The Neural Marketplace

Sarah Naomi Lewis
Department of Bioengineering, Imperial College London
Supervisor: Kenneth Harris
Co-supervisor: Claudia Clopath

Submitted for the degree of Doctor of Philosophy

June 30, 2016

Abstract

The 'retroaxonal hypothesis' (Harris, 2008) posits a role for slow retrograde signalling in learning. It is based on the intuition that cells with strong output synapses tend to be those that encode useful information; and that cells which encode useful information should not modify their input synapses too readily. The hypothesis has two parts: first, that the stronger a cell's output synapses, the less likely it is to change its input synapses; and second, that a cell is more likely to revert changes to its input synapses when the changes are followed by weakening of its output synapses. It is motivated in part by analogy between a neural network and a market economy, viewing neurons as 'entrepreneurs' who 'sell' spike trains to each other. In this view, the slow retrograde signals which tell a neuron that it has strong output synapses are 'money' and imply that what it produces is useful.

This thesis constructs a mathematical model of learning, which validates the intuition of the retroaxonal hypothesis. In this model, we show that neurons can estimate their usefulness, or 'worth', from the magnitude of their output weights. We also show that by making each cell's input synapses more or less plastic according to its worth, the performance of a network can be improved.

Contents

1 Preface

The basic biological idea in this thesis was first published in (Harris, 2008). The text of introductory sections 2 and 3 is adapted from the introduction of a manuscript that Kenneth Harris and I wrote together (Lewis and Harris, 2014). The parallel Metropolis-Hastings algorithm in Chapter 4 was my idea and the work in Chapter 6 is largely mine. The general mathematical framework of Chapter 5 we worked out together; the proofs and simulations are mine.

Thanks to Kenneth for giving me a very nice hypothesis to work with, and for many enthusiastic discussions. Thanks also to Claudia (my co-supervisor), to my labmates who were great company, and to my family and James for their loving support.

— Sarah Lewis, May 2016

2 Introduction

The brain is made of billions of neurons, which together form the world's most powerful information-processing machine. Neurons are complex devices, and are individually capable of more than was once supposed. But what is most remarkable is their ability to organize themselves into large networks that coherently guide the behavior of an animal, and constantly learn and adapt to changing circumstances.

Recordings of individual neurons show that their firing encodes diverse types of information, from simple sensory stimuli (Hubel and Wiesel, 1959) to representations of complex features such as locations in space (O'Keefe and Dostrovsky, 1971). Each neuron produces its firing pattern by building on the work of many others. This occurs with no central point of control, suggesting that local processes, carried out independently by single neurons, cause the network to automatically organize into a coherent information processing system.

In this thesis, we describe a new set of rules that could be involved in the self-organisation of neurons into functional networks. This is based on a proposal for a new form of communication between neurons, that we term the 'retroaxonal hypothesis'. The hypothesis would represent a radical new process in neuronal physiology, but is supported by substantial, if circumstantial, experimental evidence.

Before stating the hypothesis, we draw a motivating analogy with another system: the global economy. Here, too, billions of individual processing units self-organize into productive networks. The transfer of money is key to this self-organization. Money flows in the opposite direction to goods and can be conceived of as a signal that indicates to a firm that its products are useful to others. By seeking to maximize their intake of money, producers are approximately maximizing their benefit to end-consumers.

Recursive passing of money up a supply chain, with competition between suppliers at each stage, allows decisions to be decentralized and taken at multiple levels. A car manufacturer, for example, requires steel to produce a consumer product. By competing to sell the types of steel the car manufacters require, steel firms indirectly maximize their benefit to consumers. The car manufacturer does not need to understand the steel-making process, but simply to select the steel that best suits its needs; the steel manufacturer does not need to understand all details of car design. The market allows individuals of limited processing power to form networks that make products far more complex than any of them could produce

individually.

The aim of this thesis is to suggest how a similar form of self-organization could take place in the brain. We will loosely regard neurons as 'entrepreneurs' who 'sell' spike trains to each other. Analogies between the brain and market economy have been made before (Kwee et al., 2001; Balduzzi, 2014). The key novelty of the current theory is a hypothesised form of communication between neurons. We hypothesise that chemical messages passing slowly backward along axons play an analogous role to money in the economy, indicating to neurons how beneficial their output spike trains are to the organism. Although rapid communication via action potentials only occurs unidirectionally forward along axons, the fact that slow chemical messages can flow in the opposite direction has been established for many decades. In the development of the nervous system, competition for such 'retroaxonal' signals determines which neurons live and which die (Hamburger, 1992, 1993; Oppenheim, 1991; Buss et al., 2006). In the adult nervous system, neuronal death is rare, but retroaxonal signals are still conveyed (DiStefano et al., 1992; Zweifel et al., 2005). The present theory suggests that these signals promote a different form of competition between presynaptic neurons, to supply their targets with appropriate information in exchange for a 'payment' returned to them backward along the axon. We argue that this form of competition — as with competition for monetary returns in the economy — promotes the self-organization of networks to allow sophisticated information processing.

The retroaxonal hypothesis is not a learning rule on its own. Rather, it is a rule for modulating the learning rates of other plasticity rules. The retroaxonal hypothesis does not concern rapid learning of associations beteween stimuli and responses. Instead, it concerns how the brain selectively stabilises internal representations of behaviourally relevant stimuli, allowing more and more complex representations to be built up, and allowing behaviour to be tuned to relevant stimuli. In machine learning language, the retroaxonal hypothesis is about feature selection, not feature detection. The form of learning proposed in the retroaxonal hypothesis does not occur within seconds and minutes, but over days, weeks, and years. We will suggest a mechanism by which neural representations that are useful for behavior are gradually selected and consolidated, at the expense of less useful representations.

2.1 Contents of the thesis

Chapter 3 introduces the retroaxonal hypothesis. We review biological evidence that makes the retroaxonal hypothesis plausible, and describe a candidate molecular mechanism. We describe some behavioural learning phenomena which could be examples of retroaxonal learning. We also review the relationship of the retroaxonal hypothesis to other mathematical theories of plasticity in neuronal networks.

In Chapter 4 we consider any neural network (spiking or rate-based, feedforward or recurrent) whose configuration is summarized by a matrix of synaptic weights, and whose performance can be measured by a loss function. We introduce the notion of 'worth' of a cell, which measures its contribution to the performance of the network. We outline a novel learning scheme, 'parallel Metropolis-Hastings' (PMH), applicable to any such network, and define conditions for the proposed learning scheme to function. PMH implements one of the key parts of the retroaxonal hypothesis: that cells whose outputs are useful should make their input synapses less plastic.

In Chapter 5 we present a model network where cells can measure their usefulness, or 'worth', by monitoring the strengths of their output weights. We implement PMH in this network, operating with a standard unsupervised learning rule, and show that it is effective.

In Chapter 6 we discuss more complex networks with recurrent connections between cells. We consider whether cells can use slow retrograde message-passing to evaluate their usefulness, even where that usefulness is rather indirect. In the examples studied, we conclude that most cells can only compute only coarse, order-of-magnitude estimates of their effect on network performance.

3 The retroaxonal hypothesis

3.1 Introduction

In this chapter we will state the retroaxonal hypothesis and put it in its biological context. We describe candidate molecular mechanisms that could support the retroaxonal hypothesis, and we explain how certain types of behaviour or learning could be explained by the retroaxonal hypothesis. We also put the retroaxonal hypothesis in context in the machine learning literature, explaining its relation to other theories such as unsupervised learning, the backpropagation algorithm, and reinforcement learning.

3.2 Outline of the hypothesis

This thesis describes a mathematical model for the role of retroaxonal signals in adult learning, that we term the *retroaxonal hypothesis* (Harris, 2008). The retroaxonal hypothesis posits that the state of a neuron's output synapses, through the passing of retroaxonal messages, controls the stability of the same neuron's inputs. The hypothesis has two components:

1. *Neurons with weak output synapses show unstable input synapses.* A neuron with no functional output synapses serves no benefit to an organism, and such neurons do not survive in early development. In adulthood, we suggest that neurons with weak downstream synapses do not die, but rather exhibit instability in their input synapses. This instability will cause the firing correlates of the neuron to change, until it fires in a manner that causes downstream synapses to strengthen. We assume that a neuron's downstream synapses strengthen specifically when the neuron's activity provides downstream cells with useful information. If a neuron with weak outputs has unstable inputs, its firing pattern will keep changing until finds information useful for downstream cells, and thus for animal behavior.

2. *Strengthening of a neuron's downstream synapses consolidates recent changes in the same neuron's inputs.* By consolidating changes to its input synapses that are closely followed by a strengthening of its outputs, and reverting other input changes, a neuron will keep those specific changes that lead to it conveying more useful information.

In chapter 4 and 5 we will show that either (1) or (2) alone, or a mixture of the two, can be computationally viable as a learning rule.

The two statements in italics in (1) and (2) above are descriptive: they describe what you would expect to see correlated with what, in a network that learns according to the retroaxonal hypothesis. An alternative 'normative' statement of the retroaxonal hypothesis might be as follows: that

1. The less useful a neuron's firing pattern, the more plastic it should make its input synapses (so that it can find a more useful firing pattern).

2. A neuron may infer that it has a useful firing pattern if its output synapses are strong.

The first, descriptive set of statements is the one that we focus on in this chapter, asking whether there are molecular mechanisms that could produce such phenomena.

The retroaxonal hypothesis is not, on its own, a learning scheme. It proposes rules for determining *when* cells should change their input weights, but not *how*. In machine learning language, it sets rules for modulating learning rates. Retroaxonal learning cannot exist except as a component of a learning scheme where some other plasticity rules are also at play.

3.3 Candidate molecular mechanisms

The retroaxonal hypothesis represents a radical addition to the set of mechanisms usually invoked in neuronal network models. We argue for its biological plausibility by proposing candidate molecular mechanisms, while recognising that the specific mechanism we propose here is just a working hypothesis.

The classical form of communication between neurons is the action potential: an electrical impulse conducted rapidly along an axon, resulting in neurotransmitter release at a synapse. In mammalian neurons *in vivo*, action potentials travel in a strictly unidirectional manner from the presynaptic to postsynaptic cell.

Action potentials are, however, just one of many ways that neurons can communicate with each other. Like cells throughout the body, neurons release and receive many signaling molecules other than classical neurotransmitters, and the release of these substances can be controlled by intracellular events other than action potentials. And although classical neurotransmitters signal unidirectionally from the presynaptic to postsynaptic cell, other signals may

propagate retrogradely from postsynaptic to presynaptic (Harris, 2008).

The effects of such retrograde signals need not be restricted to the synapse where they are received, but can be cell-wide, propagated not by electrical conduction but by the (much slower) transport of signaling molecules backward along the axon. Typically, the destination of such messages is the soma and nucleus, where arriving signals are integrated by complex molecular networks, and the results of the computation broadcast to the entire cell.

The retroaxonal hypothesis requires the following:

- Information about synaptic strength, and about recent changes in synaptic strength, must be available at presynaptic terminals.
- Signals conveying this information must pass retrogradely along the axons of cells.
- These retrograde signals must cause the recipient cells to modulate the plasticity of their input synapses.
- Neurons must be able to revert recent changes to their input synapses.

We examine these requirements, and show that each can be plausibly met.

3.3.1 Retrosynaptic signals can communicate information about synaptic strength

Our hypothesis holds that changes in the strength of a neurons output synapses control the stability of the same neuron's inputs. But how can a cell know about the strength of its outputs, given that the induction of synaptic plasticity is mostly believed to occur in the postsynaptic cell? Hebbian plasticity, for example, requires coincident presynaptic firing and postsynaptic depolarization, which is typically detected by NMDA receptors in postsynaptic spines. Nevertheless, a great many molecules are released by the postsynaptic cell during plasticity induction, that have been shown capable of signaling information about synaptic changes to the presynaptic axon terminal. These 'retrosynaptic' signalling molecules include lipids such as cannabinoids (Sjöström et al., 2003); small-molecule gasses such as NO (Hardingham et al., 2013); and proteins, including neurotrophins such as BDNF (Edelmann et al., 2014). In hippocampus, BDNF secretion is increased by stimulation paradigms that cause LTP, and decreased by stimulation paradigms that cause

LTD (Aicardi et al., 2004). Together, the multiple molecules whose release correlates with various forms of synaptic plasticity form a 'population code' that keeps the presynaptic terminal informed of detailed changes in synaptic strength.

Experimental evidence suggests that retroaxonal messengers such as neurotrophins are not only released by tonically strong synapses, but released at particularly high rates when output synapses increase in strength (Aicardi et al., 2004). Thus, the retroaxonal flow of neurotrophins could encode both the strength of an output synapse, and its temporal derivative.

A limitation of the evidence is that we do not know if these neurotrophins are targetted to those specific cells whose output synapses are potentiated. For example, in (Aicardi et al., 2004), concentrations of BDNF were measured by ELISA of slice perfusion medium, not in individual presynaptic cells. Nor is it clear what mechanism would allow a cell with many inputs to send a different retrosynaptic signal from each of its dendrites, corresponding to the weight of each individual synapse. A clue could come from the 'synaptic tagging' theory of LTP, where late LTP is believed to depend on interaction of signals from the soma with a synapse-specific 'tag' (Frey and Morris, 1997). In that theory, the same signal, sent retrogradely from the soma to every dendrite, is differently acted on at each input synapse.

3.3.2 Signals can propagate retroaxonally

'Retroaxonal' signals are signals passing from the axon terminals of a cell, back to the soma. Retroaxonal signals play an essential role in neuronal development, where they are carried by molecules known as *neurotrophic factors*, including the *neurotrophin* family. In the developing nervous system, neurons die if they do not receive sufficient retroaxonal neurotrophic signals (Purves, 1988). In adults, neuronal death is rare, but retroaxonal communication through molecules such as neurotrophins continues (DiStefano et al., 1992).

Compared to action potentials, retroaxonal signals travel slowly, and use different mechanisms. This has been best studied for signals induced by neurotrophins, which initiate retroaxonal transport of a 'signaling endosome' — a small vesicle that carries the neurotrophin molecule, the activated receptor and other associated signaling molecules (Zweifel et al., 2005). The signaling endosome is conveyed back to the soma, where it influences gene expression, for example by activating the transcription factor CREB (cyclic AMP response element binding protein), which during development is crit-

ical for promoting neuronal survival (Riccio et al., 1999, 1997; Watson et al., 2001).

3.3.3 Retroaxonal signals can influence plasticity of a neuron's input synapses

Retroaxonal signals do not only affect cell survival, but can also control the plasticity of a neuron's input synapses (Du and Poo, 2004; Fitzsimonds et al., 1997; Tao et al., 2000; Du et al., 2009). Furthermore, trafficking of signaling endosomes does not stop at the soma, and they can be conveyed directly into neuronal dendrites, where they are able to directly modulate input synapses (Sharma et al., 2010).

3.3.4 Changes to synaptic strengths can be transient

The second part of the retroaxonal hypothesis in 3.2 says that changes to a cell's input synapses should tend to revert if they are followed by weakening of the same cell's output synapses. This requires that cells are able to revert changes to their input synapses, and that a retroaxonal signal communicating the change in downstream weights can propagate fast enough to regulate this process.

In vitro experiments suggest that changes in synaptic strength are often transient. Long-term potentiation (LTP) evoked *in vitro* typically lasts only a few hours, before decaying and leaving synapses in their prior state; this transient form of plasticity is referred to as 'early LTP'. Early LTP does not require protein synthesis, and relies instead on changes such as the phosphorylation and trafficking of AMPA receptors (Frey and Morris, 1997; Malinow and Malenka, 2002).

Early LTP can be consolidated into a permanent 'late LTP', by various cellular signals which typically require phosphorylation of the transcription factor CREB and subsequent protein synthesis (Silva et al., 1998; Barco et al., 2005). This consolidation only happens to synapses that have experienced early LTP, which exhibit a 'synaptic tag'; untagged synapses are unaffected by consolidation. If the consolidation signal is not received, tagged synapses revert to their prior state, while untagged synapses are again unaffected. A conclusion of these results is that synapses undergoing plasticity retain a memory of their previous state. If early LTP is not consolidated, synaptic strengths return after a few hours to the same value they had before potentiation began.

Recent research shows that the synaptic tag may not correspond not to any single molecule, but to a coordinated set of changes in-

cluding phosphorylation of CaM-kinase II and restructuring of the actin cytoskeleton inside a spine (Redondo and Morris, 2011; Rogerson et al., 2014). The consolidation of early LTP into late LTP occurs because tagged synapses are able to capture 'plasticity related products' (PRPs) expressed on a cell-wide basis. The capture of PRPs leads to a permanent increase in synaptic strength in tagged synapses. Several molecules may play the role of PRPs, including protein kinase M zeta, the scaffolding molecule Homer1a, and the neurotrophin BDNF (Redondo and Morris, 2011; Barco et al., 2005).

It is known that the conversion of early to late LTP can depend on dopamine signalling: selective inhibition of certain dopamine receptors in hippocampal slices can prevent late LTP without effect on early LTP (Frey and Morris, 1998). However, the timescales make it plausible that retroaxonal signals could also be involved in consolidation of synaptic tags. Recent experiments show tags lasting for 5 hours under the correct conditions *in vitro* (Li et al., 2014): long enough for a retroaxonal signal to propagate. We hypothesise that a retroaxonal signal could elicit release of PRPs from the soma.

Part of the retroaxonal hypothesis is that 'strengthening of a neuron's downstream synapses consolidates recent changes in the same neuron's inputs'. This imposes a timing requirement not just on retrograde messages, but also on synaptic plasticity in downstream cells. It proposes that one cell (the 'upstream' cell) converts early to late LTP at its *input* synapses with a probability that depends on whether its *output* synapses were strengthened following the early LTP. Thus, each *downstream* cell must make relevant changes to its input synapses within the upstream cell's 'decision window' (the period from induction of early LTP to consolidation). Either the downstream cell has a shorter decision window, or the decision whether to consolidate changes in the upstream cell is based on *early* LTP downstream, that may or may not be consolidated. We do not go into detail about either of these scenarios. However, our model of retroaxonal learning in section 4.8 is flexible enough to allow switching off this part of the retroaxonal hypothesis, while still letting it be the case that neurons with weak output synapses show unstable input synapses.

3.3.5 Candidate mechanism

The data reviewed above suggest a specific candidate mechanism for the retroaxonal hypothesis:

- Neurons release BDNF retrosynaptically across synapses which are strong, or which have been recently strengthened.

- The receipt of BDNF at an axon terminal leads to retroaxonal transport of signaling endosomes.
- These signaling endosomes have effects both at the soma, and at the dendrites.
- At the dendrites, they cause consolidation of early LTP to late LTP in tagged synapses, and suppress early LTP.
- Signaling endosomes arriving retroaxonally to the nucleus trigger gene expression of BDNF and other PRPs. These amplify the retroaxonal signal and allow consolidation of early to late LTP throughout the dentritic tree of the neuron.
- The signal can be passed on to the neuron's own presynaptic partners, forming a recursive chain that ensures that upstream activity is also stable.
- A neuron with only weak outputs receives and synthesizes little BDNF, making its inputs permanently unstable. (Consistent with this idea, a recent study showed that cell-specific knockdown of BDNF causes cells to have smaller spines, a signature of unstable synaptic inputs (English et al., 2012)).

We emphasize that this proposed mechanism remains a hypothesis, that has not been directly tested experimentally. Yet, it should at least serve to show that the phenomena we require for the retroaxonal hypothesis are within the demonstrated physiological capabilities of neurons.

3.4 Where the buck stops: 'end-consumers'

In the economic analogy, we think of neurons as producers of an information product which they 'sell' to downstream cells. But who is the final consumer? And how do these 'consumer' cells judge what information they need from producers, in order to usefully guide animal behavior?

In our framework, a consumer neuron is any cell that receives a *training signal* — i.e. a synaptically conveyed signal that rapidly and directly guides the plasticity of its other input synapses, in a manner that leads to the learning of appropriate behaviors. We refer to neurons that receive no direct training input as 'producers': in the theory, the role of these cells is to produce information required by the consumer cells.

The precise form of plasticity employed by consumer cells is not critical for the retroaxonal hypothesis, provided one condition holds:

that *consumer neurons only strengthen input synapses that provide them with useful information*. In the economic analogy, one could say that the end consumers 'buy only what they need'. Within this constraint, the retroaxonal hypothesis is agnostic about how consumers learn: dopamine-guided reinforcement learning (Schultz et al., 1997), or classical conditioning via Hebbian plasticity, are both compatible with the retroaxonal hypothesis.

Consistent with a partition of cells into 'producers' and 'consumers', training signals do target specific neuronal populations: for example, while the basal ganglia receive extremely strong dopaminergic innervation, the hippocampus and neocortex (particularly sensory cortex), receive far less (Bentivoglio and Morelli, 2005). We might regard the cells in basal ganglia that receive the dopaminergic training signal as 'consumers', and the neocortical neurons that provide their inputs as 'producers'.

3.5 Two possible examples of retroaxonal learning

To clarify how the neural marketplace theory would operate in actual brain circuits, we now consider two examples: classical fear conditioning, and reinforcement learning. In each of these examples, the retroaxonal hypothesis describes how behaviourally relevant firing patterns can be selectively stabilised.

3.5.1 Fear conditioning

Our first example is classical fear conditioning, where we suggest that retroaxonal signals could contribute to stabilizing representations of a 'conditioned stimulus' associated with fear memory.

In classical fear condition, pairing of an aversive unconditioned stimulus (US) with a conditioned stimulus (CS) leads to the CS evoking fear-related behaviors such as freezing. Animals can rapidly learn to fear multiple forms of CS, from simple sensory stimuli such as a previously neutral tone (auditory fear conditioning), to complex cue combinations such as those indicating particular spatial locations (contextual fear conditioniong). A considerable body of evidence suggests that fear conditioning occurs through Hebbian synaptic plasticity in the amygdala: coincident firing of a strong input signaling the US, together with initially weak synaptic inputs signaling a CS, leads to strengthening of the synapses carrying the CS, so that later the CS can drive a response alone (Pape and Pare, 2010). In our theory we would class the amygdalar cells exhibiting this Hebbian plasticity as 'consumers', with the strong input en-

coding the US as their training signal. Information about the CS can come from many different structures including the auditory cortex and thalamus (for auditory fear conditioning) and hippocampus (for contextual conditioning to a particular location in space). We regard cells in those areas as 'producers'.

According to the retroaxonal hypothesis, after fear conditioning has strengthened synapses carrying the CS to the amygdala, a slow retroaxonal signal will pass to the subset of presynaptic neurons whose outputs signal the CS, ensuring that the representation of the CS in these upstream areas remains stable, while representations of other, irrelevant, features are less stable. Thus, we might expect synaptic consolidation in different areas depending on the type of CS: in auditory areas when the CS is a sound, in hippocampal place cells when the CS is a location in space, and so on.

Results of contextual and auditory fear conditioning experiments are consistent with this expectation. When an animal visits a new location, place representations by hippocampal neurons form rapidly (Frank et al., 2004), but are often unstable (Kentros et al., 2004; Agnihotri et al., 2004; Kentros et al., 1998). After contextual fear conditioning, synapses from neurons representing the conditioned location onto neurons mediating fear behavior in the amygdala become strengthened (Anagnostaras et al., 2001). We suggest that this strengthening causes a retroaxonal signal to pass from the amygdala to those precise hippocampal cells encoding the conditioned location, which will stabilize their place fields and ensure that the representation of this location is retained over the long term. Indeed, a wave of CREB phosphorylation (a signature of synaptic consolidation) is found in the hippocampus several hours after contextual fear conditioning (Trifilieff et al., 2006). This delayed CREB phosphorylation is what would be expected from a slow retroaxonal signal from amygdala to hippocampus. Moreover, this wave of hippocampal CREB phosphorylation is *not* seen after auditory fear conditioning (Trifilieff et al., 2006), which suggests that it follows specifically from potentiation of inputs from the hippocampus to the amygdala.

In auditory fear conditioning, the CS is a sound, and again the pattern of plasticity is consistent with a reinforcing, retrograde signal, passing from the amygdala to the population of cells signalling the CS. Some information about the CS arrives at the amygdala from the thalamus, specifically the medial geniculate nucleus (MGN), and strengthening of synapses between MGN and amygdala is believed to underlie auditory fear conditioning (Pape and Pare, 2010). The MGN projects directly to the basolateral amyg-

dala (BLA) but not vice versa. During auditory fear conditioning in rats, there is plasticity in both BLA and MGN (Maren et al., 2001). Despite the lack of direct anterograde connections from BLA to MGN, the plasticity in MGN is suppressed if BLA is silenced during training using muscimol. We suggest that retroaxonal signals from BLA to MGN, dependent on the strength of synapses from MGN onto BLA, contribute to this process.

3.5.2 Reinforcement learning

Our second possible example of the retroaxonal hypothesis in action involves reinforcement learning and corticostriatal projections. The 'law of effect' (Thorndike, 1898a,b) states that if an animal performs a particular action in a particular situation, and this is followed by a rewarding outcome, then the probability the action will be performed in this situation increases. It is believed that dopamine signaling is fundamental to this process, with dopamine release indicating increases in expected future reward (Schultz et al., 1997). In the retroaxonal hypothesis, we regard the recipients of this dopamine signal as 'consumers', and cells providing their inputs (but receiving no dopamine signal) as 'producers'.

The basal ganglia are believed to be central to reinforcement learning (Ito and Doya, 2011), and are heavily innervated by dopamine (Bentivoglio and Morelli, 2005), whereas dopamine innervation of sensory cortex is weak. Although the exact mechanisms of reinforcement learning are still debated, most current hypotheses center on plasticity of the corticostriatal synapse (Costa, 2007; Wickens, 2009; Fee, 2014). It is well established that dopamine controls corticostriatal plasticity (Lovinger, 2010), and theoretical models have suggested how dopamine-gated plasticity might enable reinforcement learning to occur (Wickens, 2009; Fee, 2014; Ito and Doya, 2011; Frank, 2011). It is widely held that corticostriatal projection neurons encode sensory and contextual factors; that firing of specific striatal neurons causes production of specific behaviors; and that strengthening of corticostriatal synapses thereby increases the probability that a specific behavior will be performed in a specific circumstance. In the present theory, we would consider the striatal cells, which receive a dopaminergic training signal, as 'consumers'; corticostriatal projection neurons would be classed as 'producers'. Despite their lack of a direct training signal, we hypothesise that corticostriatal neurons would receive indirect, slow reinforcement, in the form of retroaxonal signals from the striatum. According to the hypothesis, strengthening of a corticostriatal synapse would

cause a retroaxonal signal to pass to the cortical neuron, stabilizing representations of those cortical neurons that encode behaviourally relevant features. Meanwhile, other cortical neurons would continue to change their input synapses until they found a behaviorally relevant variable to encode.

3.6 Unsupervised learning and the retroaxonal hypothesis

In section 3.5, we hypothesised that areas such as cortex and hippocampus contain populations of 'producer' neurons, whose plasticity is not strongly controlled by training signals such as dopamine. Instead, these producers continue to experiment with different input weights until they produce a signal that is used by the 'consumer' neurons.

How might producers select candidate input weights? For the retroaxonal learning scheme to work efficiently, a producer neuron should seek representations of its inputs that are *a priori* likely to be useful. This task — finding salient features in high-dimensional data without a direct training signal — is precisely what *unsupervised learning algorithms* are designed to accomplish. We expect the retroaxonal hypothesis to work as a modulator of learning rates for unsupervised learning rules. Unsupervised learning rules find features in data, and the retroaxonal hypothesis gives a way to selectively stabilise representations of behaviourally relevant features.

Unsupervised learning is frequently used in machine learning to form low-dimensional representations of complex data sets. Artificial unsupervised algorithms include principal component analysis, independent component analysis, and cluster analysis, all of which can be implemented neurally as Hebbian learning rules (Hinton and Sejnowski, 1999). Unsupervised learning has long been suggested as a key computational function of the cortex (Marr, 1970), and several details of cortical synaptic plasticity rules, chiefly STDP, appear consistent with a role in unsupervised learning (Cooper et al., 2004; Clopath et al., 2010; Yger and Harris, 2013; Caporale and Dan, 2008). Implementation of synaptic plasticity rules consistent with the physiology and molecular mechanisms of cortical synaptic plasticity can allow simulated recurrent networks to form unsupervised representations of speech sounds (Yger and Harris, 2013).

3.6.1 Unsupervised learning by hippocampal place cells

We now return to the example of contextual fear conditioning (section 3.5.1). We note that recent data on rat hippocampal place

cells suggest one way that unsupervised learning might operate *in vivo*, and we describe how this could interact with the retroaxonal hypothesis.

When a rat is introduced into a new spatial environment, place fields representing locations in this environment appear essentially instantaneously (Hill, 1978; Wilson and McNaughton, 1993; Frank et al., 2004). These place fields are initially coarse but over a period of minutes, they grow smaller, tighter, and more reliable (Wilson and McNaughton, 1993; Frank et al., 2004).

The initial appearance of place fields appears to result from a reduction of synaptic inhibition. When a rat is introduced to a novel environment, the firing rate of putative fast-spiking interneurons drops rapidly (Wilson and McNaughton, 1993; Nitz and McNaughton, 2004; Frank et al., 2004), but returns to baseline during subsequent exploration sessions. By allowing previously subthreshold inputs to drive spiking, this decrease in inhibition may allow the cell to fire with some degree of spatial specificity when an animal first visits a new environment, even before any synaptic plasticity has taken place. This possibility is supported by computational models, which show that a hippocampal neuron with even a randomly-weighted combination of inputs from entorhinal grid cells would show some spatial specificity (de Almeida et al., 2009), and is further supported by the observation that injecting depolarizing current reveals spatially specific firing even in hippocampal neurons that showed prior no place-related activity (Lee et al., 2012). It seems likely that the refinement of place fields happens via Hebbian LTP of each place cell's input synapses (Solstad et al., 2006; Franzius et al., 2007).

Thus, it appears that a coarse place field generated by random initial connectivity can form a 'seed' from which Hebbian plasticity sculpts a coherent place field. This unsupervised learning strategy, requiring no explicit training signal, enables 'producer' place cells of the hippocampus to form representations of space that are potentially useful.

Of relevance to the retroaxonal hypothesis, place fields are sometimes stable from one day to the next, and sometimes not (Kentros et al., 2004). The retroaxonal hypothesis suggests that useful place fields could be selectively stabilised by retroaxonal signals: cells with useful place fields form strong synapses onto downstream consumer cells and receive retroaxonal 'payment' in return, which causes them to stabilise their input synapses. In support of this idea, most of the novel place fields formed after exposure to a novel environment disappear after one or two days; furthermore, the place fields that

remain tend to be those that encode the most spatial information (Karlsson and Frank, 2008).

3.6.2 Identifying 'free memory' and preventing 'overwriting'

The retroaxonal hypothesis is not the only hypothesis that exists concerning the modulation of plasticity rates in the brain.

While the sensory cortices and hippocampus receive far less dopamine innervation than the striatum, they are strongly innervated by cholinergic fibers (Frotscher and Léránth, 1985; Eckenstein et al., 1988), which are active at times of high alertness, and lead to increased synaptic plasticity (Hasselmo, 2006). Stimuli occurring at times of high alertness are more likely to be behaviorally relevant, so these cues may allow producer cells to guess that a representation of their current inputs may be useful in future to their downstream targets. However, uniform plasticity of all neurons receiving cholinergic signals could cause the loss of existing, useful representations that are already encoded in their synaptic weights. The retroaxonal hypothesis could explain how it is that we can learn, without forgetting important things. Retroaxonal signals could dictate which cells become plastic in response to this cholinergic signal, protecting old, useful representations, yet allowing new representations to be formed. Computationally, we may think of retroaxonal signals as identifying free memory and preventing overwriting of useful information.

3.7 Relation to other theories of network plasticity

It should be clear by now that the retroaxonal hypothesis is not the same as the backpropagation algorithm, nor is it the same as reinforcement learning with a spatially diffuse reward signal received by all neurons. We now explain in more detail the relationship of the retroaxonal hypothesis to backprop, to reinforcement learning, and to theories involving feedback connections.

3.7.1 A 'retroaxonal rule' does nothing on its own

The retroaxonal hypothesis says that neurons with strong output synapses should tend to have less plastic input synapses. We call this a 'retroaxonal rule'. As already emphasised, on its own, a retroaxonal rule is not a learning scheme: it just modulates the learning rates of other plasticity rules.

In the neural marketplace hypothesis, rapid plasticity occurs through conventional mechanisms, such as Hebbian unsupervised

learning, supervised learning, and reinforcement learning. The novelty is a second, slower, 'consolidation' process, which lowers the learning rates of cells with strong output synapses, and so (we hypothesise) selectively stabilises the more useful results of those conventional learning mechanisms. The retroaxonal hypothesis is not a model of how we learn new information in minutes or hours, but rather for how skills might be built up over days, weeks, months or years.

3.7.2 Reinforcement learning

Retroaxonal learning should not be confused with reinforcement learning that uses a spatially diffuse reward signal. In reinforcement learning, synaptic plasticity is typically modulated by a single reward signal for the whole network. By contrast, in retroaxonal learning, cells use their output synaptic weights as cell-specific reward signals.

Reinforcement learning can suffer from a 'credit assignment' problem: if a large number of neurons or synapses make changes, it can be unclear which change was responsible for the improvement in performance. A number of solutions have been proposed (Friedrich et al., 2011; Urbanczik and Senn, 2009; Vasilaki et al., 2009; Izhikevich, 2007; Fremaux et al., 2010; Florian, 2007), which typically involve modulation of local synaptic plasticity by a global reinforcement signal, biologically assumed to correspond to dopamine. While this global reinforcement signal is not *cell*-specific, it can be fast, i.e. *time*-specific: immediate rewards can modulate plasticity of synapses that were recently active. If changes to different synaptic weights are separated in time, this can solve the credit assignment problem.

The present theory proposes a complementary mechanism. We show that in retroaxonal learning, retrograde signals can act as slow but cell-specific reward signals, telling each producer how important it is to the consumers. This theory may be particularly appropriate for areas such as sensory cortex, which receive little direct dopamine innervation or any other apparent training signal, but where representations develop that appear tailored to an animal's behavioral requirements (Sigala and Logothetis, 2002; Kuhl et al., 1992).

3.7.3 The backpropagation algorithm

The artificial neural network algorithm that has seen most use in real-world applications, does involve retrograde signals. This algo-

rithm is known as 'error backpropagation' or 'backprop' (Rumelhart et al., 1986). In the backprop net, the firing of activity of neurons is determined by classical anterograde transmission. However, synaptic plasticity in this network requires a training signal to flow backwards along the same connections, so that each neuron integrates a training signal from precisely those cells to which it sends axons.

In common with backprop, the retroaxonal hypothesis posits that each cell should integrate retrograde signals from its output synapses. However, the timescale and the effect of the retrograde signals is different. For a retroaxonal rule, the retrograde signals can be slow, and they do not instruct a cell whether to potentiate or depress any particular synapse. By contrast, in backprop, the retrograde signals must be fast, and they instruct cells how to strengthen or weaken each synapse to improve the network's response to the current input.

Simple versions of backprop train a layered, feedforward network with a single 'output node' in the final layer. The output node receives, as a training signal, a target output $y(t)$, while its actual output is $\hat{y}(t)$. Backpropagating signals allow each upstream neuron to measure the partial derivative of $y(t) - \hat{y}(t)$ with respect to each of its input synaptic weights, and thus to do gradient descent of the mean-square loss function $\overline{(y(t) - \hat{y}(t))^2}$.

While the backprop algorithm rapidly saw success in real-world applications, it was immediately recognized it was unlikely to be a model for neuronal plasticity in the brain (Crick, 1989). Although neurons are certainly capable of carrying retroaxonal signals, the retroaxonal signals found in real neurons are simply too slow. In the backprop algorithm, weight changes are based on the instantaneous correlation of presynaptic activity and the retroaxonal training signal, requiring that the training signal arrives while the input pattern is still being presented. This requires conduction in a matter of milliseconds, rather than the minutes or hours required for physical transport of retroaxonal chemical signals (Zweifel et al., 2005).

In the neural marketplace theory, retroaxonal signals are *selective*, not *instructive*: they tell each neuron *how useful it is*, rather than *how to be more useful*. To measure usefulness, we hypothesise that retroaxonal signals need only to encode the strength of downstream synapses, and not an instant evaluation of current firing patterns. Such selective retroaxonal signals need not be conveyed in milliseconds, but could take minutes, hours, or more, consistent with known biology of retroaxonal signals.

The difference between the retroaxonal hypothesis and the backpropagation algorithm can be illustrated using the economic anal-

ogy. In a market economy, consumers choose which products to buy; but it is rare for a consumer to explain to the supplier what changes in the manufacturing process would make a product more attractive. It is up to the producer to experiment, and discover what sells. Similarly, the retroaxonal hypothesis posits that a 'producer' cell can experiment autonomously with changes to its input synapses, and retain configurations that result in its output being 'bought' by downstream neurons.

3.7.4 Theories involving feedback connections

We have suggested that cell-specific reinforcement is conveyed to neurons by retroaxonal feedback. Could this reinforcement come instead through conventional anterograde transmission of action potentials along separate feedback pathways? There are several difficulties with this proposal. First, for some of the pathways along which we hypothesise retroaxonal signals flow — such as the corticostriatal pathway — there are no direct feedback connections. Anterograde passage of information from striatum to the sensory cortex involves an indirect, polysynaptic pathway involving multiple steps in basal ganglia, thalamus, and other cortical regions (the sensory cortex is not innervated by those thalamic regions that receive basal ganglia input). Second, even in cases where feedback connections are direct (such as from motor cortex to sensory cortex (Petreanu et al., 2012)), the feedforward and feedback projections are both highly divergent and highly convergent. Thus, if a consumer cell in motor cortex fires action potentials to signal to sensory cortex that a recent change in input provided useful information, it is entirely unclear how it would target the particular neuron in sensory cortex that made the change. Third, in cases where a role for top-down projections have been experimentally established, they appear not to be specifically involved in learning, but rather in cognitive control processes such as attention (Moore and Armstrong, 2003; Harris, 2013).

3.8 Conclusion

In this chapter we introduced the retroaxonal hypothesis (3.2), and put it in context. We explained that on its own, a retroaxonal learning rule does nothing: it is a way of modulating learning rates of other learning rules.

In (3.3), we argued that the retroaxonal hypothesis does not require any biologically implausible signalling, and presented candi-

date molecular mechanisms. We reviewed biological evidence that retrosynaptic signals can communicate information about synaptic strength; that signals can propagate retroaxonally; that retroaxonal signals can affect the plasticity of a neuron's input synapses; and that sometimes changes to synaptic strengths are transient. We presented a candidate molecular mechanism that could support retroaxonal learning.

In (3.4) we introduced the idea of a 'consumer' cell which receives a direct training signal and selects inputs from other cells. In the retroaxonal hypothesis, the retrograde signals all originate at consumers, though they may be passed on recursively from producer to producer.

In (3.5) we discussed two well-studied learning paradigms in animals, which could be examples of retroaxonal learning: fear conditioning and reinforcement learning.

In (3.6) we discussed the relationship of the retroaxonal hypothesis to unsupervised learning and explained how retroaxonal learning could interact with unsupervised learning of hippocampal place fields, selectively stabilising those which are useful.

In (3.7) we explained the relationship of retroaxonal learning to backprop, to reinforcement learning with a global reinforcement signal, and to theories involving feedback connections.

4 Learning using 'worth' in an arbitrary network

4.1 Introduction

In this chapter, we begin to turn the retroaxonal hypothesis into a mathematical model.

We begin with the second version of the hypothesis that was given in (3.2):

Claim 1 The less useful a cell's firing pattern, the more plastic its input synapses should be.

Claim 2 By measuring the strengths of its output weights, a cell can tell how useful its current firing pattern is. The stronger its output weights, the more useful its firing pattern probably is.

In this chapter, we consider claim 1 without regard to claim 2. We define a kind of learning scheme which encapsulates claim 1 — that cells should modulate their learning rates according to their usefulness — and we define conditions for this kind of learning scheme to successfully improve network performance.

In chapter 5 we will consider claim 2, and demonstrate that in a specific model network, the strength of a cell's output synapses does indeed predict its usefulness. Together, claims 1 and 2 and these two corroborating chapters complete a computational model of retroaxonal learning in a simple network, and in 5.8 we show a retroaxonal rule working in a simulation.

4.2 Related theories: budget perceptrons and optimal brain damage

Computationally, the retroaxonal hypothesis is a new approach to an old problem of attaining good performance with a limited number of neurons. This is a biologically relevant problem since maintaining a neuron costs space and energy. It is also relevant to artificial neural networks used in machine learning: networks with more nodes and more connections take longer to train and simulate, and require more RAM.

The algorithm we propose, PMH, works by systematically stabilising parts of a network that are useful, while allowing others to be plastic. To put it in context, we mention a number of other schemes which, like ours, aim to make good use of a limited pool of neurons.

The first set of algorithms are variations on the perceptron (Rosenblatt, 1958). The perceptron is an algorithm for learning a binary classification from a series of examples. We write $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots$ for the stream of examples, where $\mathbf{x}_1, \mathbf{x}_2, \ldots$ are stimuli and $y_1, y_2, \ldots$ are labels, $y_t \in \{1, -1\}$. The perceptron retains a list of past examples, used to classify each new stimulus as it arrives. When a stimulus is misclassified it is added to this list of reference examples.

The class of stimulus t is predicted to be sign $(\sum_i \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}_t))$ where K is some kernel function and α_i are weights computed online, differently for each variant of the algorithm. $K(\mathbf{x}_i, \mathbf{x}_t)$ is a measure of similarity between the 'template' $\mathbf{x}_i$ and the current stimulus $\mathbf{x}_t$; in the original perceptron, K was the standard dot product.

The perceptron can be implemented with a neural network where in one layer, each neuron computes $K(\mathbf{x}_t, \mathbf{x}_i)$ for a different i, and a readout node then computes a weighted sum of these values. This neural network implementation requires one kernel-computing neuron for each member of the 'active set' $\{i : \alpha_i \neq 0\}$.

A problem is that the size of the active set (hence, the number of neurons used) can grow without limit. Biologically, this is unrealistic; in a simulation, it can mean the algorithm grinds to a halt or runs out of memory. A number of variants of the perceptron algorithm aim to solve the problem and constrain the size of the active set. These include

- The 'budget perceptron' (Crammer et al., 2004). An example may be removed from the active set if it is correctly classified by the current classifier with a large margin.
- The 'randomised budget perceptron' (Cesa-Bianchi and Gentile, 2006; Cavallanti et al., 2007): before adding a new item to the active set, an old example is selected at random and discarded.
- The 'forgetron' (Dekel et al., 2008): on each iteration, the weight α_i of each example i in the active set is shrunk, and each time a new example is added to the active set, the *oldest* example is removed. The shrinkage means that the oldest example has a tiny weight, and its removal has little effect on the current classifier.

Computationally, changing a neuron's synaptic weights is the same as removing the neuron and replacing it with another that has different synaptic weights. Thus, one can view the randomised budget perceptron and forgetron as rules for modifying the synapses of a

fixed set of neurons, rather than adding and removing neurons. In this chapter, we propose another way of selecting which neurons to change: we make the chance of changing a cell's input weights an explicit function of the usefulness of its current output. We are agnostic as to how the new weights should be selected.

Also relevant is the work of LeCun et al. (1989) on 'optimal brain damage' (OBD). In OBD the usefulness, or 'saliency', of individual parameters, is computed, and used to decide which parameters can safely be deleted. Because it is also relevant to chapter 5, we delay further discussion of OBD to section 5.10.

4.3 Worth

The retroaxonal hypothesis involves a notion of the 'usefulness' of a cell. To construct relevant models we need to make a mathematical definition of 'usefulness'. We choose the most natural definition in the machine learning context: if a network is trying to optimise some loss function L then the usefulness, or 'worth', of a cell, is the increase in L that would result if it fell silent. The worth of a *set* of cells $\mathbb{I}$ is the increase in L that would result if all of the cells in $\mathbb{I}$ fell silent.

By 'silencing cells' we mean setting their firing rates to 0 (in a firing-rate network) or making them never spike (in a spiking network). Whether silencing cells like this is biologically plausible is moot (though TTX might do it), since our learning scheme will not involve actually silencing any cell. Rather, 'if these cells were silenced' is just a convenient baseline against which to define the usefulness of their current firing pattern.

Our definition of 'worth' is much like 'saliency' in the theory of optimal brain damage (LeCun et al., 1989), although that work deals mostly with saliency of *parameters*, not of neurons. However, we use the term 'worth' to avoid confusion with other meanings of 'saliency' in the computational neuroscience literature (a feature is 'salient' if it 'jumps out' at the observer, but may be useless to observe). The word 'worth' is chosen for its meaning of 'usefulness' or 'merit' — not the technical accounting sense of a sum of assets and liabilities.

In this chapter we show that if one cell changes its input synapses and its worth goes up, the network performance improves. We show that if neurons can independently estimate their worths, then letting each individual neuron use a running estimate of its worth as a reinforcement signal can improve performance of the network as a whole. We propose a scheme where a neuron is more likely to experiment with changes to its input synapses if its worth is low,

and where a neuron tends to revert changes that reduce its worth.

In chapter 5 we will show that in specific model networks, neurons can estimate their worths by passage of slow retroaxonal signals. Specifically, we show that the worth of a cell is large if it has strong output synapses onto a supervised output layer. In these models, cells can use retroaxonal messages to estimate their worth continuously without actually being silenced.

4.4 Notation

In this chapter we consider a very general network, whose free parameters are its synaptic weight matrix $\mathbf{W}$. The network could consist of either spiking or rate neurons, and could have a recurrent or feedforward architecture.

The quality of the network output is measured by some loss function L. L is a function only of the network activity, but since the network's only free parameters are its synaptic weight matrix $\mathbf{W}$, the loss L can be considered a function of the weights, $L(\mathbf{W})$.

Constraints on $\mathbf{W}$, such as sparsity or Dale's law, are to be encoded in a second function $p_0(\mathbf{W})$. p_0 can encode soft preferences as well as hard constraints, for example p_0 can encode a tendency to prefer small synaptic weights.

We defined the 'worth' of a cell (or set of cells) as the change in L that would arise if that cell (or those cells) were to fall silent. We introduce the special notation $\varnothing_{\mathbb{I}}$ for silencing cells $\mathbb{I}$. Thus, $L(\mathbf{W}; \varnothing_{\mathbb{I}})$ is the value the loss function takes after cells $\mathbb{I}$ are silenced We write

$$\$_{\mathbb{I}} \equiv L(\mathbf{W}; \varnothing_{\mathbb{I}}) - L(\mathbf{W})$$

for the worth of cells $\mathbb{I}$, and

$$\$_i \equiv L(\mathbf{W}; \varnothing_i) - L(\mathbf{W})$$

for the worth of a single cell i.

The synaptic weight matrix $\mathbf{W}$ has elements W_{ij}, where W_{ij} represents the strength of a synapse from cell j to cell i. For this chapter we define $\mathbf{w}_i$ to be the vector of input weights to cell i.

4.5 Cells should aim to increase their worth

This is a very simple idea, and crucial to PMH: *if a set of cells change their input synapses, then the change in their worth matches the change in the network loss function.* Explicitly, writing $\mathbb{I} = \{i : \mathbf{w}_i \neq \mathbf{w}'_i\}$, we note

$$L(\mathbf{W}) - L(\mathbf{W}') = \$_{\mathbb{I}}(\mathbf{W}') - \$_{\mathbb{I}}(\mathbf{W}) \tag{1}$$

This is because by definition,

$$\$_{\mathbb{I}}(\mathbf{W}) = L(\mathbf{W}; \varnothing_{\mathbb{I}}) - L(\mathbf{W}) \tag{2}$$
$$\$_{\mathbb{I}}(\mathbf{W}') = L(\mathbf{W}'; \varnothing_{\mathbb{I}}) - L(\mathbf{W}') \tag{3}$$

Clearly if cells $\mathbb{I}$ have been silenced, then it does not matter what their input weights were, so

$$L(\mathbf{W}; \varnothing_{\mathbb{I}}) \equiv L(\mathbf{W}'; \varnothing_{\mathbb{I}})$$

Therefore, subtracting (2) from (3) gives equation (1).

4.6 Learning in 'plasticity' and 'consolidation' phases

The retroaxonal hypothesis posits that cells experiment with changes to their input synapses, sometimes reverting changes that prove to be unhelpful. We model this by iteration of two phases: a 'plasticity' and a 'consolidation' phase. Biologically, the plasticity phase corresponds to early LTP, and the consolidation phase corresponds to the (selective) conversion of early LTP to late LTP (Redondo and Morris, 2011; Rogerson et al., 2014). In our model, in the plasticity phase, a subset of cells $\mathbb{I}$ make changes to their input synapses; in the consolidation phase, each cell in $\mathbb{I}$ has the opportunity to revert its changes.

The relation between two algorithm parameters α, β will dictate at what stage worth is taken into account. At one extreme parameter setting, worth has no effect on whether a cell will propose a change, but the decision to accept/reject a proposed change depends on how the change affected the cell's worth. At the other extreme, a cell uses its worth to decide whether to propose a change, but never rejects a change. These two extremes correspond to the two halves of the retroaxonal hypothesis as stated in 3.2.

4.7 The Metropolis-Hastings algorithm

Based on the above, we propose a model of retroaxonal learning that is closely related to the standard Metropolis-Hastings algorithm.

The Metropolis-Hastings algorithm (Metropolis et al., 1953; MacKay, 2003) is at the heart of stochastic optimization schemes like simulated annealing (Kirkpatrick et al., 1983). Like our proposed learning scheme, the generic Metropolis-Hastings algorithm cycles through two steps: 'proposing' a new value (in our case, new synaptic weights) and then 'accepting' or 'rejecting' this change. In general, given some 'target distribution' $p(\mathbf{W})$, the algorithm can

generate a random sequence of values $\mathbf{W}(1), \mathbf{W}(2), \ldots$ such that as $n \to \infty$, the probability distribution of $\mathbf{W}(n)$ tends to p.

Generic Metropolis-Hastings. The algorithm, PMH, which we will later describe is not exactly a Metropolis-Hastings algorithm, but a variant. However, by way of introduction we describe the basic Metropolis-Hastings algorithm. The standard Metropolis-Hastings algorithm is as follows. $\mathbf{W}$ is initialised at some value $\mathbf{W}(0)$. In the nth iteration, we sample a value $\mathbf{W}'$ from a **proposal distribution** q which may depend on $\mathbf{W}(n)$. Then we compute

$$a = \min\left(1, \frac{p(\mathbf{W}')q(\mathbf{W}(n)|\mathbf{W}')}{p(\mathbf{W}(n))q(\mathbf{W}'|\mathbf{W}(n))}\right)$$

With probability a we 'accept' $\mathbf{W}'$, setting $\mathbf{W}(n+1) = \mathbf{W}'$; otherwise we 'reject' the proposal and set $\mathbf{W}(n+1) = \mathbf{W}(n)$. The aim of learning is to lower the loss $L(\mathbf{W})$ while satisfying some constraints encoded in $p_0(\mathbf{W})$. Therefore, we consider a target distribution

$$p_\beta(\mathbf{W}) \propto p_o(\mathbf{W}) \exp(-\beta L(\mathbf{W}))$$

where β is some positive number. MacKay's 'practical Bayesian framework for backprop networks' works with probability distributions of this form (MacKay, 1992). Sampling from p_β will stochastically explore different values of $\mathbf{W}$, preferring values that make L small. With this choice of target distribution, the 'acceptance probability' a becomes

$$a(\mathbf{W}'; \mathbf{W}) = \min\left(1, \frac{p_0(\mathbf{W}')q(\mathbf{W}|\mathbf{W}')}{p_0(\mathbf{W})q(\mathbf{W}'|\mathbf{W})} e^{\beta(L(\mathbf{W})-L(\mathbf{W}'))}\right)$$

The factor $e^{\beta(L(\mathbf{W})-L(\mathbf{W}'))}$ favours proposals that improve the network's output. The factor $\frac{p_0(\mathbf{W}')q(\mathbf{W}|\mathbf{W}')}{p_0(\mathbf{W})q(\mathbf{W}'|\mathbf{W})}$ adjusts for bias in q: if q satisfies detailed balance for p_0 then the acceptance probability can become simply

$$a(\mathbf{W}'; \mathbf{W}) = \min\left(1, e^{\beta(L(\mathbf{W})-L(\mathbf{W}'))}\right)$$

If q already satisfies detailed balance for p_0, then just sampling $\mathbf{W}'$ from $q(\mathbf{W})$ at each timestep would generate samples from p_0 in the long run. Metropolis-Hastings takes this scheme that generates samples from p_0, and converts it to one that samples from p_β, i.e., a scheme that has a greater tendency to make L small.

The Metropolis-Hastings algorithm does not prescribe what the proposal distribution q should be, but the rate of convergence depends on selecting a good q. In the next section we will see that a

good proposal distribution q in Metropolis-Hastings corresponds to a good unsupervised learning rule in PMH.

In Metropolis-Hastings, the parameter β is known as 'inverse temperature'. When β is large, the algorithm becomes a greedy optimisation of L, whereas with $\beta = 0$ it will sample from p_0, ignoring L. The choice of β also dictates a trade-off between speed of learning and the quality of the eventual solution: as a rule of thumb, the time taken for the distribution of $\mathbf{W}(n)$ to converge to p_β increases as β increases (MacKay, 2003).

What is the advantage of a stochastic scheme over deterministic optimisation of L? First, even if we did want to simply minimise L, we might do it by first sampling from p_β. This is a standard approach: the process of running the Metropolis-Hastings algorithm while progressively raising β, known as simulated annealing, is commonly used to find global optima of analytically intractable functions (where simple greedy optimisation would likely get stuck in local optima). Second, $\mathbf{W}$ may represent one of many modules in the brain attempting to meet the same requirement, and it may be difficult to measure L exactly. A degree of diversity can improve the odds that at least some modules will succeed.

One can imagine a way for simple Metropolis-Hastings to be used for learning — but we do not suggest it. To apply simple Metropolis-Hastings, an animal experiments with a change to the weights of the synapses of multiple neurons, proposing new weights $\mathbf{W}'$ drawn from $q(\mathbf{W}'|\mathbf{W})$. Then by attempting to perform a behavioral task, the animal determines whether the loss function L has increased or decreased. If loss has decreased, all the changes are retained, but if loss has increased the weights are likely to revert to $\mathbf{W}$. A 'consolidation signal' could be encoded by global dopamine release, occurring with probability $a_\beta(\mathbf{W}'; \mathbf{W})$. Such a scheme would be compatible with numerous aspects of known neurophysiology: the 'synaptic tagging' literature shows that changes to synaptic weights are usually transient unless consolidated by a second signal; and dopamine is one of several signals able to cause this consolidation (Huang and Kandel, 1995; Sajikumar and Frey, 2004; Kentros et al., 2004).

That scheme would theoretically, eventually, converge to the desired probability distribution $p_\beta(\mathbf{W})$, but suffers from a credit assignment problem. When changes are proposed to multiple cells, these are accepted or rejected wholesale, even if some cells changed for the better and some for the worse.

4.8 Parallel Metropolis-Hastings (PMH)

We dub our proposed model of retroaxonal learning 'parallel Metropolis-Hastings' (PMH). In PMH, individual cells decide *in parallel* whether to propose, accept, or reject changes to their input synapses. Now, in a plasticity phase, each neuron may make an experimental change to its input synapses ('propose a change'), and in the subsequent consolidation phase, each neuron may decide to revert its changes ('reject') or keep them ('accept'). Because neurons decide in parallel whether to propose/accept/reject changes to their input synapses, we call this 'parallel Metropolis-Hastings' or 'PMH'. This is no longer strictly a Metropolis-Hastings algorithm, because $\mathbf{W}(n+1)$ may be neither $\mathbf{W}(n)$ nor the proposed $\mathbf{W}'$ (containing all the changes proposed by all cells), but a chimera.

For PMH, we assume we already have some unsupervised learning rule which gives rise to a reversible Markov process on $\mathbf{W}$, with stationary distribution p_0. The unsupervised learning rule corresponds to a proposal distribution for each cell: $q_i(\mathbf{w}_i; \mathbf{W})$ is the distribution of input weight vectors $\mathbf{w}_i$ at which cell i may arrive by running the unsupervised rule for some specified length of time. Randomness in q_i may arise because the unsupervised learning rule itself is stochastic, or because the unsupervised learning is driven by unpredictable stimuli.

PMH controls *when* the unsupervised learning rule runs, in a way that causes the loss L to be reduced over time. This is achieved by selectively stabilising the input weights of those cells whose outputs are useful. The unsupervised rule (which implies q_i) is responsible for finding salient features in inputs; PMH stabilises representations of *useful* features. By selectively stabilising useful features we match the experimental observation that more behaviourally relevant features are represented more stably and by larger numbers of cells, even when they are just as salient as irrelevant features (Sigala and Logothetis, 2002; Kentros et al., 2004). Just as the performance of Metropolis-Hastings depends on the proposal distribution, the performance of PMH depends on the choice of unsupervised learning rule. Convergence will be fastest when the unsupervised rule generates synaptic weight vectors that are likely to improve performance, and does so quickly.

At each step of the PMH algorithm the synaptic weight matrix $\mathbf{W}(n)$ is updated to a new value $\mathbf{W}(n+1)$, via three stages:

1. Each neuron decides independently whether to propose a change to its input weights. Cells of low worth propose changes more readily: neuron i will propose a change with probaility

$\kappa e^{-\alpha \$_i(\mathbf{W}(n))}$ (κ, α are parameters).

2. Each neuron i that is proposing a change selects new input weights $\mathbf{w}'_i$ from a proposal distribution q_i, which may depend on $\mathbf{W}$.

3. Each neuron i that proposed a change measures the change in its worth $\$_i(\mathbf{W}') - \$_i(\mathbf{W}(n))$. Cells should tend to retain changes that increase their worth: a cell consolidates the change (sets $\mathbf{w}_i(n+1) = \mathbf{w}'_i$) with probability $a_i = \min\left(1, e^{(\beta-\alpha)(\$_i(\mathbf{W}')-\$_i(\mathbf{W}))}\right)$, and otherwise rejects the change (set $\mathbf{w}_i(n+1) = \mathbf{w}_i(n)$).

We use the parameter $\alpha \in [0, \beta]$ to dictate how strongly the chance of proposing a change depends on the worth $\$_i$ (neurons of higher worth being more reluctant to propose changes), and use the parameter $\kappa > 0$ to control the total number of cells proposing changes.

Apart from the obvious intuition that a change is more likely to be beneficial to a useless cell (with low worth) than to a useful one (with high worth), there are other, subtler reasons for setting $\alpha > 0$ and making the chance of a proposal dependent on worth:

- The Metropolis-Hastings algorithm makes no guarantee as to the distribution of proposed values $\mathbf{W}'$, merely the stationary distribution of *accepted* values $\mathbf{W}(n)$. In a neural network implementing the scheme, proposed new weights *are* actually used for a short time (in order to measure their effect on performance), even if they then revert. Thus, we are likely to incur a transient penalty while evaluating changes to cells of high worth.

- There is an 'opportunity cost' to experimenting with a cell that is unlikely to improve. We will see in section 4.9 that PMH will likely fail to converge if too many cells experiment with changes at once. The limited set that do experiment with changes should include a preponderance of cells that are likely to improve, i.e., cells of low worth.

- In our implementations of PMH (section 5.8), worths are estimated rather than measured exactly, and they are estimated using slow retrograde signals, so it takes some time to arrive at a good estimate. A cell which has not changed for a long time may know its worth $\$_i$ fairly accurately. When it proposes new weights, it has only a limited time to evaluate its new worth $\$'_i$ before deciding whether to consolidate the changes. By increasing α and decreasing β we transfer emphasis from the

accept/reject decision (based on a hasty estimate of change in worth) to the proposal/no-proposal decision (based on a more considered estimate of worth).

Since PMH selectively destabilises cells with small worth, we might predict that as it runs, cells' worths become less diverse and converge around a common value. Indeed, that is what happens in the simulations in section 5.8.

4.9 Convergence of PMH

The generic Metropolis-Hastings algorithm converges because it generates a Markov chain on $\mathbf{W}$ that satisfies detailed balance for the target distribution p, that is

$$\forall \mathbf{W}, \mathbf{W}' : p(\mathbf{W})\mathbb{P}\left(\mathbf{W} \to \mathbf{W}'\right) = p(\mathbf{W}')\mathbb{P}\left(\mathbf{W}' \to \mathbf{W}\right) \tag{4}$$

In general, PMH does not satisfy equation (4) for $p = p_\beta$ and we have no guarantee that it will converge to p_β. In this section we discuss conditions that would give an analytical guarantee for PMH to satisfy equation (4) for p_β, and therefore to converge. We note that these conditions will not be met exactly in most simulated networks, let alone a real neural network. Nonetheless in chapter 5 we will show a way that the conditions for convergence can be approximately met, and in section 5.8 we show that the scheme works in simulations.

For reference, let us look at the transition probabilities of standard Metropolis-Hastings and how they satisfy detailed balance. In generic Metropolis-Hastings for p_β, the transition probabilities are

$$\mathbb{P}\left(\mathbf{W} \to \mathbf{W}'\right) = q(\mathbf{W}'|\mathbf{W})a(\mathbf{W}';\mathbf{W})$$

where

$$a(\mathbf{W}';\mathbf{W}) = \min\left(1, \frac{p_0(\mathbf{W}')q(\mathbf{W}|\mathbf{W}')}{p_0(\mathbf{W})q(\mathbf{W}'|\mathbf{W})} e^{\beta(L(\mathbf{W})-L(\mathbf{W}'))}\right)$$

This choice of a guarantees

$$\frac{a(\mathbf{W}';\mathbf{W})}{a(\mathbf{W}';\mathbf{W})} = \frac{p_0(\mathbf{W}')q(\mathbf{W}|\mathbf{W}')}{p_0(\mathbf{W})q(\mathbf{W}'|\mathbf{W})} e^{\beta(L(\mathbf{W})-L(\mathbf{W}'))}$$

so that the ratio of forward and back probabilities is given by

$$\frac{\mathbb{P}\left(\mathbf{W} \to \mathbf{W}'\right)}{\mathbb{P}\left(\mathbf{W}' \to \mathbf{W}\right)} = \frac{p_0(\mathbf{W}')e^{-\beta L(\mathbf{W}')}}{p_0(\mathbf{W})e^{-\beta L(\mathbf{W})}} = \frac{p_\beta(\mathbf{W}')}{p_\beta(\mathbf{W})}$$

That is, detailed balance for p_β is satisfied. To show when the same would be true in PMH, so that PMH obtains detailed balance for p_β, we first define 'additivity' of worths. We say a set of neurons $\mathbb{I}$ is an **additive set** if for all $\mathbb{J} \subseteq \mathbb{I}$ and for all possible $\mathbf{w}_\mathbb{I}$,

$$\$_\mathbb{J}(\mathbf{W}) = \sum_{j \in \mathbb{J}} \$_j(\mathbf{W}) \tag{5}$$

If set $\mathbb{I}$ is additive, then the worths of cells within $\mathbb{I}$ are not interdependent: for any $\mathbb{I} \supset \mathbb{J}$, additive or not, we can write

$$\begin{aligned}\$_\mathbb{I}(\mathbf{W}) &= L(\mathbf{W}; \varnothing_\mathbb{I}) - L(\mathbf{W}) \\ &= L(\mathbf{W}; \varnothing_\mathbb{I}) - L(\mathbf{W}; \varnothing_\mathbb{J}) + L(\mathbf{W}; \varnothing_\mathbb{J}) - L(\mathbf{W}) \\ &= L(\mathbf{W}; \varnothing_\mathbb{I}) - L(\mathbf{W}; \varnothing_\mathbb{J}) + \$_\mathbb{J}(\mathbf{W})\end{aligned}$$

If $\mathbb{I}$ is additive then we can also write

$$\$_\mathbb{I}(\mathbf{W}) = \sum_{i \in \mathbb{I}} \$_j(\mathbf{W}) = \$_\mathbb{J}(\mathbf{W}) + \$_{\mathbb{I} \setminus \mathbb{J}}(\mathbf{W})$$

Comparing the two we have that $\$_{\mathbb{I}\setminus\mathbb{J}}(\mathbf{W}) = L(\mathbf{W}; \varnothing_\mathbb{I}) - L(\mathbf{W}; \varnothing_\mathbb{J})$. The right hand side clearly does not depend on input weights to cells $\mathbb{J}$, hence the claim that worths are independent within an additive set.

It should not be a surprise that PMH, which is motivated by considering the usefulness of individual cells, requires additivity of worths. Additivity of worths is a necessary condition for 'credit assignment', or the 'usefulness of a single cell', to make sense. If the worth of one cell $i \in \mathbb{I}$ depends on the configuration of other cells in $\mathbb{I}$, then it does not make sense to assign worth to individual cells within the set $\mathbb{I}$.

To show how additivity of worths would let PMH satisfy detailed balance for p_β we first write down the transition probabilities $\mathbb{P}(\mathbf{W} \to \mathbf{W}'), \mathbb{P}(\mathbf{W}' \to \mathbf{W})$ for PMH. We introduce the notation $\Delta(\mathbf{W}', \mathbf{W})$ for the set of cells whose input weights differ between $\mathbf{W}$ and $\mathbf{W}'$:

$$\Delta(\mathbf{W}', \mathbf{W}) = \{i : \mathbf{w}_i \neq \mathbf{w}'_i\}$$

For the transition $\mathbf{W} \to \mathbf{W}'$ to occur, first some weights $\mathbf{W}''$ have to be proposed with $\mathbf{w}''_i = \mathbf{w}'_i$ for each $i \in \mathbb{J}$. We will write $\mathbb{J} = \Delta(\mathbf{W}', \mathbf{W})$ for the set of cells that propose and accept changes, $\mathbb{K} = \Delta(\mathbf{W}''; \mathbf{W}) \setminus \mathbb{J}$ for the set of cells that propose but reject changes, and $\mathbb{L}$ for the set of cells that do not propose changes.

The probability $\mathbb{P}(\mathbf{W} \to \mathbf{W}')$ is a product of three factors:

- the chance of proposing changes $\mathbf{w}'_j$ to cells $\mathbb{J}$:

$$\prod_{j\in\mathbb{J}} \kappa e^{-\alpha \$_j(\mathbf{W})} q_j(\mathbf{w}'_j|\mathbf{W})$$

- the chance of also proposing changes to cells $\mathbb{K}$ and no others:

$$\sum_{\mathbb{K}\subset\mathbb{J}^c} \prod_{k\in\mathbb{K}} \kappa e^{-\alpha \$_k(\mathbf{W})} \prod_{i\notin\mathbb{J}\cup\mathbb{K}} (1 - \kappa e^{-\alpha \$_i(\mathbf{W})})$$

- the chance of accepting changes to cells $\mathbb{J}$ and rejecting changes to cells $\mathbb{K}$, integrated over values of $\mathbf{w}''_{\mathbb{K}}$:

$$\int_{\mathbf{w}''_{\mathbb{K}}} \left(\prod_{k\in\mathbb{K}} q_k(\mathbf{w}''_k|\mathbf{W}) \left(1 - a_k(\mathbf{w}'_{\mathbb{J}}, \mathbf{w}''_{\mathbb{K}}, \mathbf{w}_{\mathbb{L}}; \mathbf{W})\right) \right) \times \left(\prod_{j\in\mathbb{J}} a_j(\mathbf{w}'_{\mathbb{J}}, \mathbf{w}''_{\mathbb{K}}, \mathbf{w}_{\mathbb{L}}; \mathbf{W}) \right)$$

Multiplying these gives:

$$\begin{aligned}
\mathbb{P}(\mathbf{W} \to \mathbf{W}') = & \prod_{j\in\mathbb{J}} \kappa e^{-\alpha \$_j(\mathbf{W})} q_j(\mathbf{w}'_j|\mathbf{W}) \\
& \times \sum_{\mathbb{K}\subset\mathbb{J}^c} \prod_{k\in\mathbb{K}} \kappa e^{-\alpha \$_k(\mathbf{W})} \prod_{i\notin\mathbb{J}\cup\mathbb{K}} (1 - \kappa e^{-\alpha \$_i(\mathbf{W})}) \\
& \times \int_{\mathbf{w}''_{\mathbb{K}}} \left(\prod_{k\in\mathbb{K}} q_k(\mathbf{w}''_k|\mathbf{W}) \left(1 - a_k(\mathbf{w}'_{\mathbb{J}}, \mathbf{w}''_{\mathbb{K}}, \mathbf{w}_{\mathbb{L}}; \mathbf{W})\right) \right) \\
& \times \left(\prod_{j\in\mathbb{J}} a_j(\mathbf{w}'_{\mathbb{J}}, \mathbf{w}''_{\mathbb{K}}, \mathbf{w}_{\mathbb{L}}; \mathbf{W}) \right)
\end{aligned}$$

The reverse transition probability is similarly

$$\begin{aligned}
\mathbb{P}(\mathbf{W}' \to \mathbf{W}) = & \prod_{j\in\mathbb{J}} \kappa e^{-\alpha \$_j(\mathbf{W}')} q_j(\mathbf{w}_j|\mathbf{W}') \\
& \times \sum_{\mathbb{K}\subset\mathbb{J}^c} \prod_{k\in\mathbb{K}} \kappa e^{-\alpha \$_k(\mathbf{W}')} \prod_{i\notin\mathbb{J}\cup\mathbb{K}} (1 - \kappa e^{-\alpha \$_i(\mathbf{W}')}) \\
& \times \int_{\mathbf{w}''_{\mathbb{K}}} \left(\prod_{k\in\mathbb{K}} q_k(\mathbf{w}''_k|\mathbf{W}') \left(1 - a_k(\mathbf{w}_{\mathbb{J}}, \mathbf{w}''_{\mathbb{K}}, \mathbf{w}_{\mathbb{L}}; \mathbf{W}')\right) \right) \\
& \times \left(\prod_{j\in\mathbb{J}} a_j(\mathbf{w}_{\mathbb{J}}, \mathbf{w}''_{\mathbb{K}}, \mathbf{w}_{\mathbb{L}}; \mathbf{W}') \right)
\end{aligned}$$

For the transition ratio $\frac{\mathbb{P}(\mathbf{W}\to\mathbf{W}')}{\mathbb{P}(\mathbf{W}'\to\mathbf{W})}$ to simplify neatly for PMH as it does for Metropolis-Hastings, we would require the following, rather strong conditions:

1. Whenever changes are proposed to cells $\mathbb{J}\cup\mathbb{K}$, the set $\mathbb{J}\cup\mathbb{K}$ should be additive. Then for $k\in\mathbb{K}$, the changes to $\mathbb{J}$ do not affect a_k:

   $$a_k(\mathbf{w}_\mathbb{J},\mathbf{w}''_\mathbb{K},\mathbf{w}_\mathbb{L};\mathbf{W}')\equiv a_k(\mathbf{w}'_\mathbb{J},\mathbf{w}''_\mathbb{K},\mathbf{w}_\mathbb{L};\mathbf{W})$$

   and for $j\in\mathbb{J}$, the changes to $\mathbb{K}$ do not affect a_j:

   $$a_j(\mathbf{w}_\mathbb{J},\mathbf{w}''_\mathbb{K},\mathbf{w}_\mathbb{L};\mathbf{W}')\equiv a_j(\mathbf{w}_\mathbb{J},\mathbf{w}_\mathbb{K},\mathbf{w}_\mathbb{L};\mathbf{W}')=a_j(\mathbf{W};\mathbf{W}')$$

   $$a_j(\mathbf{w}'_\mathbb{J},\mathbf{w}''_\mathbb{K},\mathbf{w}_\mathbb{L};\mathbf{W})\equiv a_j(\mathbf{w}'_\mathbb{J},\mathbf{w}_\mathbb{K},\mathbf{w}_\mathbb{L};\mathbf{W})=a_j(\mathbf{W}';\mathbf{W})$$

   Furthermore, additivity of $\mathbb{J}$ implies that the product $\prod_{j\in\mathbb{J}}a_j(\mathbf{W};\mathbf{W}')$ satisfies

   $$\frac{\prod_{j\in\mathbb{J}}a_j(\mathbf{W};\mathbf{W}')}{\prod_{j\in\mathbb{J}}a_j(\mathbf{W}';\mathbf{W})}=e^{(\beta-\alpha)(L(\mathbf{W})-L(\mathbf{W}))}$$

2. For $k\notin\mathbb{J}$ the chance of proposing the specific changes $\mathbf{w}''_k$ should be the same from $\mathbf{W}$ as from $\mathbf{W}'$:

   $$q_k(\mathbf{w}''_k|\mathbf{W}')\equiv q_k(\mathbf{w}''_k|\mathbf{W})$$

3. For $j\in\mathbb{J}$ the q_j satisfy detailed balance for p_0, in the sense that

   $$\prod_{j\in\mathbb{J}}\frac{q_j(\mathbf{w}'_j;\mathbf{W})}{q_j(\mathbf{w}_j;\mathbf{W}')}=\frac{p_0(\mathbf{W}')}{p_0(\mathbf{W})}$$

If all these conditions are met then $\frac{\mathbb{P}(\mathbf{W}\to\mathbf{W}')}{\mathbb{P}(\mathbf{W}'\to\mathbf{W})}$ becomes exactly $\frac{p_\beta(\mathbf{W}')}{p_\beta(\mathbf{W})}$.

None of the conditions is likely to be met in a real neural network, or even most artificial neural networks, but we show in chapter 5 that they can be approximately met and in section 5.8 we show this is enough to make a version of PMH work in a simulation.

Why won't these conditions be met exactly?

1. PMH lets each cell decide independently whether to propose a change, with nonzero probability. Therefore it is possible, with miniscule probability, that all cells could propose changes at once, and the additivity condition is equivalent to a 'global additivity' condition, that every set of cells, including the set of

all cells in the network, is additive. This global additivity tends to fail for any network where cells build on each others' outputs, or where multiple cells represent the same feature. However, in practice we can choose values of κ, α so that the proposing set is small. In chapter 5 we show that in some simple cases, every sufficiently small set of cells is approximately additive, and this is enough to make PMH work.

2. The only obvious way to satisfy condition 2 exactly is when $q(\mathbf{w}_i'; \mathbf{W}) \equiv q(\mathbf{w}_i'; \mathbf{w}_i)$, that is, the synaptic weights proposed by cell i do not depend on anything other than the current configuration of the input weights to cell i. This condition is likely to be violated whenever cell i operates an activity-dependent learning rule to generate proposals $\mathbf{w}_i'$, and is downstream of other plastic cells. In the feedforward networks of chapter 5 only a single layer of cells run PMH, and this condition holds exactly. In more complex networks, it is harder to guarantee.

3. This is true more or less by definition — we motivated PMH as a way to enhance a q that already generates samples from p_0. However, it assumes that the transition probabilities given by q_i generate a *reversible* Markov chain on $\mathbf{W}$.

The fact that the conditions for precise convergence of PMH to p_β are so rarely met, and the difficulty of analysing what happens when the conditions are only *approximately* met (section 4.10), means that the only way to find out if it works is to implement and simulate it. In section 5.8 we find that PMH does indeed work in a simple feedforward network, and that it greatly reduces the number of cells needed to reach a given performance level, compared to using an unsupervised rule alone. However, in chapter 6 we find a fundamental difficulty in implementing PMH in recurrent networks, that is not touched on in this chapter: cells cannot accurately measure their worths in a biologically plausible way.

4.10 Does approximate PMH approximately converge?

In practice, we only *approximately* meet the conditions required to guarantee that our implementation of PMH converges to p_β. Simulations in section 5.8 show that nonetheless, PMH can work well. For the interested reader, we record some abortive attempts to prove this is not a fluke. To the uninterested reader, apologies.

First, what should we try to prove? Write $\mathbf{P}$ for the transition probabilities on $\mathbf{W}$ that perfect PMH (satisfying all three conditions

on page 39) would give. In practice the transition probabilities are $\tilde{\mathbf{P}} \approx \mathbf{P}$, with discrepancies arising because, for example, we do not measure worth accurately. We write $\pi = p_\beta$ for the stationary distribution of $\mathbf{P}$ and $\tilde{\pi}$ for the stationary distribution of $\tilde{\mathbf{P}}$.

We wish to prove that in some sense "$\tilde{\mathbf{P}} \approx \mathbf{P} \Rightarrow \tilde{\pi} \approx p_\beta$". At least one natural interpretation of this informal statement is false: given any $\epsilon > 0$ and an arbitrary pair of probability distributions $\pi, \tilde{\pi}$ we can find $\mathbf{P}, \tilde{\mathbf{P}}$ with $\mathbf{P}\pi = \pi, \tilde{\mathbf{P}}\tilde{\pi} = \tilde{\pi}$, and $\max_{i,j} |P_{ij} - \tilde{P}_{ij}| < \epsilon$[1]. Some bounds are collected in (Haviv and Heyden, 1984), but it is hard to see how to apply them to our problem.

Much of the literature deals with *finite* Markov chains, so we might try to proceed by discretising our state-space. Specifically we index values of $\mathbf{W}$ by i, choosing a partition such that p_0 is uniform, and affinely transform L so that $\pi_i = e^{-L_i}$ corresponds to p_β.

A useful result would be that $\mathbb{E}_{\tilde{\pi}}(L) \approx \mathbb{E}_{\pi}(L)$. To that end we tried studying $|\mathbb{E}_{\mathbf{P}\tilde{\pi}}(L) - \mathbb{E}_{\tilde{\pi}}(L)|$, hoping to find an upper bound in terms of the similarity between $\tilde{\mathbf{P}}$ and $\mathbf{P}$, and a lower bound in terms of $|\mathbb{E}_{\tilde{\pi}}(L) - \mathbb{E}_{\pi}(L)|$. Detailed balance of $\mathbf{P}$ implies for any $\mathbf{p}$ that

$$\mathbb{E}_{\mathbf{Pp}}(L) - \mathbb{E}_{\mathbf{p}}(L) = \frac{1}{2}\sum_{i,j}(L_i - L_j)P_{ij}\pi_j\left(\frac{p_j}{\pi_j} - \frac{p_i}{\pi_i}\right)$$

Here we stumble: the terms could have mixed signs, so it is hard to find useful lower bounds for $|\mathbb{E}_{\mathbf{Pp}}(L) - \mathbb{E}_{\mathbf{p}}(L)|$. (Penrose, 1970) suggests another option, replacing $\mathbb{E}_{\mathbf{Pp}}(L) - \mathbb{E}_{\mathbf{p}}(L)$ with some other $\Phi(\mathbf{Pp}) - \Phi(\mathbf{p})$ and using the following theorem.

Theorem 1 (Free energy goes down)**.** *Let* $\mathbf{P}$ *be any reversible transition matrix,* π *its stationary distribution and* ϕ *any convex function (i.e.,* $\phi'' \geq 0$ *everywhere). Let* $\mathbf{p}$ *be any probability distribution. Define*

$$\Phi(\mathbf{p}) := \sum_i \pi_i \phi\left(\frac{p_i}{\pi_i}\right)$$

Then Φ *decreases under* $\mathbf{P}$*, i.e.* $\Phi(\mathbf{p}) \geq \Phi(\mathbf{Pp})$*. Moreover, we have an exact expression for* $\Phi(\mathbf{Pp}) - \Phi(\mathbf{p})$*. Define*

$$\overline{p}_i = (\mathbf{Pp})_i, \quad x_i = \frac{p_i}{\pi_i}, \quad \overline{x}_j = \frac{\overline{p}_j}{\pi_j}$$

[1]Take any $\mathbf{P}, \tilde{\mathbf{P}}$ with stationary distributions $\pi, \tilde{\pi}$ and keep taking square roots of $\mathbf{P}, \tilde{\mathbf{P}}$. Both converge towards $\mathbf{I}$ without their stationary distributions changing.

Then

$$\Phi(\mathbf{p}) - \Phi(\mathbf{Pp}) = \sum_{i,j} \pi_B P_{ij} \left(\phi(x_i) - \phi(\overline{x}_j) - (x_i - \overline{x}_j)\phi'(\overline{x}_j)\right) \quad (6)$$

$$= \sum_{i,j} \pi_j P_{ij} \left(\frac{1}{2}(x_i - \overline{x}_j)^2 \phi''(x_{ij})\right) \quad (7)$$

where x_{ij} lies in the interval between x_i and $\overline{x}_j$.

Proof. See (Penrose, 1970). □

In particular if $\phi(x) = -x \log x$ then $\phi''(x) = \frac{1}{x}$ and

$$\Phi(\mathbf{p}) - \Phi(\mathbf{Pp}) = \sum_{i,j} \frac{\pi_j P_{ij} (x_i - \overline{x}_j)^2}{2x_{ij}} \geq \sum_{i,j} \frac{\pi_j P_{ij} (x_i - \overline{x}_j)^2}{2 \max(x_i, \overline{x}_j)} \quad (8)$$

for some x_{ij} between x_i and $\overline{x}_j$. This looks more promising, but so far did not bear fruit.

4.11 Conclusion

In this chapter we began to convert the retroaxonal hypothesis to a mathematical model. We started with two statements from section 3.2: that cells which are less useful should make their input synapses more plastic; and second, that cells can infer their usefulness from the strength of their output synapses. This chapter deals with the first of those claims; the next two chapters deal with the other.

Viewed this way, the retroaxonal hypothesis is a way of making good use of a limited number of neurons. In section 4.2 we reviewed some other strategies for pruning networks or getting good performance from a network of limited size, and explained how retroaxonal learning differs from these.

We introduced the idea of 'worth' in section 4.3, a mathematical definition of what it means for a cell to be useful.

In section 4.5 we gave a very basic argument that each individual cell should aim to maximise its worth given the configuration of the other cells. This is because when one cell changes its input synapses, the change in network performance matches the change in that cell's worth. This motivates us to model retroaxonal learning as a process where cells use their worth to decide whether to make changes to their input synapses.

In section 4.6 we brought in the notion that some changes to synaptic weights are transient, and revert, while others become permanent. After proposing a change to its input synaptic weights,

we suggested that a cell could use the effect on its worth to decide whether to revert or keep the change.

We formalised this as a variation on the standard Metropolis-Hastings rule. In standard Metropolis-Hastings, described in section 4.7, changes to a variable are proposed, sometimes kept and sometimes rejected. In section 4.8 we introduced the 'parallel Metropolis-Hastings' algorithm which is our variant of Metropolis-Hastings and model of retroaxonal learning. In this scheme, individual cells are more likely to propose changes to their input synapses if they have low worth, and more likely to retain the changes if the change increases their worth. PMH is a means of taking an unsupervised learning rule, and modulating the learning rate cell-by-cell to improve the performance of the network as a whole.

In section 4.9 we explained when the algorithm of section 4.8 is analytically guaranteed to converge to its intended target distribution p_β. The conditions are strict and in practice, are not met, in simulations or in real neural networks. We will show in chapter 5 that they can be approximately met in a simple feedforward network and that this is enough to make the scheme work in a simulation (section 5.8).

As well as the convergence conditions in 4.9, the PMH algorithm requires cells to be able to estimate their worths. For this to be biologically plausible, it must be achievable using only slow retroaxonal signals or locally available information. In chapter 5 we show this is possible in a shallow feedforward network, and demonstrate using simulations that this enables PMH-based learning. In chapter 6 we consider whether cells can estimate their worths accurately in recurrent networks, for the specific case of random connectivity.

5 Worth from output weights in a feedforward network

5.1 Introduction

In Chapter 4 we considered a generic network whose performance is measured by some loss function L and we defined a learning scheme, PMH, which might work if cells can measure their individual contributions to reducing L, that is, their 'worths'. We now describe a type of network and loss function for which this can work, and in section 5.8 present a simulation using PMH. In our simulation, cells estimate their worths from their output weights, so each cell can estimate its worth without actually being silenced and without a need to transmit fast retrograde signals. The simulation is an implementation of the kind of learning scheme proposed by the retroaxonal hypothesis.

In this chapter and chapter 6, we will consider pools of 'producer' cells receiving inputs from some external source and offering their outputs as input to a layer of 'consumer' cells. Cartoons of these networks are in figure 1 (page 59). The loss L is a function of the activity only of the *consumer* cells and their input weights $\mathbf{V}$, and we assume these input weights are learned by a *supervised* learning scheme. We assume the supervised learning scheme is effective, so that the consumers choose optimal weights given the output the producers provide. Thus, the performance is really just a function of the configuration of the producers: $L = L(\mathbf{W})$, as in chapter 4 (PMH allows constraints on $\mathbf{W}$ to be introduced separately via p_0). The producers do not receive any direct training input. Section 3.5 gives examples of neural populations in the brain to which these producers and consumers might correspond.

Networks with a large pool of unsupervised (sometimes not even plastic) cells and a supervised output layer, abound in the machine learning literature. They include the liquid state machine (Maass et al., 2002), the FORCE network (Sussillo and Abbott, 2009), and networks that combine a pool of untrained or unsupervised feature-extracting neurons with a support vector machine to classify stimuli (Lazebnik et al., 2006; Henaff et al., 2011).

The innovation here is that we show how, at least in a simple case, the producers can learn using PMH: each producer will modulate its unsupervised learning rate according to its worth, and we will show that each producer can estimate its worth from its output weights to consumers.

5.2 Model network and loss function

To begin with we consider a single layer of producers offering their output to some consumers, as in the lower panel of figure 1.

Producer i has output $x_i(t)$ and consumer j has output $\hat{y}_j(t) = \sum_i V_{ji} x_i(t)$. The loss function for the network is the simple penalised least squares one,

$$L_{\mathbf{V}} := \frac{1}{2} \sum_j \overline{\left(y_j(t) - \sum_i V_{ji} x_i(t) \right)^2} + \frac{\lambda}{2} \sum_i V_{ij}^2$$

Here, $\overline{(y_j(t) - \sum_i V_{ji} x_i(t))^2}$ means a time-average of $(y_j(t) - \sum_i V_{ji} x_i(t))^2$.

Each consumer receives a training signal, for example a copy of its ideal output $y_j(t)$, which enables it to find the optimal V_{ji} given the available $x_i(t)$. The producers receive no such direct training signals, though they may adapt their input synapses using unsupervised learning rules.

We assume p producers and N samples/timesteps. We write $\mathbf{X} \in \mathbb{R}^{N \times p}$ for the 'data matrix' with $X_{ti} = x_i(t)$. Thus $\mathbf{x}_i \in \mathbb{R}^N$ is the i^{th} column of $\mathbf{X}$ and represents the output of cell i. We write $\mathbf{v}_j \in \mathbb{R}^p$ for the vector of input weights to consumer j, $\hat{\mathbf{y}}_j = \mathbf{X}\mathbf{v}_j \in \mathbb{R}^N$ for the output of consumer j, and $\mathbf{y}_j$ for the target output for consumer j. We assume that the data set is large enough to make $L_{\mathbf{V}}$ equivalent to the finite-sample loss function

$$L_{\mathbf{V}} = \sum_j \frac{1}{2N} |\mathbf{y}_j - \hat{\mathbf{y}}_j|^2 + \frac{\lambda}{2} \sum_j |\mathbf{v}_j|^2 \tag{9}$$

We define the loss *for the pool of producers* as

$$L(\mathbf{X}) = \min_{\mathbf{V}} L_{\mathbf{V}}(\mathbf{X}) \tag{10}$$

This loss function L measures the performance that we will get for a given pool of producers, *subject to the assumption that each consumer does the best it can with the products available.*

The worth of a set of producers in this feedforward network is defined as

$$\$_{\mathbb{I}} = L(\tilde{\mathbf{X}}) - L(\mathbf{X})$$

where $\tilde{\mathbf{X}}$ is the matrix $\mathbf{X}$ with the columns $\mathbf{x}_i$ for $i \in \mathbb{I}$ replaced by zeroes. Thus, the worth of producer cell i is the change in loss that will arise if cell i falls silent, and then each consumer adjusts its input weights to make best use of the remaining producers' outputs.

In this chapter we will show that the worth of a cell in the feedforward network of Figure 1 is estimated by $\$_j \approx \frac{\lambda}{2} \sum_i V_{ij}^2$. In Chapter 6 we will study more complicated networks where there are connections between producers, and only a subset $\mathbb{I}$ of producers are 'visible' to the consumers, meaning that the consumers must choose $\mathbf{V}$ subject to the constraint that $\forall i \notin \mathbb{I}, \forall j : V_{ji} = 0$. When there are connections between producers, the worth estimate $\$_j \approx \frac{\lambda}{2} \sum_i V_{ij}^2$ does not apply, but the analysis of this chapter is still relevant: it tells us the effect on performance of *hiding* a cell from the consumers. The quadratic weight penalty in L will always induce a consumer to set $V_{ij} = 0$ if $x_j \equiv 0$, whatever the connectivity between producers. Therefore, hiding a cell j has the same effect on performance as replacing its output $\mathbf{x}_j$ with all zeros in the data matrix $(\mathbf{x}_1 | \ldots | \mathbf{x}_p)$ seen by the consumer (leaving all other $\mathbf{x}_i$ unchanged). This will be useful in chapter 6.

5.3 Choice of weight penalty

The retroaxonal hypothesis is motivated by the idea that having a synapse, especially a strong synapse, is costly, and that if a synapse is present, it must be because there is something to be gained by it. Almost any neural network model includes some cost function on synaptic weights, not only because of the biological fact that space is often a tight constraint, but also because in machine learning, a cost function on synaptic weights prevents overfitting to training data.

The quadratic cost function we have chosen does not promote sparsity of connections, but our models do provide other ways of restricting the number of non-zero input weights both to producers and from producers to consumers. First, the probability density p_0 in PMH (chapter 4) can force sparsity of input weights to producers. Second, through the idea of 'hidden' and 'visible' producer cells we can enforce sparsity of connections from producers to consumers: we restrict the set of producers which are allowed to project to each consumer, and use the quadratic cost function for the remaining possible synapses.

In section 3.5 we suggested that producers and consumers could correspond to cortical cells (producers) that project to the striatum (consumers). For these projections, the space cost of adding another projecting axon from cortex to striatum is large compared to the space cost of adding more synapses to an existing projection. To model this, it makes sense to restrict the set of producers that project to consumers, rather than using a synapse-by-synapse cost

function to promote sparsity.

How do we choose the number of producers supplying input to each consumer, and how do we choose the weight penalty parameter λ in our quadratic cost function? It is estimated (Kincaid et al., 1998) that within the dendritic tree of a single striatal spiny cell, one cortical cell axon makes a maximum of 40 synapses, and that in the same volume there are approximately 2840 spiny cell bodies and 15 million corticostriatal synapses, which means that on average each striatal cell receives inputs from hundreds if not thousands of different cortical cells. In our simulations we let each consumer select input from 1000 producers.

Individual corticostriatal synapses are weak, and each striatal MSN requires a large number of excitatory inputs to fire (Carter et al., 2007). In keeping with this we choose λ so that each individual synaptic weight is $\ll 1$. In our model, the weight from producer cell i to consumer j is $\frac{1}{\lambda}\frac{\mathbf{r}_j^T \mathbf{x}_i}{N}$, where $r_j(t) = y_j - \sum_i V_{ji} x_i(t)$ is the residual error for consumer j (so $|\mathbf{r}| \leq |\mathbf{y}|$). To make our cost function dimensionally correct, λ should have units of firing rate squared and to guarantee every weight is $\ll 1$ we need $\lambda \gg |\mathbf{x}||\mathbf{y}|/N$. We will choose units so that typical firing rates are $O(1)$, so we choose $\lambda \gg 1$.

If we used a different cost function on synaptic weights, would any of the theory here still work? The specific form of the approximation for worth in terms of output weights will depend on the choice of cost function. However, the motivating intuition is very generally applicable, save a few pathological cases. If a producer has no output weights, it has no effect on performance, and no worth. Conversely if a producer has large output weights, it will usually have positive worth: if a consumer sets a nonzero input weight from a given producer, the implication is that a nonzero input weight gives better performance than zero input weight, and therefore better than the consumer could do if that producer was silenced. This argument fails in degenerate cases when multiple input weight vectors to the consumer give equivalent performance. An example is when multiple producers have identical firing patterns and the cost function for synaptic weights is a sparsity-promoting Lp norm with $p < 1$. In that case, a consumer will arbitrarily choose one of the identical producers from which to have nonzero input weight. As long as there are at least two producers with identical output, each individual producer has zero worth: if it is silenced the consumer can just select input from one of its clones, with no effect on performance. If we imagine taking the pool of identical producers and silencing the

unused ones one at a time, then at the instant we silence the last unused producer, the worth of the single producer whose output *is* being used jumps from zero to something positive, while its output weight to the consumer is unchanged. In this scenario, the output weight would not be a reliable indicator of worth. In most standard models, output weight will tell us *something* about the worth of a cell, but the quadratic cost function we work with here gives rise to an unusually tidy relationship between output weight and worth.

5.4 Ridge regression formulae

We gather here some standard formulae in the theory of ridge regression. For more details see for example (Shawe-Taylor and Cristianini, 2004; Hastie et al., 2009; Anderson, 1958). We will later use these formulae to derive equations and approximations for the worths of cells.

For simplicity of notation we consider a single consumer and simply write $\mathbf{v}$, rather than $\mathbf{v}_j$, for its vector of input weights.

The optimal $\mathbf{v}$ must satisfy

$$\begin{aligned} \nabla_{\mathbf{v}} L_{\mathbf{v}} &= (\mathbf{y} - \hat{\mathbf{y}})^T \mathbf{X} + N\lambda \mathbf{v}^T = 0 \\ \Rightarrow \mathbf{v}^T &= \frac{\mathbf{X}^T(\mathbf{y} - \hat{\mathbf{y}})}{N\lambda} \end{aligned} \tag{11}$$

By substituting the definition $\hat{\mathbf{y}} = \mathbf{X}\mathbf{v}$ into (11), we obtain a formula for optimal weights:

$$\mathbf{v} = (\mathbf{X}^T\mathbf{X} + N\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} \tag{12}$$

In simulations, these weights may be learned approximately using a delta rule, i.e. stochastic gradient descent.

Substituting (11) into (9) shows that $L = \min_{\mathbf{v}} L_{\mathbf{v}}$ is proportional to the mean product of the training signal $\mathbf{y}$ with the residual $\mathbf{y} - \hat{\mathbf{y}}$:

$$\begin{aligned} L &= \frac{1}{2N}|\mathbf{y} - \hat{\mathbf{y}}|^2 + \frac{\lambda}{2}\mathbf{v}^T\frac{\mathbf{X}^T(\mathbf{y} - \hat{\mathbf{y}})}{N\lambda} \\ &= \frac{1}{2N}(\mathbf{y} - \hat{\mathbf{y}} + \mathbf{X}\mathbf{v})^T(\mathbf{y} - \hat{\mathbf{y}}) \\ &= \frac{1}{2N}\mathbf{y}^T(\mathbf{y} - \hat{\mathbf{y}}) \end{aligned} \tag{13}$$

We will make use of the kernel matrix $\mathbf{K}$, the hat matrix $\mathbf{H}$, and the residual matrix $\mathbf{R}$, which all are real symmetric $N \times N$ matrices,

defined as follows:

$$\mathbf{K} := \mathbf{X}\mathbf{X}^T = \sum_i \mathbf{x}_i \mathbf{x}_i^T$$
$$\mathbf{H} := \mathbf{X}(\mathbf{X}^T\mathbf{X} + N\lambda\mathbf{I})^{-1}\mathbf{X}^T$$
$$\mathbf{R} := \mathbf{I} - \mathbf{H} = \left(\frac{\mathbf{K}}{N\lambda} + \mathbf{I}\right)^{-1}$$

$\mathbf{H}$ is called the 'hat matrix' because it maps the target signal $\mathbf{y}$ to the consumer neuron's actual output 'y hat': when the weights $\mathbf{v}$ take their optimal value, $\hat{\mathbf{y}} = \mathbf{H}\mathbf{y}$. Similarly, the matrix $\mathbf{R}$ maps $\mathbf{y}$ to the residual: $\mathbf{y} - \hat{\mathbf{y}} = \mathbf{R}\mathbf{y}$.

Note that $\mathbf{H}$ and $\mathbf{R}$ depend only on $\mathbf{X}$ and not on $\mathbf{y}$; they therefore summarize how well a given pattern of producer cell activity $\mathbf{X}$ prepares the consumer cell to learn any target pattern $\mathbf{y}$. $\mathbf{R}$ can be considered as a quadratic form that predicts the loss function for any given target: from (13) we have that

$$L = \frac{1}{2N}\mathbf{y}^T\mathbf{R}\mathbf{y} = \frac{1}{2N}\mathbf{y}^T\left(\frac{\mathbf{X}\mathbf{X}^T}{N\lambda} + \mathbf{I}\right)^{-1}\mathbf{y} \tag{14}$$

5.5 Estimating worth from output weights

We use the equations of 5.4 to show that in the feedforward network of figure 1, each producer cell can estimate its worth as

$$\$_i \approx \frac{\lambda}{2}\sum_j V_{ji}^2 \tag{15}$$

This approximation holds provided $[\![\mathbf{x}_i]\!]^2 \ll \lambda$, where $[\![..]\!]$ denotes the root mean square over time. The quantity $\sum_j V_{ji}^2$ is one which a cell i could reasonably compute using slow retrograde signals.

Moreover, we will show that for small sets $\mathbb{I}$ of producers, their worth is approximately additive:

$$\$_{\mathbb{I}} \approx \sum_{i\in\mathbb{I}} \$_i \approx \frac{\lambda}{2}\sum_{i\in\mathbb{I},j} V_{ji}^2 \tag{16}$$

Here a 'small set' is one with $\max_{k\in\mathbb{I}} |\mathbf{x}_k| \sum_{j\in\mathbb{I}} |\mathbf{x}_j| \ll N\lambda$, satisfied for example if each $[\![\mathbf{x}_i]\!]^2 = 1$ and $|\mathbb{I}| \ll \lambda$.

The following theorem makes the approximation (15) precise, and theorem 3 will make the approximation 16 precise.

These results are useful because they let us know which parameter regimes PMH can work in, with worths estimated from

output weights. As long as we choose parameters so that $\max_{k\in\mathbb{I}} |\mathbf{x}_k| \sum_{j\in\mathbb{I}} |\mathbf{x}_j| \ll N\lambda$ for the proposing set, then the worth estimate (15) is valid. If each cell has root mean square firing rate of order 1, this means we require $\lambda \gg$ the number of cells proposing changes at any one time.

Theorem 2 (Output weight predicts worth)**.** *The worth $\$_i$ of neuron i to a single consumer is given by*

$$\$_i = \frac{\lambda}{2} v_i^2 \left(\frac{\lambda}{\lambda - \frac{\mathbf{x}_i^T \mathbf{R} \mathbf{x}_i}{N}} \right)$$

Corollary *With multiple consumers, the worth $\$_i$ of neuron i is given by*

$$\$_i = \frac{\lambda}{2} \left(\frac{\lambda}{\lambda - \frac{\mathbf{x}_i^T \mathbf{R} \mathbf{x}_i}{N}} \right) \sum_j V_{ji}^2$$

The residual matrix $\mathbf{R}$ is a contraction, so $|\mathbf{x}_i^T \mathbf{R} \mathbf{x}_i| \leq |\mathbf{x}_i|^2$ and the approximation (15) follows when $|\mathbf{x}_i|^2 \ll N\lambda$, i.e., when $[\![x]\!]^2 \ll \lambda$.

Proof. Recall from (14) that, with a single consumer,

$$L = \frac{1}{2N} \mathbf{y}^T \mathbf{R} \mathbf{y}$$

and therefore, when $\mathbf{X}$ changes in any way (and then $\mathbf{v}$ is updated), the change in loss is

$$\Delta L = \frac{1}{2N} \mathbf{y}^T (\Delta \mathbf{R}) \mathbf{y}$$

We write $\mathbf{R}$ for the residual matrix when cell i is present and $\mathbf{R}' = \mathbf{R} + \Delta\mathbf{R}$ for the residual matrix after that cell is removed. Recall that $\mathbf{R} = \left(\frac{\mathbf{K}}{N\lambda} + \mathbf{I} \right)^{-1}$, where $\mathbf{K} = \sum_j \mathbf{x}_j \mathbf{x}_j^T$, is the kernel matrix. We can therefore use the Woodbury matrix identity to compute $\Delta\mathbf{R}$. The identity is:

$$(\mathbf{A} + \mathbf{UCV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1}$$

We use this with $\mathbf{A} = \mathbf{R}^{-1}, \mathbf{U} = \mathbf{x} = \mathbf{V}^T, \mathbf{C} = -\frac{1}{N\lambda}$ to obtain

$$\begin{aligned} \mathbf{R}' &= \left(\mathbf{R}^{-1} - \frac{\mathbf{x}_i \mathbf{x}_i^T}{N\lambda} \right)^{-1} \\ &= \mathbf{R} + \frac{\mathbf{R} \mathbf{x}_i \mathbf{x}_i^T \mathbf{R}}{N\lambda - \mathbf{x}_i^T \mathbf{R} \mathbf{x}_i} \end{aligned}$$

so that

$$\Delta \mathbf{R} = \frac{\mathbf{R}\mathbf{x}_i\mathbf{x}_i^T\mathbf{R}}{N\lambda - \mathbf{x}_i^T\mathbf{R}\mathbf{x}_i}$$

As $L = \frac{1}{2N}\mathbf{y}^T\mathbf{R}\mathbf{y}$, the change in L from removing cell i is

$$\Delta L = \frac{1}{2N}\mathbf{y}^T\left(\Delta \mathbf{R}\right)\mathbf{y} = \mathbf{y}^T\frac{\mathbf{R}\mathbf{x}_i\mathbf{x}_i^T\mathbf{R}}{2N^2(\lambda - \frac{\mathbf{x}_i^T\mathbf{R}\mathbf{x}_i}{N})}\mathbf{y} = \frac{|\mathbf{y}^T\mathbf{R}\mathbf{x}_i|^2}{2N^2(\lambda - \frac{\mathbf{x}_i^T\mathbf{R}\mathbf{x}_i}{N})}$$

Recall also that the output weight of cell i before it is silenced is $v_i = \frac{\mathbf{y}^T\mathbf{R}\mathbf{x}_i}{N\lambda}$. Thus,

$$\$_i = \Delta L = \frac{\lambda^2 v_i^2}{2\left(\lambda - \frac{\mathbf{x}_i^T\mathbf{R}\mathbf{x}_i}{N}\right)} = \frac{\lambda}{2}v_i^2\left(\frac{\lambda}{\lambda - \frac{\mathbf{x}_i^T\mathbf{R}\mathbf{x}_i}{N}}\right)$$

□

Next we show that for small sets of cells, worth is approximately additive.

Theorem 3 (Worth is approximately additive)**.** *Let $\mathbb{I}$ be a set of producer cells. Then the worth $\$_\mathbb{I}$ of set $\mathbb{I}$ satisfies*

$$\frac{\lambda}{2}\sum_{i\in\mathbb{I}} v_i^2 \leq \$_\mathbb{I} \leq \left(1 + \frac{\max_{k\in\mathbb{I}}|\mathbf{x}_k|\sum_{j\in\mathbb{I}}|\mathbf{x}_j|}{N\lambda}\right)\left(\frac{\lambda}{2}\sum_{i\in\mathbb{I}} v_i^2\right)$$

In particular, this means that if $\max_{k\in\mathbb{I}}|\mathbf{x}_k|\sum_{j\in\mathbb{I}}|\mathbf{x}_j| \ll N\lambda$ then

$$\$_\mathbb{I} \approx \frac{1}{2}\sum_{i\in\mathbb{I}} v_i^2 \approx \sum_{i\in\mathbb{I}} \$_i$$

Thus, worth in the feedforward network of Figure 1 is approximately additive for sufficiently small sets of cells.

Proof. Again, we will use

$$\Delta L = \frac{1}{2N}\mathbf{y}^T(\Delta \mathbf{R})\mathbf{y}$$

Split the producer cells into two disjoint sets $\mathbb{I}, \mathbb{J} = \mathbb{I}^c$. Write $\mathbf{X}_\mathbb{I} \in \mathbb{R}^{N\times|\mathbb{I}|}$ for the matrix of data from cells $\mathbb{I}$, $\mathbf{X}_\mathbb{J}$ for the matrix of data from cells $\mathbb{J}$. Correspondingly define

$$\mathbf{R}_\mathbb{J} = \left(\frac{\mathbf{X}_\mathbb{J}\mathbf{X}_\mathbb{J}^T}{N\lambda} + \mathbf{I}\right)^{-1}$$

$$\mathbf{R}_{\mathbb{I}\cup\mathbb{J}} = \left(\frac{\mathbf{X}_{\mathbb{I}\cup\mathbb{J}}\mathbf{X}_{\mathbb{I}\cup\mathbb{J}}^T}{N\lambda} + \mathbf{I}\right)^{-1} = \left(\frac{\mathbf{X}_{\mathbb{I}}\mathbf{X}_{\mathbb{I}}^T}{N\lambda} + \frac{\mathbf{X}_{\mathbb{J}}\mathbf{X}_{\mathbb{J}}^T}{N\lambda} + \mathbf{I}\right)^{-1}$$

so $\mathbf{R}_{\mathbb{J}}$ is the residual matrix when only cells $\mathbb{J}$ are available. Using the Woodbury matrix identity again with $A = \mathbf{R}_{\mathbb{J}}^{-1}$ and $U = V^T = \mathbf{X}_{\mathbb{I}}, C = \frac{1}{N\lambda}$, we have

$$\mathbf{R}_{\mathbb{J}} - \mathbf{R}_{\mathbb{I}\cup\mathbb{J}} = \mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(N\lambda + \mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\right)^{-1}\mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}} \tag{17}$$

It will prove convenient to define

$$\mathbf{B} := \mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}} = \mathbf{B}^T$$

and rewrite (17) as

$$\mathbf{R}_{\mathbb{J}} - \mathbf{R}_{\mathbb{I}\cup\mathbb{J}} = \mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(N\lambda + \mathbf{B}\right)^{-1}\mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}} \tag{18}$$

Therefore, the change in loss is

$$\begin{aligned}\$_{\mathbb{I}} = L_{\mathbb{J}} - L_{\mathbb{I}\cup\mathbb{J}} &= \frac{1}{2N}\mathbf{y}^T(\mathbf{R}_{\mathbb{J}} - \mathbf{R}_{\mathbb{I}\cup\mathbb{J}})\mathbf{y} \\ &= \frac{1}{2N}\mathbf{y}^T\mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(N\lambda\mathbf{I} + \mathbf{B}\right)^{-1}\mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}}\mathbf{y}\end{aligned}$$

By definition the worth is $\$_{\mathbb{I}} = L_{\mathbb{J}} - L_{\mathbb{I}\cup\mathbb{J}}$ so this means that

$$\$_{\mathbb{I}} = \mathbf{y}^T\mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(\frac{1}{2N^2\lambda}\left(\mathbf{I} + \frac{1}{N\lambda}\mathbf{B}\right)^{-1}\right)\mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}}\mathbf{y}$$

We wish to compare the worths of cells in $\mathbb{I}$ to their weights. From (11) we have

$$(N\lambda)^2\sum_{i\in\mathbb{I}} v_i^2 = \left|\mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{I}\cup\mathbb{J}}\mathbf{y}\right|^2$$

and using (17) we have

$$\begin{aligned}\mathbf{R}_{\mathbb{I}\cup\mathbb{J}}\mathbf{X}_{\mathbb{I}} &= \mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}} - \mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(N\lambda\mathbf{I} + \mathbf{B}\right)^{-1}\mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}} \\ &= \mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(\mathbf{I} - \left(N\lambda\mathbf{I} + \mathbf{B}\right)^{-1}\mathbf{B}\right) \\ &= \mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(N\lambda\mathbf{I} + \mathbf{B}\right)^{-1}\left(N\lambda\mathbf{I} + \mathbf{B} - \mathbf{B}\right) \\ &= \mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(\mathbf{I} + \frac{1}{N\lambda}\mathbf{B}\right)^{-1}\end{aligned}$$

Therefore,

$$\frac{\lambda}{2}\sum_{i\in\mathbb{I}} v_i^2 = \mathbf{y}^T\mathbf{R}_{\mathbb{J}}\mathbf{X}_{\mathbb{I}}\left(\frac{1}{2N^2\lambda}\left(\mathbf{I} + \frac{1}{N\lambda}\mathbf{B}\right)^{-2}\right)\mathbf{X}_{\mathbb{I}}^T\mathbf{R}_{\mathbb{J}}\mathbf{y}$$

$\mathbf{B}$ is real, symmetric and positive-semidefinite, so it has a full basis of orthogonal eigenvectors $\mathbf{u}_i$ and corresponding real, nonnegative eigenvalues μ_i. Multiplying any vector by $\left(\mathbf{I} + \frac{1}{N\lambda}\mathbf{B}\right)^{-1}$ scales the $\mathbf{u}_i$ component by factor $\frac{1}{1+\frac{\mu_i}{N\lambda}}$ and so it follows that

$$\frac{\lambda}{2}\sum_{i\in\mathbb{I}} v_i^2 \leq \$_{\mathbb{I}} \leq \left(1 + \frac{\max_{i\in\mathbb{I}}\mu_i}{N\lambda}\right)\frac{\lambda}{2}\sum_{i\in\mathbb{I}} v_i^2$$

Gershgorin's theorem bounds the μ_i: they must lie within the union of the 'Gershgorin discs'

$$D\left(B_{ii}, \sum_{i\neq j}|B_{ij}|\right)$$

which have centers B_{ii} and radiuses $\sum_{j\neq i}|B_{ij}|$. Now $|B_{ij}| = |\mathbf{x}_i^T\mathbf{R}_{\mathbb{J}}\mathbf{x}_j| \leq |\mathbf{x}_i||\mathbf{x}_j|$, so this implies that each μ_i satisfies

$$\mu_i \in [0, \max_{k\in\mathbb{I}}|\mathbf{x}_k|\sum_{j\in\mathbb{I}}|\mathbf{x}_j|]$$

□

The upper bound in theorem 3 is tight: we can construct an example which hits it. Suppose we have p identical cells each outputting $\mathbf{x}_i = \mathbf{y}, |\mathbf{y}|^2 = N$. Then the weights v_i satisfy

$$v_i = \frac{1}{\lambda N}(\mathbf{y} - \hat{\mathbf{y}})^T\mathbf{x} = \frac{1}{\lambda}(1 - pv_i) \Rightarrow (\lambda + p)v_i = 1$$

and the loss is

$$L = \frac{1}{2N}|\mathbf{y} - \hat{\mathbf{y}}|^2 + \frac{\lambda}{2}pv_1^2 = \frac{(1-pv_1)^2}{2} + \frac{p\lambda v_1^2}{2} = \frac{\lambda^2 v_1^2 + p\lambda v_1^2}{2}$$

If all cells had their output weights set to zero, then the loss would be $\frac{1}{2N}|\mathbf{y}|^2 = \frac{1}{2}$. Thus, the worth of the set $\{1,\ldots,p\}$ is

$$\begin{aligned}\$_{\{1,\ldots,p\}} &= \frac{1}{2} - \frac{(1-pv_1)^2}{2} - \frac{p\lambda v_1^2}{2} \\ &= \frac{2pv_1 - p^2v_1^2 - p\lambda v_1^2}{2} \\ &= \frac{\lambda + p}{2}pv_1^2 \\ &= \frac{pv_1}{2}\end{aligned}$$

This is equal to the upper bound in theorem 3:

$$\frac{\lambda + |\{1,\ldots,p\}|}{\lambda}\left(\frac{\lambda}{2}|\mathbf{v}_{\{1,\ldots,p\}}|^2\right) = \frac{\lambda+p}{\lambda}\left(\frac{\lambda}{2}pv_1^2\right) = \frac{pv_1}{2} = \$_{\{1,\ldots,p\}}$$

5.6 PMH makes the most of a small pool of producers

By selectively stabilising producers with useful outputs, PMH should cause pools of producers to concentrate on useful features, and make it possible to get equivalent performance using smaller numbers of producers.

We get a best-case estimate for the reduction in number of producers needed, by comparing two extremes: a pool of producers with completely random input weights, versus a pool of producers whose output is perfectly matched to the target $\mathbf{y}$.

We do this for the feedforward network in Figure 1, supposing that the input consists of n uncorrelated signals and that the target $\mathbf{y}$ is a random combination of the same n orthogonal signals with $|\mathbf{y}|^2 = N$. Initially the producers have random input weights, with the constraint (which can be encoded in p_0) that $|\mathbf{x}_1|^2 = \ldots = |\mathbf{x}_p|^2 = N$.

After running PMH for a long time, producers will tend to have outputs highly correlated (or anticorrelated) with the target $\mathbf{y}$. We argue that at best, this will reduce the number of producers needed for a given L by a factor of n.

In the best case, when every producer outputs an exact copy of the target $\mathbf{y}$ (or $-\mathbf{y}$) then you can show using (12) and (14) that the loss is $\frac{\lambda}{2(\lambda+p)}$.

In the 'no learning' case when the input weights to the producers are random, then for large enough p, the Marchenko-Pastur theorem implies the non-zero eigenvalues of $\mathbf{X}^T\mathbf{X}$ will be close to $\frac{pN}{n}$[2]. If we write $s_1^2, \ldots, s_N^2$ for the eigenvalues of $\mathbf{K}$, $\mathbf{u}_i$ for the corresponding eigenvectors, and write $\mathbf{y} = \sum_i \alpha_i \mathbf{u}_i$, then we find that

$$L = \frac{1}{2}\sum_{i=1}^{N} \alpha_i^2 \frac{N\lambda}{N\lambda + s_i^2} \tag{19}$$

Thus, the loss in this case is approximately $\frac{\lambda}{2(\lambda+p/n)}$. To achieve the same performance as when all producers output perfect copies of the target, we need approximately n times as many cells. In figure 2 we show that indeed, PMH substantially reduces the number of cells needed to attain a given performance L. In the example

[2] We have $\sum_i s_i^2 = \text{Trace}\, \mathbf{X}^T\mathbf{X} = pN$ and there can be only n nonzero eigenvalues. To find the interval they lie in, write $\mathbf{X} = \mathbf{S}\mathbf{W}$ where $\mathbf{S} \in \mathbb{R}^{N\times n}$ is the matrix of n uncorrelated signals and $\mathbf{W} \in \mathbb{R}^{n\times p}$ is the random weight matrix whose entries have mean 0 and variance $\frac{1}{n}$. Then $\mathbf{X}^T\mathbf{X} \approx N\mathbf{W}^T\mathbf{W}$. The Marchenko-Pastur law implies that, in the limit where $n, p \to \infty$ and n/p is fixed, the non-zero eigenvalues of $n\mathbf{W}^T\mathbf{W}$ lie between $\left(\sqrt{p} - \sqrt{n}\right)^2$ and $\left(\sqrt{p} + \sqrt{n}\right)^2$. Thus, the non-zero eigenvalues of $\mathbf{X}^T\mathbf{X}$ are in the range $\frac{N}{n}\left(\sqrt{p} \pm \sqrt{n}\right)^2 = \frac{pN}{n}\left(1 \pm \sqrt{\frac{n}{p}}\right)^2$.

simulated the stimuli to producers consist of random mixtures of 15 independent component signals, of which only 3 are of interest to consumers. Using PMH in this example reduces the number of cells needed for equivalent performance by about a factor of 5.

5.7 An implementation of PMH

We showed in theorems 2 and 3 that the worth of a cell i is approximately $\frac{\lambda}{2}\sum_j V_{ji}^2$ and that the worth of a small set of cells is approximately additive. This lets us implement PMH for the feedforward network in Figure 1.

In this implementation of PMH, each consumer j continually adjusts its input weights V_{ji} using a supervised learning rule, and producers generate new candidate input weights by an unsupervised learning rule. In the example we simulate in the next section, the supervised rule is a delta rule and the unsupervised rule does independent component analysis.

The supervised learning of the consumer input weights applies all the time, while the unsupervised learning rule only applies during a short 'proposal window' each time a cell decides to propose a change; this happens at different times for different producers.

Each producer continually monitors the strength of its output weights and uses that to infer its worth. If a producer has low worth it is more likely to change its input weights: at time t producer i proposes a change with probability proportional to $e^{-\frac{\lambda}{2}\alpha\sum_j V_{ji}^2(t)}$. The algorithm has a parameter t_{decide} which is the length of time that passes between deciding to propose a change, and deciding whether to accept it. If cell i decides at time t to propose a change, then from time t to time $t + t_{decide}$, it runs an unsupervised learning rule to update its input weights, while its output weights to consumers continue to be subject to adjustment by the consumers.

To propose a change, a producer may transiently raise the learning rate of its unsupervised rule very high — beyond the limit of stability of the unsupervised learning rule — then reduce the learning rate, with the aim of allowing the unsupervised rule to approximately converge on a new fixed point by $t + t_{decide}$.

At time $t+t_{decide}$ cell i measures $\sum_j V_{ji}^2$ again and decides whether to accept or reject the change. It accepts with probability

$$\min\left(1, \exp\left(\frac{\lambda}{2}(\beta-\alpha)\sum_j \left(V_{ji}^2(t+t_{decide}) - V_{ji}^2(t)\right)\right)\right)$$

and otherwise reverts its input weights to values remembered from

time t.

5.8 Simulation: PMH with independent component analysis

We test the algorithm described above on a form of 'cocktail party problem' (figure 3). The inputs that producers will see are linear combinations $\mathbf{s}_j = \sum_j A_{ij}\mathbf{b}_j$ of independent signals $\mathbf{b}_1, \mathbf{b}_2, \ldots$ with random weights A_{ij}. In this simulation there are 10 consumers, and the target outputs for the consumers depend only on a small subset of those hidden, independent signals — the others being distractors. This is a 'cocktail party problem' in the sense that the independent signals are like voices of many individuals, while the mixed signals $\mathbf{s}_1, \mathbf{s}_2, \ldots$ are picked up by microphones in various places. We know the consumers need to listen only to a few of these voices, but we do not know in advance which voices these will be. Each consumer j receives its ideal output $y_j(t)$ as a training signal. The loss function for the whole network is

$$L = \sum_j \left[\!\!\left[y_j - \sum_i V_{ji} x_i \right]\!\!\right]^2 + \frac{\lambda}{2}|\mathbf{v}_j|^2$$

Here, again, $[\![..]\!]$ means the root mean square.

This simulation shows how the neural marketplace framework can be used to tackle such a problem:

- Each producer uses an unsupervised learning rule to perform independent component analysis (ICA) and recover a single source $\mathbf{b}_i$. Because ICA will find a source at random, most producers will represent distractor sources.
- Each consumer j uses a delta rule to find weights $\mathbf{v}_j$ that minimise its contribution to L, namely

$$L_j = \left[\!\!\left[y_j - \sum_i v_{ji} x_i \right]\!\!\right]^2 + \frac{\lambda}{2}|\mathbf{v}_j|^2$$

- Each producer i infers from the sum of squares of its output weights, $\sum_j V_{ji}^2$, whether the source it recovered is relevant for the consumers. A small $\sum_j V_{ji}^2$ indicates irrelevance, prompting producer i to change its input weights and seek a different voice to present to the consumers.

In this simulation, each producer runs an unsupervised learning rule to perform ICA and find an independent component in the

signals it receives. When a producer decides to propose a change to its input weights, it temporarily raises the learning rate of the ICA rule very high, beyond its limit of stability, so that when the learning rate reverts to the usual small value, the neuron may settle on a different independent component.

When a producer i decides to propose a radical change, it starts a countdown timer, and remembers what its estimated worth $\frac{\lambda}{2}\sum_j V_{ji}^2$ was before it proposed the change, as well as its input weights $\mathbf{w}_i$ before proposing the change. When the countdown timer expires, the producer calculates its new estimated worth using the new values of V_{ji}, and it decides whether to keep the change, retaining the change with probability $a = \min(1, \exp((\beta - \alpha)\Delta\$_i)$ as described in Chapter 4.

Stimulus and target. The stimuli in this simulation are random linear combinations of 15 independent signals $b_1, \ldots, b_{15}$. Each $b_i(t)$ is independent and identically distributed, with $b_i + 1$ drawn from an exponential distribution with parameter 1 so that $\overline{b_i(t)} = 0, \overline{b_i(t)^2} = 1$. These independent signals are passed through a random orthonormal mixing matrix $\mathbf{A}$ before presentation to the network. The stimulus vector seen by the producers at time t is $\mathbf{s}(t) = \mathbf{A}\mathbf{b}(t)$.

The target for consumer j is

$$y_j(t) = \alpha_{j1} b_1(t) + \alpha_{j8} b_8(t) + \alpha_{j15} b_{15}(t)$$

with independently drawn random α:

$$\alpha_{j1} \sim N(0, \frac{1}{21}), \alpha_{j8} \sim N(0, \frac{4}{21}), \alpha_{j15} \sim N(0, \frac{16}{21})$$

Thus, sources 1, 8, and 15 are useful to the consumers (source 15 very important, source 1 less important), while the other 12 sources are purely distractors.

Unsupervised independent component analysis. The producers have linear activation so $x_i(t) = \mathbf{w}_i.\mathbf{s}(t)$. Each producer uses a Hebbian learning rule to perform ICA, specifically

$$\frac{d}{dt}\mathbf{w}_i \propto \frac{\mathbf{w}_i + \mu \tanh(x_i)\mathbf{s}}{|\mathbf{w}_i + \mu \tanh(x_i)\mathbf{s}|} - \mathbf{w}_i$$

This rule was described by (Hyvärinen and Oja, 1998) and works by extremising the kurtosis of $\mathbf{x}$. Its fixed points are values of $\mathbf{w}$ which make $\mathbf{x} \equiv b_i$ for some i (in fact one can perform ICA by extremising arbitrary combinations of high-order cumulants). For

the first iteration after deciding to propose a change, a neuron runs the rule with infinite learning rate μ, which means setting $\mathbf{w}_i = \frac{\mathbf{s}}{|\mathbf{s}|}$, and thereafter it uses a small positive learning rate μ. This simulates 'learning by memorisation' followed by refinement of the input weights $\mathbf{w}_i$ using Hebbian learning.

Delta rule for consumer input weights. Each consumer neuron receives a copy of its target $y(t)$ as a training signal and uses a delta rule to seek appropriate input weights $\mathbf{v}_j$:

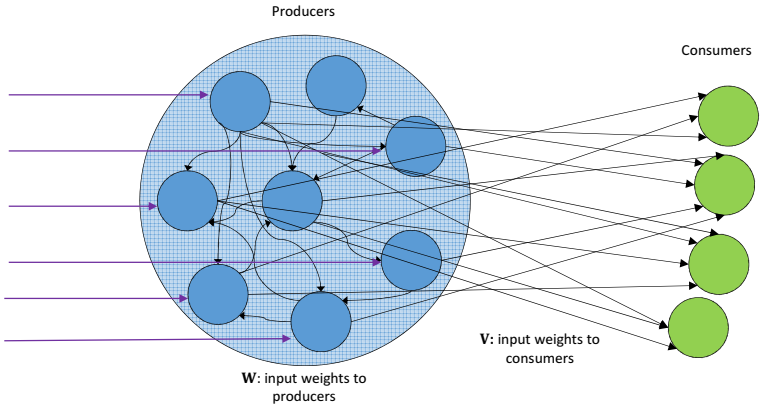
$$\frac{d}{dt}V_{ji} \propto -\lambda V_{ji} + x_i(y_j - \hat{y}_j)$$

Epochs of the simulation. So that we can separate the effect of retroaxonal learning from the effect of ICA alone, and the effect of the delta rule, we run the simulation in several stages. Initially, the input weights $\mathbf{w}_i$ to each consumer i are chosen uniformly at random from the unit hypersphere $|\mathbf{w}_i| = 1$. For the first 2000 iterations only the consumer weights $\mathbf{V}$ are plastic, and are learned using the delta rule. After 2000 iterations we switch on ICA with small positive learning rate μ. After another 2000 iterations, at the 4000th iteration, we switch on PMH, so producer neurons begin to occasionally propose radical changes to their input weights, with a probability that depends on the sum of squares of their output weights.

Figure 4 shows the learning scheme operated by producer cells once the retroaxonal rule is switched on.

Parameters. The parameters used in the simulation are as follows. 'Decision time' is the number of timesteps that elapse between a producer deciding to propose a change, and deciding whether to accept or revert the change.

Parameter	Value
β	2×10^4
α	$\beta/2$
κ	0.2
Number of unsupervised cells	1000
Number of supervised cells	10
Number of independent components	15
Decision time	3×10^4 timesteps
ICA learning rate μ	3×10^{-4}
Delta rule learning rate	3×10^{-5}



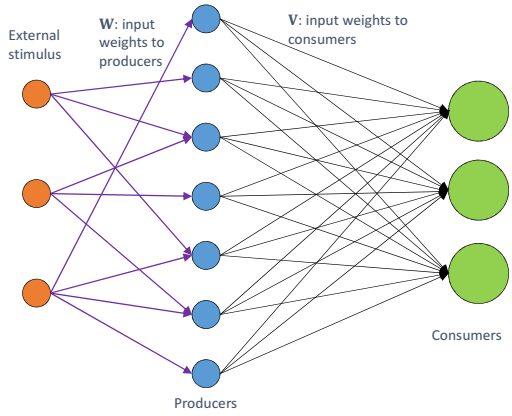


Figure 1: Pools of producers providing input to several consumers. The producers' input weights **W** are to be learned using PMH, while the consumer's input weights **V** will be learned using a standard supervised learning rule such as the delta rule. In chapter 5 we deal only with the feedforward network; in chapter 6 we discuss extension to the recurrently connected network.

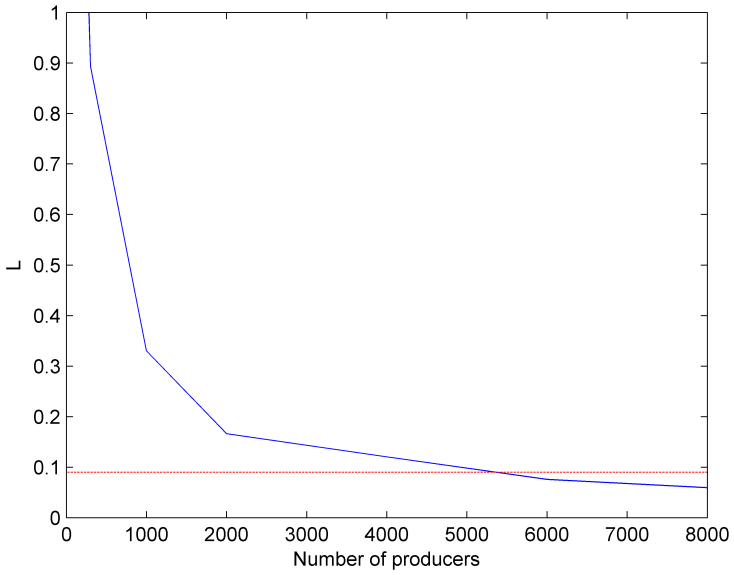


Figure 2: Performance using different numbers of producers, and ICA but no PMH, on the task of section 5.8. Parameters are the same as in section 5.8, except for the delta rule learning rate which, to avoid instability, scales with $\frac{1}{p}$ for $p > 1000$. The red horizontal line shows performance attained using 1000 cells with PMH, as in figure 6. Similar performance without PMH is not attained until there are over 5000 producers.

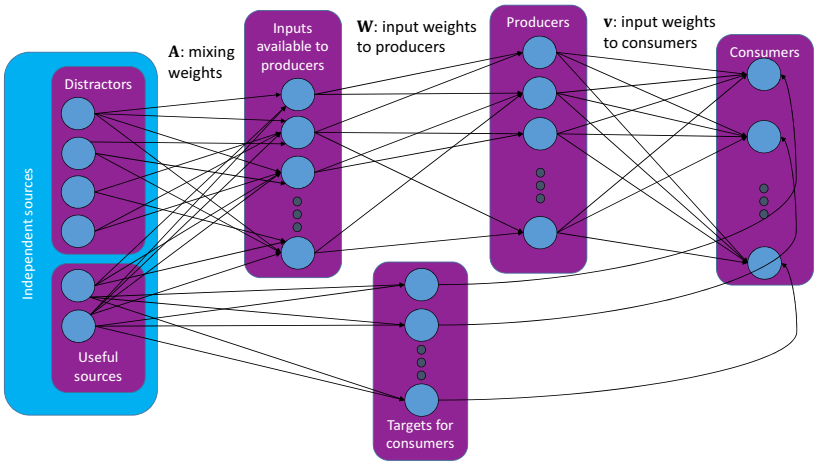


Figure 3: The 'cocktail party' problem used to test the PMH algorithm. The inputs available to the producers are generated by summing independent non-Gaussian sources (left), with dense, random weights $\mathbf{A}$. Target outputs for the consumers are generated as linear combinations of a small subset of the sources.

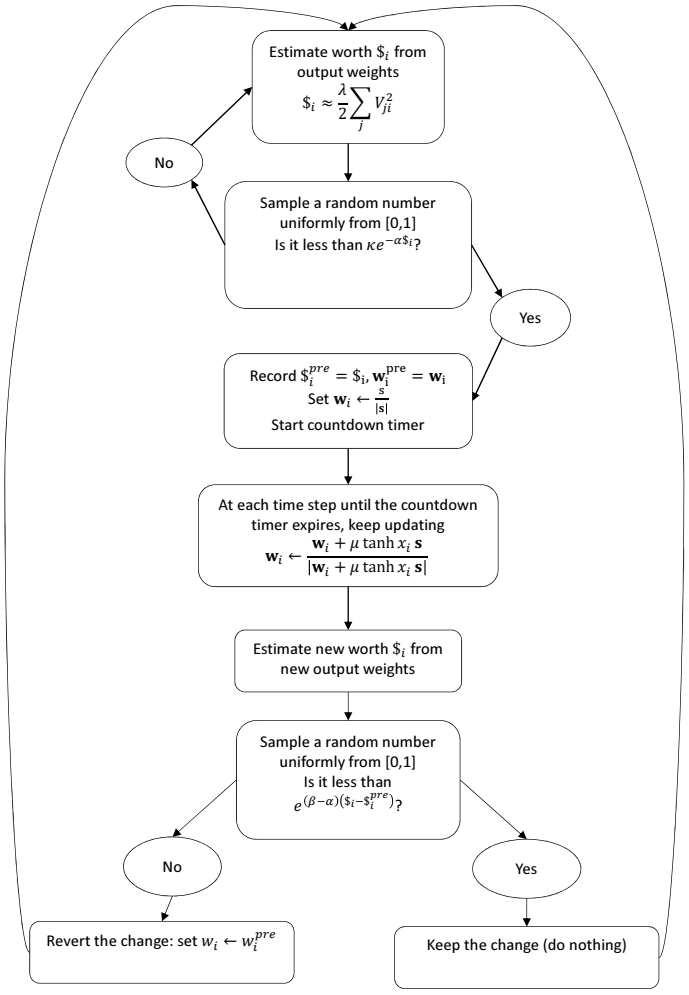


Figure 4: The version of PMH used by a single producer cell i in the simulation of section 5.8. The consumers run a delta rule continually, so that the output weights $\mathbf{V}$ adapt to the new activity of cell i when it proposes a change.

5.8.1 Results

The overall results of the algorithm are illustrated in figure 5.

PMH causes producers to concentrate on relevant features. Figure 5a illustrates the representations found by producer cells during the simulation. These are pseudocolour plots of the matrix **WA** at three different stages: at the beginning of the simulation, when the weights **W** are completely random; after ICA alone (iteration 4000); and after applying the retroaxonal rule (iteration 14000). Looking at **WA** is a way of visualising which of the independent components s contributes to the output of each producer cell. The output of producer i is $x_i(t) = \sum_j [\mathbf{WA}]_{ij} b_j(t)$. If the j^{th} row of **WA** is a row of all zeros except for ± 1 in position i then cell j outputs a pure copy of the independent component $b_i(t)$. Figure 5a shows that initially, each neuron outputs a mixture of sources. After ICA each producer neuron outputs a copy of an individual source, but with no bias towards the useful sources. After retroaxonal learning, the majority of neurons output copies of useful sources. Figure 7a shows the same thing in a different way.

PMH improves network performance. Figure 5b shows how the loss function evolves with learning. Since the consumer weights **V** are learned using a delta rule, they are not exactly optimal. Figure 5b shows a comparison of the actual network performance $L_{\mathbf{v}}$ (blue line) with the best possible loss $L = \min_{\mathbf{v}} L_{\mathbf{v}}$ given the current configuration of the producers (red dots, computed analytically from **W** and α). The two are very similar, except for a short period just after the retroaxonal rule is switched on. Here the two diverge because a large number of producers propose changes at once, and it takes some time for the consumer weights **V** to adapt.

Figure 5b shows that initially, switching on the ICA rule leads to a *decrease* in network performance, indicating the representation found by Hebbian unsupervised learning alone is actually worse than a random projection. After iteration 4000, when the retroaxonal rule is switched on, there is an improvement in performance, to a level better than was achieved before switching on ICA.

PMH allows continuing plasticity. The amount of plasticity goes down as the network learns. Figure 5c shows the number of cells performing a radical change in input weights on each iteration. A large spike occurs immediately after the retroaxonal rule is switched on. At this

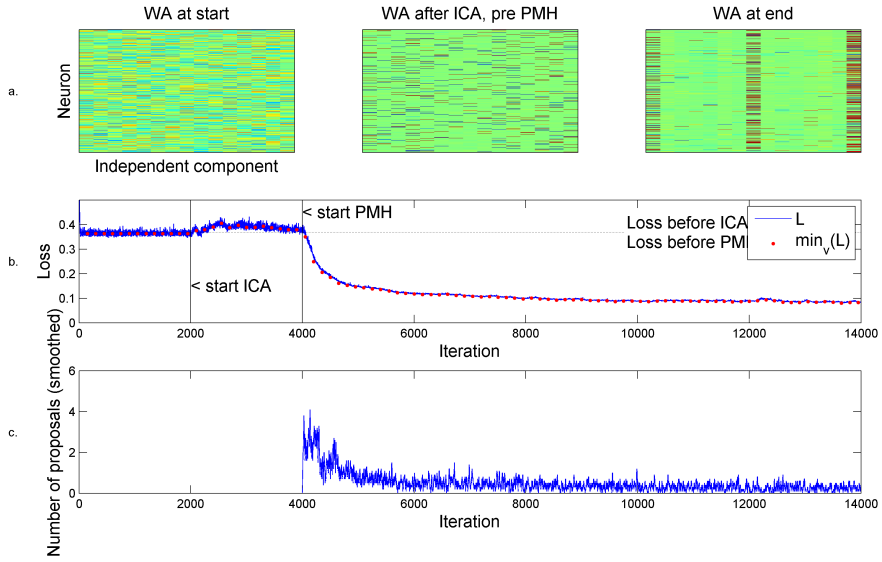


Figure 5: PMH improves the performance of a network on the cocktail party task by causing producers to focus on those independent components which are relevant to the consumers. For full explanation of this figure see section 5.8.1. *a.* Pseudocolour plot of representations found by the producer cells before learning (left); after ICA alone (middle); and after both ICA and retroaxonal learning (right). *b.* Loss function vs. iteration number (blue curve); red dots indicate the minimum loss function possible for the current producer weights $\mathbf{W}$, computed by analytically calculating the optimal $\mathbf{V}$. *c.* Number of cells experimenting with large changes to input weights at each iteration.

time a majority of producers had very little worth. The number of cells proposing changes per iteration decreases as more and more cells arrive at useful representations of their inputs; note however that the asymptotic value of this is not zero, indicating that neurons continue to occasionally change their representations even after performance has become good.

Output weights of a producer predict its worth, and worths are approximately additive for small sets of producers. Figure 6 tests the worth approximation $\$_{\mathbb{I}} \approx \frac{\lambda}{2} \sum_j \sum_{i \in \mathbb{I}} V_{ji}^2$. The approximation holds with good accuracy for sets of up to 100 neurons, except for just after the retroaxonal rule is switched on. This deviation occurs because when the retroaxonal rule is first switched on, a large number of producers change their input weights and the consumer weights $\mathbf{V}$ take some time to adapt. The worth of cell i is defined as the change in $\min_{\mathbf{v}} L_{\mathbf{V}}$ if cell i fell silent. We showed in theorem 2 that the worth of each producer can be estimated from the *optimal* values

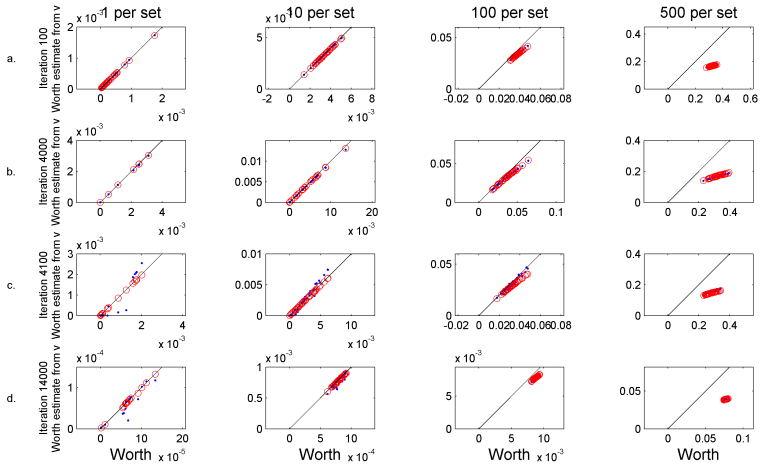


Figure 6: The approximation $\$_{\mathbb{I}} \approx \frac{\lambda}{2} \sum_j \sum_{i \in \mathbb{I}} V_{ji}^2$ holds with good accuracy during simulations of the PMH algorithm. For further discussion of this figure see section 5.8.1. Each panel shows a scatter plot of actual worth of a set of neurons (computed by calculating L with and without the set) vs. its approximated value $\frac{\lambda}{2} \sum_j \sum_{i \in \mathbb{I}} V_{ji}^2$. Blue dots indicate the approximation computed from the current weights $\mathbf{V}$ learned by the delta rule; red circles indicate the values predicted from the optimal $\mathbf{V}$ derived analytically. Each column of panels shows results for a different size of set of cells, with each point representing a randomly-drawn set of this many cells. Each row of panels corresponds to a different time point in the simulation: early in running of ICA alone; shortly before the retroaxonal rule is switched on; shortly after the retroaxonal rule is switched on; and substantially after it is switched on. The diagonal line in each plot is equality.

of the weights $\mathbf{V}$. The deviation of the blue dots from the diagonal line for single cells at iteration 4100 shows that when the consumer weights $\mathbf{V}$ are far from optimal, estimating worth from $\mathbf{V}$ can fail. We can compute the optimal weights analytically to check the result of theorem 2: the red circles on the same plot, showing worths estimated from these optimal weights, lie close to the line of equality, as we expect.

PMH makes worths converge around a common value. PMH systematically destabilises cells with low worth. It also tends to reduce the worths of the highest-worth cells, as more and more cells begin to provide the same thing they offer. Figure 7 shows how the distribution of producers' worths evolves over time.

We define the 'dominant source' for a cell i as

$$\text{argmax}_j |[\mathbf{WA}]_{ij}|$$

Figure 7a shows how the distribution of dominant sources changes during the simulation. Initially, with random weights, and after running ICA alone, the distribution of dominant sources is fairly uniform. After running the retroaxonal rule for a long time, the vast majority of producers' outputs are dominated by useful sources.

Figure 7b shows the estimated worth of each producer cell as a function of time. Lines are coloured by dominant source of the cell. After ICA is switched on, the worth of some cells increases. The cells whose worths increase are those whose activity is dominated by important sources. After the retroaxonal rule is switched on, the worth of each individual cell representing source 15 drops dramatically, the network performance improves, and the number of cells representing source 15 increases. Thus, network performance increases while certain cells' worths decrease. This is predicted by the maths of ridge regression: when p neurons represent source 15, their output weights will scale as $\frac{1}{\lambda+p}$ and individual worths scale approximately as $\frac{1}{(\lambda+p)^2}$. Thus, when large numbers of cells represent source 15, each one's worth will be lower, while network performance is overall higher.

Figure 7c shows another representation of how the distribution of worths changes over time. Since PMH systematically destabilises cells whose worths are low, over time the majority of cells' worths converge around a common value.

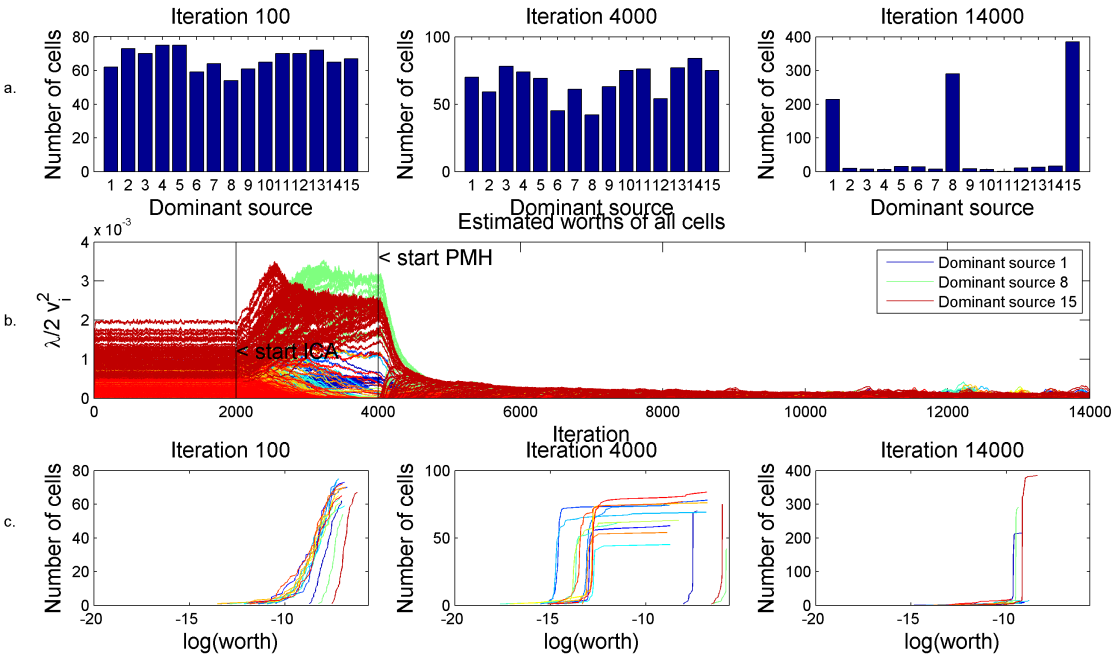


Figure 7: Increased network performance does not correspond to increased worth of individual producer cells. Instead, PMH causes all cells' worths to converge around a common value. See section 5.8.1 for further explanation of this figure. a. Histogram of the number of cells dominated by each source at different iterations. The retroaxonal rule causes the majority of producer cells to represent useful sources b. Each curve shows the estimated worth of a single producer cell as a function of iteration number; curves are colour coded by dominant sources. c. Unnormalised cumulative distribution of worths, with cells classified by dominant source (colour code as in b.). After ICA alone, cells with useful dominant sources have higher worth, but are no more numerous than, those with useless dominant sources. PMH selectively destabilises cells with low worth, with the effect that by iteration 14000, most cells represent useful sources and their worths are tightly clustered around a common value.

5.9 Simultaneous perturbations to many producers' outputs

In Chapter 6 we study the worths of cells in recurrently connected pools of producers. In those networks, silencing a single cell can have effects on many other producers' activities. We will therefore find it useful to have an expression for the general 'perturbation penalty'. This is the effect on L of adding a perturbation $\Delta\mathbf{x}_i$ to the activity of each cell i, and then letting each consumer relearn its input weights, so that the consumer input weights are optimal given the new activities of the producers.

The approximation, whose error is small whenever $\rho(\Delta\mathbf{K}) \ll N\lambda$, is

$$\Delta L = -\frac{1}{2N^2\lambda}\sum_i 2N\lambda v_i(\Delta\mathbf{x}_i^T\mathbf{r}) + |\mathbf{r}^T\Delta\mathbf{x}_i|^2 + O\left(\frac{1}{N}\left(\frac{|\mathbf{r}|\rho(\Delta\mathbf{K})}{N\lambda}\right)^2\right) \tag{20}$$

This approximation will be used in chapter 6 where we estimate the worths of producers in recurrently connected networks. To estimate the worth of cell i, we will let $\Delta\mathbf{x}_j$ be the perturbation to $\mathbf{x}_j$ that arises if cell i is silenced and $\Delta\mathbf{x}_i = -\mathbf{x}_i$. In the networks of chapter 6, apart from the cell i that is silenced, other cells j have their outputs changed by only a small amount $\Delta\mathbf{x}_j$, and (20) simplifies to

$$\$_i = \Delta L \approx \frac{\lambda}{2}v_i^2 + \sum_{j\neq i} v_j\frac{\Delta\mathbf{x}_j^T\mathbf{r}}{N}$$

To check this approximation is applicable we will need to check the spectral radius of $\Delta\mathbf{K}$. One can show that $\rho(\Delta\mathbf{K}) \leq \sum_i 2|\mathbf{x}_i^T\Delta\mathbf{x}_i| + |\Delta\mathbf{x}_i|^2$ so it suffices to show $\sum_i 2|\mathbf{x}_i^T\Delta\mathbf{x}_i| + |\Delta\mathbf{x}_i|^2 \ll N\lambda$. To show $\rho(\Delta\mathbf{K}) \leq \sum_i 2|\mathbf{x}_i^T\Delta\mathbf{x}_i| + |\Delta\mathbf{x}_i|^2$, recall that $\Delta\mathbf{K}$ is a sum over cells i:

$$\Delta\mathbf{K} = \sum_i \Delta(\mathbf{x}_i\mathbf{x}_i^T)$$

The triangle inequality for operator norms implies that

$$\begin{aligned}\rho(\Delta\mathbf{K}) &\leq \sum_i \rho(\Delta(\mathbf{x}_i\mathbf{x}_i^T)) \\ &\leq \sum_i 2\rho(\Delta\mathbf{x}_i\mathbf{x}_i^T) + \rho(\Delta\mathbf{x}_i\mathbf{x}_i^T) \\ &\leq \sum_i 2|\mathbf{x}_i^T\Delta\mathbf{x}_i| + |\Delta\mathbf{x}_i|^2\end{aligned}$$

Suppose we know that $\rho(\Delta\mathbf{K}) < N\lambda$. To derive the approximation (20), recall that

$$L = \frac{1}{2N}\mathbf{y}^T\mathbf{R}\mathbf{y} \tag{21}$$

If $\mathbf{X}$ changes to $\mathbf{X} + \Delta\mathbf{X}$ and the consequent change to $\mathbf{R}$ is $\Delta\mathbf{R}$, then

$$\Delta L = \frac{1}{2N}\mathbf{y}^T\Delta\mathbf{R}\mathbf{y}$$

We can represent $\Delta\mathbf{R}$ as a power series in terms of $\mathbf{y}, \Delta\mathbf{K}$, and $\mathbf{R}$:

$$\begin{aligned}
\mathbf{R} + \Delta\mathbf{R} &= \left(I + \frac{\mathbf{K}}{N\lambda} + \frac{\Delta\mathbf{K}}{N\lambda}\right)^{-1} \\
&= \left(\left\{I + \frac{\mathbf{K}}{N\lambda}\right\}\left\{I + \left(I + \frac{\mathbf{K}}{N\lambda}\right)^{-1}\frac{\Delta\mathbf{K}}{N\lambda}\right\}\right)^{-1} \\
&= \left(I + \mathbf{R}\frac{\Delta\mathbf{K}}{N\lambda}\right)^{-1}\mathbf{R} \\
&= \sum_{k=0}^{\infty}\left(-\mathbf{R}\frac{\Delta\mathbf{K}}{N\lambda}\right)^k\mathbf{R}
\end{aligned}$$

Since $\mathbf{R}\Delta\mathbf{K}$ is symmetric, it has operator norm equal to its spectral radius, so this power series expansion is valid provided the spectral radius $\rho\left(-\mathbf{R}\frac{\Delta\mathbf{K}}{N\lambda}\right)$ is less than 1. In fact, since $\mathbf{R}$ is always a contraction, we need simply that $\rho(\Delta\mathbf{K}) < N\lambda$, which we have by assumption. Using the power series expansion of $\mathbf{R} + \Delta\mathbf{R}$ in (21), we have

$$\Delta L = \frac{1}{2N}\mathbf{y}^T\sum_{k=1}^{\infty}\left(-\mathbf{R}\frac{\Delta\mathbf{K}}{N\lambda}\right)^k\mathbf{R}\mathbf{y} \tag{22}$$

Putting $\mathbf{r} = \mathbf{R}\mathbf{y}$ for the residual, we can write

$$\begin{aligned}
\Delta L = &-\frac{1}{2N^2\lambda}\mathbf{r}^T\Delta\mathbf{K}\mathbf{r} \\
&+ O\left(\frac{1}{N}\left(\frac{|\mathbf{r}|\rho(\Delta\mathbf{K})}{N\lambda}\right)^2\right)
\end{aligned} \tag{23}$$

The leading term is additive over cells i since $\mathbf{K} = \sum_i \mathbf{x}_i\mathbf{x}_i^T$. To expand it we use

$$\Delta\mathbf{K} = \sum_i (\mathbf{x}_i + \Delta\mathbf{x}_i)(\mathbf{x}_i^T + \Delta\mathbf{x}_i^T) - \mathbf{x}_i\mathbf{x}_i^T$$

and recall that the output weight of cell i to the consumer is given by $v_i = \frac{\mathbf{x}_i^T \mathbf{r}}{N\lambda}$, so that

$$\mathbf{r}^T \Delta \mathbf{K} \mathbf{r} = \sum_i 2N\lambda v_i (\Delta \mathbf{x}_i^T \mathbf{r}) + |\mathbf{r}^T \Delta \mathbf{x}_i|^2 \tag{24}$$

Note that when one cell i is silenced, i.e. $\Delta \mathbf{x}_i = -\mathbf{x}_i$, the above simplifies to $-(N\lambda v_i)^2$ and inserting this in (23) we recover the old estimate $\$_i \approx \frac{\lambda}{2} v_i^2$ for simple feedforward networks. More generally, putting (24) into (23) we get the required result (20).

5.10 Optimal brain damage (OBD) and worth

We noted already that our notion of 'worth' bears some relation to 'saliency' in another theory, Optimal Brain Damage (OBD) (LeCun et al., 1989). LeCun et al. (1989) showed how to estimate 'saliencies' of synaptic weights in a feedforward network that has been trained using backprop. In their work, the saliency of a synaptic weight is an estimate of how the loss function will go up if that synaptic weight is set to zero. The method is useful in artificial neural networks: it allows pruning, leaving a simpler network which takes less computing power to simulate and may generalise better.

Our worth calculation differs from the OBD calculation of saliency in that we allowed for our 'consumer' cells to relearn their input weights after a cell is deleted. By carefully defining the loss function for OBD to take this into account, and making a fairly straightforward transformation of our network, we could make our 'worth' equivalent to 'saliency' of a special dummy 'repeater' weight in the transformed network. The required transformation of the network is shown in Figure 8. If the loss function for OBD is defined appropriately, so that it takes into account the effect of a consumer relearning its weights, then the change in the worth of cell i in the original network is the saliency of its repeater weight in the transformed network.

The OBD calculation of saliency is applied by LeCun to multilayer feedforward networks, and at first sight it looks like it could enable us to calculate worths of cells in deeper networks. However, we run into multiple problems:

1. The calculation of saliency in OBD cannot be done using slow, rather than fast, retrograde signals. Like backprop, the calculation depends on measuring the correlation between a cell's input, and a (second-order) partial derivative which is calculated using retrograde signals.

2. OBD depends on every downstream weight being at a locally optimal value, achieved for example by pretraining the network using backprop.

3. OBD works for feedforward, steady-state networks, not dynamical networks with recurrent connections.

In Chapter 6 we demonstrate an alternative and also very fast way to estimate worths of cells in simulation. Like OBD, it still uses fast retrograde signals, but it does not require downstream weights to be pretrained, it operates in dynamical networks, and it does not require the network to have feedforward structure. We also demonstrate a method using *slow* retrograde signals to make very rough estimates of worth in the same networks.

5.11 Conclusion

In this chapter we presented a type of network in which cells can estimate their worths from their output weights. We introduced a network architecture and standard loss function (section 5.2). We reviewed standard results on how to find the consumer's optimal weights given this loss function.

In section 5.5 we showed that in this type of network, the worth of a producer is approximated by the sum of its squared output weights to consumers, multiplied by the weight penalty parameter: $\$_i \approx \frac{\lambda}{2} \sum_i V_{ij}^2$. We gave a bound for the error in that approximation.

PMH is designed to make good use of a limited pool of producers. In section 5.6 we made a rough prediction about relative numbers of producers needed to get the same performance with and without PMH, for a toy problem.

In section 5.7 we wrote out explicitly how the PMH algorithm would work for this specific network. In this scheme producers use an unsupervised learning rule to generate new candidate input weights, and they use their output weights to calculate running estimates of their worth. If a producer has small output weights to consumers, it deduces it has low worth and increases its unsupervised learning rate.

In section 5.8 we implemented this algorithm and showed that indeed, PMH improves the performance of the network over using the unsupervised learning rule alone. We also showed that to get the same performance using the unsupervised learning rule, without PMH, we need several times as many cells.

In section 5.9 we used some of the machinery developed to derive an estimate for the effect on performance when many producers

have their activities simultaneously perturbed. This will be crucial in the next chapter, where we try to estimate worths of producers in recurrently connected, random networks.

In section 5.10 we discussed the relationship of this work to the theory of 'optimal brain damage' (OBD). 'Saliency' in OBD and 'worth' here are related, but not identical. We identified limitations in the applicability of the OBD strategy to calculating worths of cells in neural networks that may be recurrently connected and where there are no fast retrograde signals, and in the next chapter we will try to overcome these.

The conclusion of this chapter is that there is at least one simple model network where PMH works, where cells can estimate their worths using slow retrograde messages — specifically by monitoring their output weights — and where the kind of learning scheme proposed by the retroaxonal hypothesis does indeed improve the performance of the network.

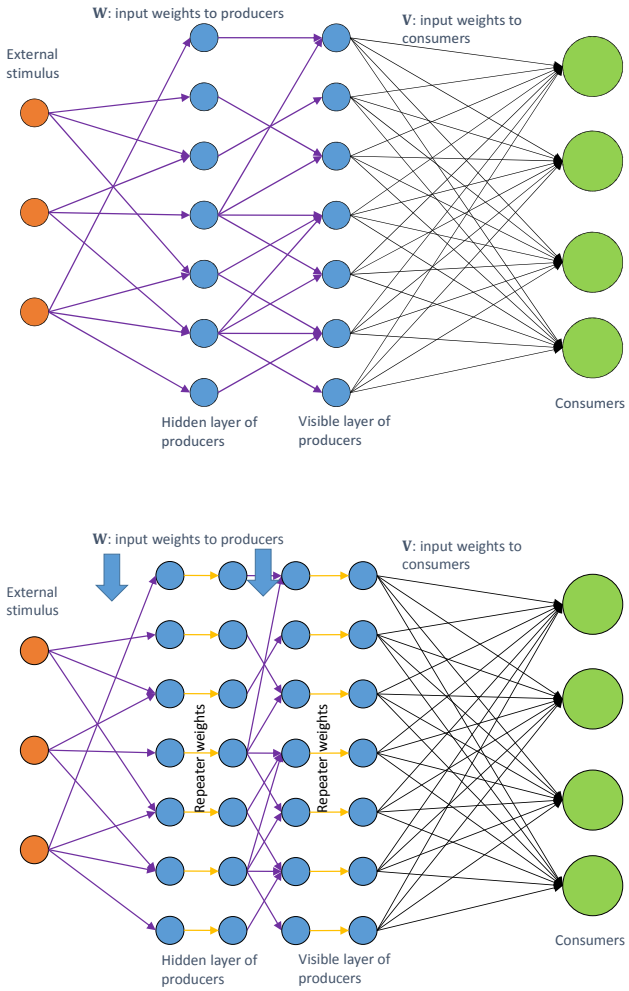


Figure 8: Transforming a network so that OBD 'saliency' becomes our 'worth'. To construct the bottom network from the top one, each producer cell (blue) is replaced by two cells. The first of the two new cells has the old cell's inputs and a unit 'repeater' weight (orange) to the second cell, while the second cell has the output synapses of the old cell. The time-averaged saliency of a repeater weight in the transformed network is the worth of the corresponding cell in the original network, provided the loss function used in OBD allows for relearning of the consumer weights when a producer changes its activity.

6 Message-passing to estimate indirect worth

6.1 Introduction

In chapter 5 we showed that when a single layer of producer cells offer their outputs to some supervised consumer cells, each unsupervised producer cell can estimate its worth from its output weights to the consumers. As described in chapter 3, in biological neural networks, consolidation of synaptic weight change may depend on retrograde signalling. Therefore, in that simple feedforward configuration, a cell can estimate its worth using retrograde signals that pass at a biologically realistic speed.

What if the synaptic weights downstream of one producer cell have not been perfectly learned, and what if there are recurrent connections? Is it true, for example, that if a producer provides strong input to cells of high worth, it probably has high worth itself?

To answer this we now look at recurrently connected pools of producers with random synaptic weights. Actual weight matrices in the cortex do not appear to be random (Song et al., 2005). However, modelling them as random allows us to use approximations and assumptions that make the calculations more tractable. This analysis therefore offers a first step towards a theoretical understanding of whether retroaxonal signals could be used to guide plasticity in recurrent cortical networks.

We will see that in these networks, it is easy to compute accurate estimates of worth using unbiological, fast retrograde signals. We also derive and test coarser estimates that can be calculated using *slow* retrograde signals. To do this, we suppose that cells pass retroaxonal signals encoding certain information e.g. synaptic weights, or estimated worths of downstream cells; and we study the 'belief' a cell might have about its worth given the retroaxonal messages it receives. 'Belief about worth' is used here to mean the conditional distribution of a cell's worth, over random weight matrices, given the retroaxonally transmitted information. By making various approximations, we derive estimates for the mean and variance of this conditional distribution, and we verify these estimates in simulation. Over the population of cells in a network, we find that the conditional coefficient of variation (CV) of worth is smaller for cells of large expected worth and larger for cells of small expected worth. Often, especially for cells with small expected worth, we find the coefficient of variation is much greater than 1, and a cell cannot reliably predict whether its worth is positive or negative. In these cases, the conditional distribution of a cell's worth given retrograde

signals provides little more than probabilistic bounds for the absolute value of the worth. Section 6.6 shows that if we abandon the attempt to compute *signed* estimates of worth using slow retrograde signals, there is a simpler way of using slow retrograde signals to compute symmetric probabilistic bounds.

6.2 Model networks

In this chapter, we continue to study networks of producers and consumers, with the same loss function we introduced in chapter 5.

Hidden and visible producer cells. We consider model networks where connections from producers to consumers are sparse. For simplicity there is no structural plasticity, so certain weights from producers to consumers are constrained always to be zero. We do this by making a minority of producer cells 'visible' and the rest 'hidden'. Hiding cell j means constraining each consumer i to have zero input from cell j, $V_{ij} = 0$. A cartoon of the network is shown in figure 9. One could think of this situation as analogous to the projections from neocortex to subcortical motor output structures such as striatum, superior colliculus or brainstem nuclei, which arise from a small subset of cortical neurons (Harris and Shepherd, 2015). The architecture could be considered a variant of the 'liquid state machine' (Maass et al., 2002).

Direct and indirect worth. In section 6.5.2 we will show how to decompose a cell's worth into 'direct' worth that comes from its direct output to consumers, and 'indirect' worth via connections to other producers. We will make use of this to derive estimates of a cell's worth given information that may be transmitted retroaxonally. Hidden cells have no direct worth, only indirect worth. Visible cells may have both direct and indirect worth. Indirect worth can be negative: for example, a hidden producer's output may degrade visible producers' outputs. In the marketplace analogy, this hidden producer with negative worth is like a supplier of faulty components that end up in consumer goods.

Neighbours. We will use 'neighbour of cell j' to mean any cell i connected to cell j by a synapse — in either direction. Cell i is a 'downstream neighbour' if cell j provides input to cell i and an 'upstream neighbour' if cell i provides input to cell j.

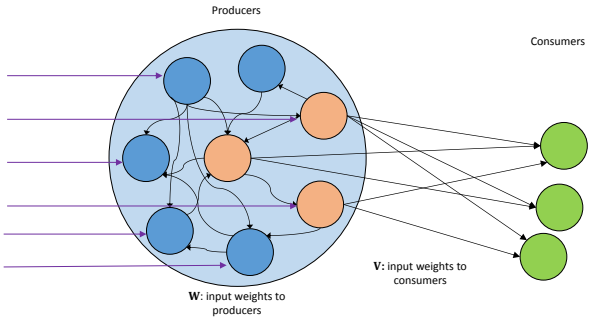


Figure 9: Recurrently connected producers serving a number of consumers. The producers in orange are 'visible' and they are the only ones from which the consumers are allowed to have nonzero input weights.

Firing rate dynamics. We model a pool of producers having continuously time-varying firing rates $x_i(t)$, obeying linear dynamics:

$$\frac{d\mathbf{x}}{dt} = \frac{1}{\tau}(\mathbf{W} - \mathbf{I})\mathbf{x} + \mathbf{s} \tag{25}$$

$x_i(t)$ is the firing rate of cell i at time t, W_{ij} is the synaptic weight from cell j to cell i, τ is a time constant and s_i is the external stimulus to cell i. $\mathbf{x} = \mathbf{x}(t)$ is a vector with one element per cell (this is not the same as the earlier notation where $\mathbf{x}_i$ was a vector of outputs for one cell i, with one element per time step). $\mathbf{I}$ is the identity matrix.

Random weights. In the brain, synaptic weights are not random in the way that we model them here (Song et al., 2005). However, we do not know the true details of learning rules used in biological neural networks, and so in this chapter we use random networks as a tractable test-case for recurrent networks. To check that the approximations we make are broadly valid, we will simulate and check them using three rather different kinds of random weight matrix. Precisely how these random weight matrices are generated is not important for the first few parts of this chapter, so the form of the random weight matrices will only be explicitly defined in section 6.4. For now, we note that our random weight matrices will have *some* structure, namely a diversity of 'cell types'. A 'cell type' is defined by a distribution of output weights: each W_{ij} is drawn independently at random from a distribution that depends on the presynaptic cell j. An 'excitatory cell' has all non-negative output weights; an 'inhibitory cell' has all non-positive output weights. No

cost function on $\mathbf{W}$ is explicitly included, but the distribution of W_{ij} for each cell type controls the strength and number of synapses. Sparse connectivity can be obtained by selecting distributions for W_{ij} with a large point mass at 0.

6.3 Estimating worth with fast retrograde signals

Before considering whether a cell can estimate its worth using only slow retrograde signals, we show how to efficiently estimate the worth of a cell in a simulation. We show that it is easy to accurately estimate the worths of cells without actually silencing them, by passing fast retrograde signals (too fast to occur in biological networks). This works in recurrent networks provided they are not too 'loopy', a term to be defined later.

This estimate of worth using fast retrograde signals is useful in two ways. First, in machine learning, it is sometimes helpful to prune a network: we have already discussed precedents such as optimal brain damage and budget perceptrons (sections 4.2, 5.10). Second, and more relevant to our core aim of evaluating the biological plausibility of the retroaxonal hypothesis for recurrent networks, we can use these fast, accurate estimates of worth to check the estimates of worth that we later compute using methods that avoid fast retrograde signalling.

The brute-force way to measure the worth of a cell in a simulated network is to remove the cell from the simulated network, then re-evaluate the network performance. This is slow: if it takes time t to evaluate the performance of the network, then to measure the worth of n cells takes time $O(nt)$. We show that it is possible to accurately estimate the worths of all the cells at once in time $O(t)$. Part of the calculation is related to a previously-described method called 'backpropagation through time' (BPTT) (Werbos, 1990), which we discuss in 6.3.3.

All the networks in this thesis are artificial, but networks that do this calculation of worth using fast retrograde signals are especially artificial: they will play back activity reversed in time and pass fast retrograde messages. Rather than being abstractions or simplifications of real neural networks, these networks have features with no counterpart in biology.

6.3.1 Worth in terms of perturbations to visible cells' outputs

When one cell in a recurrently connected network is silenced, many other cells' firing patterns are affected. This means that to estimate

the worth of a cell in a recurrently connected network of producers, we need to know how the loss L is affected by simultaneous perturbation of the outputs of multiple cells.

In this section we show how to write down the effect on L of simultaneous perturbations to many of the x_i, while accounting for the fact that the consumers may relearn their input weights after the producers' activities change. A simple linear approximation using partial derivatives does not suffice, since that only handles small perturbations to cells' outputs, but when a cell is silenced, the perturbation to that cell's output is large. In sections 6.3.2 and 6.3.4 we show, using the results of this section (6.3.1), how to compute accurate estimates of worth using fast retrograde signals.

Notation. We will use the notation $\langle x, y \rangle$ to denote the time-average of xy, so it is the inner product corresponding to the RMS norm $[\![x]\!]$. We write $r_j(t) = y_j(t) - \hat{y}_j(t)$ for the residual error of consumer cell j.

In chapter 5 we showed that when the outputs of the set $\mathbb{I}$ of visible cells are perturbed by a small amount to $x_i(t) + \Delta x_i(t)$, the effect on performance is

$$\Delta L \approx \sum_j \sum_{i \in \mathbb{I}} \langle \Delta x_i, -V_{ji} r_j \rangle - \frac{1}{2\lambda} \langle \Delta x_i, r_j \rangle^2 \tag{26}$$

Here V_{ji} is the synaptic weight from producer i to consumer j, which we assume is optimal given the x_i.

The worth of cell i is, by definition, the increase in L when cell i is silenced. In our recurrent network, silencing one cell i has a drastic effect on the output of cell i and a small effect on other cells' outputs. Therefore, we approximate the worth of cell i by

$$\$_i \approx \sum_j \left(-\frac{\lambda}{2} V_{ji}^2 + \sum_k \langle \Delta x_k, -V_{jk} r_j \rangle \right) \tag{27}$$

where Δx_k is the perturbation to the output of cell k when cell i is silenced. We do not need to restrict the sum to visible cells $k \in \mathbb{I}$, because for hidden cells, $V_{jk} \equiv 0$.

We can rewrite (27) by defining

$$b_i = -\sum_j V_{ji} r_j$$

The signal b_i is something like a partial derivative of L with respect to the activity of cell i, but *assuming that the consumer learns new optimal input weights* as cell i changes its activity.

Using the notation b_i in (27) gives

$$\$_i \approx \sum_k \langle \Delta x_k, b_k \rangle - \frac{\lambda}{2} \sum_j V_{ji}^2 \tag{28}$$

Rewriting the approximation (27) in this way will be useful in sections 6.3.2 and 6.3.4 where we show how to estimate worth from $\mathbf{b}$ by passing fast retrograde signals.

The sign of the second sum $-\frac{\lambda}{2}\sum_j V_{ji}^2$ in (28) looks surprising at first: cell i should have high worth if it has large output weights to consumers. However, for cell i the term $\langle \Delta x_k, b_k \rangle$ is $\lambda \sum_j V_{ji}^2$ and so the total contribution to (28) from cell i is $\frac{\lambda}{2} V_{ji}^2$. The term $-\frac{\lambda}{2}\sum_j V_{ji}^2$ is a quadratic-order adjustment to the linear approximation, needed only for the silenced cell i whose perturbation is large.

The equation (28) is specific to the loss function of chapter 5, but we could extend to other loss functions that have locally linear approximations by using different b_i. For example we could add a term to L which penalises large $|x_i|$, setting $L' = L + \tilde{\lambda} \sum_i [\![x_i]\!]^2$, and $b_i' = b_i - \tilde{\lambda} x_i$.

Linworth. For convenience we define 'linworth' as follows. If Δx_k are the perturbations that arise from silencing cell i, then

$$\$_i^{(\mathrm{lin})} = \sum_k \langle \Delta x_k, b_k \rangle$$

is the 'linworth' of cell i.

The expression in (28) approximates the worth of cell i as its linworth, plus a function of its output synaptic weights, $-\frac{\lambda}{2}\sum_j V_{ji}^2$. In the next sections we show that linworth can be estimated using fast retrograde messages, simultaneously for all cells, and without actually silencing any cell. This will enable us to estimate the worth of every cell in an efficient way.

6.3.2 Using fast retrograde messages to compute the effect on loss of perturbing the stimulus

As a step towards computing the linworths of cells, we first show how to compute a partial derivative a_i of $\sum_k \langle x_k, b_k \rangle$ with respect to each cell's *external stimulus*, s_i. In section 6.3.4 we show how to compute a stimulus perturbation, i.e. a Δs_i, which approximately silences cell i. For this special stimulus perturbation which we will call ζ_i, we will find that $\$_i^{(\mathrm{lin})} \approx \langle a_i, \zeta_i \rangle$.

Suppose that the stimulus is perturbed to $\mathbf{s} + \Delta\mathbf{s}$ and write $\Delta\mathbf{x}$ for the resulting change to $\mathbf{x}$. One can show that, for general real-valued $\mathbf{W}, \beta, \mathbf{s}$, if α satisfies

$$\frac{d\alpha(-t)}{dt} = \frac{1}{\tau}(\mathbf{W}^T - \mathbf{I})\alpha(-t) + \beta(-t) \tag{29}$$

and $\mathbf{x}$ satisfies

$$\frac{d\mathbf{x}}{dt} = \frac{1}{\tau}(\mathbf{W} - \mathbf{I})\mathbf{x}(t) + \mathbf{s}(t)$$

then

$$\sum_i \langle \Delta x_i, \beta_i \rangle \equiv \sum_j \langle \Delta s_j, \alpha_j \rangle \tag{30}$$

Here, α, β without subscripts represent vectors over cells. We will eventually use (30) to compute worth by setting $\beta_i = b_i$.

An easy way to prove (30) is to look at the network dynamics in the frequency domain (note that the eventual method we derive to estimate worth in simulation does not require any Fourier transform).

Network dynamics in the frequency domain. Taking the Fourier transform of (25), we obtain

$$\tilde{\mathbf{x}}(\omega) = g(\omega)\mathbf{W}\tilde{\mathbf{x}} + \tau g(\omega)\tilde{\mathbf{s}}(\omega) \Rightarrow \tilde{\mathbf{x}}(\omega) = \mathbf{G}(\omega)\tilde{\mathbf{s}}(\omega) \tag{31}$$

Here the chosen convention for Fourier transforms is

$$f(t) = \int \tilde{f}(\omega) e^{2\pi i \omega t} d\omega$$

$\tilde{\mathbf{x}}$ is, like $\mathbf{x}$, a vector with one entry per cell. The frequency-dependent 'transfer matrix' $\mathbf{G}$ is defined by

$$\mathbf{G}(\omega) := \tau g(\omega) \left(\mathbf{I} - g(\omega)\mathbf{W}\right)^{-1}, \quad g(\omega) := \frac{1}{1 + 2\pi i \tau \omega} \tag{32}$$

Series expansion of $\mathbf{G}$, valid at all frequencies since the spectral radius $\rho(\mathbf{W}) < 1$, has a natural interpretation. Equation (31) becomes

$$\tilde{x}(\omega) = \tau g(\omega) \left(\mathbf{I} + g(\omega)\mathbf{W} + g^2(\omega)\mathbf{W}^2 + \ldots\right) \tilde{s}(\omega)$$

The term $\tau g \left[g^n \mathbf{W}^n\right]_{ij}$ gives the effect on cell i of a direct stimulus to cell j, via all paths of length n.

The frequency domain representation of the network dynamics quickly leads to a proof of (30): equation (29) implies

$$\tilde{\mathbf{x}}(\omega) = \mathbf{G}(\omega)\tilde{\mathbf{s}}(\omega), \quad \tilde{\mathbf{a}}(-\omega) = \mathbf{G}^T(\omega)\tilde{\mathbf{b}}(-\omega)$$

so that

$$\tilde{\mathbf{x}}^T(\omega)\tilde{\beta}(-\omega) = \tilde{\mathbf{s}}^T(\omega)\mathbf{G}^T(\omega)\tilde{\beta}(-\omega) = \tilde{\mathbf{s}}^T(\omega)\tilde{\alpha}(-\omega)$$

We now apply Parseval's theorem, remembering that $\alpha, \beta, \mathbf{x}, \mathbf{s}$ are real-valued, so their Fourier transforms are Hermitian functions.

We call α_j the **β-weighted influence of cell** j. We will select β_i to be b_i, that is, the partial derivative of L with respect to activity x_i which we found in the previous section. We define a_i to be the corresponding α_i. This means a_j is a partial derivative of L with respect to stimulus s_j — all assuming that each time the producers change their activities x_i, the consumers learn new optimal input weights $\mathbf{V}$.

Comparing (29) with the equation (25) defining the dynamics of our network, we see that — ignoring run-in time — $\mathbf{a}(-t)$ is what the cell's activity would be if the network had all synapses reversed, and was driven by stimulus $\mathbf{b}(-t)$. A biological neural network cannot do this calculation of $\mathbf{a}$ from $\mathbf{b}$ since it depends on fast retrograde signalling and on playing back a time-reversed signal. However, we can do this in a simulation. In section 6.3.4 we show how to estimate worth from a_i, and in the simulations in section 6.4 we test the estimate of worth.

6.3.3 Related theory: backpropagation through time

Before proceeding to show how knowing a_i lets us estimate worths of cells, we note a relationship between our strategy for computing a_i from b_i and part of the backpropagation through time (BPTT) algorithm (Werbos, 1990).

The aim of BPTT is to compute partial derivatives of a loss function with respect to synaptic weights, and so facilitate supervised learning of the weights. BPTT deals with networks whose state at time $t \in \mathbb{N}$ depends on the state at a finite number of past timesteps, with different weights $W_{ij}^{(n)}$ specified for each time lag n:

$$x_i(t) = f\left(\sum_{n=0}^{n_{\max}} W_{ij}^{(n)} x_j(t-n)\right) \tag{33}$$

The function f is an activation function. For comparison, with infinitesimal dt, our network dynamics are given by

$$\mathbf{x}(t+dt) = \mathbf{x}(t) + dt\left(\frac{1}{\tau}(\mathbf{W}-\mathbf{I})\mathbf{x}(t) + \mathbf{s}(t)\right)$$

and so we could approximate the dynamics of our network in the form (33), with f an identity function.

BPTT takes a loss function $L = \sum_t L(t)$, defined as a function of a subset of network activities x_i. When f is an identity function then — though Werbos does not present it this way — BPTT works, as our scheme does, by taking a copy of the network with each synapse reversed, and stimulating the reversed network with a time-reversed $\frac{\partial L(t)}{\partial x_i(t)}$, as we stimulate the reversed network with b_i. In BPTT the resulting activities are used to calculate the gradient of loss L with respect to each weight; in contrast, we use them to calculate the worths of cells. We also had to work harder to calculate $\frac{\partial L(t)}{\partial x_i(t)}$: our x_i are being offered as inputs to consumers which adapt their weights to make best use of the available x_i, whereas in BPTT the loss function is a function directly of the x_i.

6.3.4 Finding a silencing perturbation

We now return to the task of estimating worth using fast retrograde signals.

In 6.3.1 we showed how to approximate the worth of cell i as its 'linworth', $\$_i^{(\text{lin})} = \sum_k \langle \Delta x_k, b_k \rangle$, plus a locally computable quadratic-order correction $-\frac{\lambda}{2} \sum_j V_{ji}^2$. Here, the Δx_k are the perturbations to all cells' activities — including cell i itself — when cell i is silenced.

In section 6.3.2 we showed how, for general b_i, one can compute the partial derivative a_i of $\sum_k \langle x_k, b_k \rangle$, with respect to the stimulus s_i to a single cell. That is, if the stimulus perturbations Δs_i cause activity perturbations Δx_k, then $\sum_k \langle \Delta x_k, b_k \rangle = \sum_i \langle \Delta s_i, a_i \rangle$.

We now show that an appropriately chosen stimulus perturbation Δs_i has exactly the effect of silencing cell i. For that special 'silencing perturbation', the linworth of cell j is

$$\$_j^{(\text{lin})} = \sum_k \langle \Delta x_k, b_k \rangle = \langle \Delta s_j, a_j \rangle$$

so that the worth of cell j can be written as

$$\$_j \approx \langle \Delta s_j, a_j \rangle - \frac{\lambda}{2} \sum_i V_{ij}^2 \tag{34}$$

How can we compute a silencing perturbation? Is it even guaranteed that a silencing perturbation exists? This is not entirely trivial. In a feedforward network, it is straightforward enough: the negative of the existing input to cell j is a silencing perturbation, and we

denote this ζ_j, that is

$$\zeta_j := -\left(\frac{1}{\tau}\sum_k W_{jk}x_k + s_j\right)$$

However, when there are recurrent connections via cell j, the perturbation ζ_j fails to silence cell j: consider, for example, a single cell with an excitatory autapse, or a pair of cells with mutual excitatory connections and input to one of the two. How, then, can we find a silencing perturbation in a recurrent network? We will show how to find an exact silencing perturbation, and we will argue that ζ_j acts as an approximate silencing perturbation under reasonable conditions of small 'loopiness', a term to be defined later.

We first show that in a recurrent network, there does exist an exact silencing perturbation for each cell. Specifically, cell j can be exactly silenced by the stimulus perturbation

$$\Delta\tilde{s}_j = -\frac{1}{1+\frac{\eta_j}{\tau}}\left(\frac{1}{\tau}\sum_k W_{ji}\tilde{x}_k + \tilde{s}_j\right) \tag{35}$$

where $\eta_j = \eta_j(\omega) := (\mathbf{GW})_{jj}$. We call η_j the (frequency-dependent) 'loopiness', for reasons that will be explained later. In section 6.3.6 we show that in random networks with sensible parameters, loopiness is small ($\eta_j \ll 1$), so that using $\Delta s_j = \zeta_j$ in equation 34 gives a good approximation to the worth of cell j.

To show that the perturbation in equation 35 would silence cell j, first define $\mathbf{W}_{\setminus j}$ as $\mathbf{W}$ with the output weights of cell j set to 0:

$$[\mathbf{W}_{\setminus j}]_{ik} = \begin{cases} W_{ik} & k \neq j \\ 0 & k = j \end{cases}$$

We define $\mathbf{G}_{\setminus j}$ as the corresponding transfer matrix:

$$\mathbf{G}_{\setminus j} = \tau g(\mathbf{I} - g\mathbf{W}_{\setminus j})^{-1}$$

We write $\mathbf{w}$ for the vector of output weights from cell j and $\mathbf{e}$ for the unit vector with $e_j = 1$. The Woodbury matrix identity implies that

$$\begin{aligned}\mathbf{G}_{\setminus j} &= \mathbf{G} - \frac{\frac{1}{\tau}\mathbf{G}\mathbf{w}\mathbf{e}^T}{1+\frac{\eta_j}{\tau}}\mathbf{G} \\ &= \frac{1}{1+\frac{\eta_j}{\tau}}\left(\mathbf{I} + \frac{\eta_j}{\tau} - \frac{1}{\tau}\mathbf{G}\mathbf{w}\mathbf{e}^T\right)\mathbf{G}\end{aligned}$$

Equivalently,

$$(G_{\setminus j})_{ik} = \frac{1}{1+\frac{\eta_j}{\tau}}\left(G_{ik} + \frac{\eta_j}{\tau}G_{ik} - \frac{1}{\tau}(\mathbf{Gw})_i G_{jk}\right) \tag{36}$$

$$= \frac{1}{1+\frac{\eta_j}{\tau}}\left(G_{ik} + \frac{\eta_j}{\tau}G_{ik} - \frac{1}{\tau}(\mathbf{GW})_{ij} G_{jk}\right) \tag{37}$$

and in particular

$$(\mathbf{G}_{\setminus j})_{jk} = \frac{1}{1+\frac{\eta_j}{\tau}}G_{jk}$$

Since $\tilde{\mathbf{x}} = \mathbf{G}\tilde{\mathbf{s}}$, it follows that removing the output synapses of cell j scales the ω frequency component $\tilde{x}_j(\omega)$ of its activity by a factor $\frac{1}{1+\frac{\eta_j}{\tau}}$. This means that

- Cutting the output synapses of a cell cannot silence it if it was not already silent.
- After cutting the output synapses of cell j, we can then silence cell j by adding the perturbation in (35) to its input.
- Therefore, adding the same perturbation Δs_j, even without cutting the outputs of cell j, has the effect of silencing cell j.

This completes the proof that if Δs_j has Fourier transform satisfying

$$\Delta\tilde{s}_j = -\frac{1}{1+\frac{\eta_j}{\tau}}\left(\frac{1}{\tau}\sum_k W_{ji}\tilde{x}_k + \tilde{s}_j\right) \tag{38}$$

then Δs_j is a silencing perturbation for cell j.

In section 6.3.6 we will argue that the loopiness η_j is typically small for sensible choices of parameters. Therefore, ζ_j is a good approximation to a silencing perturbation.

6.3.5 'Backwards net' recipe for estimating worth using fast retrograde messages

Combining the results above, we now have a way to estimate worths in simulated networks, using fast retrograde signals, and without actually silencing any cell. We summarise the recipe as follows:

1. Run the network on a representative set of stimuli to compute activities $\mathbf{x}$, consumer weights $\mathbf{V}$, the negative of the input to each cell ζ_j, and the residual $\mathbf{r}$

2. Calculate the partial derivative of loss with respect to each cell's output, $b_i = -\sum_j V_{ji} r_j$. This b_i allows for the consumers relearning their input weights after the activities x_i have changed.

3. Run a network — the 'backwards net' — with stimulus $\mathbf{b}(-t)$ and weights $\mathbf{W}^T$. Record the activities $\mathbf{a}(-t)$. a_i tells us something about how the input to cell i affects performance, including the knock-on effects cell i has on other producer cells' activities, and the way that consumers can adapt their input weights $\mathbf{V}$ following changes to producer outputs.

4. Use a_i together with the approximate silencing perturbation ζ_j to estimate the worth of cell j as $\$_j \approx \$_j^{(\text{lin})} - \frac{\lambda}{2}\sum_i V_{ij}^2$

The recipe here is only valid if the network is dynamically stable, and can be run for a time much longer than τ, so that initial conditions are not too important. It also requires that the 'loopinesses' η_j be small, so that the silencer ζ_i is a good approximation to the perturbation required to silence cell i. In the next section we check that we can construct networks with reasonable parameters that meet these requirements. Later, in section 6.4, we check this approximation of worth and find that it holds with very good accuracy.

6.3.6 Random weight matrices for which the fast worth estimate should work

Before jumping into simulations of the fast worth estimate derived above, we consider whether the required conditions can be met.

For our fast worth estimate to work, we noted that we require a network that is dynamically stable, and where 'loopiness' is small, that is, the numbers η_j are small. Dynamical stability is not just a mathematical convenience for the current application: it allows a network to respond robustly to noisy inputs. Small loopiness is not so intrinsically desirable — but we show that with sensible parameters, we get it, and that proves to be mathematically convenient.

In our models, each cell type is characterised by a distribution of output weights. Having fixed the distribution for each cell, we draw its output weights independently at random from that distribution. Finally, we make two small adjustments: first we disallow autapses so $\forall i : W_{ii} = 0$, and second for each cell, we add or subtract a small number from every input weight so that $\forall i : \sum_j W_{ij} = 0$. We call the condition $\forall i : \sum_j W_{ij} = 0$ 'strict input balance' ('weak input balance' would mean that the average over randomly sampled $\mathbf{W}$ of $\sum_j W_{ij}$ is zero, but the condition might not hold for individual

instances of $\mathbf{W}$). Strict input balance helps to prevent the network from behaving unstably.

The key parameters controlling stability and loopiness are the mean $\overline{w}_i$ and variance σ_i^2 of the output weights from each cell i. To understand the dependence we start with the result of Rajan and Abbott (2006). They showed that in a network similar to ours, the eigenvalues of $\mathbf{W}$ are restricted to a spectral radius $\rho(\mathbf{W}) \approx \sqrt{\sum_i \sigma_i^2}$, independent of the mean output weights per cell $\overline{w}_i$. Strict input balance is required to enforce this. Their networks differ from ours in two ways: first, they allow autapses, so W_{ii} has the same distribution as W_{ji} and can be nonzero, and second, they use networks with only two cell types — one excitatory and one inhibitory. In our network we allow multiple cell types and we remove autapses: at the stage where they subtract $\frac{1}{p}\sum_k W_{ik}$ from each W_{ij}, instead, we set $W_{ii} = 0$ and then subtract $\frac{1}{p-1}\sum_k W_{ik}$ from each remaining W_{ij}.

Stability. The spectral radius $\rho(\mathbf{W})$, which is the largest absolute value of any eigenvalue of $\mathbf{W}$, controls the dynamical stability of a linear firing-rate network. To see how, consider two copies of a network with different initial conditions $\mathbf{x}(0), \mathbf{x}'(0)$ and the same ongoing input. We have

$$\frac{d}{dt}(\mathbf{x}' - \mathbf{x}) = \frac{1}{\tau}(\mathbf{W} - \mathbf{I})(\mathbf{x}' - \mathbf{x})$$

It follows that if $\mathbf{W}$ has eigenvalues with real part greater than 1, then the magnitude of $\mathbf{x}' - \mathbf{x}$ can grow exponentially — and the magnitude of $\mathbf{x}$ or $\mathbf{x}'$ is also growing exponentially. Conversely, if every eigenvalue of $\mathbf{W}$ has real part less than 1, then the trajectories $\mathbf{x}, \mathbf{x}'$ must converge. The limit on $\rho(\mathbf{W})$ is still relevant in nonlinear networks with a saturating activation function: there, activity tends to become chaotic when synaptic weights are large (Sussillo and Abbott, 2009).

By pruning autapses, we modify the spectral radius of $\mathbf{W}$. We are effectively taking a network like Rajan and Abbott (2006) and adding a diagonal matrix with entries close to $-\overline{w}_i$; we expect therefore to find $\rho(\mathbf{W})$ approximately bounded by $\max_i |\overline{w}_i| + \sqrt{\sum_i \sigma_i^2}$. Simulations in figure 10 match this prediction. Thus, we find in simulations that setting $\sum_i \sigma_i^2 < 1 - \max_i |\overline{w}_i|$ gives a dynamically stable network.

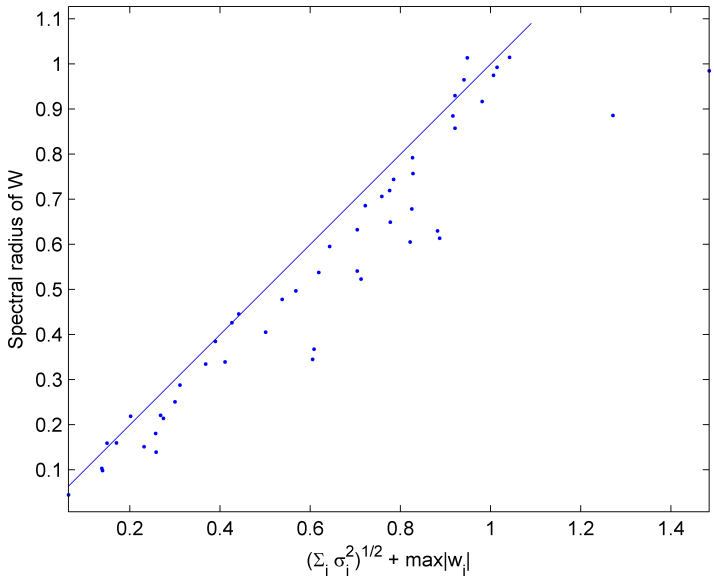


Figure 10: Spectral radius of $\mathbf{W}$ versus the prediction $\sqrt{\sum_i \sigma_i^2} + \max_i |\overline{w}_i|$, for a range of sets of values of σ_i and $\overline{w}_i$. Each of the 100 points in this graph represents an individual random network with different parameters, and we check the spectral radius of each network's weight matrix. Each network has a different value of $\sum_i \sigma_i^2$, and these values are equally spaced between 0 and 1. Each network has a random number of cells between 200 and 1200. For each network we pick a hyperparameter which specifies the standard deviation of the $\overline{w}_i$, drawing this variance from an exponential distribution, and then sample $\overline{w}_1, \ldots, \overline{w}_p$ independently from a Gaussian distribution with the given variance. Next we sample independent exponentially distributed values σ_i and then scale the σ_i uniformly so that $\sum_i \sigma_i^2$ has the right value. Finally we generate a random weight matrix with $W_{ij} - \overline{w}_j$ having Gaussian distribution.

Loopiness. The accuracy of our fast worth estimate depends on the frequency-dependent quantities η_j, which we called 'loopiness'. η_j measures the strength of recurrent connections from cell j to itself, and it dictates how close the perturbation $\Delta s_j = \zeta_j$ comes to silencing cell j.

We call η_j 'loopiness', because it comes from loops starting and ending at cell j. This is clear from the series expansion:

$$\eta_j = (\mathbf{GW})_{jj} = \tau g(\mathbf{W} + g\mathbf{W}^2 + g^2\mathbf{W}^3 + \ldots)_{jj}$$

The term $[\mathbf{W}^n]_{jj}$ is contributed by loops of length n starting and ending at cell j. For a feedforward network, $\eta_j \equiv 0$. Loopiness is small at high frequencies, where the gain g is small: this is to be expected, since in our model, each cell low-pass-filters its input.

How are the parameters $\overline{w}_i, \sigma_i$ related to loopiness? For simplicity, suppose that the values of $|\overline{w}_1|, \ldots, |\overline{w}_p|$ are all of similar magnitude, say $|\overline{w}_i| = O(\overline{w})$ for all cells in a given network, and also that each $\sigma_i = O(\sigma)$. We predict that η_j should grow roughly in proportion with $\frac{\sqrt{\sigma^2+\overline{w}^2}}{1-\sqrt{\sum_i \sigma_i^2}}$. This estimate comes from defining $T_{ij} = W_{ij} - \overline{w}_j$ for $i \neq j$ and $T_{ii} = 0$, and rewriting η_j in terms of $\mathbf{T}$ as far as possible. Because we enforced $\forall i : \sum_j W_{ij} = 0$, we have by induction on n that

$$\mathbf{W}^n = \mathbf{W}\mathbf{T}^{n-1}$$

and hence (also using the fact that $\mathbf{G}$ and $\mathbf{W}$ commute),

$$\mathbf{G}\mathbf{W} = \tau g \mathbf{W} \left(\mathbf{I} - g\mathbf{T}\right)^{-1}$$

The factor $(\mathbf{I} - g\mathbf{T})^{-1}$ is unrelated to $\overline{w}_1, \ldots, \overline{w}_p$ so the magnitude of η_j depends at most linearly on the parameters $\overline{w}_1, \ldots, \overline{w}_j$. For the estimate $\eta_j \sim \frac{\sqrt{\sigma^2+\overline{w}^2}}{1-\sqrt{\sum_i \sigma_i^2}}$, we use the fact the spectral radius of $\mathbf{T}$ is approximately $\sqrt{\sum_i \sigma_i^2}$, so that $(\mathbf{I} - g\mathbf{T})^{-1}$ has spectral radius approximately $\frac{1}{1-\sqrt{\sum_i \sigma_i^2}}$. The entries of $\mathbf{W}$, meanwhile, are of order $\sqrt{\sigma^2 + \overline{w}^2}$. We make the assumption that multiplying a random vector with entries of order $O(w)$ by a random matrix with spectral radius ρ gives a vector with entries of order $O(\rho w)$. Hence, we estimate the entries of $\mathbf{G}\mathbf{W}$ — and in particular $\eta_j = (\mathbf{G}\mathbf{W})_{jj}$ — will be of order $\frac{\sqrt{\sigma^2+\overline{w}^2}}{1-\sqrt{\sum_i \sigma_i^2}}$.

This gives a relationship between loopiness and the parameters $\sigma, \overline{w}$, but what are the relevant values of $\overline{w}, \sigma$? Sensible choices of $\overline{w}$ should be similar to or smaller than σ: otherwise, we get a rather redundant network where every cell has similar input and therefore similar activity. To see this, note the input to cell i from other cells, $\sum_j W_{ij} x_j$, can be written as $\sum_j \overline{w}_j x_j + \sum_j T_{ij} x_j$. The expected square of the first sum — which is identical for every cell — scales with $\sum_j \overline{w}_j^2$ and the expected square of the second scales with $\sum_j \sigma_j^2$, so if $\overline{w} > \sigma$ the first one dominates. Therefore, it is sensible to restrict $\overline{w} = O(\sigma)$. On making that restriction, the estimate for η_j simplifies to $\eta_j = O\left(\frac{\sigma}{1-\sqrt{\sum_i \sigma_i^2}}\right)$. If we have $\sqrt{\sum_i \sigma_i^2} = \rho$, say, then this gives $\eta_i = O\left(\frac{\rho}{\sqrt{p}(1-\rho)}\right)$: the loopiness goes to zero in the limit of a large number of cells.

In Figure 11 we check that the analysis here predicts correctly how η_j should scale with σ or $\overline{w}$, in some simulated networks. As we

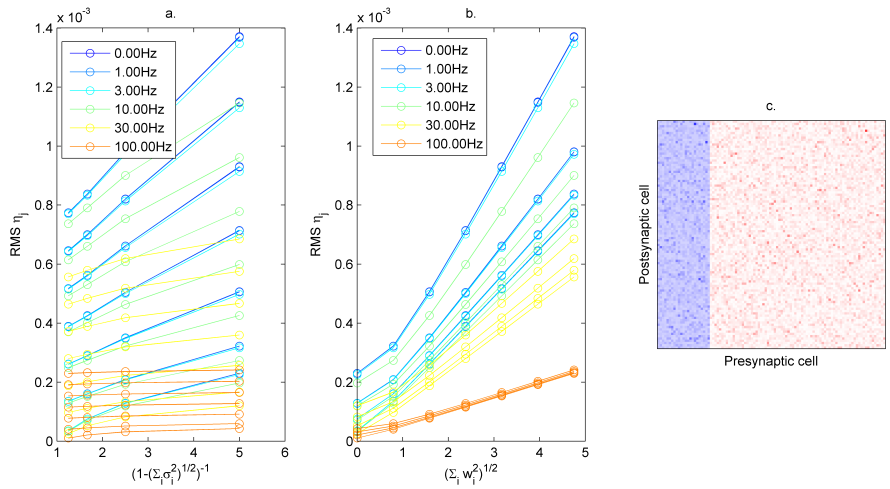


Figure 11: Root mean square over cells of the 'loopiness' η_j for a range of values of $\overline{w}_j, \sigma_j, \omega$. a. Dependence on σ. The multiple lines for each frequency correspond to different scalings of $\overline{w}_i$ in b.. b. Dependence on $\overline{w}$. The multiple lines for each frequency correspond to the different values of σ_j in a.. c. Part of the weight matrix. This is a dense network of $p = 1000$ cells of which 20% are 'inhibitory' (with negative output weights, blue) and the rest are 'excitatory' (positive output weights, pink and red). The absolute values of each cell's output weights are exponentially distributed with an offset.

expect, when $\overline{w}$ is large then η_j scales approximately linearly with $\overline{w}$ and when $\sum \sigma_i^2$ comes close to 1 then η_j scales approximately linearly with $\frac{1}{1-\sqrt{\sum_i \sigma_i^2}}$. Also, since η contains a factor of g, it is smaller for higher frequency components. These simulations show that loopiness can still be small in networks at least as strongly connected as those we simulate in section 6.4.

In section 6.4 we will simulate networks with $\rho \approx 0.8$ and $p = 1000$ so that η_i is of order 0.1 for DC signals, and smaller for higher frequency components. If loopiness is too large then our worth estimate is expected to break down because ζ_j is no longer a good approximation to a silencing perturbation for cell j. Therefore, as an extra check in those simulations, we test for each cell i in a random sample, that the approximate silencing perturbation ζ_i really does almost silence cell i.

6.4 Simulations: estimating worth with fast retrograde signals

We have derived a way of estimating worth using fast retrograde signals. In this section we use simulations of three different types

of network to check how well this method works, and to test the approximations that were made in deriving it.

Construction of the networks. The networks we study are all of the architecture shown in Figure 9, but with only a single consumer. We study three kinds of network, each with $p = 1000$ producers, and with differently constructed weight matrices:

1. A network of 80% excitatory and 20% inhibitory cells. Each $\overline{w}_i$ takes one of two values, $\overline{w}_{exc} = 0.0065 > 0$ (for excitatory cells) and $\overline{w}_{inh} = -4\overline{w}_{exc}$. Excitatory cells have sparse output weights: 10% of their output weights are nonzero and the weights are exponentially distributed with parameter $10\overline{w}_{exc}$ when non-zero. Inhibitory cells have uniform negative output weight to every other cell. This gives $\sqrt{\sum_i \overline{w}_i^2} \approx \frac{1}{2}\sqrt{\sum_i \sigma_i^2}$ which fits with the criterion of 6.3.6, that to avoid cells being highly synchronised, we should have $\sum_i \overline{w}_i^2$ of order $\sum_i \sigma_i^2$ or smaller.

2. A densely connected network of cells, where some have much stronger output synapses than others. The output weights of each individual cell are exponentially distributed, with an offset added so that the mean output weight of each cell is 0. The variance of each cell's output weights is itself drawn from an exponential distribution with mean $\frac{0.8^2}{p}$, so that $\sqrt{\sum_i \sigma_i^2} = 0.8$.

3. A sparsely connected network, with 5% connection density. Nonzero weights are all chosen from the same exponential distribution with variance $\frac{0.8^2 \times 20}{p}$ and an offset $-\sqrt{\frac{0.8^2 \times 20}{p}}$ added so that $\forall i : \overline{w}_i = 0$.

In each case the standard deviations σ_i of weights from individual cells satisfy $\sum_i \sigma_i^2 = 0.8^2$ so that the spectral radius of the weight matrix is approximately 0.8. This is a compromise between allowing a network to have reasonably long-timescale responses to transient input, and avoiding instability which can arise if the spectral radius is greater than 1, as we mentioned in section 6.3.6.

Why these particular random networks? The first network, with 20% inhibitory cells that connect promiscuously and 80% excitatory cells with sparse output weights, is meant to represent a cortical network (Wolf et al., 2014). The second network is not meant to represent a biological network, but a mathematical extreme: in this network, we deliberately make some cells far more influential than others, and therefore we expect worths to be highly diverse.

In such a network, we hypothesise that coarse estimates of worth could be particularly informative, as even a large error in estimated worth could be small compared to the population variation in worth. The third network is an alternative representation of a cortical network. Inhibitory interneurons tend to have faster time constant than principal cells (Jonas et al., 2004). The third network represents a network of sparsely interconnected excitatory cells, which all connect bidirectionally to a single inhibitory interneuron whose time constant is much faster than that of the excitatory cells, and whose activity can therefore be integrated out to produce an equivalent network where each cell has zero mean output weight.

Figure 12 shows pseudocolour plots of the three networks' recurrent weight matrices. We will later use the same three networks to test other strategies that do not employ fast retrograde signals.

Stimulus, activity, and target. To create a stimulus, we generate three independent random signals each with the same power spectrum. The stimulus to each cell is a random combination of these signals, and the target for the consumer is another combination of the same signals. Since cells low-pass-filter their inputs, no cell exactly repeats the signal it receives and the target signal cannot be represented exactly as a linear function of network activities.

Figure 13 shows the activities of a few cells, their stimuli, and the target for the consumer. The activities look qualitatively similar in all three networks, so only one network (the excitatory/inhibitory one) is shown. However, Figure 14 shows that having $\overline{w}_i \neq 0$ for some cells makes a difference to the correlation structure of the network activity, as we predicted at the end of (6.3.6): in the excitatory/inhibitory network there is positive correlation on average between the activities of different cells. In section 6.3.6 we said that we would expect the correlating part of cells' activities to become the dominant component when $\sum_i \overline{w}_i^2 > \sum_i \sigma_i^2$. Here, the excitatory/inhibitory network has $\sum_i \overline{w}_i^2 \approx \frac{1}{4} \sum_i \sigma_i^2$, so cells' activities are not too synchronised.

Silencing cells with a stimulus perturbation. Our strategey for computing the worths of cells using fast retrograde signals depends on finding a 'silencing perturbation' to the stimulus of each cell. We estimated that the stimulus perturbation

$$\Delta s_j = \zeta_j = -\left(\frac{1}{\tau} \sum_k W_{jk} x_k + s_j\right)$$

would suffice to approximately silence a cell j, provided that the loopiness η_j is small at each frequency. In Figure 15 we check that adding the silencing perturbation ζ_j to one cell's stimulus s_j, really does almost silence that cell. We use the same stimulus to run the network forwards in time, recording $x_i(t)$. Then for each one in turn of 50 randomly chosen cells i, we re-run each network replacing $s_i(t)$ with $s_i(t)+\zeta_i(t)$. Figure 15 shows that the magnitude of output from a cell is reduced by a factor of 100 or more by this treatment (right hand panel). For reference, we also show that the magnitude of output is far less drastically changed by simply removing the stimulus, i.e., setting $s_i(t) \equiv 0$: that indicates that each cell gets a substantial proportion of its input from other cells and not as direct stimulus.

Estimating worth with fast retrograde signals. Having satisfied ourselves that ζ_i would roughly silence cell i, which was a key part of the derivation of the recipe in section 6.3.5, we now check that the recipe works, i.e. we can estimate worth using fast retrograde signals. In Figure 16 we show that the estimate computed using the recipe of section 6.3.5 is accurate. We estimate the worth of every cell using that recipe, and then for each of 50 randomly chosen cells i, we check the estimate. To check the estimate for a given cell i, we re-run the simulation setting $\forall j : W_{ij} = W_{ji} = 0$; this silences cell i. We record the activites $x_j(t)$ in the absence of cell i, recompute optimal consumer weights $\mathbf{V}$ given these new $x_j(t)$, and measure the new loss. The increase in loss relative to the baseline when no cells are silenced is, by definition, the worth of cell i. Figure 16 shows a plot of the actual worth versus the estimate. Our strategy with fast retrograde signals provides an extremely accurate estimate of worth, whether the cell in question is hidden or visible. The estimate is much quicker to compute than the exact measurement made by excising one cell at a time: the estimate is computed simultaneously for every cell in the network, in the same time it takes to measure the exact worth of one single cell. Since this estimate of worth is accurate and fast to compute, we will use it later to check the performance of other more realistic schemes, that do not use fast retrograde signals.

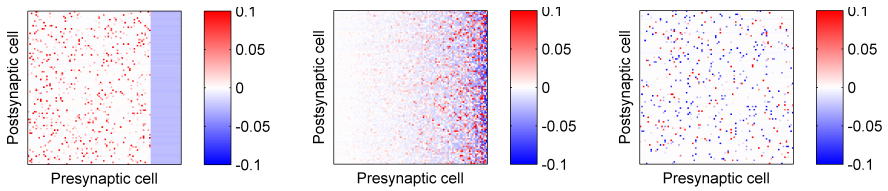


Figure 12: Weight matrices for three different networks. Pairwise weights between a random subset of 100 neurons are shown. Left: a network with excitatory and inhibitory cells. Excitatory cells have sparse weights which are exponentially distributed when non-zero, excitatory connection density 0.1. Inhibitory cells have output connection density 1 and uniform negative output weight to every other cell. Middle: densely connected network with a gradation from 'strong' to 'weak' cells, strong cells having larger output weights. The output weights of each cell are exponentially distributed, with an offset added so that the mean output weight of each cell is 0. Right: sparse network, with connection density 0.05. Nonzero weights are chosen from the same exponential distribution for every cell and an offset added so that the mean output weight is 0.

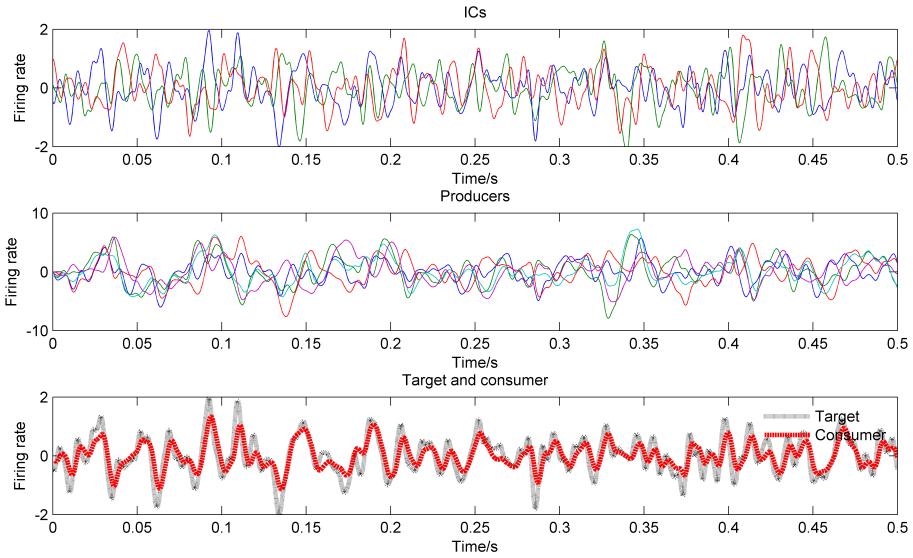


Figure 13: Stimulus, activities and target in one of our simulated networks. The stimulus to each cell is a sum of the three ICs shown in the top panel. A few producer activities are shown in the middle panel. The bottom panel shows the target for the consumer and the actual activity of the consumer when it has optimal input weights. This is for the excitatory/inhibitory network. In the other two networks, activities of individual cells look similar but the mean correlation between pairs of cells is smaller, as shown in 14.

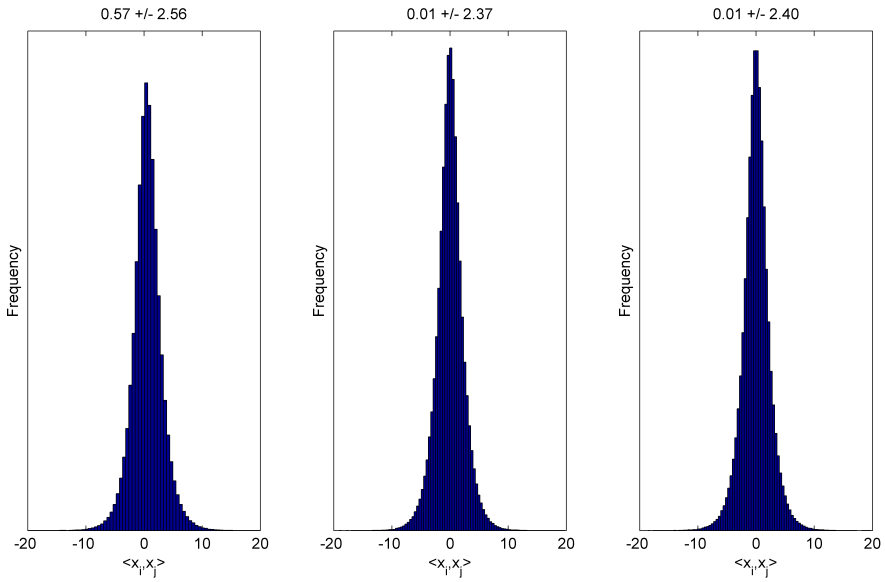


Figure 14: Histograms of correlations between activities of different cells for the three networks shown in figure 12. Titles give the mean and standard deviation of $\langle x_i, x_j \rangle$ for all pairs of cells i, j. As predicted the non-balanced network (left), with excitatory and inhibitory cells, has a positive correlation on average between activities of pairs of cells.

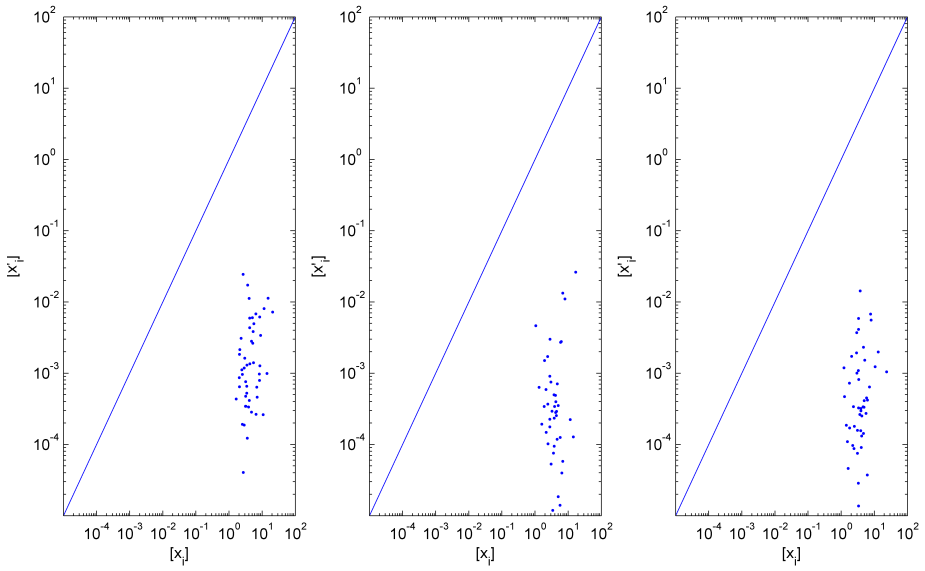


Figure 15: Adding the silencing perturbation ζ_i almost silences cell i, or at least reduces its mean square firing rate by orders of magnitude. The three plots correspond to the three networks in Figure 12 and section 6.4. In each plot there is one dot for each of a random sample of 50 cells. On the horizontal axis, $[\![x_i]\!]$ is root mean square over time of the firing rate of cell i. On the vertical axis, $[\![x'_i]\!]$ is the root mean square of the firing rate of the same cell after replacing s_i with $s_i + \zeta_i$ where ζ_i is the silencing perturbation computed in section 6.3, then re-running the network with all other stimuli unchanged. x'_i is typically several orders of magnitude smaller than x_i. This works because our networks have small 'loopiness'.

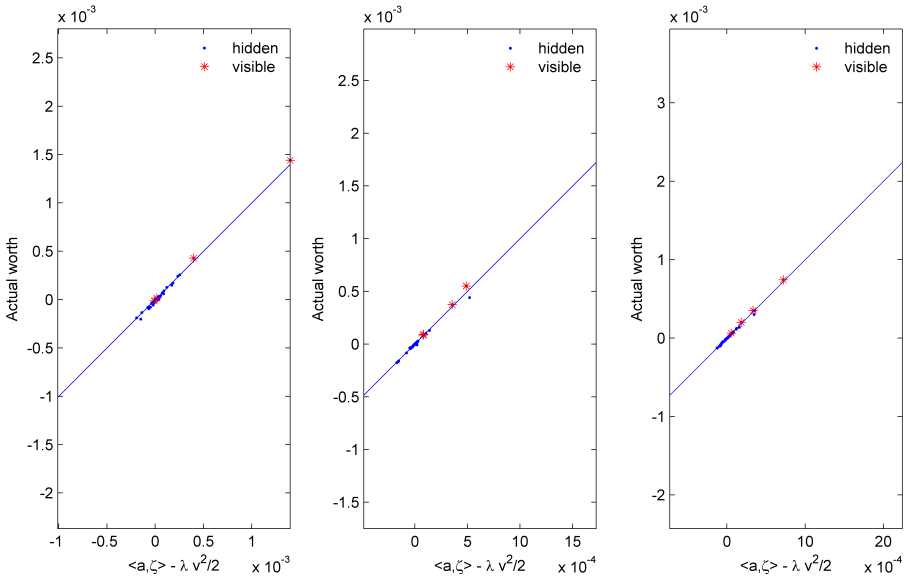


Figure 16: Checking the estimate of worth using 'backwards network' recipe of section 6.3 for the three networks of figure 12. The horizontal axis is the estimate using that recipe, while the vertical axis is the actual worth of each cell measured by removing the cell from the network and reevaluating the network performance. Each network has $p = 1000$ cells; a random selection of 50 cells are shown here.

6.5 Estimating worth with slow retrograde signals

6.5.1 Introduction

We have shown that using fast retrograde signals, we can accurately estimate the worths of cells in a recurrently connected, random network. To what extent is it possible to estimate worths without fast retrograde signalling? We consider two cases — either using intracellular retrograde signalling only (so a cell can 'see' its output weights but nothing else about its downstream neighbours), or using a recursive scheme in which ech cell gains information not only about the strength of its output synapses, but also about the worths of its downstream neighbours.

Intuitively it seems like estimating worth using slow retrograde signals could be possible, even in recurrent networks. A cell with no output weights, or which never fires, has zero worth, so we can infer something about worth from a cell's output weights and firing rate alone. Recursive message-passing also seems promising: if your downstream neighbours have high worth, perhaps that is because you supplied them with useful information, and you can deduce that you yourself also have high worth. We might try to propagate estimates of worth using slow, recursive, retrograde signals, with each cell reporting its worth to its upstream neighbours.

The aim of this section is to see how these intuitive ideas convert to mathematical claims, and test them. The strategy will be to assume that cell j has some information, and work out what it can then infer about its own worth. We consider two cases:

Case 1: Cell j knows information only about itself, $\Theta_j^{(\text{cell})}$:

- its own output synaptic weights, to other producers and to consumers
- its mean square firing rate $[\![x_j]\!]^2$

Case 2: Cell j knows $\Theta_j^{(\text{cell})}$ and also knows some things about its downstream neighbours

- an estimate of the linworth of each downstream neighbour
- for each downstream neighbour i, the quantity

$$F_{ij} = \frac{\frac{1}{\tau^2} W_{ij}^2 [\![x_j]\!]^2}{[\![\zeta_i]\!]^2}$$

which measures what proportion of the input to cell i comes from cell j.

We write $\Theta_j^{(\mathrm{rec})}$ for the information available to cell j in Case 2, which includes $\Theta_j^{(\mathrm{cell})}$.

For case 1 and case 2, we will try to characterise the 'belief' of a cell about its worth, given the information it gets. We will estimate the mean and variance of the conditional distribution of one cell's worth, over random weight matrices $\mathbf{W}$, given $\Theta_j^{(\mathrm{rec})}$ or $\Theta_j^{(\mathrm{cell})}$.

We can make some basic claims without doing any algebra:

- If cell j provides the only input to cell i, then all the worth of cell i is attributable to cell j. We expect to see a clear relationship between $\$_j$ on $\$_i$.
- Conversely if a single downstream cell i has a large number of similarly weighted inputs, then there is no clear way to assign different credit to each input (figure 17 shows a cartoon of such a network). We do not expect $\$_i$ to provide much information about $\$_j$.

It is less obvious what to expect when there are many-to-many or recurrent connections. With many-to-many connections, each individual output gives little information about the worth of cell j, but adding up the estimated contributions from each output may give something with small percentage error if these errors are uncorrelated.

In this section we estimate the mean and variance of worth given the retrogradely transmitted information $\Theta_j^{(\mathrm{rec})}$ or $\Theta_j^{(\mathrm{cell})}$, up to some constants of proportionality. The calculation uses constants of proportionality which we have to find in simulations. The results indicate that the variance and the expected value of *indirect* worth are linearly related. Therefore, the coefficient of variation (CV) of indirect worth scales with the inverse square root of the expected indirect worth. For *total* worth — including both direct and indirect worth — the variance and expected value again appear, in simulations, to be linearly related. Cells of low expected worth have a high coefficient of variation, making it impossible to determine the sign of their actual worth. For cells with high expected worth, the expected worth is a (relatively) more accurate estimate of their actual worth.

In sections 6.5.4 and 6.5.6 we use simulations to check the form of these estimates, and to find the constants of proportionality. The networks simulated are the same as in 6.4. We find that for the types of network simulated, these constants of proportionality take values which make for a large coefficient of variation (CV) in the

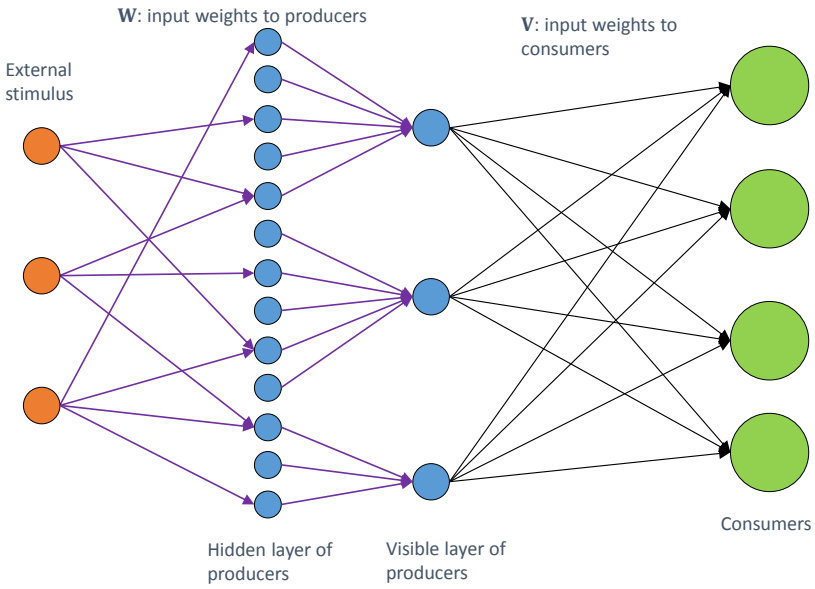


Figure 17: A network in which many hidden cells provide input to each visible producer. If the W_{ij} all have similar magnitude, there is no way to use worths of visible cells and W_{ij} to assign different credit to each hidden cell

conditional belief about worth given Θ_j or $\Theta_j^{(\text{cell})}$, except for cells of high expected worth. Thus, while a high expected worth indicates that a cell probably has positive worth, a low expected worth does not indicate whether the true worth is positive or negative.

We also find that when every cell has a large number of input synapses, uncertainty about the worth of cell j is similar given Θ_j or $\Theta_j^{(\text{cell})}$: knowing the linworths of its downstream neighbours does not help a cell much in estimating its own worth. This is analytically predicted and then confirmed in the simulations.

6.5.2 Direct and indirect worth

We now show how to decompose the worth of a cell into 'direct worth' (from direct connections to consumers) and 'indirect worth' (via connections to other producers). Making this decomposition will help us to understand what information about the worth of cell j can be inferred from $\Theta_j^{(\text{rec})}$ or $\Theta_j^{(\text{cell})}$.

First, note that we can write the linworth $\$_j^{(\text{lin})}$ in terms of a sum over downstream cells. To do this we use Parseval's theorem and

$\tilde{\zeta}_j = \frac{1}{\tau g}\tilde{x}_j$:

$$\begin{aligned}
\$_j^{(\text{lin})} &= \int \tilde{\zeta}_j \tilde{a}_j d\omega = \int \tilde{\zeta}_j \left(\sum_i g W_{ij} \tilde{a}_i + \tau g \tilde{b}_j \right) d\omega \\
&= -\int \frac{1}{\tau g} \tilde{x}_j \left(\sum_i g W_{ij} \tilde{a}_i + \tau g \tilde{b}_j \right) d\omega \\
&= -\frac{1}{\tau} \left\langle x_j, \sum_i W_{ij} a_i \right\rangle - \langle x_j, b_j \rangle \qquad (39) \\
&= -\sum_i \left\langle \frac{1}{\tau} W_{ij} x_j, a_i \right\rangle + \lambda \sum_i V_{ij}^2 \qquad (40)
\end{aligned}$$

Recall a_i was a partial derivative of loss with respect to the input to cell i, and $\frac{1}{\tau} W_{ij} x_j$ is the input cell i gets from cell j. Since $\$_j \approx \$_j^{(\text{lin})} - \frac{\lambda}{2} \sum_k V_{kj}^2$ it follows that

$$\$_j \approx \sum_i \aleph_{ij} + \frac{\lambda}{2} \sum_k V_{kj}^2 \qquad (41)$$

where we define $\aleph_{ij}$ to be the 'worth of cell j via cell i':

$$\aleph_{ij} := -\left\langle \frac{1}{\tau} W_{ij} x_j, a_i \right\rangle$$

In the decomposition (41),

$$\$_j^{(\text{indirect})} := \sum_i \aleph_{ij}$$

is considered to be the indirect worth of cell j, via other producers, and

$$\$_j^{(\text{direct})} := \frac{\lambda}{2} \sum_k V_{kj}^2$$

is the direct worth, measuring direct utility to consumers. From chapter 5 we know that this direct worth $\$_j^{(\text{direct})}$ approximates the change in performance that would result if we hid cell j from the consumers, leaving all other producers' outputs unchanged.

6.5.3 Estimating worth from cellular information

We asked what cell j can infer about its worth from its own root mean square activity and output weights, $\Theta_j^{(\text{cell})}$. As explained previously in 6.5.1, our approach is to consider the belief of a cell j

about its worth given this information, meaning the conditional distribution of its worth over random $\mathbf{W}$ given $\Theta_j^{(\text{cell})}$. In symbols, we are trying to estimate

$$\mathbb{E}_{\mathbf{W}}\left(\$_i^{(\text{lin})}|\Theta_j^{(\text{cell})}\right)$$

where $|\Theta_j^{(\text{cell})})$ denotes conditionality on $\Theta_j^{(\text{cell})}$ and $\mathbb{E}_{\mathbf{W}}(..)$ denotes expectation over random $\mathbf{W}$.

We also estimate

$$\text{Var}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right)$$

which will become useful in the next section.

These estimates are hard to make analytically, so our approach is to derive estimates via a heuristic argument, and then use simulations to test what we deduce. The key approximations are that we treat each $\aleph_{ij}$ as having Gaussian distribution given $\Theta_j^{(\text{cell})}$, and we assume that each cell has a large number of inputs, which are approximately pairwise uncorrelated.

Recall that the cellular information $\Theta_j^{(\text{cell})}$ includes $[\![x_j]\!]$ and the output weights of cell j, both to other producers and to consumers. The direct worth is easy to compute from the output weights to consumers V_{ij} as $\$_j^{(\text{direct})} = \frac{\lambda}{2}\sum_i V_{ij}^2$; the indirect worth is more difficult. What about indirect worth? Although $\Theta_j^{(\text{cell})}$ does not include any explicit information about the indirect worth of cell j, the two are not statistically independent. For example, if cell j sends no output to producer i then it can have no indirect worth via i:

$$W_{ij}\,[\![x_j]\!] = 0 \Rightarrow \aleph_{ij} = 0$$

If cell i has no activity, it has no worth, and indeed no direct worth, indirect worth, or linworth:

$$[\![x_i]\!] = 0 \Rightarrow \$_i = \$_i^{(\text{lin})} = \$_i^{(\text{indirect})} = \$_i^{(\text{direct})} = 0$$

We will argue that given our assumptions and approximations, more generally, the expected indirect worth of cell j given cellular information $\Theta_j^{(\text{cell})}$ is proportional to $\sum_i [\![W_{ij}x_j]\!]^2$:

$$\mathbb{E}_{\mathbf{W}}\left(\$_j^{(\text{indirect})}|\Theta_j^{(\text{cell})}\right) \propto \sum_i [\![W_{ij}x_j]\!]^2$$

We also argue that the variance of the worth via a given downstream neighbour, $\aleph_{ij}$, scales with the mean square output to that neighbour: $\text{Var}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right) \propto [\![W_{ij}x_j]\!]^2$. The justification for these

is as follows. We approximate the conditional distribution of each $\aleph_{ij}$ given $\Theta_j^{(\text{cell})}$ as Gaussian. The cellular information $\Theta_j^{(\text{cell})}$ fixes $[\![W_{ij}x_j]\!]$, and we assume that $[\![W_{ij}x_j]\!]$ is all that is relevant to $\aleph_{ij}$ in $\Theta_j^{(\text{cell})}$.

Conditional expectation of worth via a neighbour given cellular information. For the conditional expectation of $\aleph_{ij}$ we note that since $\aleph_{ij} = \left\langle a_i, -\frac{1}{\tau}W_{ij}x_j \right\rangle$, we have

$$\mathbb{E}\left(\aleph_{ij}|W_{ij}x_j\right) = \left\langle -\frac{1}{\tau}W_{ij}x_j, \mathbb{E}\left(a_i|W_{ij}x_j\right) \right\rangle$$

Recall that in the frequency domain,

$$\tilde{a}_i = g\sum_j W_{ji}\tilde{a}_j + \tau g\tilde{b}_i$$

Since $b_i = \frac{1}{\lambda}\sum_j \langle x_i, r_j\rangle r_j$, where r_j is the residual error for consumer j, the signal b_i for each cell depends approximately linearly on the activity of the same cell, x_i. Therefore, a_i has a component that depends linearly on the activity of cell i, and therefore also linearly on the input to cell i. Hence, we estimate $\mathbb{E}_{\mathbf{W}}\left(a_i|W_{ij}x_j\right) \propto W_{ij}x_j$ and deduce

$$\mathbb{E}_{\mathbf{W}}\left(\aleph_{ij}|W_{ij}x_j\right) \propto [\![W_{ij}x_j]\!]^2$$

This means we expect there is some constant of proportionality k_m — dependent on the distribution from which $\mathbf{W}$ is drawn — for which we can approximate

$$\mathbb{E}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right) = \mathbb{E}_{\mathbf{W}}\left(\aleph_{ij}|\,[\![W_{ij}x_j]\!]\right) \approx \frac{k_m}{\tau^2}[\![W_{ij}x_j]\!]^2 \tag{42}$$

Conditional variance of worth via a neighbour given cellular information. For the conditional variance of $\aleph_{ij}$, we note that in signal space, $\aleph_{ij} := -\left\langle \frac{1}{\tau}W_{ij}x_j, a_i \right\rangle$ takes the projection of a_i in the direction of x_j and scales it by $W_{ij}[\![x_j]\!]$. Therefore, we expect to find approximately that

$$\mathrm{Var}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right) = \mathrm{Var}_{\mathbf{W}}\left(\aleph_{ij}|W_{ij}[\![x_j]\!]\right) \propto [\![W_{ij}x_j]\!]^2$$

That is, there is some other constant of proportionality k_σ with

$$\mathrm{Var}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right) \approx \frac{k_\sigma^2}{\tau^2}[\![W_{ij}x_j]\!]^2 \tag{43}$$

Conditional coefficient of variation (CV) of worth via a neighbour given cellular information. One prediction from (42) and (43) is that the conditional CV of $\aleph_{ij}$ given $\Theta_j^{(\text{cell})}$ is large when $W_{ij}x_j$ is small:

$$\frac{\text{Std}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right)}{\mathbb{E}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right)} \approx \frac{k_\sigma \tau}{k_m \left[\!\left[W_{ij}x_j\right]\!\right]}$$

If we use the expected value $\frac{k_m}{\tau^2}\left[\!\left[W_{ij}x_j\right]\!\right]^2$ as an estimate of $\aleph_{ij}$, then the percentage error should get smaller as $\left[\!\left[W_{ij}x_j\right]\!\right]$ increases.

Estimating the constants of proportionality. Analytically estimating the constants of proportionality k_m, k_σ is hard: they are expected magnitudes of correlations between random signals, and we do not know what the distribution of these signals is like. Instead, we estimate k_m, k_σ in simulations:

- We estimate k_m by least squares linear regression of $\frac{\aleph_{ij}}{\frac{1}{\tau}\left[\!\left[W_{ij}x_j\right]\!\right]}$ against $\frac{1}{\tau}\left[\!\left[W_{ij}x_j\right]\!\right]$. The division by $\frac{1}{\tau}\left[\!\left[W_{ij}x_j\right]\!\right]$ is to transform the regression problem to a homoscedastic one. The choice of least squares regression implicitly uses the assumption that $\aleph_{ij}|\Theta_j^{(\text{cell})}$ is Gaussian.

- We take k_σ to be the standard deviation of $\frac{\aleph_{ij} - \frac{k_m}{\tau^2}\left[\!\left[W_{ij}x_j\right]\!\right]^2}{\frac{1}{\tau}\left[\!\left[W_{ij}x_j\right]\!\right]}$. k_σ will not be used here, but in the next section 6.5.5.

Does this let neurons estimate their worth? The answer depends on k_m, k_σ. If these are stable parameters, then conceivably they could be known and used by each cell to approximate a belief about its worth given the information $\Theta_j^{(\text{cell})}$. Then, each cell could estimate its indirect worth using the expected value

$$\mathbb{E}_{\mathbf{W}}\left(\$_j^{(\text{indirect})}|\Theta_j^{(\text{cell})}\right) \approx k_m \left[\!\left[x_j\right]\!\right]^2 \sum_i \frac{1}{\tau^2} W_{ij}^2 \tag{44}$$

and then estimate its worth as

$$\mathbb{E}_{\mathbf{W}}\left(\$_j|\Theta_j^{(\text{cell})}\right) \approx k_m \left[\!\left[x_j\right]\!\right]^2 \sum_i \frac{1}{\tau^2} W_{ij}^2 + \frac{\lambda}{2}\sum_i V_{ij}^2$$

However, if k_σ is not sufficiently small compared to k_m, then the difference between this expected value and the actual worth is expected to be large. These are cases where the coefficient of variation

in $\$_j$ given $\Theta_j^{(\text{cell})}$ is large, meaning that a cell receiving information $\Theta_j^{(\text{cell})}$ still has a very uncertain belief about its worth.

In sections 6.5.4 and 6.5.6 we calculate values of k_m, k_σ in a variety of simulated networks and compute the CV of indirect worth given $\Theta_j^{(\text{cell})}$. For the examples simulated, we find that the two constants of proportionality k_m, k_σ have similar order of magnitude, and $[\![x_i]\!]$ is of order 1, with the effect that the conditional CV of $\$_j$ given $\Theta_j^{(\text{cell})}$ is of order $\frac{1}{|W_{ij}|}$, i.e. $\sqrt{p}$. This means that $\mathbb{E}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right)$ would be a very inaccurate estimate of $\aleph_{ij}$, with an error of order $\sqrt{p}$ times the estimate. The coefficient of variation of $\$_j^{(\text{indirect})}$ given $\Theta_j^{(\text{cell})}$ is found to be $O(1)$ in the same examples (smaller, since errors in the individual $\aleph_{ij}$ cancel out). This means that if the expression in (44) were used as an estimate of $\$_j^{(\text{indirect})}$, the errors would be of the same order of magnitude as the estimate itself.

6.5.4 Simulations: estimating worth from cellular information

In section 6.5.3 we discussed what a cell can infer about its worth using only the cellular information $\Theta_j^{(\text{cell})}$, namely its root mean square activity $[\![x_j]\!]$ and its output weights $W_{1j}, \ldots, W_{pj}$. We now test the predictions and estimates made in section 6.5.3, by simulating multiple instances of three different types of random recurrent network. The network types are the same ones introduced in the simulations of section 6.4.

Section 6.5.3 made use of the decomposition of worth into direct worth plus indirect worth via each downstream neighbour, which we derived in section 6.5.2:

$$\$_i \approx \frac{\lambda}{2}\sum_i V_{ij}^2 + \sum_j \aleph_{ij}$$

where

$$\aleph_{ij} = \left\langle \frac{1}{\tau} W_{ij} x_j, a_i \right\rangle$$

measures the indirect worth of cell j via cell j. We studied the conditional distribution of $\aleph_{ij}$ given the cellular information $\Theta_j^{(\text{cell})}$, over random weight matrices $\mathbf{W}$: this conditional distribution is the belief of cell j about its worth via each neighbour if it gets information $\Theta_j^{(\text{cell})}$. We predicted that the conditional mean of indirect worth given $\Theta_j^{(\text{cell})}$ would be linear in $\sum_i [\![x_j]\!]^2 W_{ij}^2$. We also predicted that each constituent $\aleph_{ij}$ of that indirect worth would have

conditional variance linear in the corresponding $[\![x_j]\!]^2 W_{ij}^2$, so that, barring strong correlation effects between the different $\aleph_{ij}$ for a given cell j, the conditional variance of indirect worth given $\Theta_j^{(\text{cell})}$ would be, like the conditional expectation, linear in $\sum_i [\![x_j]\!] W_{ij}^2$.

Obtaining 'true' values of $\$_j^{(\text{lin})}$, etc. We used the fast retrograde messaging scheme to obtain accurate estimates of $\aleph_{ij}, \$_j, \$_j^{(\text{lin})}$, etc. in each network. We know from simulations in section 6.4 that this gives accurate estimates, and it does so without incurring the enormous computational cost of actually silencing each individual cell and measuring the effect on performance.

Estimating k_m. We used the simulations to calculate the constant of proportionality k_m defined in (42). To estimate k_m we use the linear coefficient from least squares regression of $\frac{\aleph_{ij}}{[\![\frac{1}{\tau} W_{ij} x_j]\!]}$ against $[\![\frac{1}{\tau} W_{ij} x_j]\!]$, dividing $\aleph_{ij}$ and the predictor $[\![\frac{1}{\tau} W_{ij} x_j]\!]^2$ by $[\![\frac{1}{\tau} W_{ij} x_j]\!]$ to make the regression problem homescedastic. The values of k_m we obtained are given in Table 1 (page 116).

Expected worth and indirect worth given cellular information. We computed values for the conditional expected worth and indirect worth given cellular information $\Theta_j^{(\text{cell})}$, using

$$\mathbb{E}_{\mathbf{W}} \left(\$_j^{(\text{indirect})} | \Theta_j^{(\text{cell})} \right) \approx \frac{k_m}{\tau^2} [\![x_j]\!]^2 \sum_i W_{ij}^2$$

and

$$\mathbb{E}_{\mathbf{W}} \left(\$_j | \Theta_j^{(\text{cell})} \right) \approx \frac{k_m}{\tau^2} [\![x_j]\!]^2 \sum_i W_{ij}^2 + \frac{\lambda}{2} \sum_i V_{ij}^2$$

Results: worth. Figure 18 summarises the relationship between $\mathbb{E}_{\mathbf{W}} \left(\$_j | \Theta_j^{(\text{cell})} \right)$ computed this way, and the actual values of $\$_j$. Data are aggregated over 10 instances of each type of network. We sort the values of expected worth into bins with approximately 5% of cells in each bin, and for each bin we calculate a conditional mean, standard error on the mean, and variance, of the corresponding true values of worth. The top row verifies our estimated expected value for worth given $\Theta_j^{(\text{cell})}$. The bottom row shows that the conditional variance of worth given $\Theta_j^{(\text{cell})}$ is approximately linearly related to the expected value.

Results: indirect worth. Figure 19 shows a similar analysis to 18, but for *indirect* worth, that is, excluding direct worth. This figure shows that the fit in Figure 18 is not purely down to good estimation of direct worth from output weights to consumers: the expected indirect worth, calculated from output weights to other producers, is also meaningful. We predicted that the conditional variance of each component $\aleph_{ij}$ (the worth of j via i) would be linear in $W_{ij}\left[\!\left[x_j\right]\!\right]^2$ so that barring strong correlations between the $\aleph_{ij}$ given $\Theta_j^{(\text{cell})}$, we could also expect a linear relationship between the conditional variance of indirect worth and $\sum_i \left[\!\left[W_{ij}x_j\right]\!\right]^2$. The bottom row of figure 19 matches this prediction.

Results: worth and indirect worth for a single instance of each network. Figure 20 shows data for a single instance of each type of network. This shows that within a single network, $\Theta_j^{(\text{cell})}$ helps to predict which cells are likely to have high worth and which cells are likely to have low worth. Table 2 shows Spearman rank correlation coefficients between expected and actual worth of cells in individual networks, and supports the same conclusion: the correlation coefficients range from 0.22 to 0.32 for the three types of network simulated, and the rank correlations are highly significant for all three networks.

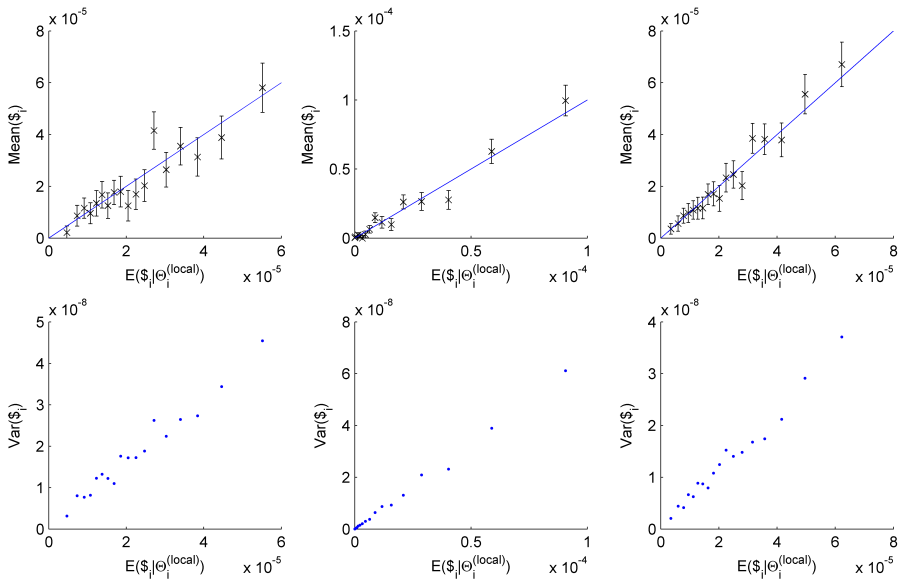


Figure 18: Conditional distribution of worth given cellular information $\Theta_j^{(\text{cell})}$. The three columns are for the three types of network in figure 12. Data are aggregated over 10 instances of each type of network. For each cell in each network we have an actual worth and the expected value of worth given cellular information. We sort the expected values into bins with approximately 5% of data points in each bin, and then calculate the actual mean worth, standard error on the mean, and standard devaition of worth for each of those bins. **Top.** Mean of actual worth for each bin. The error bars are standard error on the mean and blue lines are equality. There is a clear correspondence between our predicted mean worth and measured mean worth in each bin. **Bottom.** Variance of actual worth in each bin versus expected value of worth given cellular information. The variance rises linearly with the expected value. This matches our prediction that the conditional coefficient of variation in worth is smallest when the expected value of worth is large.

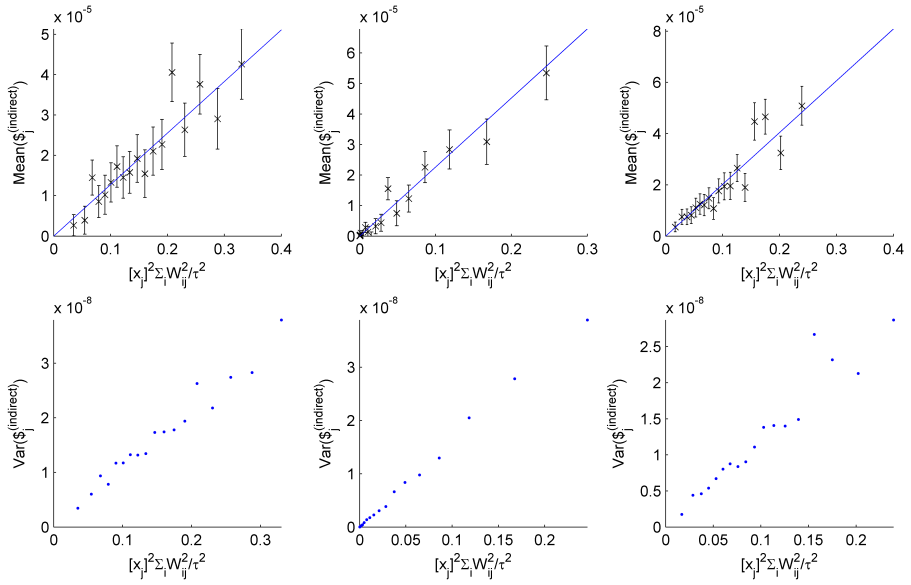


Figure 19: Conditional distribution of indirect worth given cellular information $\Theta_j^{(\text{cell})}$, and specifically as a function of the total output from a cell, $\sum_i W_{ij}^2 [\![x_j]\!]^2$. The three columns are for the three types of network in figure 12. Data are aggregated over 10 instances of each type of network. For each cell in each network we have an actual indirect worth and the value $\sum_i W_{ij}^2 [\![x_j]\!]^2$; we predict that both the conditional mean and conditional variance of indirect worth should be linear in $\sum_i W_{ij}^2 [\![x_j]\!]^2$. Values of $\sum_i W_{ij}^2 [\![x_j]\!]^2$ are sorted into bins and summary statistics computed for each bin as in figure 18. **Top.** Mean indirect worth is linearly related to $\sum_i W_{ij}^2 [\![x_j]\!]^2$. **Bottom.** Variance of indirect worth is linearly related to $\sum_i W_{ij}^2 [\![x_j]\!]^2$, so the conditional coefficient of variation in indirect worth is smallest when the expected value of indirect worth is large.

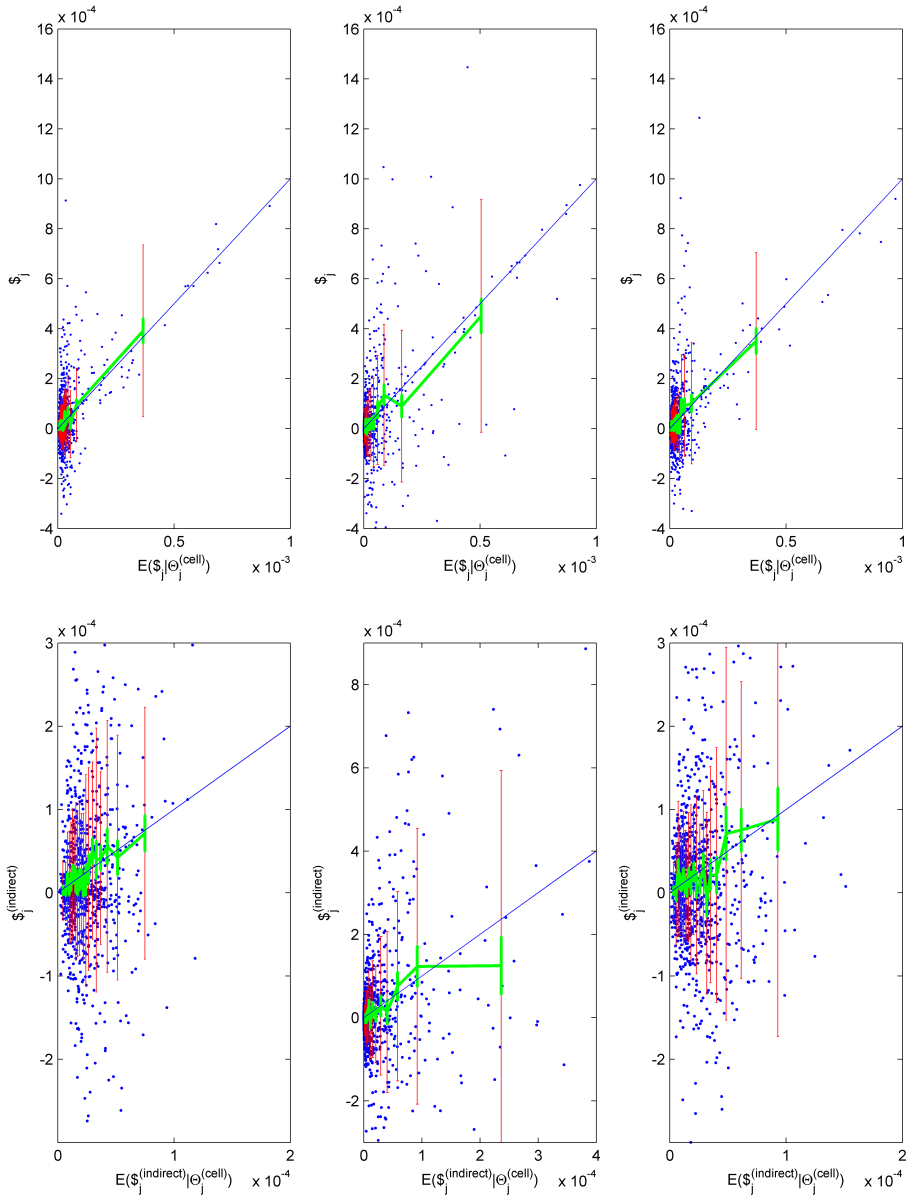


Figure 20: Expected worth and indirect worth, given cellular information $\Theta_j^{(\text{cell})}$, in individual networks. The expected value is compared to the actual value for each cell. Each scatter plot has one point per cell for a single instance of each of the three random networks in figure 12. **Top.** Worth $\$_j$ in the three networks, versus the expected value $\mathbb{E}_{\mathbf{W}}\left(\$_j|\Theta_j^{(\text{cell})}\right)$. For the overlaid lines, expected values are sorted into bins, with approximately 5% of cells in each bin, and statistics of the corresponding values of true worth $\$_j$ computed in each bin. The lines show the mean (green) and standard deviation (red error bars) of the actual worth in each bin, as well as the standard error on the mean (green error bars). The expected values are approximately correct but the spread of true values around the expected values is large. **Bottom.** Same, but showing indirect worth instead of worth.

Results: worth via a single neighbour. Finally in Figure 21, we check our prediction that the conditional expected value of $\aleph_{ij}$ and its variance should be linear in $[\![W_{ij}x_j]\!]^2$. We note that the conditional coefficient of variation of worth via a single neighbour, $\aleph_{ij}$, given the amount of output sent to that neighbour, $[\![x_j]\!]^2 W_{ij}^2$, is much greater than that of indirect worth given $\sum_i [\![x_j W_{ij}]\!]^2$. The conditional CV of $\aleph_{ij}$ given $\Theta_j^{(\text{cell})}$ is of order 100 in each bin. This means that if we used $\mathbb{E}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j^{(\text{cell})}\right)$ as an estimate of $\aleph_{ij}$ we could expect to get errors a hundred times the size of the estimate itself. The smaller conditional CV for indirect worth given $\Theta_j^{(\text{cell})}$ depends on the fact it aggregates data from many downstream neighbours.

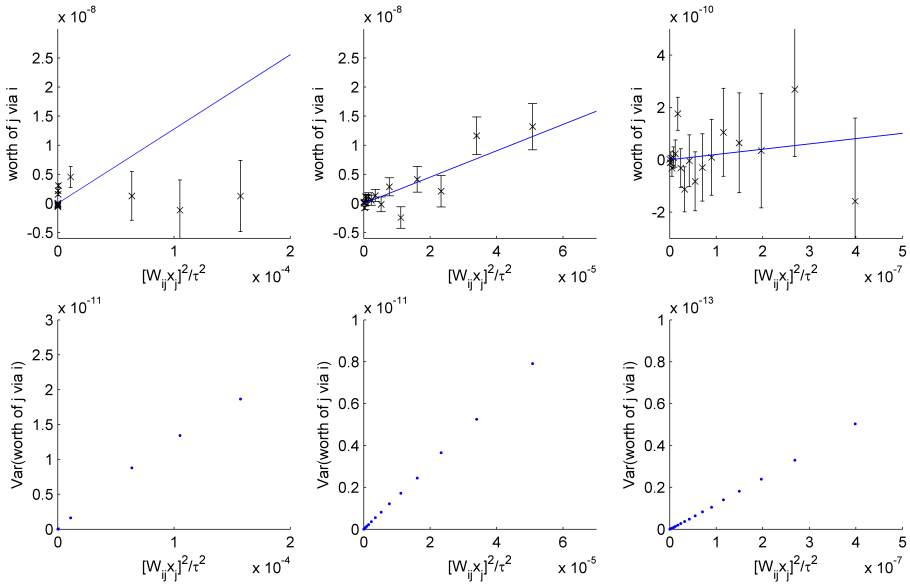


Figure 21: Worth via each individual neighbour, versus the magnitude of output to that neighbour. Values of $[\![W_{ij}x_j]\!]^2$ are binned and statistics computed for each bin as in figures 18 and 19. **Top.** Conditional mean of via each neighbour $\aleph_{ij}$ given the amount of output to that neighbour $[\![W_{ij}x_j]\!]$. Error bars show standard error on the mean for each bin. The straight lines are the prediction $\frac{k_m}{\tau^2}[\![W_{ij}x_j]\!]^2$. As discussed, the CV of worth via a neighbour given $\Theta_j^{(\text{cell})}$ is large, so that these lines are a poor fit, except for the middle network which was constructed specially to have cells of very diverse influence. **Bottom.** Conditional variance of $\aleph_{ij}$ given $\Theta_j^{(\text{cell})}$ scales linearly with $[\![W_{ij}x_j]\!]^2$.

6.5.5 Estimating worth with slow recursive retrograde signals

In section 6.5.3 we estimated the expected value and conditional variance of a cell's worth $\$_j$ given cellular information $\Theta_j^{(\text{cell})}$. We derived what the form of the expected value and variance should be as a function of $\Theta_j^{(\text{cell})}$ but left a parameter k_m to be measured in simulations. In section 6.5.4 we checked this estimate and measured k_m for three types of random recurrent network. We found that the conditional coefficient of variation in worth given $\Theta_j^{(\text{cell})}$ can be large, meaning that a cell in one of these networks cannot use cellular information alone to accurately and reliably estimate its worth.

We now ask whether cells can obtain much more precise information about their worth by passing *recursive* slow retrograde messages. These could be supported by chemical retroaxonal messages in the brain (see section 3.3). Motivated by the idea that a cell whose downstream neighbours have high worths, probably has high

worth itself, we consider a message-passing scheme where each cell receives an estimate of each downstream neighbour's linworth. If one cell can accurately deduce its linworth $\$_j^{(\text{lin})}$ given its neighbours' linworths, then this scheme could let neurons propagate estimates of worth using slow recursive retrograde message-passing.

We will write $\Theta_j^{(\text{rec})}$ for the augmented information cell j receives in this scheme. $\Theta_j^{(\text{rec})}$ includes $\Theta_j^{(\text{cell})}$, plus, for each downstream neighbour i:

- an estimate Ω_i of the linworth of cell i, which will be defined recursively as $\mathbb{E}_{\mathbf{W}}\left(\$_i^{(\text{lin})}|\Theta_i^{(\text{rec})}\right)$
- the quantity $F_{ij} = \frac{\frac{1}{\tau^2}W_{ij}^2[\![x_j]\!]^2}{[\![\zeta_i]\!]^2}$ for each downstream neighbour i.

Note F_{ij} measures the proportion of input to cell i that comes from cell j, since $-\zeta_i = \left(\frac{1}{\tau}\sum_k W_{jk}x_k + s_j\right)$ is the total input to cell i. Computing F_{ij} does not introduce a requirement for fast retrograde signalling: $[\![\zeta_i]\!]^2$ and $[\![W_{ij}x_j]\!]^2$ are time-averaged, local quantities.

From $\Theta_j^{(\text{rec})}$, cell j will compute an estimate Ω_j of its own linworth, namely $\Omega_j = \mathbb{E}_{\mathbf{W}}\left(\$_j^{(\text{lin})}|\Theta_j^{(\text{rec})}\right)$, and pass this on to upstream cells. We can implement this recursive estimate by initialising each Ω_j at $\mathbb{E}_{\mathbf{W}}\left(\$_j^{(\text{lin})}|\Theta_j^{(\text{cell})}\right)$ (as estimated in section 6.5.3) and then repeatedly updating Ω_j using the current value Ω_i from each downstream neighbour i. We will see that for the examples studied, this procedure converges to a unique set of estimates Ω.

Deductions using exact values of neighbours' linworths. To work out $\mathbb{E}_{\mathbf{W}}\left(\$_j^{(\text{lin})}|\Theta_j^{(\text{rec})}\right)$ we start by considering what is known about $\$_j^{(\text{lin})}$ when each downstream neighbour's linworth is known exactly. We write Θ_j for $\Theta_j^{(\text{rec})}$ with each downstream neighbour's Ω_i replaced by an exact measurement of $\$_i^{(\text{lin})}$. Our philosophy, as before, is to consider the belief cell j about its worth given Θ_j i.e. the conditional distribution of $\$_j^{(\text{lin})}$ over random weight matrices $\mathbf{W}$, given Θ_j. To estimate the mean and variance of this conditional distribution, we start with the belief of cell j given $\Theta_j^{(\text{cell})}$ and then update that belief in the light of the downstream neighbours' linworths and the F_{ij}. We will see that knowing Θ_j does not reduce uncertainty much compared to having only cellular information $\Theta_j^{(\text{cell})}$. Therefore, recursive message-passing of this type does not tell cells much more

about their worth than they could deduce from cellular information alone.

To update the belief about $\$_j^{(\text{lin})}$ given $\Theta_j^{(\text{cell})}$ to a belief given Θ_j, we first show that one can write each downstream neighbour's linworth $\$_i^{(\text{lin})}$ as $\aleph_{ij}$ (the worth of j via i), plus some other terms. Hence, $\$_i^{(\text{lin})}$ can be used as a noisy measurement of $\aleph_{ij}$. To do this we use $-\zeta_i = \frac{1}{\tau}\sum_j W_{ij}x_j + s_i$ and take the sum outside the inner product in $\$_i^{(\text{lin})} = \langle \zeta_i, a_i \rangle$:

$$\begin{aligned}
\$_i^{(\text{lin})} := \langle \zeta_i, a_i \rangle &= -\left\langle \frac{1}{\tau}\sum_j W_{ij}x_j + s_i, a_i \right\rangle \\
&= -\frac{1}{\tau}\sum_j W_{ij}\langle x_j, a_i \rangle - \langle s_i, a_i \rangle \\
&= \sum_j \aleph_{ij} - \langle s_i, a_i \rangle
\end{aligned}$$

We regard $\$_i^{(\text{lin})}$ as providing a noisy measurement of $\aleph_{ij}$, and we assume that all the dependence between $\$_j$ and $\$_i^{(\text{lin})}$ is via $\aleph_{ij}$. The 'error' $\$_i^{(\text{lin})} - \aleph_{ij}$ is the correlation with a_i of all the input cell i gets from sources other than cell j. If cell i gets all of its input from cell j, then $\$_i^{(\text{lin})} = \aleph_{ij}$: all the worth of cell i is due to cell j, matching the common-sense prediction we made at the beginning of 6.3, without doing any algebra.

How can we use these noisy measurements to update beliefs about worth via each neighbour? Let X, Y be random variables whose distributions are the conditional distributions of $\aleph_{ij}, \$_i^{(\text{lin})} - \aleph_{ij}$ given $\Theta_j^{(\text{cell})}$. The joint distribution of $X, X+Y$ expresses belief of cell j about $\aleph_{ij}, \$_i^{(\text{lin})}$ given cellular information $\Theta_j^{(\text{cell})}$. We now want to update the belief about $\aleph_{ij} = X$ using a measurement of $\$_i^{(\text{lin})} = X+Y$. In section 6.5.3 we approximated the conditional distribution of $\aleph_{ij}$ given $\Theta_j^{(\text{cell})}$ as Gaussian. We now also approximate X, Y as being independent and Gaussian, so we can use the following standard formulae:

$$\begin{aligned}
\text{Var}\,(X|X+Y) &= \left(\frac{1}{\text{Var}\,(X)} + \frac{1}{\text{Var}\,(Y)}\right)^{-1} \\
&= \frac{\text{Var}\,(X)\,\text{Var}\,(Y)}{\text{Var}\,(X+Y)}
\end{aligned}$$

and

$$\mathbb{E}\left(X - \mathbb{E}\left(X\right)|X + Y = z + \mathbb{E}\left(X + Y\right)\right) = \frac{z\mathrm{Var}\left(X\right)}{\mathrm{Var}\left(X + Y\right)}$$

For our X and Y, we have

$$\mathbb{E}\left(\aleph_{ij}|\Theta_j\right) = \mathbb{E}\left(X|X + Y\right)$$

and

$$\mathrm{Var}\left(\aleph_{ij}|\Theta_j\right) = \mathrm{Var}\left(X|X + Y\right)$$

To use the above we need an estimate of $\frac{\mathrm{Var}(X)}{\mathrm{Var}(X+Y)}$. We know $X, X + Y$ are the correlations of two different signals $-\frac{1}{\tau}W_{ij}x_j, \zeta_i$ with the same a_i. Therefore, we estimate $\frac{\mathrm{Var}(X)}{\mathrm{Var}(X+Y)}$ to be the ratio of squared magnitudes of $\frac{1}{\tau}W_{ij}x_j$ and ζ_i, which we called F_{ij}:

$$F_{ij} := \frac{\frac{1}{\tau^2}W_{ij}^2\left[\!\left[x_j\right]\!\right]^2}{\left[\!\left[\zeta_i\right]\!\right]^2} \approx \frac{\mathrm{Var}\left(X\right)}{\mathrm{Var}\left(X + Y\right)}$$

We can now write

$$\mathrm{Var}_{\mathbf{W}}\left(\aleph_{ij}|\Theta_j\right) \approx (1 - F_{ij})\mathrm{Var}\left(X\right) \tag{45}$$

$$\mathbb{E}_{\mathbf{W}}\left(\aleph_{ij} - \frac{k_m}{\tau^2}\left[\!\left[W_{ij}x_j\right]\!\right]^2|\Theta_j\right) \approx F_{ij}\left(\$_i^{(\mathrm{lin})} - m\right) \tag{46}$$

where $m = \mathbb{E}\left(X + Y\right)$ is the average linworth of a cell, and $\frac{k_m}{\tau^2}\left[\!\left[W_{ij}x_j\right]\!\right]^2$ is, as in section 6.5.3, the expected value of $\aleph_{ij}$ given $\Theta_j^{(\mathrm{cell})}$. As with k_m, the parameter m will be estimated by simulating a number of random networks with weights drawn from the same distribution.

Summing (46) over downstream neighbours i for a single cell j gives

$$\mathbb{E}\left(\$_j^{(\mathrm{lin})}|\Theta_j\right) \approx \sum_i F_{ij}(\$_i^{(\mathrm{lin})} - m) + \lambda\sum_k V_{kj}^2 + k_m\left[\!\left[x_j\right]\!\right]^2\sum_i\frac{W_{ij}^2}{\tau^2} \tag{47}$$

Recursive estimate. The estimated expected value on the left of (47) is linear in each downstream linworth on the right, so (47) converts directly to a rule for propagating values of *expected* linworth. If each downstream neighbour i discloses its estimated expected linworth Ω_i, then cell j's expected linworth is approximated by

$$\Omega_j = \sum_i F_{ij}(\Omega_i - m) + \lambda\sum_i V_{ij}^2 + k_m\left[\!\left[x_j\right]\!\right]^2\sum_i\frac{W_{ij}^2}{\tau^2} \tag{48}$$

Provided $\mathbf{F}$ is a contraction, we can arrive at a unique solution of (48) by repeatedly assigning

$$\Omega_j \leftarrow \sum_i F_{ij}(\Omega_i - m) + \lambda \sum_i V_{ij}^2 + k_m \left[\!\left[x_j\right]\!\right]^2 \sum_i \frac{W_{ij}^2}{\tau^2} \tag{49}$$

We can efficiently initialise each Ω_j at $\mathbb{E}_{\mathbf{W}}\left(\$_j^{(\text{lin})} | \Theta_j^{(\text{cell})}\right)$, and the repeated update will converge on the unique solution

$$\begin{pmatrix} \Omega_1 \\ \vdots \\ \Omega_p \end{pmatrix} = (\mathbf{I} - \mathbf{F}^T)^{-1} \times \left(-\mathbf{F}^T \begin{pmatrix} m \\ \vdots \\ m \end{pmatrix} + \lambda \begin{pmatrix} \sum_i V_{i1}^2 \\ \vdots \\ \sum_i V_{ip}^2 \end{pmatrix} + \frac{k_m}{\tau^2} \begin{pmatrix} \left[\!\left[x_1\right]\!\right]^2 \sum_i W_{i1}^2 \\ \vdots \\ \left[\!\left[x_p\right]\!\right]^2 \sum_i W_{ip}^2 \end{pmatrix} \right) \tag{50}$$

Remaining uncertainty about linworth given recursive retrograde messages. When each cell gets a large number of inputs, Ω_j is not much more precise as an estimate of linworth than the expected value of linworth given cellular information alone, $\mathbb{E}_{\mathbf{W}}\left(\$_j^{(\text{lin})} | \Theta_j^{(\text{cell})}\right)$. This conclusion comes from (45) and (46). There, the coefficients F_{ij} measure what proportion of the input to cell i comes from cell j. If each cell receives $p' \gg 1$ inputs all of similar magnitude, then F_{ij} is approximately $\frac{1}{p'} \ll 1$. In that case,

- (45) says that

  $$\text{Var}_{\mathbf{W}}\left(\aleph_{ij} | \Theta_j^{(\text{cell})}\right) \approx \text{Var}_{\mathbf{W}}\left(\aleph_{ij} | \Theta_j\right)$$

  i.e., knowing the linworth of its downstream neighbour i does not greatly reduce uncertainty of cell j about $\aleph_{ij}$

- (46) says that the difference between $m_{ij} = \mathbb{E}\left(\aleph_{ij} | \Theta_j^{(\text{cell})}\right)$ and $\mathbb{E}\left(\aleph_{ij} | \Theta_j\right)$ is small.

Thus (46) and (45) quantify the statement that if cell i gets most of its input from sources other than j, then the worth of cell i does not tell us much about the worth of cell j.

In conclusion, we have a way to propagate estimates of expected linworth using recursive retrograde message-passing, with each cell

calculating an expected linworth from downstream neigbour's expected linworths. However, this may not give much advantage over using only the cellular information $\Theta_j^{(\text{cell})}$. Comparing the simulations in sections 6.5.4 and 6.5.6 confirms this conclusion.

6.5.6 Simulations: estimating worth with slow recursive retrograde signals

In this section we check the predictions of section 6.5.5.

In section 6.5.5 we derived a scheme using slow recursive retrograde messages to estimate an expected linworth for each cell in a network, using the expected linworth of each downstream neighbour. We built up the recursive message-passing scheme by taking the belief of cell j about its worth given cellular information $\Theta_j^{(\text{cell})}$ — derived in section 6.5.3 and validated in simulations in section 6.5.4 — and then updating the belief using estimates of downstream neighbours' linworths. However, we predicted that in the networks we study, where each cell gets a large number of inputs, this belief is similar to the belief given $\Theta^{(\text{cell})}$ only, and we predicted that the residual uncertainty about $\$_j^{(\text{lin})}$ given the recursive retrograde messages is similar to the residual uncertainty given only cellular information $\Theta_j^{(\text{cell})}$.

Estimating m, the mean linworth of a cell. As well as the parameter k_m derived and used in 6.5.3 and 6.5.4 we also need to estimate the parameter m, which is the mean linworth of a cell. To estimate m we simulate ten instances of each type of network, use the fast retrograde scheme to accurately estimate the linworth of every cell, and take the mean m over all cells and all instances for each network type. The parameters found are in table 1.

Network type	Mean linworth m	k_m
1	6.9×10^{-5}	1.3×10^{-4}
2	6.1×10^{-5}	2.3×10^{-4}
3	7.2×10^{-5}	2.0×10^{-4}

Table 1: Values of k_m, m found for the three networks of figure 12

Expected linworth using recursive retrograde messages. We use the values found for k_m, m, together with equation 50, to compute an expected value Ω_j for the linworth of each cell j. This Ω_j is the expected value of the linworth of cell j given slow recursive retrograde signals that communicate each cell's expected linworth to

its upstream neighbours. $\Omega_j - \$_j^{(\text{direct})}, \Omega_j - 2\$_j^{(\text{direct})}$ approximate the expected value of a cell's worth and indirect worth respectively, given the same information. We predicted that these would give correct mean values for $\$_j, \$_j^{(\text{indirect})}$, but that variance of $\$_j, \$_j^{(\text{indirect})}$ given the recursive messages would be similar to the variance of $\$_j, \$_j^{(\text{indirect})}$ given cellular information only.

Results: worth. Figure 22 summarises how the expected worth given recursive retrograde messages, $\Omega_j - \frac{\lambda}{2}\$_j^{(\text{direct})}$, relates to actual values of worth. Data are aggregated over 10 instances of each type of network. We sort values of $\Omega_j - \$_j^{(\text{direct})}$ into bins with approximately 5% of data in each bin, and compute statistics of the corresponding actual values of $\$_j$ for each bin: a mean, variance, and standard error on the mean. The mean values of $\$_j$ are shown to approximately match the prediction $\Omega_j - \$_j^{(\text{direct})}$. However, as we also predicted, the variances in $\$_j$ given $\Omega_j - \$_j^{(\text{direct})}$ are large (as they were given $\Theta_j^{(\text{cell})}$). This means that like $\Theta_j^{(\text{cell})}$, $\Omega_j - \$_j^{(\text{direct})}$ has limited value as an estimate of $\$_j$ when the estimated worth is small. Since both mean and variance of indirect worth are linear in $\Omega_j - \$_j^{(\text{direct})}$, the CV goes down as the expected value goes up. Practically this means that if a cell believes its worth to be small, it has a large percentage uncertainty in that estimate, and has little confidence in even correctly estimating the sign of its worth. For larger values of expected worth the percentage error is smaller, but even for the largest values of expected worth in these simulations, the CV remains above 1.

Results: indirect worth. Figure 23 is like Figure 22 but shows *indirect* worth only. Again, the mean values $\Omega_j - 2\$_j^{(\text{direct})}$ are shown to be approximately correct, but the remaining uncertainty in indirect worth given the recursive retrograde messages is large. The conditional variance in indirect worth scales approximately linearly with the expected value.

Results: worth and indirect worth in individual networks Figure 24 shows results for individual instances of each type of network. The top row compares actual worth to expected worth given slow recursive retrograde signals, and the bottom row shows the same analysis for indirect worth only. The plots show one dot per cell. Expected values are sorted into bins, and for the actual values in each bin, we compute a mean (green line), standard deviation (red

error bars), an standard error on the mean (green error bars). This shows that the results in Figures 22 and 23 are relevant to individual networks, and not just statistics aggregated over multiple networks. Even within a single network, slow recursive retrograde signals give some information as to which cells are likely to have high worth and which cells are not.

Results: correlation between expected and actual worth. Table 2 compares expected worth given recursive slow retrograde messages, $\Omega_j - \$_j^{(\text{direct})}$, with expected worth given only cellular information, $\mathbb{E}_{\mathbf{W}}\left(\$_j|\Theta_j^{(\text{cell})}\right)$, as predictors of actual worth. The table shows correlation coefficients of actual worth with the two expected values. Since we predicted the CV of worth given its expected value will be large for cells of small expected worth, we also separately compute the correlation coefficients for cells of low expected worth and cells of high expected worth. As expected, the correlation coefficients are larger for cells of high expected worth. Table 3 compares the Spearman correlation coefficient between expected and actual worth for the two different estimation methods — using cellular information alone, and using recursive slow retrograde messages. For cells of low expected worth the recursive message-passing scheme does not do significantly better than the cellular one, in any of the three types of network. In network type 3 the recursive scheme has statistically significant advantage over the cellular one, and it seems this advantage is all to do with cells of higher expected worth.

		Network 1	Network 2	Network 3
Using $\Theta_j^{(\text{cell})}$	all	0.25 ($p = 1.86 \times 10^{-15}$)	0.32 ($p < 10^{-307}$)	0.22 ($p = 4.02 \times 10^{-12}$)
	high	0.36 ($p = 1.00 \times 10^{-16}$)	0.36 ($p = 4.32 \times 10^{-17}$)	0.28 ($p = 1.80 \times 10^{-10}$)
	low	0.07 ($p = 1.04 \times 10^{-01}$)	0.16 ($p = 4.19 \times 10^{-04}$)	0.04 ($p = 3.44 \times 10^{-01}$)
Using $\Theta_j^{(\text{rec})}$	all	0.25 ($p = 4.53 \times 10^{-15}$)	0.32 ($p < 10^{-307}$)	0.27 ($p = 1.75 \times 10^{-17}$)
	high	0.43 ($p < 10^{-307}$)	0.38 ($p < 10^{-307}$)	0.37 ($p < 10^{-307}$)
	low	-0.01 ($p = 8.54 \times 10^{-01}$)	0.14 ($p = 1.79 \times 10^{-03}$)	0.04 ($p = 3.19 \times 10^{-01}$)

Table 2: Spearman rank correlation coefficient of actual worth versus the expected value given cellular information (top) or the expected value given recursive slow retrograde messages (bottom), which is $\Omega_j - \$_j^{(\text{direct})}$. Correlation coefficients and p-values are computed for a single instance of each type of network, treating each cell's data as an independent sample. 'All' includes all the cells in the network, 'high' means the cells with 50% highest expected worths, and 'low' means the cells with 50% lowest expected worths. The rank correlation appears to be mostly due to cells of high expected worth, consistent with the fact that the conditional CV of worth given expected worth is smaller for cells with large expected worth.

	Network 1	Network 2	Network 3
all	$0.121(0.197, 0.227)$	$0.678(0.295, 0.299)$	$0.045(0.243, 0.279)$
high	$0.140(0.280, 0.330)$	$0.473(0.371, 0.380)$	$0.000(0.304, 0.380)$
low	$0.140(0.024, -0.008)$	$0.791(0.027, 0.022)$	$0.186(0.027, -0.003)$

Table 3: Rank-sum test of the null hypothesis that recursive messages, and cellular information alone, give the same median Spearman correlation coefficient between expected and actual worth. The three columns correspond to the three kinds of network in Figure 12. This analysis uses 10 instances of each type of network. Each table entry gives a p value, and in brackets, the mean Spearman correlation coefficient between expected and actual worth, first given cellular information, then given recursive slow retrograde signals. 'All' includes all cells; 'low' includes only cells whose expected worth is below the median expected worth of cells of in the same network, and 'high' includes only cells whose expected worth is above the median.

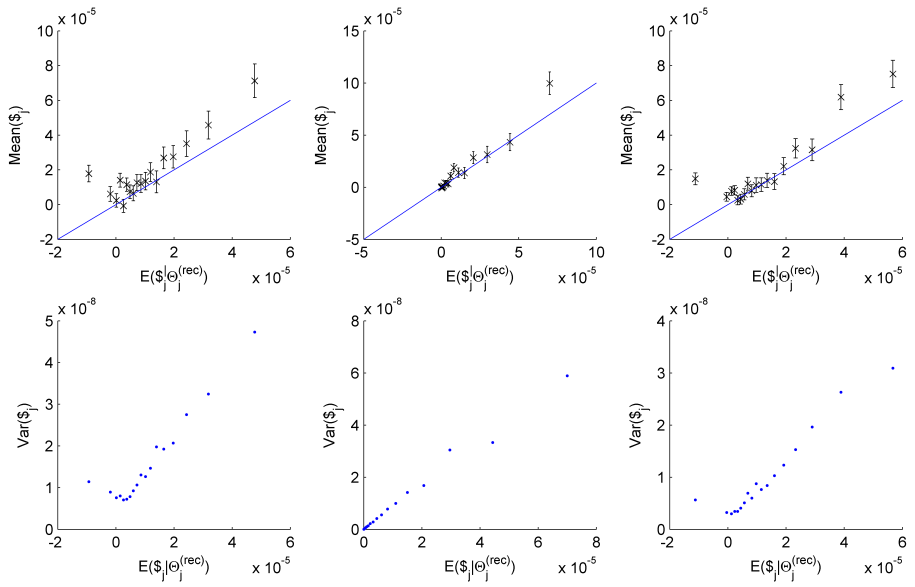


Figure 22: Conditional mean and variance of worth given recursive retrograde messages. The three columns correspond to the three networks in Figure 12. Data are aggregated over 10 instances of each network type. Values of expected worth are sorted into bins with approximately 5% of data in each bin, and statistics computed for corresponding actual values of worth in each bin. **Top** Actual mean (crosses) and standard error on the mean (error bars) for each bin. The blue line is equality. This shows the expected value to be approximately correct. **Bottom** Variance in actual worth for each bin. The variance is largest when the expected worth is large. For positive values of expected worth, variance is approximately linearly related to the expected worth.

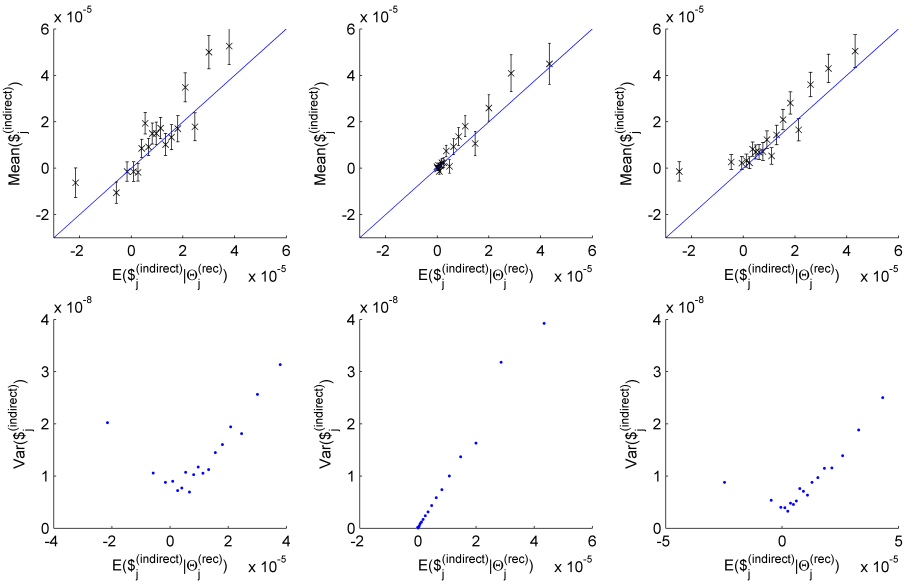


Figure 23: Conditional mean and variance of indirect worth given recursive retrograde messages. Data are aggregated over ten instances of each type of network. The three columns correspond to the networks in Figure 12. We sort the expected values into bins with approximately 5% of data in each bin, and compute statistics for the actual indirect worth in each bin. **Top** Actual mean indirect worth for each bin, versus the expected value $\Omega_j - 2\$_j^{(\text{indirect})}$, computed using slow recursive retrograde messages. The error bars are standard error on the mean for each bin and the blue line is equality. The expected value is shown to be a good predictor of the actual mean indirect worth in each bin. **Bottom** Variance of actual indirect worth versus the same expected values. Note (bottom left, bottom right) that the conditional variance of indirect worth given expected indirect worth is large for large negative expected values, as well as large positive expected values. Cells with large negative expected indirect worth are cells with strong output weights to downstream neighbours of lower-than-average worth; these cells have a large effect on downstream activity, but cannot be sure whether the effect is good or bad.

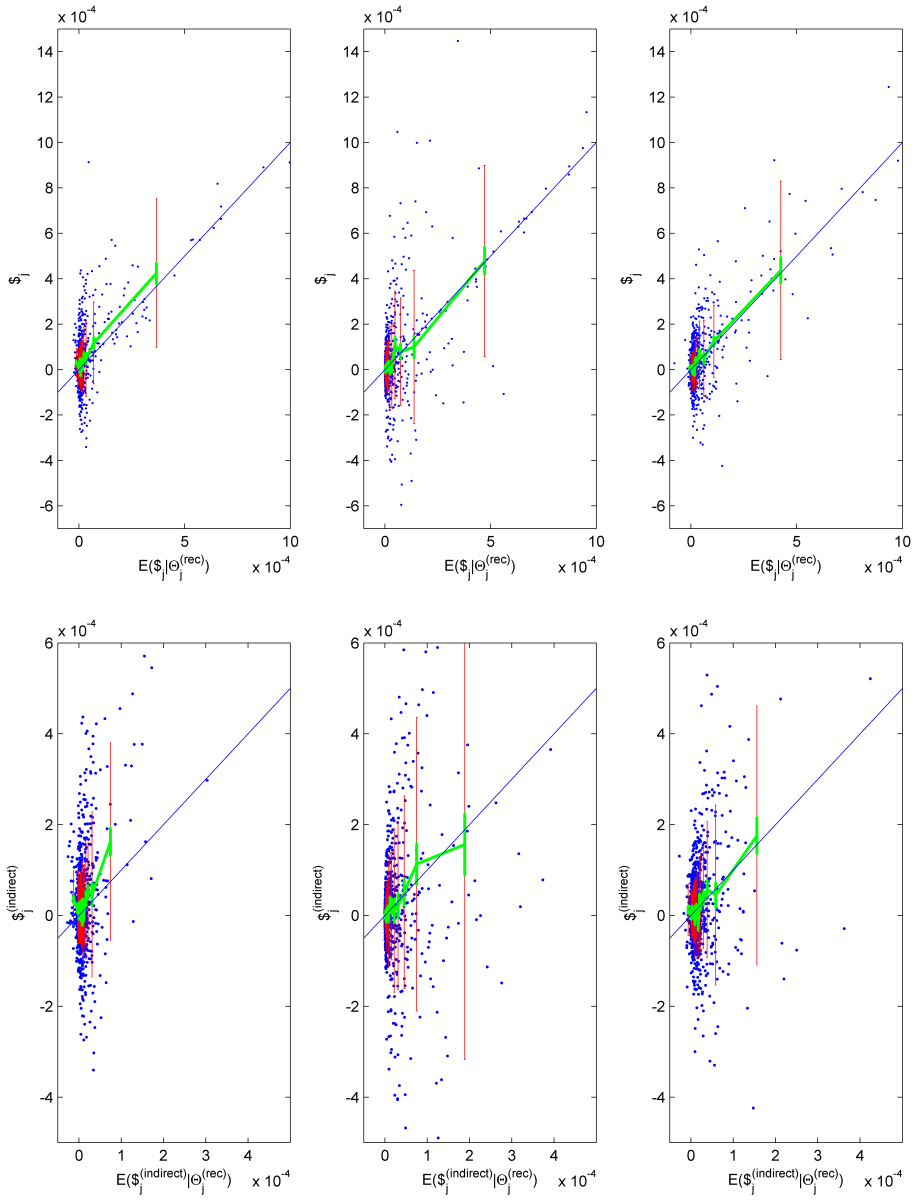


Figure 24: Actual versus expected worth and indirect worth in a single instance of each type of network. The expected values are computed using slow recursive retrograde messages, passing the expected linworth of each cell to each of its upstream neighbours. Each scatter plot has one point per cell for a single instance of each of the three random networks in figure 12. Blue lines are equality. **Top** Worth in the three networks, versus the expected value. Expected values are sorted into bins, and statistics of the corresponding values of true worth $\$_j$ computed in each bin. The lines show the mean (green) and standard deviation (red error bars) of the actual worth in each bin, as well as the standard error on the mean (green error bars). The expected values are approximately correct but the spread of true values around the expected values is large. **Bottom** Same, but with indirect worth instead of worth: not such a close fit, as it excludes the easy-to-estimate direct worth.

6.6 Rough bounds for worth using slow retrograde signals

In the previous section we considered how accurately a cell can estimate its worth using certain slow retrograde signalling schemes. We found that each cell could compute a conditional expected value and variance for its worth given the information it would receive from such signals, but that the coefficient of variation would typically remain greater than 1, so that few cells could confidently predict even whether their worth is positive or negative. One might argue that these schemes provide each cell with not much more than roughly symmetric, probabilistic bounds on its worth; and that such coarse symmetric bounds could be obtained more simply. Here we show a simpler way to get symmetric probabilistic bounds for each cell's worth. This simpler scheme still uses recursive retrograde messages, but without requiring the precalculation step to estimate the parameter k_m, and without depending so heavily on assumptions of Gaussianity and approximately zero mean output weights from each cell.

The linworth of cell i, $\$_i^{(\mathrm{lin})} = \langle \zeta_i, a_i \rangle$, depends on the correlation of one signal, a_i, with another, ζ_i. Components of a_i orthogonal to ζ_i do not contribute to this correlation, but could still contribute to the worth of an upstream cell j. Therefore, instead of propagating bounds for $\$^{(\mathrm{lin})}$ directly, we first propagate estimates for $|a_i|$ and then use this to bound linworth. A bound for a_i implies a bound for linworth via the Cauchy-Schwartz inequality: $\$_i^{(\mathrm{lin})^2} = \langle \zeta_i, a_i \rangle^2 \leq [\![a_i]\!]^2 [\![\zeta_i]\!]^2$.

We will not use any Fourier transform in the scheme we derive, but to derive it we start with the frequency-domain description:

$$\tilde{\mathbf{a}} = g\mathbf{W}^T\tilde{\mathbf{a}} + \tau g\tilde{\mathbf{b}}$$

We will use this to work out how to propagate estimates of $|a_i|^2$.

To see how the mean output weight of each cell affects the calculation, we decompose $\mathbf{W}$ into $W_{ij} = \overline{w}_i + T_{ij}$, so the T_{ij} are independent random variables with zero mean. In this notation

$$\tilde{a}_j = g\sum_i W_{ij}\tilde{a}_i + \tau g\tilde{b}_j = g\overline{w}_j \sum_i \tilde{a}_i + \sum_i T_{ij}\tilde{a}_i + \tau g\tilde{b}_j$$

Taking a sum over j on each side and using $\sum_j W_{ij} = 0$ we have that $\sum_j \tilde{a}_j = \tau g\tilde{b}_j$ and so $\sum_j \tilde{a}_j = \tau g \sum_j \tilde{b}_j$ is a function of $\tilde{b}_j$ alone. Therefore we can define

$$\tilde{C} = \sum_j \tilde{a}_j = \tau g \sum_j \tilde{b}_j$$

and write

$$\tilde{a}_j = g\overline{w}_j\tilde{C} + g\sum_i T_{ij}\tilde{a}_i + \tau g\tilde{b}_j \tag{51}$$

The T_{ij} are uncorrelated, therefore the expected value of $|\tilde{a}_j|^2$ given knowledge of $|\tilde{a}_i|^2$,T_{ij},$|\tau g\tilde{b}_j + g\overline{w}_j\tilde{C}|$ is approximately

$$\mathbb{E}\left(|\tilde{a}_j|^2\right) \approx \left|\tau g\tilde{b}_j + g\overline{w}_j\tilde{C}\right|^2 + \sum_i |g|^2 T_{ij}^2 \left|\tilde{a}_i\right|^2$$

If each downstream neighbour i were able to report the expected value $\tilde{A}_i^2$ of its $|\tilde{a}_i|^2$ given information it receives, then cell j could compute an expected value $\tilde{A}_j^2$ of its own $|\tilde{a}_j|^2$ using

$$\tilde{A}_j^2 = |\tau g\tilde{b}_j + g\overline{w}_j\tilde{C}|^2 + \sum_i |g|^2 T_{ij}^2 \tilde{A}_i^2$$

This would eventually converge on

$$\begin{pmatrix} \tilde{A}_1^2 \\ \vdots \\ \tilde{A}_p^2 \end{pmatrix} = (\mathbf{I} - |g|^2(\mathbf{T}\circ\mathbf{T})^T)^{-1} \begin{pmatrix} |\tau g\tilde{b}_1 + g\overline{w}_1\tilde{C}|^2 \\ \vdots \\ |\tau g\tilde{b}_p + g\overline{w}_p\tilde{C}|^2 \end{pmatrix} \tag{52}$$

This needs simplification, since to directly implement a message-passing scheme where cells compute and relay messages about $\tilde{A}_i^2$, each cell would have to compute a power spectrum of $\tau g\tilde{b}_i$, and separately transmit messages about each frequency band. However, we can simplify. The synaptic gain $|g| \leq 1$ at all frequencies, and the Hadamard product $(\mathbf{T}\circ\mathbf{T})$ has all positive entries and spectral radius less than 1, so that if

$$\begin{pmatrix} A_1^2 \\ \vdots \\ A_p^2 \end{pmatrix} = (\mathbf{I} - (\mathbf{T}\circ\mathbf{T})^T)^{-1} \begin{pmatrix} |\tau b_1 + \overline{w}_1 C|^2 \\ \vdots \\ |\tau b_p + \overline{w}_p C|^2 \end{pmatrix} \tag{53}$$

then for each i, we have

$$\int_\omega \tilde{A}_i(\omega)^2 \leq A_i^2 \tag{54}$$

The A_i^2 in (53) can be computed by recursive retrograde message-passing if each cell repeatedly updates

$$A_j^2 \leftarrow \sum_i T_{ij}^2 A_i^2 + |\tau b_j + \overline{w}_j C|^2 \tag{55}$$

The inequality (54) lets us attach meaning to the resulting values A_i. Recall that $[\![a_i]\!]$ measures the magnitude of effect a tweak to cell i's input can have on the performance of the network as a whole, once consumer cells have relearned their input weights, and we can estimate worth of a cell i as $\langle a_i, \zeta_i \rangle - \$_i^{(\text{direct})}$. In (54), $\int_\omega \tilde{A}_i(\omega)^2$ is the expected value of $[\![a_i]\!]^2 = \int_\omega |\tilde{a}_i(\omega)|^2$ that cell i *would* compute if each cell could compute a power spectrum of b and pass separate estimates of power in a at each frequency. A_i^2 is an upper bound for that expected value of $[\![a_i]\!]^2$, and can be computed without doing any spectrum analysis, using the recursive update (55). With Chebyshev's theorem, A_i^2 gives bounds on the probability that the mean square of a_i will exceed any particular value:

$$\forall k > 0 : \mathbb{P}\left([\![a_i]\!]^2 > k^2 A_i^2\right) \leq \frac{1}{k} \tag{56}$$

From the probabilistic bounds for $[\![a_i]\!]$ in (56), Cauchy's inequality lets us derive probabilistic bounds for linworth $\$_i^{(\text{lin})} = \langle a_i, \zeta_i \rangle$:

$$\forall k > 0 : \mathbb{P}\left(\$_i^{(\text{lin})^2} > k^2 A_i^2 \, [\![\zeta_i]\!]^2\right) \leq \frac{1}{k}$$

We now have a way to calculate probabilistic bounds for linworth, using a simpler recursive retrograde message-passing scheme (55) than the one in 6.5.5.

Figure 25 compares the probabilistic bound $A_i^2 \, [\![\zeta_i]\!]^2$ to the actual value of linworth squared, for the same three simulated networks as in sections 6.4, 6.5.4 and 6.5.6. We introduced slack at several stages in deriving A_i^2 — for example, g can be much smaller than 1 — so it is not surprising to find that in our simulations the linworth squared never exceeds $A_i^2 \, [\![\zeta_i]\!]^2$. Figure 25 also shows that $A_i^2 \, [\![\zeta_i]\!]^2$ significantly discriminates between cells of large worth and cells of small worth. The Spearman rank correlation coefficients between $A_i^2 \, [\![\zeta_i]\!]^2$ and $\$_i^{(\text{lin})^2}$ are given in table 4. These are all positive and highly statistically significant.

	Network 1	Network 2	Network 3
all	$0.45(p < 10^{-307})$	$0.76(p < 10^{-307})$	$0.48(p < 10^{-307})$
high	$0.45(p < 10^{-307})$	$0.59(p < 10^{-307})$	$0.49(p < 10^{-307})$
low	$0.20(p = 6.62 \times 10^{-6})$	$0.54(p < 10^{-307})$	$0.21(p = 2.48 \times 10^{-6})$

Table 4: Spearman rank correlation coefficient between $\$_i^{(\mathrm{lin})^2}$ and $A_i^2 [\![\zeta_i]\!]^2$. The three networks are the three types of network in Figure 12 and data are shown for a single instance of each type of network. 'All' means all cells, 'low' means cells with below-median values of $A_i^2 [\![\zeta_i]\!]^2$, and 'high' means cells with above-median values of $A_i^2 [\![\zeta_i]\!]^2$. All correlations are highly significant.

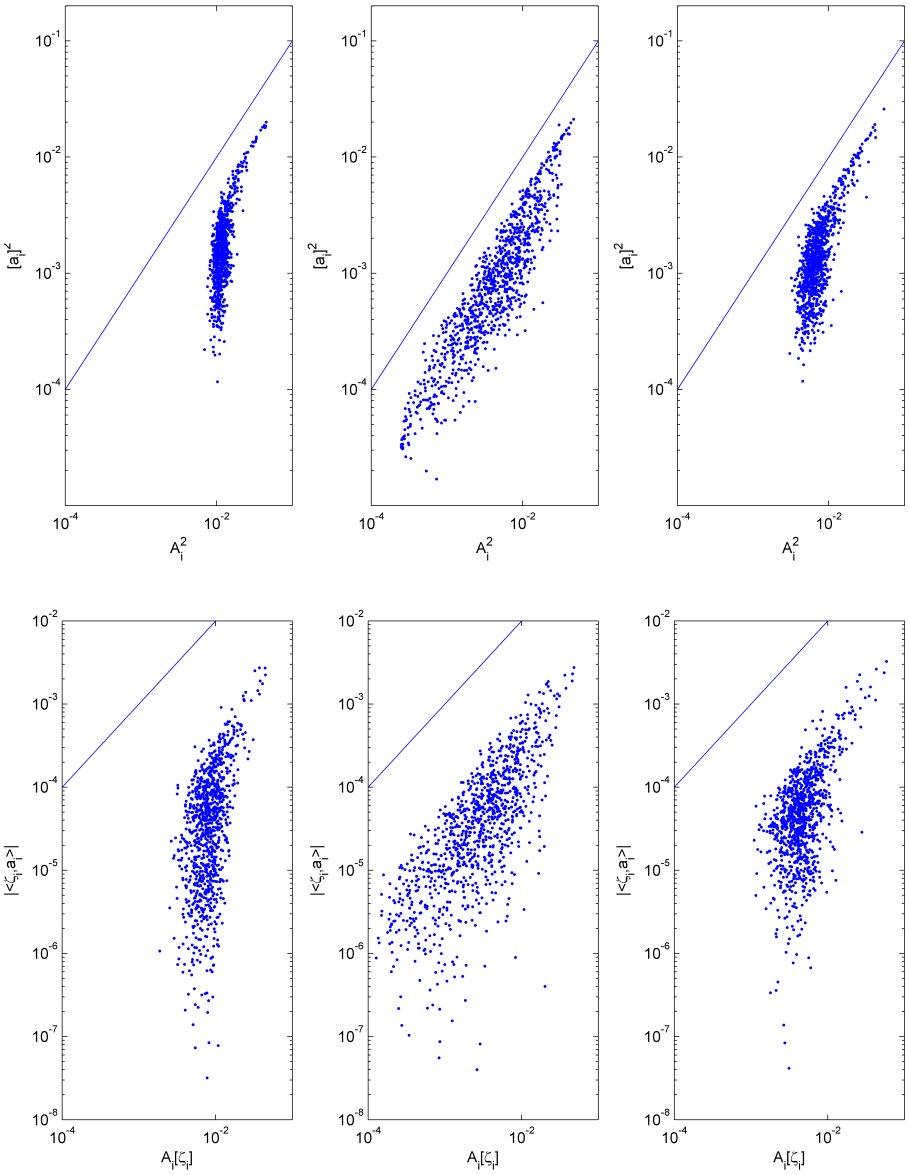


Figure 25: Checking the probabilistic bounds for $[\![a_i]\!]$ and $\$_i^{(\mathrm{lin})}$ computed using the equations in section 6.6. The three columns are for the three networks shown in Figure 12. The top panels show the actual values of $[\![a_i]\!]^2$ versus the probabilistic bound A_i^2. The bottom panels show the actual values of linworth, $\$_i^{(\mathrm{lin})} = \langle \zeta_i, a_i \rangle$, versus the probabilistic bounds $A_i\,[\![\zeta_i]\!]$ which follow from the bound on a_i by Cauchy's inequality. The straight lines are equality. In all cases the actual values are well below the probabilistic bound, and a relationship can be seen between the bound and the actual value.

6.7 Conclusion

In this chapter we asked what a cell can infer about its own worth if it lies in a random, recurrently connected network of producers.

First we found a way to accurately estimate worths of cells in a simulated network (6.3). We used simulations of three different types of random network to test the method. The method is fast to simulate and gives accurate estimates of worth (6.4), but biologically implausible because it depends on fast retrograde signalling.

We then asked (6.5) what can be inferred by a cell about its worth using *slow* retrograde signals. We considered two scenarios: one where cells attempt to estimate their worth using cellular information alone, and another where cells pass recursive slow retrograde messages to estimate worth. In the recursive case, the retrograde signals tell each cell an estimated linworth for each of its downstream neighbours. We tried to approximate 'the belief of cell j about its worth' given the available information in each case. The work in (6.3) using fast retrograde signals was useful in two ways: as a benchmark against which to measure the accuracy of these other estimates, and as a starting point for the mathematical analysis. We derived strategies for estimating worth given cellular information, or using recursive slow retrograde messages. We implemented the strategies in simulations in sections 6.5.4 and 6.5.6 and we showed that, consistent with the analysis, they give correct expected values for worth, but leave large uncertainty about the worth of each cell. The uncertainty about a cell's worth is similar whether we use cellular information alone, or whether we use recursive retrograde signals that report to each cell the expected linworths of all its downstream neighbours. For cells with high expected worth, the estimates are more informative than for cells with low worth: the conditional coefficient of variation in actual worth is largest when the expected worth is small.

Finally in section 6.6 we looked at a simpler way of getting probabilistic, symmetric bounds for cells' worths, still using slow recursive retrograde messages. In this simpler scheme, we do not even attempt to estimate the sign of each cell's worth. In simulations we found this gave a reliable bound on the absolute value of each cell's worth, and that the bounds were significantly correlated with the absolute values of cells' true worths. In a real, biological neural network that is well adapted to its task, and with a loss function that really corresponds to reproductive success, there should not be many cells with large negative worths: these are cells which not only serve no purpose, but are actually harmful. It might be ar-

gued, therefore, that bounding the absolute value of a cell's worth and assuming the worth is nonnegative gives a good *signed* esitmate of worth. However, the analysis of section 6.6 assumed independent random synaptic weights and a rather specific form of loss function, so we cannot deduce that without further analysis.

The conclusion of this chapter is that, in random networks where each cell has a large number of inputs, it is hard to accurately estimate the indirect worths of cells using slow retrograde signals. However, it is possible to arrive at probablistic bounds or an order-of-magnitude estimate. Adding recursive retrograde signalling — so that each cell can monitor the worths of downstream cells to which it provides input — may offer marginally better worth estimates than simply letting each cell monitor the strengths of its output weights.

A question not addressed in this analysis is how accurate an estimate of worth needs to be to support PMH, or to support some other learning scheme where learning rates are modulated according to a cell's worth. If we start with a random network, use expected worth given slow retrograde messages as an estimate of worth, and modulate learning rates according to that estimate, can we expect to make the network perform better? The fact that there is significant correlation between expected and actual worth — especially for cells of high worth — makes this plausible, but it is not trivial to implement. We found that simple implementations (not shown here) quickly led to unstable networks, probably because in the model studied, the expected worth of a cell increases as its root mean square firing rate increases. Some kind of homeostasis, or a cost function that penalises large firing rates, is probably required, both for computational efficacy and for biological relevance.

The analysis in this chapter was confined to networks with independent, random synaptic weights. As soon as any 'learning' happens, the weights cease to be independent random variables. This includes 'learning' that consists of selectively destabilising the input synaptic weights of cells with small expected worth. Having independent, random weights was a key assumption in our analysis of how to compute expected worths from slow retrograde signals. The troubling conclusion is that as soon as we use these estimates of worth to guide learning, the estimates themselves may become invalid. However, it still seems plausible that a learning rule using estimated worth, computed from slow retrograde signals, could work to improve network performance.

7 Discussion

This thesis develops an intuitive, biological idea — the retroaxonal hypothesis — into a mathematical framework — parallel Metropolis-Hastings, using estimation of 'worth' by passage of slow retroaxonal signals. The intention is to show that the retroaxonal hypothesis corresponds to a class of learning algorithms which should, and do, work, and also to explain some limitations. We use mathematical models that are necessarily reductive and abstract, but aim not to violate the constraints of biological systems.

We began in chapter 3 by stating the retroaxonal hypothesis and reviewing the biological evidence. This evidence is circumstantial, but shows that the hypothesis does not demand anything biologically implausible. Rather, the retroaxonal hypothesis offers a coherent explanation for disparate experimental observations: reversibility of synaptic changes; over-representation of behaviourally relevant information; and the existence of slow retrograde signals which can modulate synaptic plasticity. Still, the retroaxonal hypothesis lacks direct experimental evidence.

In chapter 4, we converted the retroaxonal hypothesis to a mathematical model. The retroaxonal hypothesis hinges on the idea of cells' output spike trains being more or less useful, so we made a precise mathematical definition of the 'worth' of a cell, in a very general model neural network. The retroaxonal hypothesis is reminiscent of the standard Metropolis-Hastings algorithm, often used for stochastic optimisation. We showed that the retroaxonal hypothesis can be translated into a varation on Metropolis-Hastings which we call 'parallel Metropolis-Hastings', or PMH.

We described conditions that would analytically guarantee convergence of PMH. These will not be met exactly in any reasonably sophisticated simulation, let alone a real neural network, but in chapter 5 we showed that they can be approximately met in a simple feedforward network with a quadratic cost function. In chapter 5 we also showed that in this kind of network, cells can estimate their worths from their output weights. We implemented PMH, with cells estimating their worths from their output weights and making their rates of plasticity dependent on their worth. Here, PMH was effective in making a population of cells concentrate on useful features, and it improved performance of the network on the given task. Chapters 4 and 5 together validate the two key intuitions of the retroaxonal hypothesis: that cells with strong output synapses tend to be those that encode something useful, and that useful cells should not change their input synapses too readily. However, this

validation was in a very simple feedforward network. The algorithm was successfully deployed on a toy 'cocktail party' problem.

PMH, and the retroaxonal hypothesis, depend crucially on cells being able to estimate how useful their output is by measuring their output synaptic weights. Chapter 5 showed this is possible in a simple feedforward network. In Chapter 6 we explored how this may extend to recurrently connected networks, and investigated the use of slow retrograde signals to estimate worth in these networks. We considered networks with independent random synaptic weights, and we studied the belief of a cell about its worth given information it might receive via retrograde signals. We derived strategies for estimating worth with slow retrograde signals, and simulated them.

Consistent with the analysis in Chapter 6, we found that — assuming a netowrk with independent, random synaptic weights — when using slow retroaxonal signals to estimate worth, the conditional coefficient of variation in worth is large for cells of small expected worth, and greater than 1 even for cells with the largest expected worths. Using slow retrograde signals, cells can learn about the absolute values of their worths, but few cells (especially not those with small estimated worth) can tell whether their worth is positive or negative.

7.1 What does it mean for the brain?

In this thesis we set out to formalise the retroaxaonal hypothesis mathematically, and to show that it corresponds to some kind of computationally useful learning rule.

Chapters 4 and 5 suggest that for feedforward structures, 'retroaxonal learning' as proposed by the retroaxonal hypothesis is computationally sound. Thus, the retroaxonal hypothesis might be a model for how hippocampal cells in CA1 that project to the subiculum — and which do not have many excitatory connections to each other — could modulate the plasticity rate of their input synapses.

In recurrent networks, the results are less clear. The argument of Chapter 4 still applies: if cells could discover their worths, then cells of low worth should make their input synapses more plastic than cells of high worth. However, Chapter 6 shows that in a recurrently connected network, it becomes difficult to accurately estimate the worths of cells in a biologically plausible way.

The analysis in Chapter 6 was of networks with independent random synaptic weights, which is obviously a poor approximation to the brain, but allows some initial calculations to be done. In those networks there are some cells of *negative* worth, corresponding to

cells that are not only unhelpful but actively hinder the survival of the host. All of the slow retroaxonal messaging schemes we investigated leave a large proportion of cells highly uncertain as to whether their worth is positive or negative. However, we did find that cells with large estimated worth were relatively less uncertain about their worth, so that some highly useful cells are correctly identified as such. Even without the stochasticity of the PMH algorithm, it seems that if a learning scheme uses slow retrograde messages to estimate worth and, say, selectively change the input synapses of cells that have negative worth, it is bound to also change some cells that have positive worth. It is possible that in a network that is already well adapted, there are fewer cells of negative worth, and therefore estimating the magnitude of each cell's worth is as good as estimating the signed worth. Then, learning rates modulated by worth, with worth estimated by slow retrograde signals, could become a better proposition. However, since we only studied ways of estimating worth in recurrent networks with *random* weights, we do not know if this is the case.

The difficulty of estimating a neuron's worth from its output weights brings us back to the economic analogy that motivated the retroaxonal hypothesis. Thinking about this analogy makes it unsuprising that it is hard to estimate a cell's worth from its output weights in a random recurrent network. In the retroaxonal hypothesis, each cell is considered as an entrepreneur, 'selling' spike trains to other cells. For example, the random recurrent networks of chapter 6, with supervised output layer, correspond to a strange dystopia where end-consumers make intelligent buying decisions, while all the suppliers randomly mix and match components from each other. The retroaxonal hypothesis says that if a cell has many strong output weights, we may infer that what it produces is probably useful. This is like assuming that if producer X sells many items, what it makes is probably good i.e. that end-consumers are better off than they would be if producer X did not exist or ceased production. It is only a good assumption if something forces the recipients of producer X's output to make buying decisions that benefit end-consumers: for example, every consumer must know what will benefit himself or herself. This is not reality. Guessing, from the revenue of a manufacturer, how much worse off people would be if it disappeared, will always be unreliable. By analogy it is not surprising that it is hard to guess from the 'revenue' of a neuron how much worse off the 'end-consumers' of the network would be if that neuron fell silent. The message-passing scheme of 6.5.5 estimates the indirect worth of a cell to be large if it provides a lot of input to other

cells that have above-average worth, but with large uncertainty. In the economic analogy, a producer whose output is frequently used in more-popular-than-average products, probably makes a good thing — but we cannot be too sure.

To convincingly prove or disprove the retroaxonal hypothesis, biological experiments are required. To prove the retroaxonal hypothesis, an experiment (or experiments) would need to do two things. First, it must show not just that strong output weights tend to coincide with low input synaptic plasticity, but that strong output synaptic weights *cause* a reduction in input synaptic plasticity. Second, it must show that strong output weights indicate high utility to the host.

To show that strong (or weak) output weights cause reduced (or increased) input synaptic plasticity, Kenneth Harris has suggested disabling the output synapses of a subset of pyramidal cells by knocking out the vesicular glutamate transporter VGLUT1 in those cells. This would prevent them from releasing neurotransmitter at their output synapses, but should not affect their electrical properties and should not directly affect plasticity of their input synapses. However, it will prevent postsynaptically initiated, activity-dependent plasticity of their output synapses; and according to the retroaxonal hypothesis it should increase plasticity of the same cells' input synapses. The experiment was attempted *in vitro*, using glutamate uncaging to measure synaptic weights, but failed because of problems with the viral vector; it seems worth another try.

The second experimental task involves measuring utility of a neuron to its host organism, which is another interesting problem. Usually this is done not directly, by measuring the behavioural effect of killing or silencing cells, but by measuring the activity of neurons while the host performs some behavioural task, guided by sensory stimuli. If doing the task depends on recognising some particular sensory stimulus, and a neuron is found to be selectively responsive to that stimulus, then we guess that the neuron is useful in performance of that task. This is supported by (Sigala and Logothetis, 2002) which shows in a vision experiment that after learning, more behaviourally relevant features are represented by larger numbers of cells. If we accept this method of estabilishing utility of a cell, then 'all' we have to do is measure each cell's responsiveness to a variety of stimuli while doing a behavioural task and while also measuring the same cells' output weights, or the plasticity rates of their input synapses. It may be possible to test for input plasticity not electrophysiologically, but histologically, by staining for some long-lasting

byproduct of input synaptic plasticity. None of this is impossible with modern techniques, but the number of variables involved may make it hard to get adequate statistical power. A totally different approach would be to try to selectively kill or silence populations of cells with strong output synapses, or cells whose input synapses are undergoing plasticity, and measure the effect on behaviour. The retroaxonal hypothesis would predict that the cells with plastic input synapses or weak output synapses tend to be useless cells, and can be silenced without great detriment to the host, while others are less dispensable.

References

Agnihotri, N. T., Hawkins, R. D., Kandel, E. R., and Kentros, C. (2004). The long-term stability of new hippocampal place fields requires new protein synthesis. *Proceedings of the National Academy of Sciences of the United States of America*, 101(10):3656–61.

Aicardi, G., Argilli, E., Cappello, S., Santi, S., Riccio, M., Thoenen, H., and Canossa, M. (2004). Induction of long-term potentiation and depression is reflected by corresponding changes in secretion of endogenous brain-derived neurotrophic factor. *Proceedings of the National Academy of Sciences of the United States of America*, 101(44):15788–15792.

Anagnostaras, S. G., Gale, G. D., and Fanselow, M. S. (2001). Hippocampus and contextual fear conditioning: Recent controversies and advances. *Hippocampus*, 11(1):8–17.

Anderson, T. W. (1958). *An Introduction to Multivariate Statistical Analysis*. Wiley series in probability and statistics. Wiley.

Balduzzi, D. (2014). Cortical Prediction Markets. In *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems*.

Barco, A., Patterson, S., Alarcon, J. M., Gromova, P., Mata-Roig, M., Morozov, A., and Kandel, E. R. (2005). Gene expression profiling of facilitated L-LTP in VP16-CREB mice reveals that BDNF is critical for the maintenance of LTP and its synaptic capture. *Neuron*, 48(1):123–137.

Bentivoglio, M. and Morelli, M. (2005). Chapter I The organization and circuits of mesencephalic dopaminergic neurons and the distribution of dopamine receptors in the brain. *Handbook of Chemical Neuroanatomy*, 21:1–107.

Buss, R. R., Sun, W., and Oppenheim, R. W. (2006). Adaptive roles of programmed cell death during nervous system development. *Annual Review of Neuroscience*, 29:1–35.

Caporale, N. and Dan, Y. (2008). Spike timing-dependent plasticity: a Hebbian learning rule. *Annu Rev Neurosci*, 31:25–46.

Carter, A. G., Soler-Llavina, G. J., and Sabatini, B. L. (2007). Timing and location of synaptic inputs determine modes of subthreshold integration in striatal medium spiny neurons. *The Journal of Neuroscience*, 27(33):8967–8977.

Cavallanti, G., Cesa-Bianchi, N., and Gentile, C. (2007). Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167.

Cesa-Bianchi, N. and Gentile, C. (2006). Tracking the best hyperplane with a simple budget perceptron. In *Learning Theory*, pages 483–498. Springer.

Clopath, C., Buesing, L., Vasilaki, E., and Gerstner, W. (2010). Connectivity reflects coding: a model of voltage-based STDP with homeostasis. *Nature Neuroscience*, 13(3):344–352.

Cooper, L. N., Intrator, N., Blais, B. S., and Shouval, H. Z. (2004). *Theory of Cortical Plasticity*. World Scientific.

Costa, R. M. (2007). Plastic corticostriatal circuits for action learning. *Annals of the New York Academy of Sciences*, 1104(1):172–191.

Crammer, K., Kandola, J., and Singer, Y. (2004). Online classification on a budget. *Advances in neural information processing systems*, 16:225.

Crick, F. (1989). The recent excitement about neural networks. *Nature*, 337(6203):129–132.

de Almeida, L., Idiart, M., and Lisman, J. E. (2009). The input–output transformation of the hippocampal granule cells: from grid cells to place fields. *Journal of Neuroscience*, 29(23):7504–7512.

Dekel, O., Shalev-Shwartz, S., and Singer, Y. (2008). The forgetron: a kernel-based perceptron on a budget. *Siam J comput*, 37(5):1342–1372.

DiStefano, P. S., Friedman, B., Radziejewski, C., Alexander, C., Boland, P., Schick, C. M., Lindsay, R. M., and Wiegand, S. J. (1992). The neurotrophins BDNF, NT-3, and NGF display distinct patterns of retrograde axonal transport in peripheral and central neurons. *Neuron*, 8(5):983–993.

Du, J.-L. and Poo, M.-M. (2004). Rapid BDNF-induced retrograde synaptic modification in a developing retinotectal system. *Nature*, 429(6994):878–883.

Du, J.-l., Wei, H.-p., Wang, Z.-r., Wong, S. T., and Poo, M.-m. (2009). Long-range retrograde spread of LTP and LTD from optic tectum to retina. *Proceedings of the National Academy of Sciences of the United States of America*, 106(45):18890–18896.

Eckenstein, F. P., Baughman, R. W., and Quinn, J. (1988). An anatomical study of cholinergic innervation in rat cerebral cortex. *Neuroscience*, 25(2):457–474.

Edelmann, E., Leßmann, V., and Brigadski, T. (2014). Pre-and postsynaptic twists in bdnf secretion and action in synaptic plasticity. *Neuropharmacology*, 76:610–627.

English, C. N., Vigers, A. J., and Jones, K. R. (2012). Genetic evidence that brain-derived neurotrophic factor mediates competitive interactions between individual cortical neurons. *Proceedings of the National Academy of Sciences of the United States of America*, 109(47):19456–61.

Fee, M. S. (2014). The role of efference copy in striatal learning. *Current Opinion in Neurobiology*, 25:194–200.

Fitzsimonds, R. M., Song, H. J., and Poo, M. M. (1997). Propagation of activity-dependent synaptic depression in simple neural networks. *Nature*, 388(6641):439–448.

Florian, R. V. (2007). Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Computation*, 19(6):1468–1502.

Frank, L. M., Stanley, G. B., and Brown, E. N. (2004). Hippocampal plasticity across multiple days of exposure to novel environments. *Journal of Neuroscience*, 24(35):7681–7689.

Frank, M. J. (2011). Computational models of motivated action selection in corticostriatal circuits. *Current Opinion in Neurobiology*, 21(3):381–386.

Franzius, M., Vollgraf, R., and Wiskott, L. (2007). From grids to places. *Journal of Computational Neuroscience*, 22(3):297–299.

Fremaux, N., Sprekeler, H., and Gerstner, W. (2010). Functional requirements for reward-modulated spike-timing-dependent plasticity. *Journal of Neuroscience*, 30(40):13326–13337.

Frey, U. and Morris, R. G. (1997). Synaptic tagging and long-term potentiation. *Nature*, 385(6616):533–536.

Frey, U. and Morris, R. G. (1998). Synaptic tagging: implications for late maintenance of hippocampal long-term potentiation. *Trends in neurosciences*, 21(5):181–188.

Friedrich, J., Urbanczik, R., and Senn, W. (2011). Spatio-Temporal credit assignment in neuronal population learning. *PLoS Computational Biology*, 7(6).

Frotscher, M. and Léránth, C. (1985). Cholinergic innervation of the rat hippocampus as revealed by choline acetyltransferase immunocytochemistry: a combined light and electron microscopic study. *The Journal of Comparative Neurology*, 239(2):237–246.

Hamburger, V. (1992). History of the discovery of neuronal death in embryos. *Journal of Neurobiology*, 23(9):1116–1123.

Hamburger, V. (1993). The history of the discovery of the nerve growth factor. *Journal of Neurobiology*, 24(7):893–897.

Hardingham, N., Dachtler, J., and Fox, K. (2013). The role of nitric oxide in pre-synaptic plasticity and homeostasis. *Frontiers in Cellular Neuroscience*, 7:190.

Harris, K. D. (2008). Stability of the fittest: organizing learning through retroaxonal signals. *Trends in Neuroscience*, 31(3):130–136.

Harris, K. D. (2013). Top-Down control of cortical state. *Neuron*, 79(3):408–410.

Harris, K. D. and Shepherd, G. M. (2015). The neocortical circuit: themes and variations. *Nature neuroscience*, 18(2):170–181.

Hasselmo, M. E. (2006). The role of acetylcholine in learning and memory. *Current Opinion in Neurobiology*, 16(6):710–5.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer, 2 edition.

Haviv, M. and Heyden, L. V. D. (1984). Perturbation bounds for the stationary probabilities of a finite Markov chain. *Advances in Applied Probability*, 16(4):804–818.

Henaff, M., Jarrett, K., Kavukcuoglu, K., and LeCun, Y. (2011). Unsupervised learning of sparse features for scalable audio classification. In *ISMIR*, volume 11, page 445.

Hill, A. J. (1978). First occurrence of hippocampal spatial firing in a new environment. *Experimental Neurology*, 62(2):282–297.

Hinton, G. E. and Sejnowski, T. J. (1999). *Unsupervised learning: foundations of neural computation*. MIT press.

Huang, Y. Y. and Kandel, E. R. (1995). D1/D5 receptor agonists induce a protein synthesis-dependent late potentiation in the CA1 region of the hippocampus. *Proceedings of the National Academy of Sciences of the United States of America*, 92(7):2446–2450.

Hubel, D. H. and Wiesel, T. N. (1959). Receptive fields of single neurones in the cat's striate cortex. *Journal of Physiology*, 148:574–591.

Hyvärinen, A. and Oja, E. (1998). Independent component analysis by general nonlinear hebbian-like learning rules. *Signal Processing*, 64(3):301–313.

Ito, M. and Doya, K. (2011). Multiple representations and algorithms for reinforcement learning in the cortico-basal ganglia circuit. *Current Opinion in Neurobiology*, 21(3):368–373.

Izhikevich, E. M. (2007). Solving the distal reward problem through linkage of STDP and dopamine signaling. *Cerebral Cortex*, 17(10):2443–2452.

Jonas, P., Bischofberger, J., Fricker, D., and Miles, R. (2004). Interneuron diversity series: Fast in, fast out–temporal and spatial signal processing in hippocampal interneurons. *Trends in neurosciences*, 27(1):30–40.

Karlsson, M. P. and Frank, L. M. (2008). Network dynamics underlying the formation of sparse, informative representations in the hippocampus. *Journal of Neuroscience*, 28(52):14271–81.

Kentros, C., Hargreaves, E., Hawkins, R. D., Kandel, E. R., Shapiro, M., and Muller, R. V. (1998). Abolition of long-term stability of new hippocampal place cell maps by NMDA receptor blockade. *Science*, 280(5372):2121–6.

Kentros, C. G., Agnihotri, N. T., Streater, S., Hawkins, R. D., and Kandel, E. R. (2004). Increased attention to spatial context increases both place field stability and spatial memory. *Neuron*, 42(2):283–295.

Kincaid, A. E., Zheng, T., and Wilson, C. J. (1998). Connectivity and convergence of single corticostriatal axons. *The Journal of neuroscience*, 18(12):4722–4731.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Kuhl, P. K., Williams, K. A., Lacerda, F., Stevens, K. N., and Lindblom, B. (1992). Linguistic experience alters phonetic perception in infants by 6 months of age. *Science*, 255(5044):606–608.

Kwee, I., Hutter, M., and Schmidhuber, J. (2001). Market-based reinforcement learning in partially observable worlds. In *Proceedings of the International Conference on Arti Neural Networks (ICANN-2001*, pages 865–873. Springer.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE.

LeCun, Y., Denker, J. S., Solla, S. A., Howard, R. E., and Jackel, L. D. (1989). Optimal brain damage. In *NIPs*, volume 89.

Lee, D., Lin, B.-J., and Lee, A. K. (2012). Hippocampal place fields emerge upon single-cell manipulation of excitability during behavior. *Science*, 337(6096):849–53.

Lewis, S. N. and Harris, K. D. (2014). The neural marketplace: I. general formalism and linear theory. *bioRxiv*, page 013185.

Li, Q., Rothkegel, M., Xiao, Z. C., Abraham, W. C., Korte, M., and Sajikumar, S. (2014). Making synapses strong: Metaplasticity prolongs associativity of long-term memory by switching synaptic tag mechanisms. *Cerebral Cortex*, 24(2):353–363.

Lovinger, D. M. (2010). Neurotransmitter roles in synaptic modulation, plasticity and learning in the dorsal striatum. *Neuropharmacology*, 58(7):951–961.

Maass, W., Natschlager, T., and Markram, H. (2002). Real-time computing without stable states: a new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560.

MacKay, D. J. (1992). A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472.

MacKay, D. J. C. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press.

Malinow, R. and Malenka, R. C. (2002). AMPA receptor trafficking and synaptic plasticity. *Annual Review of Neuroscience*, 25:103–126.

Maren, S., Yap, S. a., and Goosens, K. a. (2001). The amygdala is essential for the development of neuronal plasticity in the medial geniculate nucleus during auditory fear conditioning in rats. *Journal of Neuroscience*, 21(6):RC135.

Marr, D. (1970). A theory for cerebral neocortex. *Proceedings of the Royal Society of London. Series B, Containing papers of a Biological character. Royal Society (Great Britain)*, 176(43):161–234.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

Moore, T. and Armstrong, K. M. (2003). Selective gating of visual signals by microstimulation of frontal cortex. *Nature*, 421(6921):370–373.

Nitz, D. and McNaughton, B. (2004). Differential modulation of CA1 and dentate gyrus interneurons during exploration of novel environments. *Journal of Neurophysiology*, 91(2):863–872.

O'Keefe, J. and Dostrovsky, J. (1971). The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain research*, 34(1):171–175.

Oppenheim, R. W. (1991). Cell death during development of the nervous system. *Annual Review of Neuroscience*, 14:453–501.

Pape, H.-C. and Pare, D. (2010). Plastic synaptic networks of the amygdala for the acquisition, expression, and extinction of conditioned fear. *Physiological Reviews*, 90(2):419–463.

Penrose, O. (1970). *Foundations of statistical mechanics*. Pergamon Press.

Petreanu, L., Gutnisky, D. A., Huber, D., Xu, N.-l., O'Connor, D. H., Tian, L., Looger, L., and Svoboda, K. (2012). Activity in motor-sensory projections reveals distributed coding in somatosensation. *Nature*, 489(7415):299–303.

Purves, D. (1988). *Body and brain: a trophic theory of neural connections*. Harvard University Press.

Rajan, K. and Abbott, L. F. (2006). Eigenvalue spectra of random matrices for neural networks. *Physical Review Letters*, 97(18):188104.

Redondo, R. L. and Morris, R. G. M. (2011). Making memories last: the synaptic tagging and capture hypothesis. *Nature Reviews Neuroscience*, 12(1):17–30.

Riccio, A., Ahn, S., Davenport, C. M., Blendy, J. A., and Ginty, D. D. (1999). Mediation by a CREB family transcription factor of NGF-dependent survival of sympathetic neurons. *Science*, 286(5448):2358–2361.

Riccio, A., Pierchala, B. A., Ciarallo, C. L., and Ginty, D. D. (1997). An NGF-TrkA-mediated retrograde signal to transcription factor CREB in sympathetic neurons. *Science*, 277(5329):1097–1100.

Rogerson, T., Cai, D. J., Frank, A., Sano, Y., Shobe, J., Lopez-Aranda, M. F., and Silva, A. J. (2014). Synaptic tagging during memory allocation. *Nature Reviews Neuroscience*, 15(3):157–69.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Sajikumar, S. and Frey, J. U. (2004). Late-associativity, synaptic tagging, and the role of dopamine during LTP and LTD. *Neurobiology of Learning and Memory*, 82(1):12–25.

Schultz, W., Dayan, P., and Montague, P. R. (1997). A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599.

Sharma, N., Deppmann, C. D., Harrington, A. W., St. Hillaire, C., Chen, Z. Y., Lee, F. S., and Ginty, D. D. (2010). Long-Distance Control of Synapse Assembly by Target-Derived NGF. *Neuron*, 67(3):422–434.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge University Press.

Sigala, N. and Logothetis, N. K. (2002). Visual categorization shapes feature selectivity in the primate temporal cortex. *Nature*, 415(6869):318–20.

Silva, A. J., Kogan, J. H., Frankland, P. W., and Kida, S. (1998). CREB and memory. *Annual Review of Neuroscience*, 21:127–148.

Sjöström, P. J., Turrigiano, G. G., and Nelson, S. B. (2003). Neocortical LTD via coincident activation of presynaptic NMDA and cannabinoid receptors. *Neuron*, 39(4):641–654.

Solstad, T., Moser, E. I., and Einevoll, G. T. (2006). From grid cells to place cells: A mathematical model. *Hippocampus*, 16(12):1026–1031.

Song, S., Sjöström, P. J., Reigl, M., Nelson, S., and Chklovskii, D. B. (2005). Highly nonrandom features of synaptic connectivity in local cortical circuits. *PLoS Biol*, 3(3):e68.

Sussillo, D. and Abbott, L. F. (2009). Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63(4):544–57.

Tao, H., Zhang, L. I., Bi, G., and Poo, M. (2000). Selective presynaptic propagation of long-term potentiation in defined neural networks. *Journal of Neuroscience*, 20(9):3233–3243.

Thorndike, E. (1898a). Some experiments on animal intelligence. *Science*, 7(181):818–824.

Thorndike, E. L. (1898b). Animal intelligence: an experimental study of the associative processes in animals. *The Psychological Review: Monograph Supplements*, 2(4):i.

Trifilieff, P., Herry, C., Vanhoutte, P., Caboche, J., Desmedt, A., Riedel, G., Mons, N., and Micheau, J. (2006). Foreground contextual fear memory consolidation requires two independent phases of hippocampal ERK/CREB activation. *Learning and Memory*, 13(3):349–58.

Urbanczik, R. and Senn, W. (2009). A gradient learning rule for the tempotron. *Neural Computation*, 21(2):340–352.

Vasilaki, E., Fremaux, N., Urbanczik, R., Senn, W., and Gerstner, W. (2009). Spike-based reinforcement learning in continuous state and action space: when policy gradient methods fail. *PLoS Comput Biol*, 5(12):e1000586.

Watson, F. L., Heerssen, H. M., Bhattacharyya, A., Klesse, L., Lin, M. Z., and Segal, R. A. (2001). Neurotrophins use the Erk5 pathway to mediate a retrograde survival response. *Nature Neuroscience*, 4(10):981–988.

Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Wickens, J. R. (2009). Synaptic plasticity in the basal ganglia. *Behavioural Brain Research*, 199(1):119–128.

Wilson, M. A. and McNaughton, B. L. (1993). Dynamics of the hippocampal ensemble code for space. *Science*, 261(5124):1055–1058.

Wolf, F., Engelken, R., Puelma-Touzel, M., Weidinger, J. D. F., and Neef, A. (2014). Dynamical models of cortical circuits. *Current Opinion in Neurobiology*, 25:228 – 236. Theoretical and computational neuroscience.

Yger, P. and Harris, K. D. (2013). The Convallis Rule for Unsupervised Learning in Cortical Networks. *PLoS Computational Biology*, 9(10).

Zweifel, L. S., Kuruvilla, R., and Ginty, D. D. (2005). Functions and mechanisms of retrograde neurotrophin signalling. *Nature Reviews Neuroscience*, 6(8):615–625.