Chapter 9: Plasticity as optimization

Neural plasticity refers broadly to adaptive change over time in the brain. Much of the work on neural plasticity has focused on synapses as the site of learning and memory. This chapter conceptualizes synaptic plasticity in terms of optimization: modifications of synaptic strength lead to improvements in performance (as measured by an objective function). The most efficient algorithms for improving performance are based on gradient descent, delivering vector feedback that assigns credit to individual synapses. We review ways that synaptic plasticity might approximate gradient descent, either explicitly or implicitly.

Change happens everywhere in the brain, but not all of it is adaptive. For example, thermodynamics places limits on the stability of molecular structures, and mechanical perturbations (stretching, compression) place limits on the stability of cellular structures. It is therefore non-trivial to claim that some forms of change reflect adaptations—i.e., they confer advantages upon an organism that ultimately improve evolutionary fitness. We will refer to these adaptive changes as *neural plasticity*.

Here *learning* will refer to adaptive changes in behavior at the level of the organism. The existence of synaptic plasticity is uncontroversial, but its contribution to behavioral adaptation has been disputed, as discussed further below.

The most well-studied form of neural plasticity is experience-dependent synaptic modification (or synaptic plasticity for short), which plays an important role in learning. This chapter analyzes synaptic plasticity from a normative perspective: given an objective function, how can the brain adapt to efficiently optimize the objective? Efficiency means several things in this context, including sample complexity (how much data are needed to achieve a particular level of performance?), time complexity (how much computation is required as a function of network size and other parameters?), and space complexity (how much information needs to be stored in order to implement the algorithm?). Another important consideration is scalability: how well does the algorithm work in practice on realistically large-scale problems?

In addition to efficiency and scalability considerations, we need to also consider biological constraints: can synapses implement the algorithm based only on the information available to it? Will the algorithm work for spiking neurons? As we will see, these considerations strongly constrain the space of efficient optimization algorithms plausibly realizable by the brain.

1 A brief tour of synaptic plasticity

Before turning to the normative analysis, it will be useful to lay some groundwork for how synaptic plasticity works in the brain. This will help us see connections between the technical ideas elaborated below and the relevant empirical phenomena.

Much of what we know about synaptic plasticity comes from protocols in which the axon of a presynaptic neuron is electrically stimulated while recording from its postsynaptic partner. At baseline, a brief pulse of electrical stimulation induces a measurable change in the membrane potential of the postsynaptic neuron—an *excitatory postsynaptic potential* (EPSP). A high-frequency train of stimulation is then applied, and the EPSP in response to the same test pulse used at baseline is measured. The classic finding, now known as *long-term potentiation* (LTP), is that the resulting EPSP is higher compared to baseline (Bliss and Lømo, 1973), indicating an increase in synaptic strength. The "long-term" designation refers to the persistence of this change, in some cases lasting up to several months (Abraham, 2003).

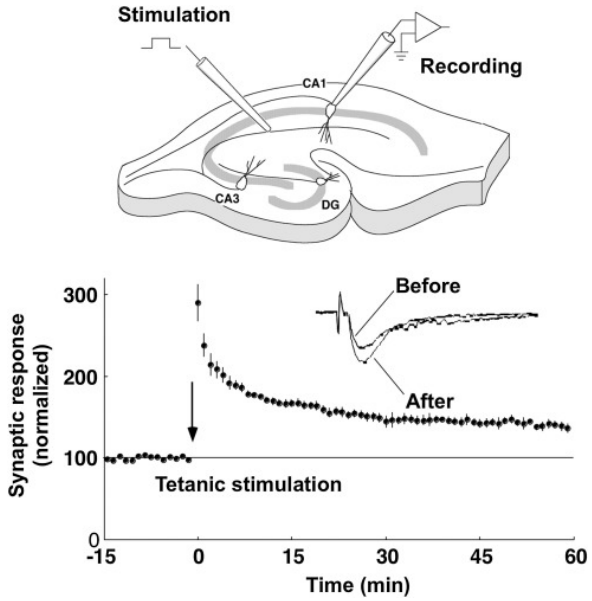

Figure 1: **Long-term potentiation**. (Top) Experimental protocol applied to synapses in the hippocampus. (Bottom) Change in the synaptic response following repetitive high-frequency (tetanic) stimulation is used to measure potentiation. Reproduced from Hayashi (2022).

Several cellular changes underlie LTP. At relatively short timescales (minutes to hours), there are modifications of existing receptors, such as phosphorylation of AMPA receptors by protein kinases, converting them to a high-conductance state (Soderling and Derkach, 2000). These kinases are activated by calcium influx into the cell (more on this below). At relatively long timescales (hours to days), signaling cascades reach the nucleus, activating gene expression that ultimately results in the synthesis of plasticity-related proteins and the traffick-

ing of receptors to the postsynaptic membrane.

Studies of LTP have supported an associative view of synaptic plasticity, often attributed to Hebb (1949), who postulated that "when an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased." The simplest formalization of this idea, applied to firing rates, is a simple coactivation rule:

As noted in Chapter 1, Hebb's postulate can be summarized by the mantra "neurons that fire together, wire together" (Shatz, 1992).

$$\Delta w \propto xy, \tag{1}$$

This is sometimes referred to as the *basic Hebb rule*.

where x is the presynaptic firing rate, y is the postsynaptic firing rate, and w is the synaptic strength connecting the two neurons, such that y is a monotonically increasing function of the product wx. Support for the associative view comes from studies showing that both presynaptic and postsynaptic activity is necessary to induce LTP. For example, hyperpolarization (i.e., deactivation) of the postsynaptic neuron prevents LTP induction (Malinow and Miller, 1986).

The associative nature of LTP fits broadly with associative models of learning. According to these models (Pearce and Bouton, 2001), changes in behavior during associative learning tasks (e.g., Pavlovian conditioning; see next chapter) arise from changes in associations between stimuli. For example, in a Pavlovian fear conditioning task, an animal is repeatedly exposed to a neutral stimulus (e.g., tone) paired with an aversive stimulus (e.g., footshock). The animal learns to produce a conditioned response (freezing) when it hears the tone, a process that depends on the amygdala. The conditioned response can be inactivated by inducing LTD at amygdala synapses, and reactivated by inducing LTP (Nabavi et al., 2014), indicating that synaptic plasticity plays a causal role in associative learning. We'll have more to say about Pavlovian conditioning in the next chapter.

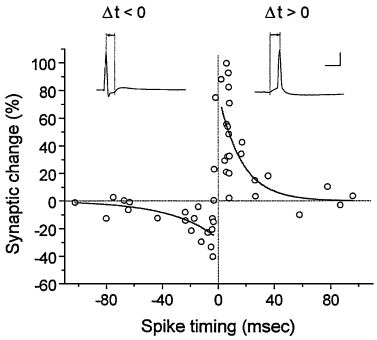

Figure 2: **Spike timing-dependent plasticity**. The x-axis show the relative timing between presynaptic and postsynaptic spikes, as illustrated by the inset postsynaptic voltage traces. The y-axis shows the change in synaptic strength, ranging from negative values (LTD) to positive values (LTP). Reproduced from Bi and Poo (2001), based on data from Bi and Poo (1998).

Hebb's postulate is actually stronger than mere coactivation: it stipulates that the presynaptic neuron "takes part in firing" the postsynaptic neuron. In other words, it is a causal statement. One prerequisite of causality is that it is temporally asymmetric: a cause (presynaptic firing) must occur prior to an effect (postsynaptic firing), typically within a relatively short time window. This is consistent with the phenomenon of *spike timing-dependent plasticity* (STDP; Bi and Poo, 2001), the requirement that presynaptic spikes must occur within 50 ms *before* postsynaptic spikes in order to produce LTP (Figure 2). If they instead occur within 50 ms *after* postsynaptic spikes, a reduction in synaptic strength, known as *long-term depression* (LTD), is obtained. Outside this $\pm$50 ms window, the synaptic strength does not change at all.

The essential mechanisms underlying STDP can be captured with

a relatively simple calcium-dependent model (Graupner and Brunel, 2012). Postsynaptic calcium influx plays a crucial role in the induction of LTP and LTD. One way calcium enters neurons is through NMDA receptors, which are highly permeable to calcium when they bind to the excitatory transmitter glutamate. At the resting potential, magnesium ions block NMDA receptors, preventing calcium influx; when the neuron is depolarized, the magnesium block is removed and calcium can enter. Depolarization can also activate voltage-dependent calcium channels (VDCCs).

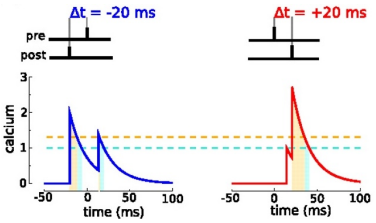

Figure 3: **Model of calcium dynamics underlying STDP**. The turquoise line shows the depression threshold, and the orange line shows the potentiation threshold. Δt denotes the temporal gap between presynaptic and postsynaptic spikes. The shaded regions show the time spent above each threshold. Reproduced from Graupner and Brunel (2012).

In the model of Graupner and Brunel (Figure 3), calcium levels that exceed a depression threshold induce LTD, and levels exceeding a higher potentiation threshold induce LTP. Calcium transients produced by weak inputs are able to cross the depression threshold, but require larger potential changes produced by postsynaptic spiking in order to cross the potentiation threshold. The temporal asymmetry arises from the fact that NMDA receptors have slower kinetics than VDCCs; as a consequence, presynaptic and postsynaptic spiking are more likely to synergistically push the calcium level above the potentiation threshold when presynaptic spikes (driving NMDA receptor activation) occur prior to postsynaptic spikes (driving VDCC activation). When presynaptic spikes occur shortly after postsynaptic spikes, the calcium transient due to VDCC activation may have already decayed, leaving the calcium level above the depression threshold but below the potentiation threshold.

The Graupner-Brunel model also helps resolve a fundamental problem with Eq. 1: synaptic strength can potentially increase without bound, which is obviously problematic for biological synapses. The postulate that low levels of synaptic activity produce LTD, while high levels produce LTP, provides an important stabilizing force. However, plasticity thresholds do not automatically prevent runaway plasticity; it is necessary for the thresholds to grow more quickly than the changes in postsynaptic firing rate. This leads to models with sliding thresholds, where the threshold update is a supralinear function of the firing rate. For example, according to BCM theory (Bienenstock et al., 1982):

$$\Delta w \propto x(y - \theta), \tag{2}$$

where θ is a plasticity threshold updated according to a quadratic function of the firing rate:

$$\Delta\theta \propto y^2 - \theta. \tag{3}$$

Unlike the Graupner-Brunel model, BCM theory only has one threshold separating LTP ($y > \theta$) from LTD ($y < \theta$).

Evidence for a sliding threshold comes from experiments on visual deprivation. A rodent reared in the dark will have weaker activation of neurons in visual cortex, and therefore is predicted to have a lower

threshold compared to normally reared rodents. Consistent with this hypothesis, LTP can be induced with weaker stimulation in dark-reared rodents (Kirkwood et al., 1996).

Another shortcoming of Eq. 1 is that it assumes a strongly local form of synaptic plasticity (only depending on presynaptic and postsynaptic activity), whereas in fact there is considerable evidence that synaptic plasticity depends on additional variables, notably neuromodulators like dopamine, serotonin, and norepinephrine (Kuśmierz et al., 2017). This leads to "three-factor" rules of the following form:

$$\Delta w \propto xy\rho, \tag{4}$$

where ρ is the third factor (e.g., a neuromodulator). We will see below, as well as in the next chapter, how three-factor rules arise from normative considerations. We will also see how the third factor might not be a simple scalar signal, as is usually assumed in the case of neuromodulators, but might instead be vector-valued, conveyed by feedback projections.

2 *Optimization and the credit assignment problem*

Our aim now is to lay out, from first principles, what the goal of synaptic plasticity might be, and how this goal can be achieved algorithmically. Our central claim is that synaptic plasticity optimizes an objective (also known as the *loss* or *error*) function, which scores how well the network is doing on some task. The core problem for any optimization system is *credit assignment*: given a scalar score, how does the system know which parameter to change, and what should the parameter be changed to? In practice, many parameters may need to be changed simultaneously, prohibiting a brute force trial-and-error approach. As we will see, the brain may implement a more sophisticated approach based on gradients.

2.1 *Objective functions*

The material in this section is an elementary introduction to some central machine learning concepts. See Shalev-Shwartz and Ben-David (2014) for a more in-depth treatment.

We will denote the objective function by L, which depends on the network outputs y and an instructive signal (as explained below). The network outputs are in turn are dependent on parameters w. In this chapter, we will take w to be a set of synaptic weights, but more generally this can include any parameters governing the neural network. We will use $y = \phi_w(x)$ to denote the input-output mapping, where x denotes the inputs and y denotes the outputs.

Broadly speaking, objective functions fall into three categories:

- **Supervised objectives** assume that the instructive signal takes the form of a target (or "label") vector y^*. For example, a commonly

used objective is the squared error: $L(y, y^*) = \sum_i (y_i - y_i^*)^2$. Supervised objectives encompass both discrete (categorical) targets as in the last chapter, as well as continuous (regression) targets.

- **Unsupervised objectives** evaluate a network based on how well it matches the input data. A typical example is an autoencoder network that maps the input data to outputs of the same domain (i.e., the inputs are the target outputs, $y^* = x$); the objective function in this case corresponds to a "reconstruction error" such as squared error, $L(y, x) = \frac{1}{2} \sum_i (y_i - x_i)^2$.
- **Reward objectives** evaluate a network based on how well it predicts/improves a scalar reward (or cost). This leads to reinforcement learning algorithms, which will be studied in the next two chapters.

For the purposes of this chapter, we will remain agnostic about the form of the objective function. Our goal is to formulate optimization algorithms that can operate generically on many different kinds of objective functions.

We assume that the agent is exposed to training data $\{x_m, y_m^*\}_{m=1}^M$ sampled from a distribution $p(x, y^*)$, and the objective function is evaluated on this dataset. The agent is thus optimizing a random variable, the *empirical risk*:

$$\hat{L}(w) = \frac{1}{M} \sum_{m=1}^{M} L(\phi_w(x_m), y_m^*). \tag{5}$$

Optimizing the expectation of this random variable (the *risk*, $\mathbb{E}[\hat{L}(w)]$) is typically the ultimate goal, but the agent does not have direct access to this expectation.

Because the empirical risk is a noisy estimate of the risk, optimizing it directly can lead to suboptimal generalization (performance on new inputs). Thus, it is important to introduce an inductive bias that prevents the network from "overfitting" the data (i.e., fitting noise in the data, which degrades generalization to new inputs). This is often dealt with by optimizing *regularized* empirical risk functions of the form $E(w) = \hat{L}(w) + \Omega(w)$, where $\Omega(w)$ is a regularization function. For example, L2 regularization penalizes the Euclidean norm of the weights, $\Omega(w) \propto ||w||^2$ (intuitively, large weights are penalized). Theoretical analyses of generalization error typically require some inductive bias in order to guarantee bounded generalization error (see Shalev-Shwartz and Ben-David, 2014).

Recall from the previous chapter that the representer theorem assumes a regularized empirical risk of this form.

2.2 *Gradient descent: the best game in town*

There are many ways to optimize an objective function. For example, one could try randomly perturbing the synaptic weights, evaluating the objective function, and then accepting or rejecting the perturbed weights based on whether they improve the objective. This form of perturbation + selection is the basis of many different optimization algorithms, such as simulated annealing and genetic algorithms (indeed, natural selection essentially works this way). While these algorithms can work in principle, they face several difficulties. In addition to the difficulty of implementing the selection operation in a biological circuit, these algorithms are hopelessly inefficient: random perturbations to a large network are vanishingly unlikely to improve the objective. They might be good models for evolution, but they're probably not good models for learning in the brain (at least in their simplest forms).

This same basic issue (random-walking in high dimensional space) afflicts some of the Markov chain Monte Carlo algorithms discussed in Chapter 5.

A more efficient approach is to follow the (negative) gradient of the objective function, which specifies the direction of steepest descent in the parameter space. In its simplest form, the weight update takes the following form:

$$\Delta w \propto -\nabla_w E. \tag{6}$$

In practice, it is often desirable to compute the gradient on a subset of examples, possibly just one at a time. Repeatedly sampling subsets and applying the weight update is known as *stochastic gradient descent*. It still converges to the same weights, but now Δw is a random variable. From a neuroscience perspective, it makes sense to think about the "streaming" form of this algorithm, where samples enter sequentially and the update is applied after each one.

Non-streaming algorithms might appear in the brain through *replay*, the reactivation of neural patterns encoding past experiences (Hayes et al., 2021). The reactivated patterns can be recycled into gradient updates.

There are many variations of stochastic gradient descent, too many to cover here. The important point is that they have proven to be the most effective form of learning algorithm in artificial systems, which is why they are ubiquitous in machine learning applications, and also why they may be a plausible hypothesis for learning in the brain. The challenge we address next is how to compute the gradient in a biologically plausible way. At the end of the chapter, we will discuss learning beyond gradients.

3 *Perturbation methods*

A simple way to approximate the gradient is using a variation on the weight perturbation idea that we dismissed in the previous section. We again randomly perturb the weights (e.g., with Gaussian noise), but instead of accepting or rejecting the new weights, we use the

perturbation to estimate the gradient:

This is a form of *finite difference* approximation.

$$\nabla_w E = \frac{1}{\sigma^2}\mathbb{E}[(E(\tilde{w}) - E(w))(\tilde{w} - w)], \tag{7}$$

where $\tilde{w} \sim \mathcal{N}(w, \sigma^2)$ denotes the perturbed weights. We can use this identity to devise a *stochastic approximation* algorithm that uses random perturbations to update the weights:

$$\Delta w \propto -\frac{1}{\sigma^2}(E(\tilde{w}) - E(w))(\tilde{w} - w). \tag{8}$$

The expected update for this algorithm is equal to the exact gradient descent update.

This kind of model has interesting neurobiological implications. Synapses are known to be highly unreliable: an action potential typically produces neurotransmitter release less than half of the time (Allen and Stevens, 1994; Branco and Staras, 2009). Why would a neuron go to the trouble of spiking if the signal frequently fails to be propagated? One answer (see Seung, 2003) is that stochastic release provides information about the gradient through perturbation. If we think of σ^2 as a proxy for synaptic unreliability, one implication is that changes in synaptic strength should be bigger for more unreliable synapses, consistent with experimental data (Bolshakov and Siegelbaum, 1995).

While weight perturbation "works" by providing an unbiased estimate of the gradient, this estimate can have extremely high variance, limiting its practical usefulness—learning via weight perturbation can be orders of magnitude slower than gradient descent (Werfel et al., 2003). An alternative, *node perturbation*, follows the same logic, but perturbs neural activity rather than the weights, taking advantage of the stochastic nature of neural activity (Mazzoni et al., 1991). Because the dimensionality of this perturbation is smaller (there are fewer neurons than synapses), it tends to have lower variance. However, this method is still suffers from high variance compared to direct computation of the gradient.

4 *Backpropagation and its approximations*

Instead of using finite difference approximations, most engineers directly compute the gradient by utilizing the chain rule of calculus. This is particularly apt for deep neural networks, where computations are arranged in chains. The same is true for recurrent networks, which can be "unrolled" into deep networks (as we mentioned in Chapter 8). The backpropagation algorithm (Rumelhart et al., 1986) is essentially an application of the chain rule to deep neural networks

Suppose we have a chain of computations, $x \to y \to z$, where x, y, and z are variables. The chain rule states that $\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}$.

which enables efficient computation of gradients at all layers of the network. It is the workhorse of modern machine learning.

The key idea is to recursively compute gradients, starting at the output layer and then passing the gradients down to the next layer. Let $h_j = \sigma(\mu_j)$ denote the activation of neuron j, where $\sigma(\cdot)$ is an activation function and $\mu_j = \sum_i w_{ij} h_i$ is the total input to neuron j. To make things easier to follow, we'll write the algorithm down in terms of scalar derivatives rather than gradients:

In general the activation function does not strictly need to have this functional form, but it simplifies the math a bit.

$$\frac{dE}{dw_{ij}} = h_i e_j, \qquad e_j = \sigma'(\mu_j) \sum_k e_k w_{jk} \tag{9}$$

where $\sigma'(\mu_j)$ is the partial derivative of the activation function with respect to μ_j. We can think of the e_j terms as "error signals" propagating backwards along the chain of computation. For output neurons, the error signals are given by the gradient of the loss with respect to the output activations, $e_j = \frac{dE}{dh_j}$. As in other gradient descent algorithms, the gradient is used to update the weights:

$$\Delta w_{ij} \propto -\frac{dE}{dw_{ij}}. \tag{10}$$

One way to implement backpropagation (Figure 4) is to construct feedback connections (carrying errors) that mirror the feedforward connections (carrying activations). The errors modulate a "Hebbian" plasticity rule, which depends on the correlation between a presynaptic term, h_i, and a postsynaptic term, $\sigma'(\mu_j)$. In addition, the weight update depends on a third factor—the error signals.

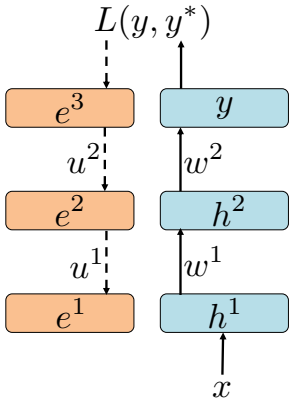

Figure 4: **Backpropagation**. Feedforward projections are shown by solid black lines; feedback projections are shown by dashed lines.

A number of problems vex the biological plausibility of backpropagation:

- **The weight transport problem**: the feedback weights need to be mirror images of the feedforward weights—i.e., if we denote u_{ji} as the feedback weight carrying error signals from unit j to unit i, then we require that $u_{ji} = w_{ij}$. This constraint is not generally satisfied in the brain (Grossberg, 1987).

- **The sign problem**: errors need to be signed (both positive and negative), but neural activity is always non-negative (Lillicrap et al., 2020). One could try to resolve this problem by introducing a threshold (e.g., the baseline firing rate) above which activity is considered positive and below which activity is considered negative. However, this introduces additional complexities, such as the need to carefully tune the threshold and communicate its value to receiving neurons. Another solution could be to use inhibitory neurons to represent negative errors.

- **The magnitude problem**: errors can sometimes vary over multiple orders of magnitude, particularly in recurrent or very deep networks, producing exploding or vanishing gradients. Yet the firing rates of biological neurons typically vary over only about one order of magnitude.

In machine learning, the magnitude problem has motivated solutions based on cut-offs (clipping) and soft constraints (Pascanu et al., 2013).

- **The update locking problem**: weight updates cannot occur in real time, because they have to wait until both the forward and backward passes have been completed (Jaderberg et al., 2017). This implies that somehow a memory trace of the feedforward activations must be maintained until the forward pass has been completed and the backward pass has arrived back at the synapse.

We will not comprehensively address all of these problems, but the next sections provide an overview of approaches that address some of them.

4.1 Learning without weight symmetry

Surprisingly, it turns out that weight symmetry is not necessary for backpropagation to be effective. In fact, backpropagation can work well even when feedback weights are random. There are two important insights into this observation:

- **Sign concordance**: matching the magnitudes of the feedforward and feedback weights doesn't matter much, but matching the signs matters much more (Liao et al., 2016). This implies that there can be a certain degree of "sloppiness" (i.e., the magnitude problem can be "solved" by essentially ignoring it), but somehow the network must still achieve at least partial sign concordance.

In particular, the weights will move in the direction of the gradient as long as on average $\sum_{i,j} e_i e_j w_{ij} u_{ji} > 0$.

- **Feedback alignment**: with random feedback weights, backpropagation naturally adjusts the feedforward weights to achieve "soft" alignment with the feedback weights (Lillicrap et al., 2016). In essence, the feedforward weights learn to partially compensate for the asymmetries created by the random feedback weights.

While random feedback is useful, it would be even better if the feedback connections were also plastic. This possibility was explored by Akrout et al. (2019) using a Hebbian rule (Eq. 1) that converges to symmetric weights. They showed that feedback learning allowed the network to keep pace with the performance of backpropagation.

4.2 Dendritic segregation of feedforward and feedback signals

All of the approaches we've discussed so far assume that there are separate pathways conveying feedforward and feedback signals, and

that feedback signals only affect the feedforward weights (not the activity of the feedforward neurons). We will defer discussion of the second assumption until later. The first assumption is problematic because there is no evidence that feedback pathways have the intricate organization required by the algorithms we've discussed, namely precise pairing between specific feedforward and feedback neurons and their weights. While this is possible in principle, there is no direct evidence for such organization.

An alternative approach is to utilize the known segregation of inputs at the level of dendritic compartments (Guerguiev et al., 2017; Sacramento et al., 2018; Richards and Lillicrap, 2019), as illustrated in Figure 5. At a coarse scale, dendrites on pyramidal neurons (the principal class of excitatory neurons in cortex) can be divided into "apical" (emanating from the apex of the pyramid-shaped cell body) and "basal" (emanating from the base of the pyramid). Apical dendrites primarily receive input from feedback projections, whereas the basal dendrites primarily receive input from feedforward projections. Because of their spatial separation, the activation of these different compartments are electrotonically segregated (i.e., electrical signals generated in one compartment will not passively spread to the other). Importantly, the apical dendrites have a region dense with VDCCs, which can generate "plateau potentials" (a temporally broad voltage change) when inputs to the apical dendrites coincide with somatic spiking (Larkum et al., 1999). This in turn produces high-frequency bursts that are required for plasticity in the basal dendrites (Pike et al., 1999). The bursts thus play the role of the third factor.

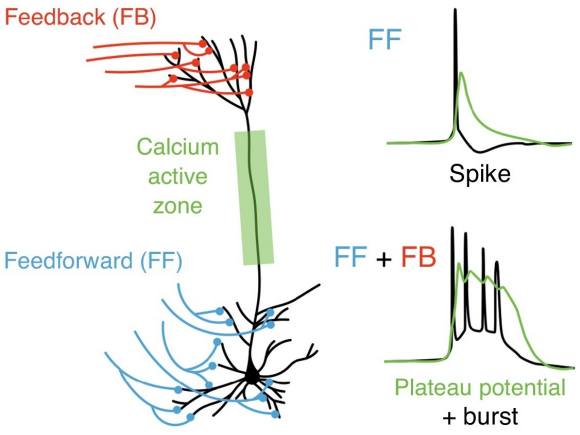

Figure 5: **Dendritic mechanisms supporting credit assignment**. Reproduced from Richards and Lillicrap (2019).

Returning to the backpropagation algorithm, we can identify the feedforward signals with the inputs to the basal dendrites, and the error signals with the inputs to the distal apical dendrites. Plasticity of the feedforward weights is regulated by the error signals through

the cooperative generation of plateau potentials and burst spiking.

4.3 *Target propagation*

Instead of conveying gradients, feedback projections might alternatively convey target patterns of activity. To build some intuitions, let's assume a layered architecture as shown in Figure 4. Suppose the feedforward projections are invertible, so that the pattern of activity at one layer of the network can be inverted to recover the pattern of inputs that produced it. This means that a target pattern of activity at the outputs can be propagated backwards to provide targets for all the earlier layers. Formally, the inverse function is defined as $\hat{h}_{l-1} = g_l(\hat{h}_l)$, where $\hat{h}_l$ is the reconstructed activity at layer l.

The feedforward weights can then be learned in a local manner to match each layer's feedback targets—i.e., to minimize an objective function $L(h_l, \hat{h}_l)$. The inverse functions must also be learned to minimize reconstruction errors. In essence, this *target propagation* algorithm decouples a complex non-local credit assignment problem into a collection of local problems that can be solved more easily. As pointed out by Bengio (2014), what we've described here is a form of *autoencoder*—a network that reconstructs its inputs. In this case, the autoencoder architecture is utilized to solve a general credit assignment problem.

In what sense is target propagation an approximation of the backpropagation algorithm? After all, it departs fundamentally from the core idea of propagating gradients. Lee et al. (2015) showed, under some assumptions (including invertibility), that the weight updates for backpropagation and target propagation typically point in similar directions. Thus, target propagation is doing something *implicitly* like backpropagation, but without propagating gradients.

This algorithm is idealized due to its assumption of invertible projections. In practice, projections are rarely invertible. Lee et al. (2015) developed a variant called *difference target propagation* which corrects for imperfect inversion (i.e., it enforces local consistency between feedforward and feedback mappings):

$$\hat{h}_{l-1} = h_{l-1} + g_l(\hat{h}_l) - g_l(h_l). \tag{11}$$

This guarantees that as the inversion improves in higher layers, it also improves in lower layers. Specifically, it satisfies the following stability condition (by construction):

$$h_l = \hat{h}_l \Rightarrow h_{l-1} = \hat{h}_{l-1}. \tag{12}$$

To see this, note that $\hat{h}_{l-1} - h_{l-1} = g_l(\hat{h}_l) - g_l(h_l)$.

While there is little direct evidence for the update used by difference target propagation, the idea of propagating target patterns is broadly

consistent with predictive coding theories reviewed in Chapter 5. All of these theories have in common the idea that feedback projections convey predictions about feedforward activity, and that plasticity of feedforward weights should reduce the discrepancy (prediction error) between feedforward activations and feedback predictions.

Indeed, predictive coding networks can be used to approximate backpropagation (Whittington and Bogacz, 2017; Song et al., 2020).

Target propagation and its variants take a step towards addressing the sign problem and the magnitude problem. Because only activity *differences* matter (i.e., between the feedback and feedforward patterns), these patterns can be shifted and scaled without fundamentally impairing the algorithm's effectiveness.

4.4 Gradient descent with spiking neurons

So far, we have been assuming that neural activity is rate-based. This is convenient because firing rates are differentiable as long as the activation function $\sigma(\cdot)$ is differentiable. However, spike-based synaptic plasticity phenomena, such as STDP, compel us to think about how backpropagation could work with spiking neurons, despite the fact that their discreteness makes them non-differentiable. There are several different approaches to this problem.

Another argument for spike-based plasticity models is that plasticity based on firing rates requires temporal integration, which precludes learning based on fast changes in neural activity.

One approach is to replace the hard spike threshold with a soft nonlinearity that varies smoothly with input current and approximates the discrete spike generation process (Lee et al., 2016; Huh and Sejnowski, 2018). This renders the membrane potential a differentiable function of the input current. Applying gradient descent to this model leads to a three-factor Hebbian plasticity rule: a product of presynaptic and postsynaptic firing rates modulated by a third factor reflecting the gradient of the objective function with respect to the membrane potential. The main drawback of this approach is that it dispenses with the discreteness characteristic of real neural activity.

Another approach is based on the concept of *surrogate gradients*. The basic idea is to smooth the gradient itself rather than the spike-generation process, permitting the use of more biologically realistic spiking neurons. For example, Zenke and Ganguli (2018) approximated the gradient as the product of temporally filtered presynaptic spike trains (a proxy for the neurotransmitter concentration arriving at the postsynaptic membrane) and a smooth nonlinear function of the postsynaptic membrane potential, modulated by an error signal derived from a (possibly random) feedback projection. Once again, we find a three-factor Hebbian plasticity rule. This algorithm is capable of learning precisely timed spikes in multilayer networks (Figure 6).

Surprisingly, this approach is robust to many different choices of surrogate gradient (Zenke and Vogels, 2021).

A third approach is based on a fundamentally different assumption about the representational primitives. Instead of using rates or

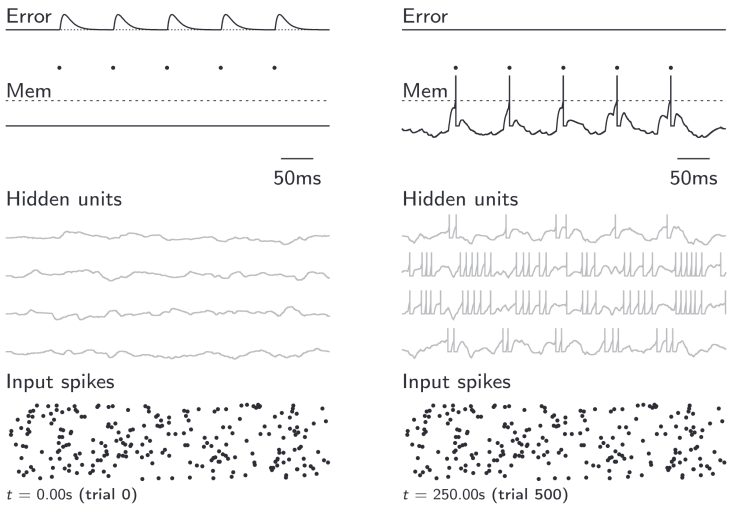

Figure 6: **Gradient descent in a multilayer spiking network using surrogate gradients**. The network is trained to emit a spike at the times indicated by the dots in the top row. The membrane potential of the output neuron is shown in the second row (firing threshold is indicated by a dashed line). The lower rows show the membrane potentials of 4 "hidden" neurons in intermediate layers, along with the spike times of the input neurons (Poisson spike trains that repeated every 500 ms). The left column shows activity before training; the right column shows activity after training. Reproduced from Zenke and Ganguli (2018).

spike counts to convey signals, one can alternatively use spike times as the primitives. This has a number of advantages: it can operate on relatively fast times scales, conveys more information than spikes, and can support various forms of analog computation (e.g., Hopfield, 1995). For present purposes, the continuous nature of spike times affords differentiability and therefore is amenable to gradient descent. An example of this approach is the work of Mostafa (2017), who used the timing of the first spike (i.e., a spike latency code) as the representational primitive.

See Chapter 2 for more on the computational function of spike timing.

5 *Beyond the synapse*

While we have focused on synaptic plasticity, there is considerable evidence that other forms of plasticity also occur in the brain. One example is plasticity of cellular excitability (Debanne et al., 2019). This is "cell-intrinsic" in the sense that what's changing is the cell's overall responsiveness to its integrated inputs rather than to inputs from a specific synapse (hence it is often referred to as *intrinsic plasticity*). For example, the same protocol that produces LTP (high-frequency stimulation of presynaptic axons) increases the probability that a postsynaptic EPSP will produce a spike (Chavez-Noriega et al., 1990). This change in EPSP-spike coupling is distinct from the synapse specific changes underlying LTP.

The learning rules for intrinsic plasticity might, like synaptic plasticity, follow optimization principles. One hypothesis is that the goal of intrinsic plasticity is to maximize information transmission (Stemmler and Koch, 1999). Recall that the mutual information can be decomposed into the difference between entropy and cross-

entropy terms:

$$\mathcal{I}[x; y] = \mathcal{H}[y] - \mathcal{H}[y|x], \tag{13}$$

where again x is the neural input and y is the output. If we make the simplifying assumption that the neuron is near-deterministic, then $\mathcal{H}[y|x]$ goes to 0. Maximizing information transmission then reduces to maximizing output entropy, $\mathcal{H}[y]$. For a fixed average firing rate, output entropy is maximized by an exponential distribution of firing rates, as observed for example in inferotemporal cortex (Baddeley et al., 1997). Stemmler and Koch (1999) derived a learning rule that adapts membrane conductance to maximize output entropy, which could be implemented by phosphorylation of ion channels or changes in gene expression of ion subunits.

A fixed average firing rate implies a homeostatic optimization principle: neurons should adapt their sensitivity to inputs in order to maintain a firing rate set point. Evidence for firing rate homeostasis comes from studies of neural responses in visual cortex after deprivation of ocular input. After initial reduction in responding, neural activity rebounds back to baseline (Hengen et al., 2013). Firing rate homeostasis can help address some of the problems challenging backpropagation in the brain. It supplies a fixed reference for signed errors (positive = above the set point, negative = below the set point), reduces runaway plasticity by preventing the size of Hebbian updates from getting too large, and similarly reduces exploding/vanishing gradients by preventing error signals from becoming too small or large.

More radical proposals have suggested that neurons may learn and remember more complex forms of information using cell-intrinsic mechanisms that go beyond changes in excitability (Gallistel, 2017; Gershman, 2023). For example, modification of polynucleotide sequences like RNA could be a learning mechanism, consistent with (controversial!) evidence that memories can be transferred between organisms via RNA (Jacobson et al., 1966; Bédécarrats et al., 2018). This hypothesis might also explain why memories in some species can survive dramatic synaptic remodeling, such as during metamorphosis, hibernation, and even decapitation (Blackiston et al., 2015).

6 Conclusion

In summary, this chapter has argued that synaptic plasticity can be viewed as approximate gradient descent under biological constraints. We also considered alternative mechanisms potentially that extend optimization principles beyond the synapse.

It's unlikely that any single learning principle will be sufficient to explain all forms of learning in the brain. Nonetheless, the reverse engineering approach introduced in Chapter 1 instructs us to consider what learning principles are optimal or close to optimal—and that work in practice. Research on machine learning has overwhelmingly favored a relatively small set of learning principles based on gradient descent, and its efficient implementation in the form of the backpropagation algorithm. This is why we devoted so much space in this chapter to discussing biological approximations of backpropagation. These approximations enable mutilayered networks to learn complex tasks, sometimes at levels of performance comparable to humans.

Most work in this area has applied gradient descent to synaptic plasticity, in keeping with the widespread belief that synapses are the site of long-term memory storage underlying biological learning. This is surely an important part of the story, but probably not the whole story. Non-synaptic learning mechanisms likely play important roles, as discussed in the last section. We are still only at the beginning of understanding these roles well enough to model them.

Study questions

1. Why is gradient descent considered "the best game in town" compared to methods like weight or node perturbation? What trade-offs exist in terms of biological plausibility?
2. How does the unreliability of neurotransmitter release (synaptic stochasticity) provide information that can be harnessed for gradient estimation?
3. Why do findings of memory transfer and persistence following brain damage challenge synaptic learning models? How might such findings be reconciled with synaptic learning models?

References

Abraham, W. C. (2003). How long will long-term potentiation last? *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 358:735–744.

Akrout, M., Wilson, C., Humphreys, P., Lillicrap, T., and Tweed, D. B. (2019). Deep learning without weight transport. *Advances in Neural Information Processing Systems*, 32.

Allen, C. and Stevens, C. (1994). An evaluation of causes for unrelia-

bility of synaptic transmission. *Proceedings of the National Academy of Sciences*, 91:10380–10383.

Baddeley, R., Abbott, L. F., Booth, M. C., Sengpiel, F., Freeman, T., Wakeman, E. A., and Rolls, E. T. (1997). Responses of neurons in primary and inferior temporal visual cortices to natural scenes. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 264:1775–1783.

Bédécarrats, A., Chen, S., Pearce, K., Cai, D., and Glanzman, D. L. (2018). RNA from trained Aplysia can induce an epigenetic engram for long-term sensitization in untrained Aplysia. *eneuro*, 5.

Bengio, Y. (2014). How auto-encoders could provide credit assignment in deep networks via target propagation. *arXiv preprint arXiv:1407.7906*.

Bi, G.-q. and Poo, M.-m. (1998). Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of Neuroscience*, 18:10464–10472.

Bi, G.-q. and Poo, M.-m. (2001). Synaptic modification by correlated activity: Hebb's postulate revisited. *Annual Review of Neuroscience*, 24:139–166.

Bienenstock, E. L., Cooper, L. N., and Munro, P. W. (1982). Theory for the development of neuron selectivity: orientation specificity and binocular interaction in visual cortex. *Journal of Neuroscience*, 2:32–48.

Blackiston, D. J., Shomrat, T., and Levin, M. (2015). The stability of memories during brain remodeling: a perspective. *Communicative & Integrative Biology*, 8:e1073424.

Bliss, T. V. and Lømo, T. (1973). Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path. *The Journal of Physiology*, 232(2):331–356.

Bolshakov, V. Y. and Siegelbaum, S. A. (1995). Regulation of hippocampal transmitter release during development and long-term potentiation. *Science*, 269:1730–1734.

Branco, T. and Staras, K. (2009). The probability of neurotransmitter release: variability and feedback control at single synapses. *Nature Reviews Neuroscience*, 10:373–383.

Chavez-Noriega, L., Halliwell, J., and Bliss, T. (1990). A decrease in firing threshold observed after induction of the EPSP-spike (ES) component of long-term potentiation in rat hippocampal slices. *Experimental Brain Research*, 79:633–641.

Debanne, D., Inglebert, Y., and Russier, M. (2019). Plasticity of intrinsic neuronal excitability. *Current Opinion in Neurobiology*, 54:73–82.

Gallistel, C. R. (2017). The coding question. *Trends in Cognitive Sciences*, 21:498–508.

Gershman, S. J. (2023). The molecular memory code and synaptic plasticity: A synthesis. *Biosystems*, 224:104825.

Graupner, M. and Brunel, N. (2012). Calcium-based plasticity model explains sensitivity of synaptic changes to spike pattern, rate, and dendritic location. *Proceedings of the National Academy of Sciences*, 109:3991–3996.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11:23–63.

Guerguiev, J., Lillicrap, T. P., and Richards, B. A. (2017). Towards deep learning with segregated dendrites. *eLife*, 6:e22901.

Hayashi, Y. (2022). Molecular mechanism of hippocampal long-term potentiation—towards multiscale understanding of learning and memory. *Neuroscience Research*, 175:3–15.

Hayes, T. L., Krishnan, G. P., Bazhenov, M., Siegelmann, H. T., Sejnowski, T. J., and Kanan, C. (2021). Replay in deep learning: Current approaches and missing biological elements. *Neural Computation*, 33:2908–2950.

Hebb, D. (1949). *The Organization of Behavior*. John Wiley.

Hengen, K. B., Lambo, M. E., Van Hooser, S. D., Katz, D. B., and Turrigiano, G. G. (2013). Firing rate homeostasis in visual cortex of freely behaving rodents. *Neuron*, 80:335–342.

Hopfield, J. J. (1995). Pattern recognition computation using action potential timing for stimulus representation. *Nature*, 376:33–36.

Huh, D. and Sejnowski, T. J. (2018). Gradient descent for spiking neural networks. *Advances in Neural Information Processing Systems*, 31.

Jacobson, A. L., Fried, C., and Horowitz, S. D. (1966). Planarians and memory: I. Transfer of learning by injection of ribonucleic acid. *Nature*, 209:599–601.

Jaderberg, M., Czarnecki, W. M., Osindero, S., Vinyals, O., Graves, A., Silver, D., and Kavukcuoglu, K. (2017). Decoupled neural interfaces using synthetic gradients. In *International Conference on Machine Learning*, pages 1627–1635. PMLR.

Kirkwood, A., Rioult, M. G., and Bear, M. F. (1996). Experience-dependent modification of synaptic plasticity in visual cortex. *Nature*, 381:526–528.

Kuśmierz, Ł., Isomura, T., and Toyoizumi, T. (2017). Learning with three factors: modulating Hebbian plasticity with errors. *Current Opinion in Neurobiology*, 46:170–177.

Larkum, M. E., Zhu, J. J., and Sakmann, B. (1999). A new cellular mechanism for coupling inputs arriving at different cortical layers. *Nature*, 398:338–341.

Lee, D.-H., Zhang, S., Fischer, A., and Bengio, Y. (2015). Difference target propagation. In *Proceedings of the 2015th European Conference on Machine Learning and Knowledge Discovery in Databases-Volume Part I*, pages 498–515.

Lee, J. H., Delbruck, T., and Pfeiffer, M. (2016). Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10:508.

Liao, Q., Leibo, J., and Poggio, T. (2016). How important is weight symmetry in backpropagation? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Lillicrap, T. P., Cownden, D., Tweed, D. B., and Akerman, C. J. (2016). Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7:13276.

Lillicrap, T. P., Santoro, A., Marris, L., Akerman, C. J., and Hinton, G. (2020). Backpropagation and the brain. *Nature Reviews Neuroscience*, 21:335–346.

Malinow, R. and Miller, J. P. (1986). Postsynaptic hyperpolarization during conditioning reversibly blocks induction of long-term potentiation. *Nature*, 320:529–530.

Mazzoni, P., Andersen, R. A., and Jordan, M. I. (1991). A more biologically plausible learning rule for neural networks. *Proceedings of the National Academy of Sciences*, 88:4433–4437.

Mostafa, H. (2017). Supervised learning based on temporal coding in spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 29:3227–3235.

Nabavi, S., Fox, R., Proulx, C. D., Lin, J. Y., Tsien, R. Y., and Malinow, R. (2014). Engineering a memory with LTD and LTP. *Nature*, 511:348–352.

Pascanu, R., Mikolov, T., and Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318. Pmlr.

Pearce, J. M. and Bouton, M. E. (2001). Theories of associative learning in animals. *Annual Review of Psychology*, 52:111–139.

Pike, F. G., Meredith, R. M., Olding, A. W., and Paulsen, O. (1999). Postsynaptic bursting is essential for "Hebbian' induction of associative long-term potentiation at excitatory synapses in rat hippocampus. *The Journal of physiology*, 518:571–576.

Richards, B. A. and Lillicrap, T. P. (2019). Dendritic solutions to the credit assignment problem. *Current Opinion in Neurobiology*, 54:28–36.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323:533–536.

Sacramento, J., Ponte Costa, R., Bengio, Y., and Senn, W. (2018). Dendritic cortical microcircuits approximate the backpropagation algorithm. *Advances in Neural Information Processing Systems*, 31.

Seung, H. S. (2003). Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. *Neuron*, 40:1063–1073.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press.

Shatz, C. J. (1992). The developing brain. *Scientific American*, 267:60–67.

Soderling, T. R. and Derkach, V. A. (2000). Postsynaptic protein phosphorylation and LTP. *Trends in Neurosciences*, 23:75–80.

Song, Y., Lukasiewicz, T., Xu, Z., and Bogacz, R. (2020). Can the brain do backpropagation?—exact implementation of backpropagation in predictive coding networks. *Advances in Neural Information Processing Systems*, 33:22566–22579.

Stemmler, M. and Koch, C. (1999). How voltage-dependent conductances can adapt to maximize the information encoded by neuronal firing rate. *Nature Neuroscience*, 2:521–527.

Werfel, J., Xie, X., and Seung, H. (2003). Learning curves for stochastic gradient descent in linear feedforward networks. *Advances in Neural Information Processing Systems*, 16.

Whittington, J. C. and Bogacz, R. (2017). An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity. *Neural Computation*, 29:1229–1262.

Zenke, F. and Ganguli, S. (2018). Superspike: Supervised learning in multilayer spiking neural networks. *Neural Computation*, 30:1514–1541.

Zenke, F. and Vogels, T. P. (2021). The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *Neural Computation*, 33:899–925.