.fftshift(filt)))

# Meaning and setting the variance of the resulting image to one
filteredim -= np.mean(filteredim)
filteredim /= np.std(filteredim)
```

B.0.3 Spike-Triggered Averaging

In order to decode the receptive field tunings of spiking neurons in both experimental and modelling studies, a commonly used approach is Spike-Triggered Averaging (STA). In short, the spiking response of a neuron to a stream of input stimulation is collected for some time. Following this, the input stimulation which preceded each action potential produced by the neuron of interest is collected and these are all averaged to form an approximation of its receptive field.

In the chapters of this thesis, we apply this method to decode the receptive fields of spiking neurons in models of early visual processing. All such STA analyses are carried out without any synaptic plasticity upon networks after they have been trained. In these cases, data is collected by setting the firing rates of all input neurons to 30Hz (based upon which random Poisson sampled spike times are produced) and thereafter our networks are simulated for tens of thousands of seconds. By giving all input neurons the same firing rate and allowing them

to randomly draw spikes based upon this firing rate, we ensure that the input stimulation is random and does not bias the decoded the receptive field.

Following such simulation, we collect all input neuron and output neuron spike times. For every output neuron spike, we count the number of spikes produced by each input neuron in a 10ms window prior to this output spike. We collect this count for every spike of our chosen output neuron. These are then averaged and used produce a per input neuron weighting. This forms a vector of length the number of input neurons with values which indicate the average number of spikes which a given input neuron emits prior to an output neuron spike.

We can express this process mathematically. Take a set of N input neurons, indexed n, with spike times t_n^f, where f indexes the individual spike times of neuron n of N. Given these spike times we can construct a spike train, S_n, for every neuron such that,

$$S_n(t) = \sum_f \delta(t - t_n^f),$$

where $\delta(\cdot)$, is the Dirac delta function.

Similarly, take a single output neuron with spike times t_{out}^f, where f indexes the output neuron spikes up to a maximum of F spikes. The average contribution to this output cell's receptive field from a particular input neuron n, W_n, is then,

$$W_n = \frac{1}{F} \sum_{f=1}^{F} \int_{t_{out}^f - d}^{t_{out}^f} S_n(t),$$

where d is the duration of the time window over which we collect the contributions to the output neuron (10ms). The set of values W_n thereafter represent the average number of spikes from input neuron n which occur in a time window d prior to the output neuron action potentials. Across all N neurons, these values provide a measure of the receptive field of our output neuron relative to the input neuron space.

B.0.4 STA Decoded Receptive Field Tunings for Trained Inhibitory Only Neuron Networks

In Section 4.4.1.2 we described the receptive field tunings produced in a network of densely connected inhibitory neurons receiving feedforward excitation from a simulated visual input. See Chapter 4 for details.

Figure 4.6d shows receptive field tunings of neurons in the proposed inhibitory neuron network after training. These receptive fields were plot based upon the weight vector inputs to the neurons in the network, however the true receptive field tunings are also affected by the lateral inhibitory input from neighbouring neurons.

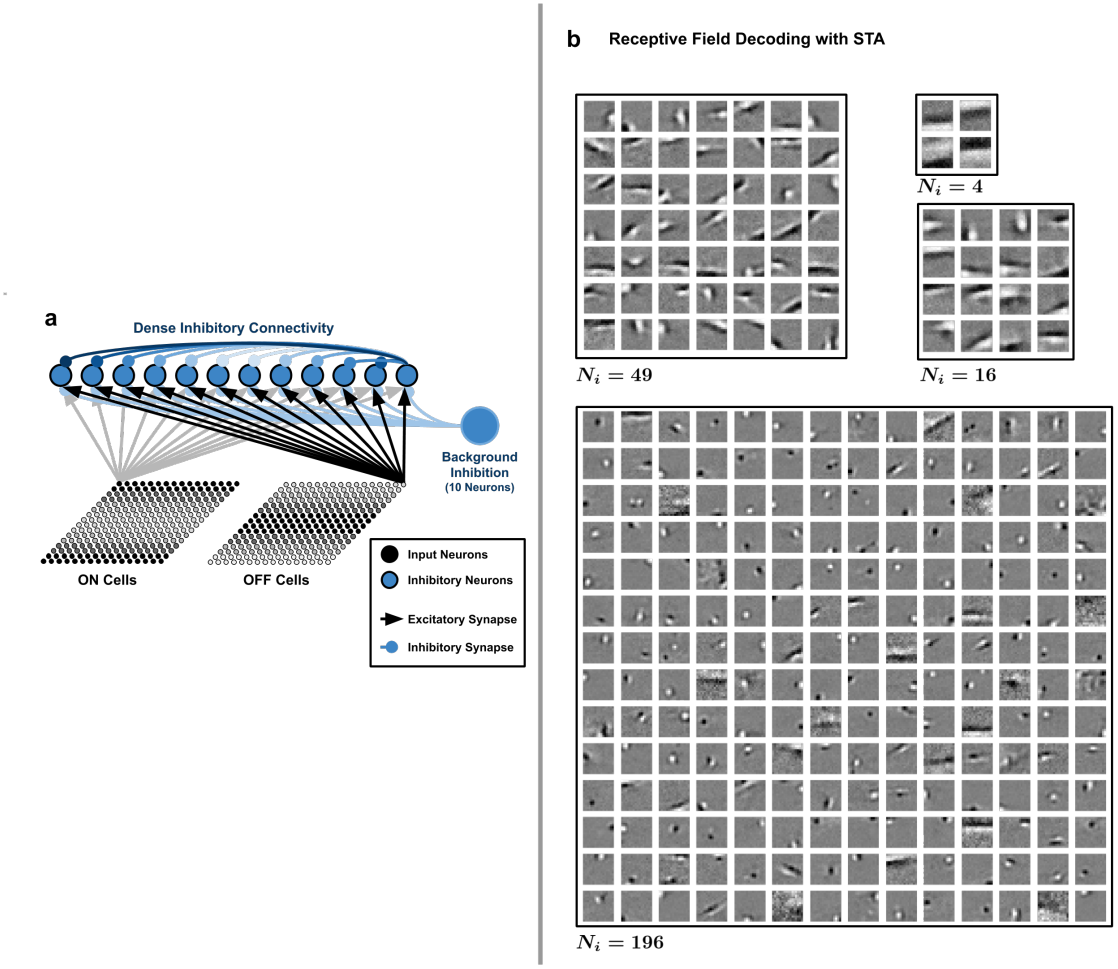

Figure B.2: **Results of training dense competitive networks of inhibitory**

neurons with patches from natural images. (a) A depiction of the network structure. Input is provided by two 16x16 input sources which represent the ON and OFF cell responses of retinal ganglion cells to a single 16x16 image patch (see Methods Section 4.3.4.2). These input populations provide excitatory input to a layer of densely connected inhibitory neurons. Here we analyse the trained networks presented in Figure 4.6 such that every input neuron is consistently active, producing Poisson distributed spikes at a rate of 25Hz. These drive the output neurons which are also affected by the lateral inhibitory connections. The neurons are stimulated for 2000s and the spike times of input and output neurons collected. **(b)** The receptive field tunings of neurons in the same trained networks as described in Figure 4.6 decoded using Spike-Triggered Averaging, see Section B.0.3. Network sizes are presented below the receptive field grids, $N_{\mathrm{i}} = 4, 16, 49, 196$.
A

Figure B.2, shows the receptive field tunings of these same neurons (organised as in Figure 4.6d), when decoded using the Spike-Triggered Averaging protocol described in Section B.0.3. These can be qualitatively observed to be very similar to the weight vector decoded receptive field tunings but with greater levels of noise. Neurons were stimulated with 25Hz random Poisson input from all input neurons for 2000 seconds in order to produce these receptive fields.

Bibliography

[1] R Abbasi-Asl et al. “Do retinal ganglion cells project natural scenes to their principal subspace and whiten them?” In: *2016 50th Asilomar Conference on Signals, Systems and Computers*. Nov. 2016, pp. 1641–1645.

[2] W C Abraham et al. “Heterosynaptic metaplasticity in the hippocampus in vivo: a BCM-like modifiable threshold for LTP”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 98.19 (Sept. 2001), pp. 10924–10929.

[3] Nasir Ahmad et al. “Harmonic Training and the Formation of Pitch Representation in a Neural Network Model of the Auditory Brain”. en. In: *Front. Comput. Neurosci.* 10 (Mar. 2016), p. 24.

[4] Katrin Amunts and Karl Zilles. “Architectonic Mapping of the Human Brain beyond Brodmann”. en. In: *Neuron* 88.6 (Dec. 2015), pp. 1086–1107.

[5] J S Anderson, M Carandini, and D Ferster. “Orientation tuning of input conductance, excitation, and inhibition in cat primary visual cortex”. en. In: *J. Neurophysiol.* 84.2 (Aug. 2000), pp. 909–926.

[6] A Artola and W Singer. “Long-term depression of excitatory synaptic transmission and its relationship to long-term potentiation”. en. In: *Trends Neurosci.* 16.11 (Nov. 1993), pp. 480–487.

[7] R Baddeley et al. “Responses of neurons in primary and inferior temporal visual cortices to natural scenes”. en. In: *Proc. Biol. Sci.* 264.1389 (Dec. 1997), pp. 1775–1783.

[8] H Barlow. “Redundancy reduction revisited”. In: *Network: Computation in Neural Systems* 12.3 (Jan. 2001), pp. 241–253.

[9] H B Barlow. “Possible principles underlying the transformations of sensory messages”. In: *Sensory Communication*. Ed. by W A Rosenblith. MIT Press, 1961, pp. 217–234.

[10] Andre M Bastos et al. “Canonical microcircuits for predictive coding”. en. In: *Neuron* 76.4 (Nov. 2012), pp. 695–711.

[11] Marina Bedny et al. “Language processing in the occipital cortex of congenitally blind adults”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 108.11 (Mar. 2011), pp. 4429–4434.

[12] Anthony J Bell and Terrence J Sejnowski. “The “independent components” of natural scenes are edge filters”. In: *Vision Res.* 37.23 (Dec. 1997), pp. 3327–3338.

[13] Alberto Bernacchia and Xiao-Jing Wang. “Decorrelation by recurrent inhibition in heterogeneous neural circuits”. en. In: *Neural Comput.* 25.7 (July 2013), pp. 1732–1767.

[14] Joshua G Bernstein and Andrew J Oxenham. “Pitch discrimination of diotic and dichotic tone complexes: Harmonic resolvability or harmonic number?” In: *The Journal of the Acoustical Society of America* 113.6 (2003), pp. 3323–3334.

[15] M J Berry, D K Warland, and M Meister. “The structure and precision of retinal spike trains”. en. In: *Proc. Natl. Acad. Sci. U. S. A.* 94.10 (May 1997), pp. 5411–5416.

[16] Sarah F Beul and Claus C Hilgetag. “Towards a “canonical” agranular cortical microcircuit”. en. In: *Front. Neuroanat.* 8 (2014), p. 165.

[17] M Beyeler et al. "CARLsim 3: A user-friendly and highly optimized library for the creation of neurobiologically detailed spiking neural networks". In: *2015 International Joint Conference on Neural Networks (IJCNN)*. ieeexplore.ieee.org, July 2015, pp. 1–8.

[18] Michael Beyeler et al. "Sparse coding and dimensionality reduction in cortex". en. In: *bioRxiv* (June 2017), p. 149880.

[19] G Q Bi and M M Poo. "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type". en. In: *J. Neurosci.* 18.24 (Dec. 1998), pp. 10464–10472.

[20] E L Bienenstock, L N Cooper, and P W Munro. "Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex". en. In: *J. Neurosci.* 2.1 (Jan. 1982), pp. 32–48.

[21] Brian S Blais et al. "Receptive Field Formation in Natural Scene Environments: Comparison of Single Cell Learning Rules". In: *Advances in Neural Information Processing Systems 10*. Ed. by M I Jordan, M J Kearns, and S A Solla. MIT Press, 1998, pp. 423–429.

[22] Ralph Bourdoukan and Sophie Denève. "Enforcing balance allows local supervised learning in spiking recurrent networks". In: *Advances in Neural Information Processing Systems 28*. Ed. by C Cortes et al. Curran Associates, Inc., 2015, pp. 982–990.

[23] Ralph Bourdoukan et al. "Learning optimal spike-based representations". In: *Advances in Neural Information Processing Systems 25*. Ed. by F Pereira et al. Curran Associates, Inc., 2012, pp. 2285–2293.

[24] Romain Brette and Wulfram Gerstner. "Adaptive exponential integrate-and-fire model as an effective description of neuronal activity". en. In: *J. Neurophysiol.* 94.5 (Nov. 2005), pp. 3637–3642.

[25] Romain Brette and Dan F M Goodman. "Simulating spiking neural networks on GPU". en. In: *Network* 23.4 (Oct. 2012), pp. 167–182.

[26] André R Brodtkorb, Trond R Hagen, and Martin L Sætra. "Graphics processing unit (GPU) programming strategies and trends in GPU computing". In: *J. Parallel Distrib. Comput.* 73.1 (Jan. 2013), pp. 4–13.

[27] N Brunel. "Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons". en. In: *J. Comput. Neurosci.* 8.3 (May 2000), pp. 183–208.

[28] Engin Bumbacher and Vivienne L Ming. "PITCH-SENSITIVE COMPONENTS EMERGE FROM HIERARCHICAL SPARSE CODING OF NATURAL SOUNDS". In: *Proceedings of the 1st International Conference on Pattern Recognition Applications and Methods*. Vilamoura, Algarve, Portugal: SciTePress - Science, 2012, pp. 219–229.

[29] G T Buracas et al. "Efficient discrimination of temporal patterns by motion-sensitive neurons in primate visual cortex". en. In: *Neuron* 20.5 (May 1998), pp. 959–969.

[30] Anthony N Burkitt, Hamish Meffin, and David B Grayden. "Spike-timing-dependent plasticity: the relationship to rate-based learning for models with weight dynamics determined by a stable fixed point". en. In: *Neural Comput.* 16.5 (May 2004), pp. 885–940.

[31] Peter A Cariani and Bertrand Delgutte. "Neural correlates of the pitch of complex tones. I. Pitch and pitch salience". In: *Journal of Neurophysiology* 76.3 (1996), pp. 1698–1716.

[32] C Nikoosh Carlo and Charles F Stevens. "Structural uniformity of neocortex, revisited". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 110.4 (Jan. 2013), pp. 1488–1493.

[33] Hideyuki Câteau, Katsunori Kitano, and Tomoki Fukai. "An accurate and widely applicable method to determine the distribution of synaptic strengths formed by the spike-timing-dependent learning". In: *Neurocomputing* 44-46 (June 2002), pp. 343–351.

[34] Romain Daniel Cazé, Mark Humphries, and Boris Gutkin. "Passive dendrites enable single neurons to compute linearly non-separable functions". en. In: *PLoS Comput. Biol.* 9.2 (Feb. 2013), e1002867.

[35] Claudia Clopath and Wulfram Gerstner. "Voltage and Spike Timing Interact in STDP - A Unified Model". en. In: *Front. Synaptic Neurosci.* 2 (July 2010), p. 25.

[36] Claudia Clopath, André Longtin, and Wulfram Gerstner. "An online Hebbian learning rule that performs Independent Component Analysis". In: *Advances in Neural Information Processing Systems 20*. Ed. by J C Platt et al. Curran Associates, Inc., 2008, pp. 321–328.

[37] Claudia Clopath et al. "Connectivity reflects coding: a model of voltage-based STDP with homeostasis". en. In: *Nat. Neurosci.* 13.3 (Mar. 2010), pp. 344–352.

[38] Claudia Clopath et al. "Receptive field formation by interacting excitatory and inhibitory synaptic plasticity". en. In: *bioRxiv* (Jan. 2016), p. 066589.

[39] Michael A Cohen, Stephen Grossberg, and Lonce L Wyse. "A spectral network model of pitch perception". In: *The Journal of the Acoustical Society of America* 98.2 (1995), pp. 862–879.

[40] Nuno Maçarico da Costa and Kevan A C Martin. "Whose Cortical Column Would that Be?" en. In: *Front. Neuroanat.* 4 (May 2010), p. 16.

[41] M Cynader. “Prolonged sensitivity to monocular deprivation in dark-reared cats: effects of age and visual exposure”. en. In: *Brain Res.* 284.2-3 (June 1983), pp. 155–164.

[42] James A D’amour and Robert C Froemke. “Inhibitory and excitatory spike-timing-dependent plasticity in the auditory cortex”. en. In: *Neuron* 86.2 (Apr. 2015), pp. 514–528.

[43] R C deCharms and A Zador. “Neural representation and the cortical code”. en. In: *Annu. Rev. Neurosci.* 23 (2000), pp. 613–647.

[44] Javier DeFelipe, Lidia Alonso-Nanclares, and Jon I Arellano. “Microstructure of the neocortex: comparative aspects”. en. In: *J. Neurocytol.* 31.3-5 (Mar. 2002), pp. 299–316.

[45] Sophie Denève and Christian K Machens. “Efficient codes and balanced networks”. en. In: *Nat. Neurosci.* 19.3 (Mar. 2016), pp. 375–382.

[46] Anja L Dorrn et al. “Developmental sensory experience balances cortical excitation and inhibition”. en. In: *Nature* 465.7300 (June 2010), pp. 932–936.

[47] R J Douglas and K A Martin. “A functional microcircuit for cat visual cortex”. en. In: *J. Physiol.* 440 (1991), pp. 735–769.

[48] Rodney J Douglas and Kevan A C Martin. “Neuronal circuits of the neocortex”. en. In: *Annu. Rev. Neurosci.* 27 (2004), pp. 419–451.

[49] Rodney J Douglas, Kevan A C Martin, and David Whitteridge. “A Canonical Microcircuit for Neocortex”. In: *Neural Comput.* 1.4 (Dec. 1989), pp. 480–488.

[50] Akihiro Eguchi et al. “The emergence of polychronization and feature binding in a spiking neural network model of the primate ventral visual system”. en. In: *Psychol. Rev.* 125.4 (July 2018), pp. 545–571.

[51] Jochen Martin Eppler et al. “PyNEST: A Convenient Interface to the NEST Simulator”. en. In: *Front. Neuroinform.* 2 (2008), p. 12.

[52] Nafise Erfanian Saeedi et al. "An integrated model of pitch perception incorporating place and temporal pitch codes with application to cochlear implant research". en. In: *Hear. Res.* 344 (Feb. 2017), pp. 135–147.

[53] Nafise Erfanian Saeedi et al. "Learning Pitch with STDP: A Computational Model of Place and Temporal Pitch Perception Using Spiking Neural Networks". en. In: *PLoS Comput. Biol.* 12.4 (Apr. 2016), e1004860.

[54] O V Favorov, M E Diamond, and B L Whitsel. "Evidence for a mosaic representation of the body surface in area 3b of the somatic cortex of cat". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 84.18 (Sept. 1987), pp. 6606–6610.

[55] Daniel E Feldman. "Synaptic mechanisms for plasticity in neocortex". en. In: *Annu. Rev. Neurosci.* 32 (2009), pp. 33–55.

[56] A K Fidjeland et al. "NeMo: A Platform for Neural Modelling of Spiking Neurons Using GPUs". In: *2009 20th IEEE International Conference on Application-specific Systems, Architectures and Processors*. July 2009, pp. 137–144.

[57] Peter Foldiak. "Sparse coding in the primate cortex". en. In: *The Handbook of Brain Theory and Neural Networks, Second Edition*. Ed. by Michael A Arbib. MIT Press, 2003, pp. 1064–1068.

[58] Peter Földiák. "Adaptive network for optimal linear feature extraction". In: *International 1989 Joint Conference on Neural Networks*. 1989, 401–405 vol.1.

[59] Peter Földiák and Malcom P Young. "Sparse Coding in the Primate Cortex". In: *Handbook of Brain Theory and Neural Networks*. Ed. by Michael A Arbib. MIT Press, 1995, pp. 1–1064.

[60] Katrin Franke et al. "Inhibition decorrelates visual feature representations in the inner retina". en. In: *Nature* 542.7642 (Feb. 2017), pp. 439–444.

[61] Karl Friston. "The free-energy principle: a unified brain theory?" en. In: *Nat. Rev. Neurosci.* 11.2 (Feb. 2010), pp. 127–138.

[62] Robert C Froemke, Michael M Merzenich, and Christoph E Schreiner. "A synaptic memory trace for cortical receptive field plasticity". en. In: *Nature* 450.7168 (Nov. 2007), pp. 425–429.

[63] K Fukushima. "Cognitron: a self-organizing multilayered neural network". en. In: *Biol. Cybern.* 20.3-4 (Nov. 1975), pp. 121–136.

[64] Juan M Galeazzi et al. "A self-organizing model of the visual development of hand-centred representations". en. In: *PLoS One* 8.6 (June 2013), e66272.

[65] Surya Ganguli and Haim Sompolinsky. "Compressed sensing, sparsity, and dimensionality in neuronal information processing and data analysis". en. In: *Annu. Rev. Neurosci.* 35 (Apr. 2012), pp. 485–508.

[66] Dileep George and Jeff Hawkins. "Towards a mathematical theory of cortical micro-circuits". en. In: *PLoS Comput. Biol.* 5.10 (Oct. 2009), e1000532.

[67] W Gerstner et al. "A neuronal learning rule for sub-millisecond temporal coding". en. In: *Nature* 383.6595 (Sept. 1996), pp. 76–81.

[68] Wulfram Gerstner et al. *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. en. Cambridge University Press, July 2014.

[69] Julijana Gjorgjieva et al. "A triplet spike-timing-dependent plasticity model generalizes the Bienenstock-Cooper-Munro rule to higher-order spatiotemporal correlations". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 108.48 (Nov. 2011), pp. 19383–19388.

[70] Brian R Glasberg and Brian CJ Moore. "Derivation of auditory filter shapes from notched-noise data". In: *Hearing research* 47.1 (1990), pp. 103–138.

[71] Julius L Goldstein. "An optimum processor theory for the central formation of the pitch of complex tones". In: *The Journal of the Acoustical Society of America* 54.6 (1973), pp. 1496–1516.

[72] Dan F M Goodman and Romain Brette. "Learning to Localise Sounds with Spiking Neural Networks". In: *Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1*. NIPS'10. Vancouver, British Columbia, Canada: Curran Associates Inc., 2010, pp. 784–792.

[73] Dan F M Goodman and Romain Brette. "The brian simulator". en. In: *Front. Neurosci.* 3.2 (Sept. 2009), pp. 192–197.

[74] Daniel J Graham, Damon M Chandler, and David J Field. "Can the theory of "whitening" explain the center-surround properties of retinal ganglion cell receptive fields?" en. In: *Vision Res.* 46.18 (Sept. 2006), pp. 2901–2913.

[75] Charles G Gross. "Genealogy of the "grandmother cell"". en. In: *Neuroscientist* 8.5 (Oct. 2002), pp. 512–518.

[76] Charles G Gross. "Single neuron studies of inferior temporal cortex". en. In: *Neuropsychologia* 46.3 (Feb. 2008), pp. 841–852.

[77] Stephen Grossberg. "Competitive learning: From interactive activation to adaptive resonance". In: *Cogn. Sci.* 11.1 (Jan. 1987), pp. 23–63.

[78] Jan Hahne et al. "Integration of Continuous-Time Dynamics in a Spiking Neural Network Simulator". en. In: *Front. Neuroinform.* 11 (May 2017), p. 34.

[79] Bilal Haider et al. "Neocortical network activity in vivo is generated through a dynamic balance of excitation and inhibition". en. In: *J. Neurosci.* 26.17 (Apr. 2006), pp. 4535–4545.

[80] Martha N Havenith et al. “Synchrony makes neurons fire in sequence, and stimulus properties determine who is ahead”. en. In: *J. Neurosci.* 31.23 (June 2011), pp. 8570–8584.

[81] D O Hebb. “The organization of behavior: A neuropsychological approach”. In: (1949).

[82] Keith B Hengen et al. “Neuronal Firing Rate Homeostasis Is Inhibited by Sleep and Promoted by Wake”. en. In: *Cell* 165.1 (Mar. 2016), pp. 180–191.

[83] Guillaume Hennequin, Everton J Agnes, and Tim P Vogels. “Inhibitory Plasticity: Balance, Control, and Codependence”. en. In: *Annu. Rev. Neurosci.* (June 2017).

[84] Guillaume Hennequin, Tim P Vogels, and Wulfram Gerstner. “Optimal control of transient dynamics in balanced networks supports generation of complex movements”. en. In: *Neuron* 82.6 (June 2014), pp. 1394–1406.

[85] Suzana Herculano-Houzel. “The human brain in numbers: a linearly scaled-up primate brain”. en. In: *Front. Hum. Neurosci.* 3 (Nov. 2009), p. 31.

[86] M L Hines and N T Carnevale. “The NEURON simulation environment”. en. In: *Neural Comput.* 9.6 (Aug. 1997), pp. 1179–1209.

[87] G E Hinton, J L McClelland, and D E Rumelhart. “Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1”. In: ed. by David E Rumelhart, James L McClelland, and Corporate PDP Research Group. Cambridge, MA, USA: MIT Press, 1986. Chap. Distributed Representations, pp. 77–109.

[88] Judith A Hirsch and Luis M Martinez. “Laminar processing in the visual cortical column”. en. In: *Curr. Opin. Neurobiol.* 16.4 (Aug. 2006), pp. 377–384.

[89] Tomás Hromádka, Michael R Deweese, and Anthony M Zador. "Sparse representation of sounds in the unanesthetized auditory cortex". en. In: *PLoS Biol.* 6.1 (Jan. 2008), e16.

[90] D H Hubel and T N Wiesel. "Receptive fields and functional architecture of monkey striate cortex". en. In: *J. Physiol.* 195.1 (Mar. 1968), pp. 215–243.

[91] Nathan Intrator and Joshua I Gold. "Three-Dimensional Object Recognition Using an Unsupervised BCM Network: The Usefulness of Distinguishing Features". In: *Neural Comput.* 5.1 (Jan. 1993), pp. 61–74.

[92] Jeffry S Isaacson and Massimo Scanziani. "How inhibition shapes cortical activity". en. In: *Neuron* 72.2 (Oct. 2011), pp. 231–243.

[93] Matias J Ison et al. "Selectivity of pyramidal cells and interneurons in the human medial temporal lobe". en. In: *J. Neurophysiol.* 106.4 (Oct. 2011), pp. 1713–1721.

[94] Alekseı Ivakhnenko and V G Lapa. *Cybernetic predicting devices*. CCM Information Corporation, 1965.

[95] E M Izhikevich. "Simple model of spiking neurons". en. In: *IEEE Trans. Neural Netw.* 14.6 (2003), pp. 1569–1572.

[96] Eugene M Izhikevich. "Polychronization: computation with spikes". en. In: *Neural Comput.* 18.2 (Feb. 2006), pp. 245–282.

[97] Kimberle M Jacobs. "Brodmann's Areas of the Cortex". In: *Encyclopedia of Clinical Neuropsychology*. Ed. by Jeffrey S Kreutzer, John DeLuca, and Bruce Caplan. New York, NY: Springer New York, 2011, pp. 459–459.

[98] Edward G Jones and Pasko Rakic. "Radial columns in cortical architecture: it is the composition that counts". en. In: *Cereb. Cortex* 20.10 (Oct. 2010), pp. 2261–2264.

[99] Jean-Sébastien Jouhanneau, Jens Kremkow, and James F A Poulet. "Single synaptic inputs drive high-precision action potentials in parvalbumin expressing GABA-ergic cortical neurons in vivo". en. In: *Nat. Commun.* 9.1 (Apr. 2018), p. 1540.

[100] Bahadir Kasap and A John van Opstal. "Dynamic parallelism for synaptic updating in GPU-accelerated spiking neural network simulations". In: *Neurocomputing* 302 (Aug. 2018), pp. 55–65.

[101] Narayanan Kasthuri et al. "Saturated Reconstruction of a Volume of Neocortex". en. In: *Cell* 162.3 (July 2015), pp. 648–661.

[102] Tara Keck et al. "Integrating Hebbian and homeostatic plasticity: the current state of the field and future research directions". en. In: *Philos. Trans. R. Soc. Lond. B Biol. Sci.* 372.1715 (Mar. 2017), p. 20160158.

[103] M M Khan et al. "SpiNNaker: Mapping neural networks onto a massively-parallel chip multiprocessor". In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. June 2008, pp. 2849–2856.

[104] Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, and Timothée Masquelier. "Bio-inspired Unsupervised Learning of Visual Features Leads to Robust Invariant Object Recognition". In: (Apr. 2015). arXiv: `1504.03871 [cs.CV]`.

[105] Paul D King, Joel Zylberberg, and Michael R DeWeese. "Inhibitory interneurons decorrelate excitatory cells to drive sparse code formation in a spiking model of V1". en. In: *J. Neurosci.* 33.13 (Mar. 2013), pp. 5475–5485.

[106] A Kirkwood, M C Rioult, and M F Bear. "Experience-dependent modification of synaptic plasticity in visual cortex". en. In: *Nature* 381.6582 (June 1996), pp. 526–528.

[107] James C Knight and Thomas Nowotny. "GPUs Outperform Current HPC and Neuromorphic Solutions in Terms of Speed and Energy When Simulating a Highly-Connected Cortical Model". en. In: *Front. Neurosci.* 12 (Dec. 2018), p. 941.

[108] Teuvo Kohonen. "Self-organized formation of topologically correct feature maps". In: *Biol. Cybern.* 43.1 (Jan. 1982), pp. 59–69.

[109] Romesh D Kumbhani, Mark J Nolt, and Larry A Palmer. "Precision, reliability, and information-theoretic analysis of visual thalamocortical neurons". en. In: *J. Neurophysiol.* 98.5 (Nov. 2007), pp. 2647–2663.

[110] Kelly Lambert et al. "Optimizing brain performance: Identifying mechanisms of adaptive neurobiological plasticity". en. In: *Neurosci. Biobehav. Rev.* 105 (Oct. 2019), pp. 60–71.

[111] Jonathan Laudanski, Yi Zheng, and Romain Brette. "A structural theory of pitch". In: *eneuro* 1.1 (2014), ENEURO–0033.

[112] Peter Lennie. "The Cost of Cortical Computation". In: *Curr. Biol.* 13.6 (Mar. 2003), pp. 493–497.

[113] W B Levy and R A Baxter. "Energy efficient neural codes". en. In: *Neural Comput.* 8.3 (Apr. 1996), pp. 531–543.

[114] Charl Linssen et al. *NEST 2.16.0*. Aug. 2018. DOI: 10.5281/zenodo.1400175. URL: https://doi.org/10.5281/zenodo.1400175.

[115] Bao-Hua Liu et al. "Visual receptive field structure of cortical inhibitory neurons revealed by two-photon imaging guided recording". en. In: *J. Neurosci.* 29.34 (Aug. 2009), pp. 10520–10532.

[116] Michael London et al. "Sensitivity to perturbations in vivo implies high noise and suggests rate coding in cortex". en. In: *Nature* 466.7302 (July 2010), pp. 123–127.

[117] W Maass. "To Spike or Not to Spike: That Is the Question". In: *Proc. IEEE* 103.12 (Dec. 2015), pp. 2219–2224.

[118] C von der Malsburg. "Self-organization of orientation sensitive cells in the striate cortex". en. In: *Kybernetik* 14.2 (Dec. 1973), pp. 85–100.

[119] Adam H Marblestone, Greg Wayne, and Konrad P Kording. "Toward an Integration of Deep Learning and Neuroscience". en. In: *Front. Comput. Neurosci.* 10 (2016).

[120] H Markram et al. "Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs". en. In: *Science* 275.5297 (Jan. 1997), pp. 213–215.

[121] D Marr and T Poggio. "From understanding computation to understanding neural circuitry". In: (1976).

[122] Timothée Masquelier and Simon J Thorpe. "Unsupervised learning of visual features through spike timing dependent plasticity". en. In: *PLoS Comput. Biol.* 3.2 (Feb. 2007), e31.

[123] K Meier. "A mixed-signal universal neuromorphic computing system". In: *2015 IEEE International Electron Devices Meeting (IEDM)*. Dec. 2015, pp. 4.6.1–4.6.4.

[124] L von Melchner, S L Pallas, and M Sur. "Visual behaviour mediated by retinal projections directed to the auditory pathway". en. In: *Nature* 404.6780 (Apr. 2000), pp. 871–876.

[125] Hanno S Meyer et al. "Inhibitory interneurons in a cortical column form hot zones of inhibition in layers 2 and 5A". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 108.40 (Oct. 2011), pp. 16807–16812.

[126] K D Miller. "Synaptic economics: competition and cooperation in synaptic plasticity". en. In: *Neuron* 17.3 (Sept. 1996), pp. 371–374.

[127] M J Mišić, Đ M Đurđević, and M V Tomašević. "Evolution and trends in GPU computing". In: *2012 Proceedings of the 35th International Convention MIPRO*. May 2012, pp. 289–294.

[128] Kenji Mizuseki and György Buzsáki. "Preconfigured, skewed distribution of firing rates in the hippocampus and entorhinal cortex". en. In: *Cell Rep.* 4.5 (Sept. 2013), pp. 1010–1021.

[129] Brian CJ Moore, Brian R Glasberg, and Robert W Peters. "Relative dominance of individual partials in determining the pitch of complex tones". In: *The Journal of the Acoustical Society of America* 77.5 (1985), pp. 1853–1860.

[130] Rubén Moreno-Bote and Jan Drugowitsch. "Causal Inference and Explaining Away in a Spiking Network". en. In: *Sci. Rep.* 5 (Dec. 2015), p. 17531.

[131] Abigail Morrison et al. "Advancing the boundaries of high-connectivity network simulation with distributed computing". en. In: *Neural Comput.* 17.8 (Aug. 2005), pp. 1776–1801.

[132] MOUNTCASTLE and V. "An organizing principle for cerebral function : the unit module and the distributed system". In: *The Mindful Brain* (1978).

[133] V B Mountcastle. "The columnar organization of the neocortex". en. In: *Brain* 120 (Pt 4) (Apr. 1997), pp. 701–722.

[134] Mikhail Mukovski et al. "Detection of active and silent states in neocortical neurons from the field potential signal during slow-wave sleep". en. In: *Cereb. Cortex* 17.2 (Feb. 2007), pp. 400–414.

[135] Jayram Moorkanikara Nageswaran et al. "A configurable simulation environment for the efficient simulation of large-scale spiking neural networks on graphics processors". en. In: *Neural Netw.* 22.5-6 (July 2009), pp. 791–800.

[136] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. "Surrogate Gradient Learning in Spiking Neural Networks". In: (Jan. 2019). arXiv: `1901.09948 [cs.NE]`.

[137] Guangjian Ni et al. "Modelling cochlear mechanics". en. In: *Biomed Res. Int.* 2014 (July 2014), p. 150637.

[138] Wilten Nicola and Claudia Clopath. "Supervised learning in spiking neural networks with FORCE training". en. In: *Nat. Commun.* 8.1 (Dec. 2017), p. 2208.

[139] Cuda Nvidia. "Cuda C programming guide v8. 0". In: *Nvidia Corporation* (2017).

[140] E Oja. "Subspace Methods of Pattern Recognition". In: *Pattern Recognition and Image Processing Series* 6 (1983).

[141] Erkki Oja. "Simplified neuron model as a principal component analyzer". In: *J. Math. Biol.* 15.3 (Nov. 1982), pp. 267–273.

[142] Michael Okun and Ilan Lampl. "Instantaneous correlation of excitation and inhibition during ongoing and sensory-evoked activities". en. In: *Nat. Neurosci.* 11.5 (May 2008), pp. 535–537.

[143] Aude Oliva and Antonio Torralba. "Building the gist of a scene: the role of global image features in recognition". en. In: *Prog. Brain Res.* 155 (2006), pp. 23–36.

[144] B A Olshausen and D J Field. "Sparse coding with an overcomplete basis set: a strategy employed by V1?" en. In: *Vision Res.* 37.23 (Dec. 1997), pp. 3311–3325.

[145] B A Olshausen and D J Field. "Sparse coding with an overcomplete basis set: a strategy employed by V1?" en. In: *Vision Res.* 37.23 (Dec. 1997), pp. 3311–3325.

[146] Bruno A Olshausen and David J Field. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images". In: *Nature* 381 (June 1996), p. 607.

[147] Bruno A Olshausen and David J Field. "Sparse coding of sensory inputs". en. In: *Curr. Opin. Neurobiol.* 14.4 (Aug. 2004), pp. 481–487.

[148] Andrew J Oxenham. "Pitch perception and auditory stream segregation: implications for hearing loss and cochlear implants". In: *Trends in amplification* 12.4 (2008), pp. 316–331.

[149] Andrew J Oxenham et al. "Pitch perception beyond the traditional existence region of pitch". In: *Proceedings of the National Academy of Sciences* 108.18 (2011), pp. 7629–7634.

[150] Hélène Paugam-Moisy, Régis Martinez, and Samy Bengio. "Delay learning and polychronization for reservoir computing". In: *Neurocomputing* 71.7–9 (2008), pp. 1143–1158.

[151] Cengiz Pehlevan. "A Spiking Neural Network with Local Learning Rules Derived From Nonnegative Similarity Matching". In: (Feb. 2019). arXiv: `1902.01429 [cs.NE]`.

[152] Jean-Pascal Pfister and Wulfram Gerstner. "Triplets of spikes in a model of spike timing-dependent plasticity". en. In: *J. Neurosci.* 26.38 (Sept. 2006), pp. 9673–9682.

[153] R P Rao and D H Ballard. "Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects". en. In: *Nat. Neurosci.* 2.1 (Jan. 1999), pp. 79–87.

[154] Martin Rehn and Friedrich T Sommer. "A network that uses few active neurones to code visual input predicts the diverse shapes of cortical receptive fields". en. In: *J. Comput. Neurosci.* 22.2 (Apr. 2007), pp. 135–146.

[155] P Reinagel and R C Reid. "Temporal coding of visual information in the thalamus". en. In: *J. Neurosci.* 20.14 (July 2000), pp. 5392–5400.

[156] M Riesenhuber and T Poggio. "Hierarchical models of object recognition in cortex". en. In: *Nat. Neurosci.* 2.11 (Nov. 1999), pp. 1019–1025.

[157] Dario L Ringach. "Mapping receptive fields in primary visual cortex". en. In: *J. Physiol.* 558.Pt 3 (Aug. 2004), pp. 717–728.

[158] Dario L Ringach. "Spatial structure and symmetry of simple-cell receptive fields in macaque primary visual cortex". en. In: *J. Neurophysiol.* 88.1 (July 2002), pp. 455–463.

[159] L Robles, M A Ruggero, and N C Rich. "Two-tone distortion in the basilar membrane of the cochlea". en. In: *Nature* 349.6308 (Jan. 1991), pp. 413–414.

[160] A J Rockel, R W Hiorns, and T P Powell. "The basic uniformity in structure of the neocortex". en. In: *Brain* 103.2 (June 1980), pp. 221–244.

[161] A W Roe et al. "Visual projections routed to the auditory pathway in ferrets: receptive fields of visual neurons in primary auditory cortex". en. In: *J. Neurosci.* 12.9 (Sept. 1992), pp. 3651–3664.

[162] B Roerig and B Chen. "Relationships of local inhibitory and excitatory circuits to orientation preference maps in ferret visual cortex". en. In: *Cereb. Cortex* 12.2 (Feb. 2002), pp. 187–198.

[163] E T Rolls and T Milward. "A model of invariant object recognition in the visual system: learning rules, activation functions, lateral inhibition, and information-based performance measures". en. In: *Neural Comput.* 12.11 (Nov. 2000), pp. 2547–2572.

[164] E T Rolls and M J Tovee. "Sparseness of the neuronal representation of stimuli in the primate temporal visual cortex". en. In: *J. Neurophysiol.* 73.2 (Feb. 1995), pp. 713–726.

[165] Edmund T Rolls and T Milward. "A model of invariant object recognition in the visual system: learning rules, activation functions, lateral inhibition, and information-based performance measures". In: *Neural Computation* 12.11 (2000), pp. 2547–2572.

[166] Edmund T Rolls and Simon M Stringer. "Invariant global motion recognition in the dorsal visual system: a unifying theory". en. In: *Neural Comput.* 19.1 (Jan. 2007), pp. 139–169.

[167] Cyrille Rossant et al. "Sensitivity of noisy neurons to coincident inputs". en. In: *J. Neurosci.* 31.47 (Nov. 2011), pp. 17193–17206.

[168] M C van Rossum, G Q Bi, and G G Turrigiano. "Stable Hebbian learning from spike timing-dependent plasticity". en. In: *The Journal of* 20.23 (Dec. 2000), pp. 8812–8821.

[169] Sébastien Royer and Denis Paré. "Conservation of total synaptic weight through balanced synaptic depression and potentiation". In: *Nature* 422.6931 (2003), pp. 518–522.

[170] Christopher J Rozell et al. "Sparse coding via thresholding and local competition in neural circuits". en. In: *Neural Comput.* 20.10 (Oct. 2008), pp. 2526–2563.

[171] J Rubin, D D Lee, and H Sompolinsky. "Equilibrium properties of temporally asymmetric Hebbian plasticity". en. In: *Phys. Rev. Lett.* 86.2 (Jan. 2001), pp. 364–367.

[172] M. Rudnicki and W. Hemmert. "Cochlea: inner ear models in Python". In: *https://github.com/mrkrd/cochlea* (2014).

[173] Michelle Rudolph-Lilith, Mathieu Dubois, and Alain Destexhe. "Analytical integrate-and-fire neuron models with conductance-based dynamics and

realistic postsynaptic potential time course for event-driven simulation strategies". en. In: *Neural Comput.* 24.6 (June 2012), pp. 1426–1461.

[174] M A Ruggero. "Distortion in those good vibrations". en. In: *Curr. Biol.* 3.11 (Nov. 1993), pp. 755–758.

[175] David E Rumelhart and David Zipser. "Feature Discovery by Competitive Learning*". In: *Cogn. Sci.* 9.1 (Jan. 1985), pp. 75–112.

[176] Setsuko Sahara et al. "The fraction of cortical GABAergic neurons is constant from near the start of cortical neurogenesis to adulthood". en. In: *J. Neurosci.* 32.14 (Apr. 2012), pp. 4755–4761.

[177] Jason Sanders and Edward Kandrot. *CUDA by Example: An Introduction to General-Purpose GPU Programming*. 1st. Addison-Wesley Professional, 2010.

[178] Hajime Sano and B Keith Jenkins. "A neural network model for pitch perception". In: *Computer Music Journal* (1989), pp. 41–48.

[179] Jürgen Schmidhuber. "My First Deep Learning System of 1991 + Deep Learning Timeline 1962-2013". In: (Dec. 2013). arXiv: `1312.5548 [cs.NE]`.

[180] Jan Schnupp, Israel Nelken, and Andrew King. *Auditory Neuroscience: Making Sense of Sound*. en. MIT Press, 2011.

[181] Trevor M Shackleton and Robert P Carlyon. "The role of resolved and unresolved harmonics in pitch perception and frequency modulation discrimination". In: *The Journal of the Acoustical Society of America* 95.6 (1994), pp. 3529–3540.

[182] Shihab Shamma and David Klein. "The case of the missing pitch templates: How harmonic templates emerge in the early auditory system". In: *The Journal of the Acoustical Society of America* 107.5 (2000), pp. 2631–2644.

[183] Shihab A Shamma. "Topographic organization is essential for pitch perception". In: *Proceedings of the National Academy of Sciences of the United States of America* 101.5 (2004), pp. 1114–1115.

[184] Yosef Singer et al. "Sensory cortex is optimized for prediction of future input". en. In: *Elife* 7 (June 2018).

[185] Tanya Sippy and Rafael Yuste. "Decorrelating action of inhibition in neocortical networks". en. In: *J. Neurosci.* 33.23 (June 2013), pp. 9813–9830.

[186] T S Skoglund, R Pascher, and C H Berthold. "Heterogeneity in the columnar number of neurons in different neocortical areas in the rat". en. In: *Neurosci. Lett.* 208.2 (Apr. 1996), pp. 97–100.

[187] Christopher J Smalt et al. "Distortion products and their influence on representation of pitch-relevant information in the human brainstem for unresolved harmonic complex tones". en. In: *Hear. Res.* 292.1-2 (Oct. 2012), pp. 26–34.

[188] W R Softky and C Koch. "The highly irregular firing of cortical cells is inconsistent with temporal integration of random EPSPs". en. In: *J. Neurosci.* 13.1 (Jan. 1993), pp. 334–350.

[189] Henning Sprekeler. "Functional consequences of inhibitory plasticity: homeostasis, the excitation-inhibition balance and beyond". en. In: *Curr. Opin. Neurobiol.* 43 (Apr. 2017), pp. 198–203.

[190] C F Stevens and Y Wang. "Facilitation and depression at single central synapses". en. In: *Neuron* 14.4 (Apr. 1995), pp. 795–802.

[191] Marcel Stimberg, Dan F M Goodman, and Thomas Nowotny. "Brian2GeNN: a system for accelerating a large variety of spiking neural networks with graphics hardware". en. In: *bioRxiv* (Oct. 2018), p. 448050.

[192] Marcel Stimberg et al. “Equation-oriented specification of neural models for simulations”. In: *Front. Neuroinform.* 8 (Feb. 2014), p. 6.

[193] Carsen Stringer et al. “Inhibitory control of correlated intrinsic variability in cortical networks”. en. In: *Elife* 5 (Dec. 2016).

[194] Yujiao J Sun et al. “Fine-tuning of pre-balanced excitation and inhibition during auditory cortical development”. en. In: *Nature* 465.7300 (June 2010), pp. 927–931.

[195] M Sur, P E Garraghty, and A W Roe. “Experimentally induced visual projections into auditory thalamus and cortex”. en. In: *Science* 242.4884 (Dec. 1988), pp. 1437–1441.

[196] Ian Taylor and Mike Greenhough. “Modelling pitch perception with adaptive resonance theory artificial neural networks”. In: *Connection Science* 6.2-3 (1994), pp. 135–154.

[197] Tom Tetzlaff et al. “Decorrelation of neural-network activity by inhibitory feedback”. en. In: *PLoS Comput. Biol.* 8.8 (Aug. 2012), e1002596.

[198] M Tommerdahl et al. “Minicolumnar activation patterns in cat and monkey SI cortex”. en. In: *Cereb. Cortex* 3.5 (Sept. 1993), pp. 399–411.

[199] Mcw Van Rossum, G Q Bi, et al. “Stable Hebbian learning from spike timing-dependent plasticity”. In: *The Journal of* (2000).

[200] Julien Vitay, Helge Ü Dinkelbach, and Fred H Hamker. “ANNarchy: a code generation approach to neural simulations on parallel hardware”. en. In: *Front. Neuroinform.* 9 (July 2015), p. 19.

[201] T P Vogels et al. “Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks”. en. In: *Science* 334.6062 (Dec. 2011), pp. 1569–1573.

[202] Tim P Vogels and L F Abbott. “Signal propagation and logic gating in networks of integrate-and-fire neurons”. en. In: *J. Neurosci.* 25.46 (Nov. 2005), pp. 10786–10795.

[203] Tim P Vogels, Kanaka Rajan, and L F Abbott. “Neural network dynamics”. en. In: *Annu. Rev. Neurosci.* 28 (2005), pp. 357–376.

[204] C van Vreeswijk and H Sompolinsky. “Chaos in neuronal networks with balanced excitatory and inhibitory activity”. en. In: *Science* 274.5293 (Dec. 1996), pp. 1724–1726.

[205] G Wallis and E T Rolls. “Invariant face and object recognition in the visual system”. en. In: *Prog. Neurobiol.* 51.2 (Feb. 1997), pp. 167–194.

[206] Daniel Walters, Simon Stringer, and Edmund Rolls. “Path integration of head direction: updating a packet of neural activity at the correct speed using axonal conduction delays”. en. In: *PLoS One* 8.3 (Mar. 2013), e58330.

[207] Michael Wehr and Anthony M Zador. “Balanced inhibition underlies tuning and sharpens spike timing in auditory cortex”. en. In: *Nature* 426.6965 (Nov. 2003), pp. 442–446.

[208] W Bryan Wilent and Diego Contreras. “Dynamics of excitation and inhibition underlying stimulus selectivity in rat somatosensory cortex”. en. In: *Nat. Neurosci.* 8.10 (Oct. 2005), pp. 1364–1370.

[209] B Willmore and D J Tolhurst. “Characterizing the sparseness of neural codes”. In: *Network: Computation in Neural Systems* 12.3 (Jan. 2001), pp. 255–270.

[210] Ben D B Willmore, James A Mazer, and Jack L Gallant. “Sparse coding in striate and extrastriate visual cortex”. en. In: *J. Neurophysiol.* 105.6 (June 2011), pp. 2907–2919.

[211] Nathaniel C Wright and Ralf Wessel. "Network activity influences the subthreshold and spiking visual responses of pyramidal neurons in the three-layer turtle cortex". en. In: *J. Neurophysiol.* 118.4 (Oct. 2017), pp. 2142–2155.

[212] Guangying K Wu et al. "Lateral sharpening of cortical frequency tuning by approximately balanced inhibition". en. In: *Neuron* 58.1 (Apr. 2008), pp. 132–143.

[213] Mingshan Xue, Bassam V Atallah, and Massimo Scanziani. "Equalizing excitation-inhibition ratios across visual cortical neurons". en. In: *Nature* 511.7511 (July 2014), pp. 596–600.

[214] Daniel L K Yamins et al. "Performance-optimized hierarchical models predict neural responses in higher visual cortex". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 111.23 (June 2014), pp. 8619–8624.

[215] Yang Yang et al. "Millisecond-scale differences in neural activity in auditory cortex can drive decisions". en. In: *Nat. Neurosci.* 11.11 (Nov. 2008), pp. 1262–1263.

[216] Esin Yavuz, James Turner, and Thomas Nowotny. "GeNN: a code generation framework for accelerated brain simulations". en. In: *Sci. Rep.* 6 (Jan. 2016), p. 18854.

[217] Izzet B Yildiz, Nima Mesgarani, and Sophie Deneve. "Predictive Ensemble Decoding of Acoustical Features Explains Context-Dependent Receptive Fields". en. In: *J. Neurosci.* 36.49 (Dec. 2016), pp. 12338–12350.

[218] Takashi Yoshida and Kenichi Ohki. "Robust representation of natural images by sparse and variable population of active neurons in visual cortex". en. In: *bioRxiv* (July 2018), p. 300863.

[219] William A Yost. "Pitch of iterated rippled noise". In: *The Journal of the Acoustical Society of America* 100.1 (1996), pp. 511–518. DOI: 10.1121/1.415873. URL: http://dx.doi.org/10.1121/1.415873.

[220] Christos Zarras et al. "Frequency and Pitch Representation Using Self-Organized Maps". In: *Proceedings of the 12th International Conference on Music Perception and Cognition* (2012).

[221] Friedemann Zenke, Everton J Agnes, and Wulfram Gerstner. "Diverse synaptic plasticity mechanisms orchestrated to form and retrieve memories in spiking neural networks". en. In: *Nat. Commun.* 6 (Apr. 2015), p. 6922.

[222] Friedemann Zenke and Wulfram Gerstner. "Hebbian plasticity requires compensatory processes on multiple timescales". en. In: *Philos. Trans. R. Soc. Lond. B Biol. Sci.* 372.1715 (Mar. 2017), p. 20160259.

[223] Friedemann Zenke and Wulfram Gerstner. "Limits to high-speed simulations of spiking neural networks using general-purpose computers". en. In: *Front. Neuroinform.* 8 (Sept. 2014), p. 76.

[224] Friedemann Zenke, Guillaume Hennequin, and Wulfram Gerstner. "Synaptic plasticity in neural networks needs homeostasis with a fast rate detector". en. In: *PLoS Comput. Biol.* 9.11 (Nov. 2013), e1003330.

[225] L I Zhang et al. "A critical window for cooperation and competition among developing retinotectal synapses". en. In: *Nature* 395.6697 (Sept. 1998), pp. 37–44.

[226] Mengchen Zhu and Christopher J Rozell. "Modeling Inhibitory Interneurons in Efficient Sensory Coding Models". en. In: *PLoS Comput. Biol.* 11.7 (July 2015), e1004353.

[227] Muhammad S A Zilany et al. “A phenomenological model of the synapse between the inner hair cell and auditory nerve: long-term adaptation with power-law dynamics”. en. In: *J. Acoust. Soc. Am.* 126.5 (Nov. 2009), pp. 2390–2412.

[228] Muhammad SA Zilany and Ian C Bruce. “Modeling auditory-nerve responses for high sound pressure levels in the normal and impaired auditory periphery”. In: *The Journal of the Acoustical Society of America* 120.3 (2006), pp. 1446–1466.

[229] Muhammad SA Zilany, Ian C Bruce, and Laurel H Carney. “Updated parameters and expanded simulation options for a model of the auditory periphery”. In: *The Journal of the Acoustical Society of America* 135.1 (2014), pp. 283–286.

[230] Robert S Zucker and Wade G Regehr. “Short-term synaptic plasticity”. en. In: *Annu. Rev. Physiol.* 64 (2002), pp. 355–405.

[231] Joel Zylberberg, Jason Timothy Murphy, and Michael Robert DeWeese. “A sparse coding model with synaptically local plasticity and spiking neurons can account for the diverse shapes of V1 simple cell receptive fields”. en. In: *PLoS Comput. Biol.* 7.10 (Oct. 2011), e1002250.