Laboratory 4: Concept formation, synaptic modification and error correction

Goals

1. To understand category-supervised synaptic modification; in category-supervised modification, a neuron learns how to recognize a category based on the weighting of features associated with that category;
2. To understand self-supervised synaptic modification; in self-supervised modification, a neuron creates its own feature weighting based on the correlation between features;
3. To understand how an error-correction synaptic modification mechanism can produce improvement in a neuron's performance by adjusting synaptic strengths.

Laboratory 4 is divided into three parts which will be covered in three, consecutive lab sessions. In Part 1, you will create networks capable of category-supervised and linear, self-supervised synaptic modification. Part 2 considers nonlinear, self-supervised synaptic modification and threshold modification via error correction. Finally, Part 3 applies error-corrected synaptic modification to a very simple form of associative learning.

In the first three labs, you learned how to create a simple neural network that could recognize different patterns and how the rate constant ε influences performance of a moving averager. As we mentioned at the end of Laboratory 3, information encoding by a synapse is more complex than a simple moving averager. Despite the increased complexity, there are notable similarities, e.g., the important role played by the rate constant ε. Taken together with this lab, you have all the ideas needed to build pattern recognition neural networks that specify their own synaptic weights. To build such networks, moving averagers have been promoted from devices that approximate simple means (e.g., `Ave[x]` as in Lab 3) to devices that incorporate pairwise statistics (e.g., a conditional statistic such as `Ave[x|y]` or a correlation `Ave[`x_1`,y]`).

Introduction

Now we can finally get to the interesting stuff: *adaptive synaptic modification*. *Synaptic modification,* and especially associative synaptic modification, is at the heart of nearly all neural computation and at the heart of the biological basis of most learning. When we talk about encoding associations, you should think specifically of changes at a synapse (see Fig. 4.1 for a schematic view of the underlying biology).

What do we mean by *associative* synaptic modification? Synaptic modification depends upon coordinated changes (i.e., associated in time) of the activities of the presynaptic and the postsynaptic neurons. Specifically, the strengthening of these synapses depends critically upon *coactivation* in the sense of contemporariness or nearly contemporariness. Such co-activation dependent modification is often called Hebbian.

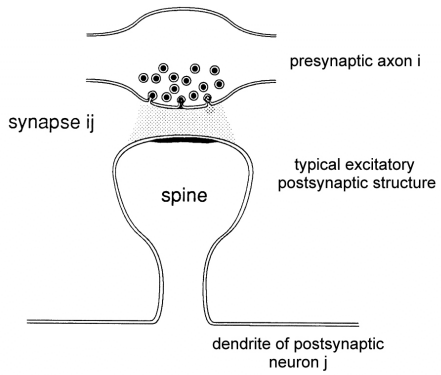


Figure 4.1. A schematic view of an excitatory synapse formed between presynaptic axon **i** and a dendritic spine on postsynaptic neuron **j**. In the cerebral cortex, most synapses are formed between a presynaptic axon and a postsynaptic dendritic spine. You can think of a dendritic spine as a protrusion from the parent dendrite. As shown in Fig. 4.1, a dendritic spine often has a stem, that connects it with its parent dendrite and the remainder of the neuron, and a head, where the synapse itself is located. The **synapse** is where modification takes place. In this schematic, packets of neurotransmitter (•) are being released from synaptic vesicles in the presynaptic axon. The transmitter diffuses across the synaptic cleft and binds to transmitter receptors typically associated with a postsynaptic density (shown as a thickening on the surface of the postsynaptic spine). These binding events lead to a series of biochemical and electrical events in the postsynaptic spine and neuron. Spinous axodendritic synapses in forebrain cortical systems are almost always excitatory.

Because correlated activity directs synaptic changes, synaptic modification is a modification that reflects the correlated history of the two elements that make up the synapse—that is, the correlated history of a presynaptic and a postsynaptic neuron.

Modification rules

Here we introduce three different rules for associative modification of existing synapses:

1. category-supervision
2. self-supervision (unsupervised)
3. error-correction (feedback supervised)

These rules are related in at least two ways. First, variables being associated are, ultimately, *local* to the synapse being modified. Second, and arising from this spatial localization, all three rules encode something about the correlation between pre- and postsynaptic information.

What do we mean by the word *local*? The synaptic change which stores information can only be used (in a sense, read out) at the very same location. That is, reusing a modified synapse means reading out its stored information. Read-out and use of a memory by signal flow is in contrast to the way one uses memory when programming a digital computer. In a computer, two numbers can interact in the CPU even though they are spatially segregated in memory; for two synaptic weights to be added together, they must be part of the same postsynaptic neuron, and their activation must be close together in this.

Because a synapse is a two part structure consisting of a presynaptic and a postsynaptic element, we can divide local processes into two categories: 1) the recent activity history, or some function of this recent activity history, of the presynaptic neuron, and 2) the activity history, or some function of this recent activity history, of some part of the postsynaptic neuron. As a result of locality, each synapse has a certain degree of independence of operation from all other synapses. Thus it is critical to designate the name of a synapse via the pre/postsynaptic designation already used, i.e., $w_{pre\ post}$ or w_{ij}. Both locality and the analog nature of neural computation imply that synaptic memory is only accessed when the synapse is presynaptically activated.

Finally, all three rules only modify synapses of just one postsynaptic neuron. This characteristic is another example of the principle of locality—that is, multiple applications of a rule are necessary for multiple postsynaptic neurons.

In contrast to these commonalities, there are some important computational and biological differences between these three synaptic modification rules. The first two rules (category-supervision and self-supervised synaptic modification) are postsynaptic excitation-based, *associative* synaptic modification rules in the sense that a postsynaptic value of zero vetos modification while the third form of synaptic modification (error-correction) is vetoed by a presynaptic value of zero.

In simple feedforward networks, all three rules are guaranteed to converge in the sense that, if we wait long enough, the average change of a synaptic weight converges to zero (the stochastic convergence observed in Laboratory 3). Because of this similarity with the moving averager, we can understand the first two associative, Hebbian synaptic modification rules by elaborating on the simple moving averager of the previous lab.

Now we need to make these general statements quantitative. At any particular point in time, the local, associative synaptic modification rules have the generalized form:

new weight = current weight + change in weight

where

change in weight = postsynaptic state g **presynaptic state**

Consider the weight of the synapse between presynaptic neuron *i* and postsynaptic neuron *j*, w_{ij}, as it changes from time *t* to the next time step, *t*+1. The new synaptic weight equals the sum of the current synaptic weight plus the change in synaptic weight based on the newest sample; that is,

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t). \qquad (4.0)$$

In other words, the strength of synapse w_{ij} at time step *t*+1 equals the sum of its initial strength and the change in its strength which, for computational purposes, occurs at the end of time step *t* to *t*+1.

For each synaptic modification rule, a different process defines $\Delta w_{ij}(t+1)$. Despite this, the associative synaptic modification rules are similar because they describe the changes in the strength of a single synapse, and this change is, in some sense, based on a multiplicative interaction between the pre- and postsynaptic neurons forming the synapse w_{ij}. Thus, the operation, presynaptic$_i$ g postsynaptic$_j$, occurs in all three synaptic modification equations.

In the next sections, we consider the category and the self-supervised modification rules. For both of these rules, the generic form of the change equation is

change in weight = postsynaptic state g **(input excitation-current weight)**

Laboratory 4, Part 1: Concept formation and synaptic modification

Section 4.1.1. Creating a category-supervised network

People learn to identify objects from their features, and in this regard, there is often a particular name that goes along with a particular set of features. A similar process associating a particular category with its associated features occurs at the level of an individual neuron.

Category-supervised synaptic modification is a local, associative form of synaptic modification. A category-supervised neuron has a two-compartment dendrite, which is distinguished by a different class of inputs for each compartment. The inputs to one compartment, when thresholded, define the category each neuron recognizes. Thus, in a sense, this set of thresholded inputs names the category a neuron recognizes. The inputs to the other compartment define the features that are associated with the category name. Sometimes the features and category name are available at about the same time. Sometimes the features are available without the category name. When the features and category occur in temporal contiguity, learning can occur. When only the features are present, the learning is tested to see if the correlation is remembered.

The following example may help you to better understand what we mean by *category* and *features* of a category. Imagine that you are teaching a young child to name animals. You and the child visit the zoo, and as you walk around the exhibits, you see various kinds of animals. First you go to visit the elephants. Even before you speak, the child looks at one of the elephants, seeing its trunk, big feet, and big ears. She looks at the elephant while you say the word 'elephant,' and an association begins forming between the features she sees and the name 'elephant.' Next you walk by the giraffes—once again the child looks at a giraffe, observing its long neck and spots, and you tell her, "It's a giraffe." Eventually she will see a giraffe (i.e., she will observe its features) and will say the category to you: "Look, a giraffe!"

Let's set up a simple neuron that distinguishes the two classes of inputs (see Fig. 4.2). The two-class distinction is fundamental; e.g., the word 'elephant' is distinct and of a different class than the sensory impressions that are the features of an elephant. Such distinctive classes of inputs seem to be segregated at the neuronal level—different input classes are spatially segregated across a neuron's dendritic domain, similar to what has been drawn in Fig. 4.2. In this case, the class of inputs closer to the cell body, the input vector x, is the set of features belonging to the named pattern (e.g., what an elephant looks like); call the state of such a feature vector X. The second class of inputs defines the name itself—i.e., the spoken word 'elephant.' For example, the auditory input when we hear the word "elephant" and the visual input when we see the image of an elephant are coded, and these codes converge on the dendrite. The category inputs called z signify the absence or presence {0,1} of a category name (e.g., elephants). This category naming input will always fire the postsynaptic neuron on which it synapses.

Sometimes both classes of inputs, Z and X, are simultaneously active (or, to say it more biologically, the features are active slightly before the category name is active). Sometimes the

X vector is positively valued in an appropriate way, *X* is active, but the *Z* input is subthresholded for firing a particular neuron, is not. In this case, the goal for the neural network is to guess which category is present, just as if the child went to the zoo alone or looked at the picture book she brought back from her trip to the zoo and had to name the animals on her own. Thus the elephant neuron can be fired in one of two ways, by its name or by its features.

To build a system that can recognize more than one category, we need more postsynaptic neurons. Here, in the interest of simplicity rather than biologically appropriate coding, one postsynaptic neuron stands for each category (such a code is called a *grandmother code*—the more plausible distributed code will be introduced in the coming weeks).

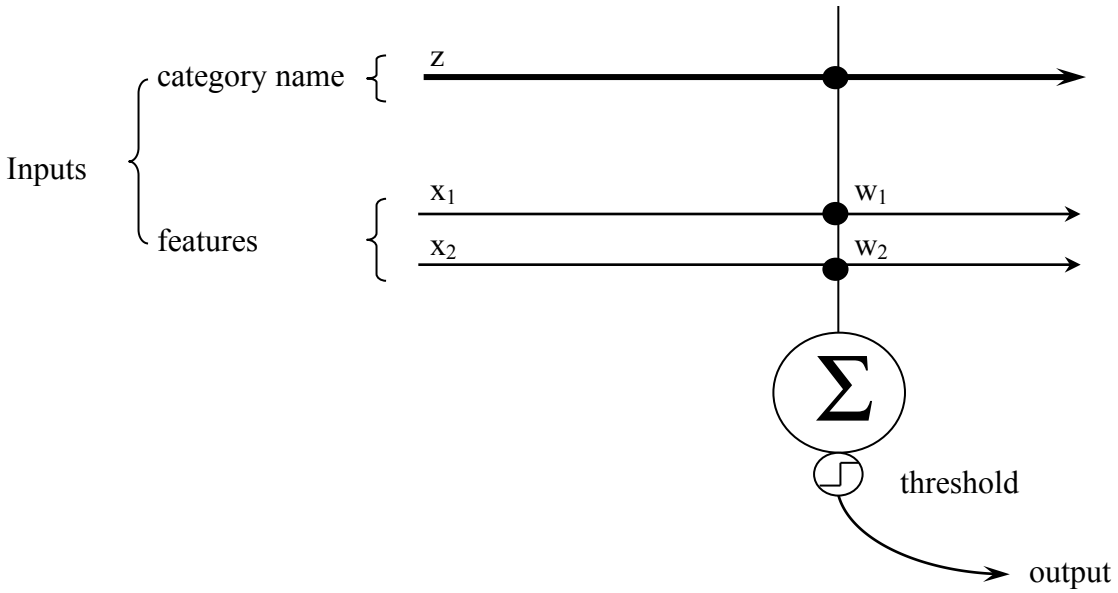


Fig. 4.2. The category-supervised synaptic modification rule governs changes in weight of the synapses, w_{1j} and w_{2j} formed by inputs x_1 and x_2 on neuron j. The output is determined by the weighted sum of the feature inputs or by the value of z. The category line (z) acts like a teacher, telling the neuron when a particular feature set should be incorporated into its averaging processes. Importantly, all synapses here are excitatory and therefore are positively valued. A category name is a complex object but we simplify it here.

A most important distinction between the two classes of inputs is that the category-class controls modifications of the feature-class synapses, but the reverse does not happen—the feature-class inputs do NOT affect the category-class synapses. Rather, the category-class inputs are self-modifying.

Section 4.1.2. Synaptic modification equations for a category-supervised neuron

The neuron you'll create here will have synapses that are modified moving averagers much like averagers in Lab 3. The synaptic weight of a feature input will converge to its approximate average conditioned on the presence of the category input to the neuron being active—this is called a *conditional average*. By conditional, we mean that the average that is encoded at the synapse depends on the context of the input. Using our earlier notation, each synapse *i* encodes $Ave\left[x_i \left| z_j = 1 \right. \right]$ where x_i is a value of a feature input *i* and z_j is the value of neuron *j's* category (absent or present). (Read the vertical stroke as "given.")

The first step is to create a single category input whose value is represented by the variable z_j. When the category *j* is present, z_j=1, e.g., a giraffe neuron fires when the child hears the word 'giraffe' even if no giraffe is being viewed. The z_j input is very special because the postsynaptic neuron, also called *j*, fires whenever z_j=1. When the category *j* is absent, z_j=0.

In addition to responding to the presence or absence of a category name, a category-supervised neuron gets feature inputs. In keeping with the learning and memory paradigm, initially a neuron does not fire to the giraffe features. Suppose the features are visual while the category name is spoken out loud when presented. Moreover, suppose there are other non-giraffe visual features present. Thus, the auditory input $\mathbf{z_j} \in$ {not giraffe, giraffe} signals via correlation and averaging which features to ignore and not ignore. Denote the state of the i^{th} feature input x_i. The feature inputs are represented by the column vector **x**. Just as for the simple moving averager, define a rate constant, $\varepsilon \in (0,1)$. Now we can write the modification rule for category-supervised synaptic modification as

$$\Delta w_{ij}(t) = \varepsilon \cdot z_j(t) \cdot \left(x_i(t) - w_{ij}(t) \right). \tag{4.1}$$

We usually leave time as an implicit variable and inevitably write this equation as

$$\Delta w_{ij} = \varepsilon z_j \left(x_i - w_{ij} \right). \tag{4.2}$$

Note that z_j is a binary variable and can be valued either zero (no giraffe present) or one (giraffe is present). Thus, we denote the allowed values of z_j, $z_j \in \{0,1\}$.

Now we examine these two cases; suppose z_j=1. Then equation (4.2) is exactly our simple moving averager,

$$\Delta w_{ij} = \varepsilon \left(x_i - w_{ij} \right). \tag{4.3}$$

Alternatively, when z_j is not one, no modification occurs. That is, substituting z_j=0 into the category-supervised synaptic modification rule gives Δw_{ij}=0. As a result, when z_j=0, $w_{ij}(t+1) = w_{ij}(t)$; thus, when $z_j = 0$, the synaptic modification of feature *j* ignores the value of the feature vector. This process of updating the synaptic weight allows the neuron to encode the correlations between a category name and each feature.

Based on these relationships, we can just feed the sequence values of $x_i(t+1)$ that coexist with $z_j(t+1)=1$ into the simple moving averager previously studied. In turn, we know what is encoded at each synapse because the moving averager is just that—a process that converges to the average. More precisely, the category-supervised modification rule results in each synapse "learning" the moving average of x_i given that $z_j=1$.

Note that for multiple features i, each weight w_{ij} is updated independently of all other x_i's and w_{ij}'s. Thus this modification rule assumes conditional independence between the feature input values x_i.

Another point to note is that ε is no longer a simple, i.e., time-dependent, rate constant.

Homework 4.1.1. Category Supervised Synaptic Modification

Use input environment 2 from Homework 2.7.1. #5 of Laboratory 2 (where there are three classes (= categories) of inputs on the unit circle and no overlap between classes of inputs). Draw inputs from the set of possible values in some kind of random fashion, write function. Use three postsynaptic neurons, one for each category—i.e., each neuron should be category-activated by only one categorical input so that you will have a grandmother code for all sets of inputs. Let the three categories be equiprobable.

Note that during training the inputs are three-dimensional, category, feature one and feature two. Data arrives as a vector $\begin{bmatrix} z \\ x_1 \\ x_2 \end{bmatrix}$. That is, a category state is paired with a feature state.

Randomly select input data vectors while allowing synaptic modification of the feature synapses. All three postsynaptic neurons receive both of the input features, x_1 and x_2. In MatLab, you can implement the effect of the category states z_j by multiplication as in Equation 4.2 or with an "if/then" conditional statement. (But remember, linear algebra commands execute much, much faster.)

(see next page for continuation)

Homework 4.1.1. continued

(*Specific Assignments*)

A. Start the synaptic weights at zero, and watch them change as a function of successive input presentations. After sufficient synaptic modification, i.e., when the synaptic values have nearly converged, test the network's response to the input features alone (i.e., without category information).
   1) Describe how you define convergence. Graphically illustrate with two examples.
   2) Quantitatively describe the response of each neuron to the various feature vectors.
   3) Describe data that indicate convergences.

Hint: What should happen

Each postsynaptic neuron will have its synapses modified without regard to the other synapses, and each of its synaptic weights will converge to the appropriate conditional average

B. In your write-up, compare the values of the synapses to the conditional averages of each feature given each particular category. Compare the conditional averages of each of x_i of each j to the unconditional averages of each x_i. Compare/contrast how ε should be set if you want an average convergence to be the same for a conditioned and unconditioned averager.

Finally a hard question: Assuming ε is the same for all three postsynaptic neurons, what will be the effect of nonequiprobable categories on convergence rates of synaptic values? Hint: look at your program. What is the effect on changing the probability of categories?

Why is this an example of learning and memory or both? Why/where is the memory, where is the learning? In what sense is this an example of associative learning and memory? Why does this form of synaptic modification qualify as Hebbian?

C. Here we will allow you to set thresholds in a *post hoc* manner. You should use thresholds that give best performance after synaptic weight convergence. Describe the thresholds you use; describe how they affect performance; and describe why you choose them. You should plot the error rate as a function of threshold to show how you determined the optimal threshold. Have your neurons predict (retrodict) the category based on the features. Consistent with a grandmother code, choose the category with the largest excitation (implicitly we will assume some kind of inhibitory feedback in producing this winner-take-all process).

Section 4.1.3. Creating a linear, self-supervised neuron

Sometimes we must learn concepts without a supervising teacher, as I hope you will when you are doing the homework in this course. We all have this ability, although some prefer not to use it. Certainly any of us can go to the zoo alone, e.g., in an exotic foreign country, and learn to discriminate the different animals even though we wouldn't be able to read or hear the animals' names. The same is true for microscopic neuronal computation. A simple neuron can learn something interesting from an input set that is all features and no category name. Such a neuron (or dendritic compartment) lacks a category input, and thus, it is not supervised. Thus we can call the synaptic modification of this neuron *unsupervised* or *self-supervised*. The latter makes particular sense when we understand its synaptic modification rule in comparison to the supervised rule.

In contrast to a category-supervised neuron with its two distinct classes of inputs and corresponding dendritic compartments, a self-supervised neuron has a single compartment dendrite (see Fig. 4.3 for a schematic view). Instead of using an external category input like the binary variable *z* that the category-supervised neuron uses, the self-supervised neuron uses the feature-generated postsynaptic excitation at time *t* as the multiplier of our synapse-specific moving averager. In other words, such neurons use their internal excitation to determine the postsynaptic multiplying factor at each time step.

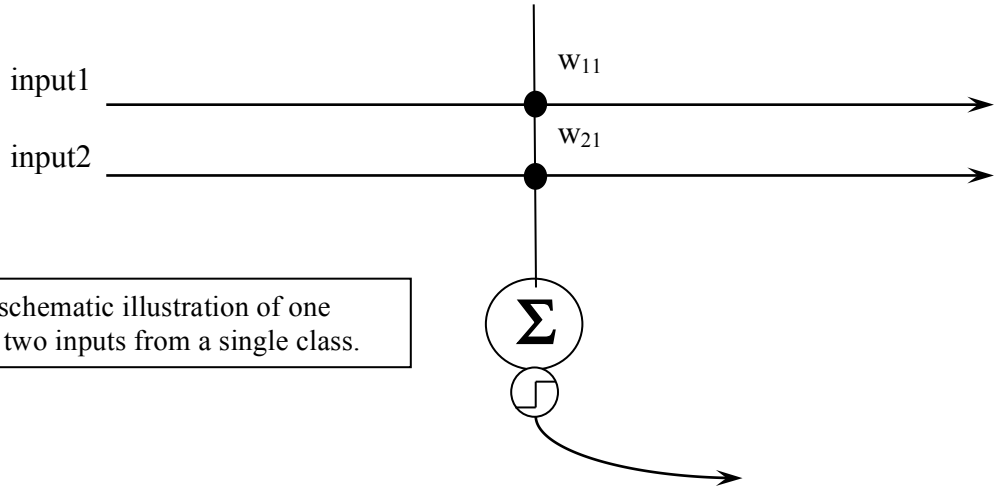


Fig. 4.3. A schematic illustration of one neuron with two inputs from a single class.

As in the case of all synaptic modification rules, we start with the general statement

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t). \tag{4.4}$$

And then define the specific rule

Δw = (postsynaptic excitation) g(presynaptic excitation − current weight)

As before we let $x_i(t)$ stand for the activity of input *i* at time *t*. This time, however, the inputs being modified supply their own postsynaptic excitation. Thus, we must define this postsynaptic excitation, and the simple linear relationship will do:

$$y_j(t) = \sum_i x_i(t) w_{ij}(t). \quad (4.5)$$

Substituting into (4.4) gives the weight modification rule as

$$\Delta w_{ij}(t) = \varepsilon y_j(t)\left(x_i(t) - w_{ij}(t)\right), \quad (4.6)$$

where we note that y_j is multiplying a moving average-like process.

Homework 4.1.2.

In this exercise, begin with one postsynaptic neuron and two inputs and a self-supervised modification rule. Use one of the three input sets used in Environment 2 of homework exercise 2.7.2. from Laboratory 2 as the input to a single self-supervised neuron.

In this exercise, you need to show that, with sufficient training on the input set, the starting point (i.e., the initialization values of the synaptic weights) is irrelevant to the final values of the weights. Show that you can start the synaptic weights anywhere (you can initialize them to random starting values), and they will converge to the same final values. Note that you'll need to cycle through the input set many times here, just as you needed many samples in the moving averager. Of course, the actual number of samples you need is a function of your rate constant, ε. You can just keep resampling the same data set to obtain the many samples you need.

Initialize the weights and training for at least four different initialization values, all within the first quadrant. Illustrate the developing changes in the neuron's synaptic weight vector. You will need to save the successive updatings of the synaptic weights in order to plot this set of updates (Fig. 4.4B shows such a plot; it is called a state-space diagram).

Comment on where the synaptic weights converge relative to the input patterns in the training set and relative to the initialization values of the weights. What you should observe is the result of a modification process that is *globally, asymptotically convergent* on average. Global means the starting point does not matter. Asymptotic convergence means that the values only approach a final value. How do your observations fit (or not fit) with this phrase?

A B

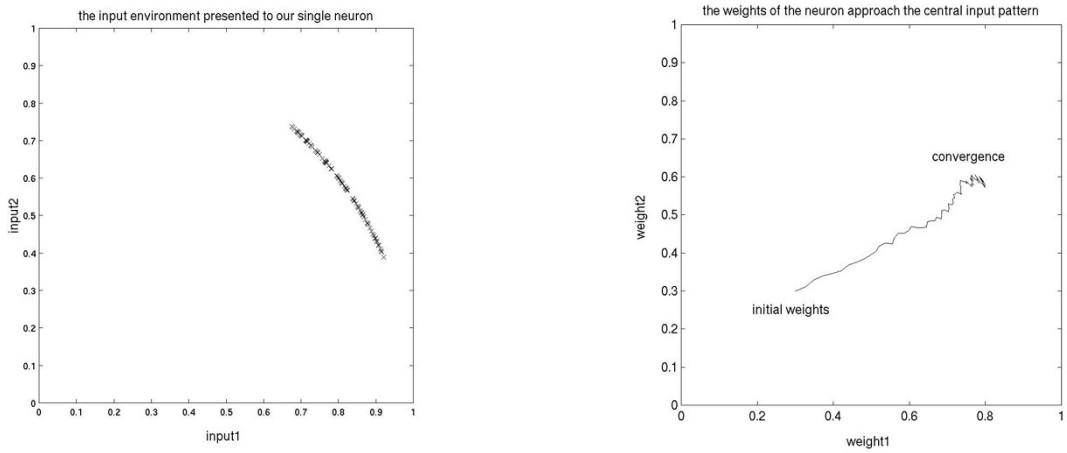


Fig. 4.4. A state space diagram of a weight vector over time (right). Here the two synaptic weights approach the central pattern of the input set (left) despite the wide spread of input patterns. The weights of the neuron approach the mean of the input set.

Homework 4.1.3.

Use Environment 1 of homework exercise 2.7.2. and combine the two input sets to create your input set for this exercise.

To have a clearer understanding of how a weight vector modifies when its inputs are presented with such a mixture of two sets, it will be necessary to intersperse your sequence of inputs that are drawn from each of the two pattern sets. First present a single input from data set one, then one from data set two and just continue alternating back and forth (the `mod` command in MatLab can be used to determine which time-steps are even or odd, to intersperse the data sets).

First, be sure to plot the training data points. Repeat exercise 4.1.2 and examine where the synaptic weight vectors converge as a function of starting point. Do you think the current version of the self-supervised synaptic modification rule can be used to learn to recognize just one of these two input sets? Why or why not?

Comment on your assessment of the self-supervised synaptic modification rule based on what you observe for these two input sets. Compare the self-supervised rules to the category-supervised synaptic modification rule in terms of feature-based category recognition.

As part of your discussion, compare equations (4.6) and (4.2), noting similarities and differences.

Next week we will study and fix the problem with the linear self-supervised synaptic modification rule.

Include in your lab report the following:

1. Input generating function
2. Plot the inputs.

Homework 4.1.4.

Rule 2 guarantees weights on the unit circle. Compare this rule and the simpler one already described. Show that they produce weights that point in the same direction. Then compare the Euclidean length produced by the two rules. To answer these questions, use a three (or higher) dimensional input.

Compare the three self-supervised modification rules.

1. $\Delta w_{ij} = \varepsilon y_j \left(x_i - w_{ij} \right)$
2. $\Delta w_{ij} = \varepsilon y_j \left(x_i - w_{ij} y_j \right)$
3. $\Delta w_{ij} = \varepsilon y_j \left(x_i - w_{ij} \left(1^T w_j \right) \right)$

where 1 is a column vector of all ones, y_j has its usual definition, w_j is *j's* weight vector, and $x_i \in [0,1]$.

Use a three-dimensional input to create an example showing that Rule 2 is self-normalizing. That is, describe your data set and what the weights converge to. Calculate the length of the weight vector. Now double all the values of the input set. What happens?

Laboratory 4, Part 2: Concept formation and synaptic modification (continued)

In the second part of the lab we show an example in which the self-supervised rule can perform quite poorly. Part 2 has itself two parts. First, we work through in class exercises that help explain what the unsupervised rule makes a postsynaptic neuron learn and why this particular _____ is learned. Then we examine the benefit of a nonlinear postsynaptic term in the self-supervised synaptic modification rule.

Section 4.2.1. Where and why do self-supervised weight modifications converge? A brief introduction to eigenvectors

Towards a deeper understanding of what self-supervised synaptic modification is encoding

A note to the instructor: Perform this section as an in class exercise. This section requires a lecture on eigenvalues and eigenvectors.

Form a 4-by-4 random matrix of positive values.

```
>>rmatrix=rand(4,4)
```

Using this matrix, make a new matrix that is symmetric.

```
>>symmatrix=rmatrix*rmatrix'
```

Allow MatLab to note some observations about your symmetric matrix.

```
>>tracesymmat=trace(symmatrix)
>>determinantsymmat=det(symmatrix)
>>[eigvectors,eigvaluesdiag]=eig(symmatrix)
```

Now compare the trace and determinant of the matrix `eigvaluesdiag` to the earlier ones using
the `trace()` and `det()` commands in MatLab.

```
>>trace(eigvaluesdiag)-tracesymmat
>>det(eigvaluesdiag)-determinantsymmat
```

Now recover the original symmetric matrix from the eigvectors and eigvalues

```
>>eigvectors*eigvaluesdiag*eigvectors'-symmatrix
```

Now check to see if one eigenvector is normalized

```
>>eigvectors(1)*eigvectors(1)'
```

Now check to see if two eigenvectors are orthogonal

```
>>eigvectors(1)*eigvectors(2)'
```

Now do both checks across all eigenvectors

```
>>eigvectors*eigvectors'
```

Record your observations.

Lord Rayliegh's power method

Here is an algorithm that is related to self-supervised modification.

```
>>format long
>>oldvector=ones(4,1)
            %initialize vector
>>for timestep=1:20
            %recursion
      newvector=symmatrix*oldvector;
      oldvector=newvector/norm(newvector)
  end
            %end recursion
```

Note how quickly convergence occurs. Repeat the loop and graph successive values.
Compare the vector formed to the eigenvectors of symmatrix. Scale and subtract this eigenvector—that is, create a new matrix.

```
newmatrix = symmatrix-eigvaluesdiag(1,1) *eigvectors(:,4)*eigvectors(:,4)'
```

Now repeat the recursive algorithm on the new matrix. How accurate is this method?

Compare the converged vector to the value found using the MatLab function eig( ).

A hint about the convergence

Note that a fundamental property of eigenvectors is their orthogonal relationships. That is, all the eigenvectors are at right angle to each other—or, to put it another way, the cosine of angle between any two eigenvectors is zero. Now consider what happens as a weight vector moves towards the dominant eigenvector—it must also be moving away from all the other eigenvectors. Thus, on average, these other eigenvectors, which live implicitly in the correlation matrix, exert a smaller and smaller pull on the evolving synaptic weight vector as successive modifications are made.

Homework 4.2.1. Unit circle weights

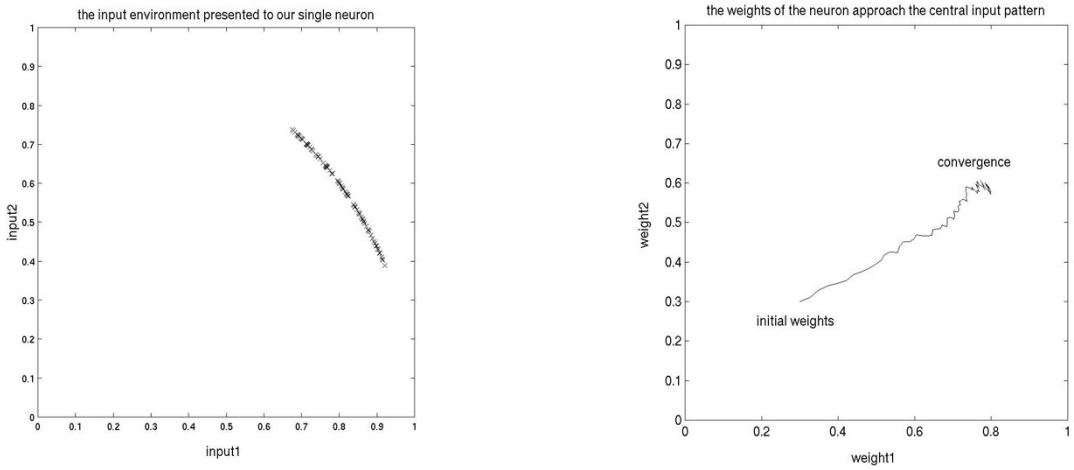
Optional exercise

As in the previous homework section, us the input set to demonstrate that all these rules produce a weight vector that aligns with the dominant eigenvector of the correlation matrix XX^T where X is the set of input (column) vectors.

Using an input set from your investigations of an unsupervised synaptic modification rule, use MatLab to find the eigenvectors of XX^T where X is the set of column vectors, each of which is an input vector. Compare the eigenvector associated with the largest eigenvalue to the weights encoded by an unsupervised modification rule.

Section 4.2.2

Now we will look at some state space graphs of the weight modification of self-supervised neurons. As a reminder of what you did in Part 1, Figures 4.5 and 4.6 show a single neuron with two inputs trained on a single pattern set. Figures 4.7 and 4.9 look at cases in which a self-supervised neuron is trained on two pattern sets.



Figs. 4.5 and 4.6. A state space diagram (Figure 4.6 on the right) in which the two synaptic weights approach the central pattern of the input set (Figure 4.5 on the left) despite the wide spread of input patterns. The weights of the neuron approach the mean of the input set.

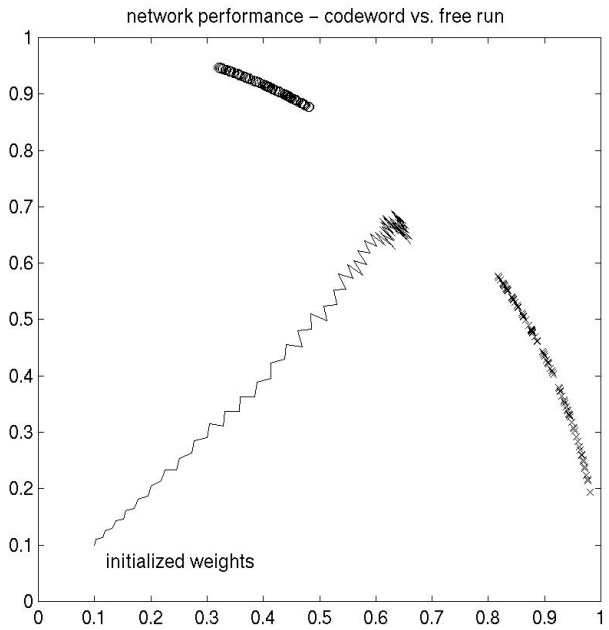


Fig. 4.7. A single linear self-supervised neuron trained on two input sets. The neuron converges towards a part of the input space that does not contain any input patterns. This is analogous to seeing a cat and a dog, and instead of storing their separate features, coming away with a memory of some theory in between (a cog? or is it a dat?).

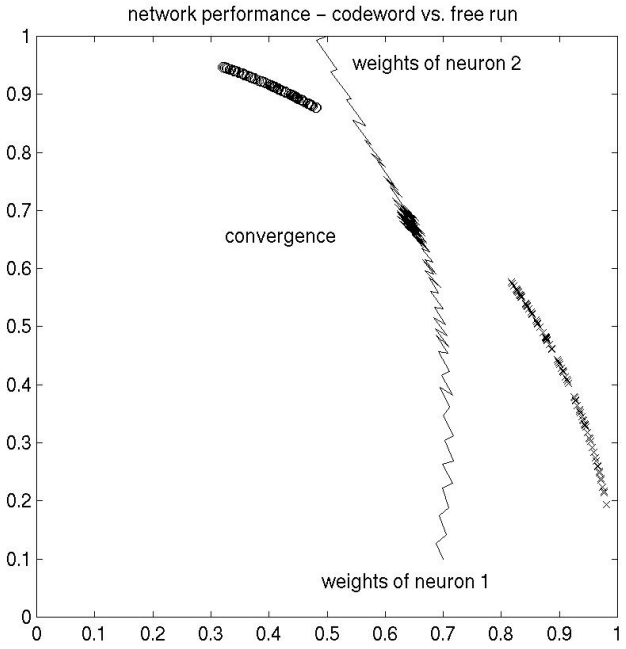


Fig. 4.8. Two different linear self-supervised neurons trained on a symmetric mixture of two data sets. Each weight vector converges to the same place even though initialization values differ.

A self-supervised neuron is teacherless. This presents some problems—suppose we try to train a self-supervised neuron with two distinct data sets, and suppose the neuron is trained first on all samples of one set then trained on all samples of the other set. In this situation, the weight vector will first approach one pattern set, then move towards the other. On the other hand, if sampling alternates between two non-overlapping pattern sets, then the weight vector goes somewhere between the two sets.

Section 4.2.3. Nonlinear self-supervised synaptic modification

In the exercises in Part 1, you discovered that it is difficult for a single neuron to learn more than one input pattern set using the self-supervised synaptic modification rule. *Expand on the reasons for this in your laboratory write-up.*

You can improve this situation by introducing a nonlinearity into the postsynaptic term of the synaptic modification rule. With this nonlinearity, excitation below a certain level is ignored by the postsynaptic neuron. If we rewrite the self-supervised modification rule and replace postsynaptic excitation y_j with a function $f\left(y_j\right)$, where the function is nonnegative and nondecreasing in y_j, then

$\Delta w(t) = \varepsilon f(y(t))*(x(t) - w(t))$.

Fig. 4.8 shows two such functions.

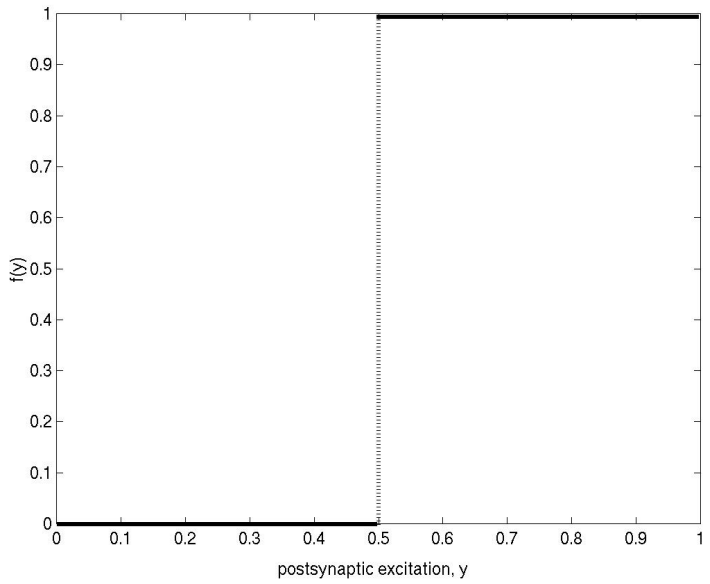


Fig. 4.9A. A binary postsynaptic excitation rule. Instead of the linear postsynaptic excitation we have been using, consider a synaptic modification rule that thresholds this excitation, as zero or one. The postsynaptic term of the synaptic modification equation (ordinate) goes to zero if excitation falls below a certain threshold, and is one if the threshold is exceeded. The threshold illustrated is 0.5 but any other value from zero to one could be selected.

Exercise: COULD NOT READ PAGE 25 of comparison doc

Use the binary thresholded synaptic modification rule

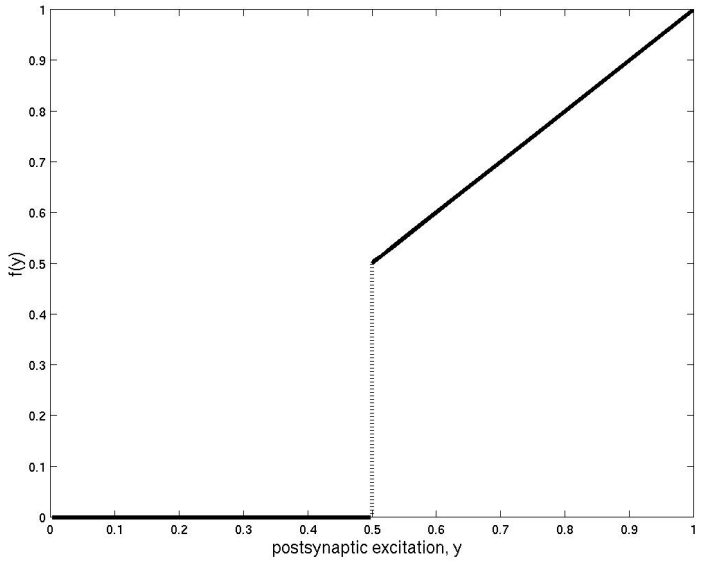


Fig. 4.9B. Another nonlinear excitation function for the self-supervised neuron. This postsynaptic term of the synaptic modification equation (ordinate) goes to zero if linear excitation (abscissa) falls below a threshold (in this case the threshold is 0.5). The postsynaptic excitation function might be called threshold-linear because once threshold is reached, the postsynaptic excitation term in the synaptic modification equation increases linearly with *y*.

A threshold-based nonlinear excitation function is easy to implement in MatLab. Insert an 'if' statement into your code after the postsynaptic excitation is calculated and before the weight change is calculated. The choice of the cutoff point below which excitation is zero will be determined by your input sets and your initial weights. For a nonlinear, self-supervised neuron, a difference of 0.05 in the cutoff point will sometimes be the difference between a successful simulation and a failed one (note that we take advantage of MatLab's facility for vector-based operations in calculating the pre-thresholded postsynaptic excitation of the neuron.)

Homework 4.2.2.

A. Take two distinct data sets on the unit circle and compare the final weights when you use the binary rule and the threshold linear rule.

```
>>for i = 1:99
      output(i) = (input(:,i)' * weights(:,i)); %run 99 cycles of the input
      if output(i) <= 0.8     %set threshold at 0.8 by zeroing
            output(i) = 0;        %values less than 0.8.
      end
                                  %...modify weights here as %before...
  end
```

B. Using old data sets, compare and contrast the ability of a pair (or three) of category-supervised neurons versus thresholded self-supervised neurons to discriminate feature sets. Be sure to consider what happens with feature sets overlap versus when there is no overlap.

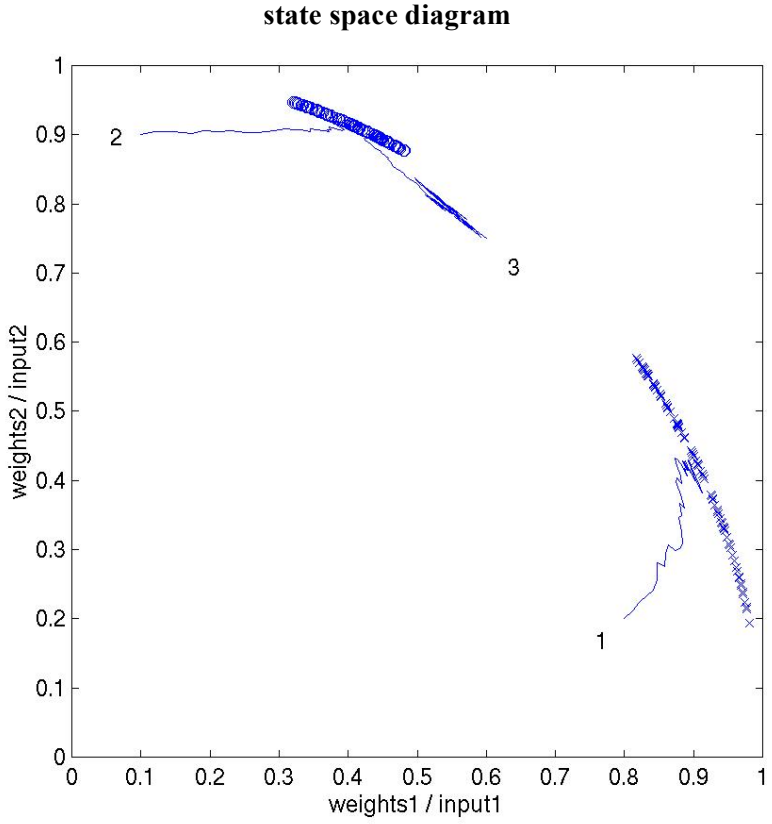


Fig. 4.10. A self-supervised neuron with a nonlinear postsynaptic excitation function can selectively learn one of two pattern sets. Such neurons are sensitive to initial weights and to the threshold of the postsynaptic excitation function. Indeed, the neurons can be very sensitive to these parameters, so set your initial parameters carefully.

Hints for setting the threshold linear rule

There are two failure modes for these thresholded, unsupervised modification rules. The threshold might be set too high and nothing happens or the threshold might be set too low. When the threshold is too high, relative to the initial weight value, $\Delta w_{ij}(t)$ will be zero for all *ij* and for all *t*. When the threshold is too low, then the neuron will fail to discriminate between the close-by and nonoverlapping data sets.

Homework 4.2.4.

For the following, use two non-overlapping data sets as in Fig 4.10.
Suggested data generation: Be sure to confirm there is no overlap.

Nonlinear rules to examine

- Binary (Fig 4.9A)
- threshold linear (Fig 4.9B)
- a nonlinear rule with a smoothly increasing postsynaptic function $f(y)$. (suggestions given in class)

Suppose there are two distinct states of the world, call them (a) and (b). Each state generates samples in a probabilistic fashion. Show how non-linear self-supervised neurons can create a stable weight vector which depends on the initial value of w. Explain why dependence on initial value might allow the formation of postsynaptic neurons that recognize different pattern sets. Under what conditions will your network fail in the sense that two neurons learn the same set of patterns?

Laboratory 4, Part 3: Error Correction

Section 4.3.1. Error-corrected synaptic modification

There is a third form of associative synaptic that is clearly distinct from the first two covered in this chapter. All the forms of synaptic modification can be called associative and are an outcome of the submicroscopic multiplication between a single presynaptic and a single postsynaptic term. However, the nature of the postsynaptic term in an error-corrected synaptic modification rule is quite different than the functions used for category-supervised and self-supervised synaptic modification rules. For all three rules, the multiplication is synaptically localized so that the information of individual input states $x_{ij}(t)$ can be correlated with some function of the postsynaptic neuron *j*. That is, the expectation $E\left[x_i \cdot f\left(y_j\right)\right]$ is implicit in all three synaptic encodings. However, in contrast to the category-supervised and self-supervised synaptic modification rule, the postsynaptic scalar variable $y_j(t)$ requires another set of neurons; this other set calculates the error of the most recent response and communicates this error to to neuron *j*. Specifically, this remotely computed, scalar message is an error signal. In the brain there are at there are two primary places that use error-corrected synaptic modification: the cerebellum and the basal ganglia. In the case of the cerebellum, errors are computed by the inferior olive and lateral reticular nucleus. The error signal is transmitted by axons from these regions that have a special name, climbing fibers (because of the way they make synapses that climb up the dendrite of each purkinje cell of the cerebellar cortex).

For linear systems with Gaussian noise, the optimal form of prediction is a linear regression, and the error-corrected modification rule provides information to do just that. That is, the synaptic weights converge to values that minimize the variance of the error signal.

The general form of this synaptic modification rule is $\Delta w_{ij} \propto x_i \cdot error_j$ where (i) x_i is the state of the i^{th} input and $error_j$ is the difference between the desired result (which is influenced, if not controlled by neuron *j* ,and thus, implicitly predicted by *j*'s output) and (ii) the outcome that has actually occurred. This error calculation requires combining information that arises at different times. Such time differences require careful implementation of synaptic modification algorithm.

(In fact, similar delays exist in the other two classes of synaptic modification rules but have been ignored. Here, eventually, a major part of the exercises is to get the delay right while taking into account physical delays introduced by the input and output circuitry.)

Prediction using linear regression is ideal of for Gaussian variables that combine in a linear fashion. Moreover, if the prediction problem is limited to interpolation between earlier correlations and if the events being interpolated are close enough together, than nonlinear regression can be performed with good results by patching together a sequence of these linear regressions. (It is notable that all predictions, taken at their full dimension, are either interpolation or extrapolation because no event every repeats exactly in the same way.)

In statistics, the relationship between continuously valued variables is often expressed by a regression curve. When this is the case and because it is continuous, interpolation and extrapolation are straightforward. Thus, from a finite number of sample points, we can predict an infinite number of possible values. Suppose we have two variables x and y, then the general linear relationship between them is $y = ax + m$ where a and m are parameters that must be discovered by sampling. We can plot sample values and might get something like,

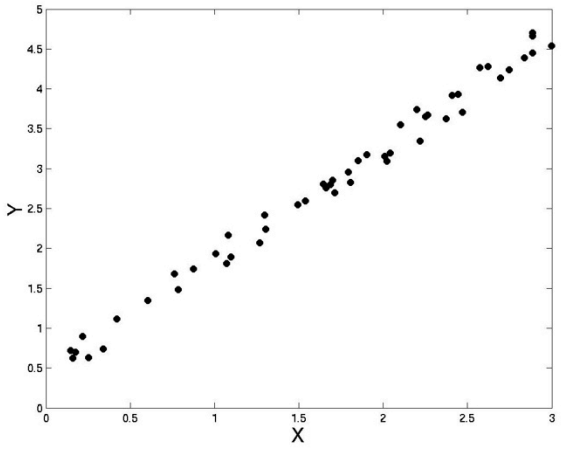


Fig. 4.11. Scalar diagram. The data points will fit by a straight line

where each point is a particular pair of values of the two variables. The method of linear regression allows us to draw the optimal relationship between the two variables (this optimization assumes a Euclidean measure of error, in addition to assuming the linear relationship).

The optimal line minimizes the sum of the Euclidean distances of all the data points to the line where the distances are perpendicular lines from each point to the regression line. This idea can be extended to a multidimensional vector X predicting a scalar y. For us, this means using a vector input to a neuron to predict the neuron's best excitation, where best minimizes the average squared error.

The error-correcting modification rule creates synaptic weights that produce a prediction based on a *linear regression* of the inputs. It does so adaptively—that is, it uses each new sample as it arrives and adjusts the weight after getting, with a small delay, the feedback error signal.

Error-Corrected Synaptic Modification Depends on the Neuron Being Modified

We will use the associative error-correction mechanism with two different postsynaptic neurons—an excitatory neuron and an inhibitory neuron—and this pair of neurons will have opposite effects on the response of interest. For both types of neurons, the input synapses are excitatory; in fact, these inputs are identical, just axonal branches of the same neuron. Because the inputs are the same and because error-correcting synaptic modification depends on the output

commanded by the postsynaptic neuron, the sign of the synaptic modification equation depends on the sign of the output neuron's effect. Thus, you will need two forms of error-correction equations at the end of this section when you simulate learning mediated by the cerebellar cortex. These two forms only differ by a sign, plus versus minus.

There is another innovation required in building toward this cerebellar model; we must conjecture a different neural code for category membership. While we still require no more than a binary decision—category-absent versus category-present—instead of the {0,1} alphabet we have used in the pyramidal neuron model, we will use the binary alphabet {-1,1} for the Purkinje cell of the cerebellum in our exercises (in a more realistic setting, a continuous output range, [-1,+1] should be used). A possible biological justification or even a biological mechanism corresponding to the distinction of binary states hinges on the spontaneous activity of each neuron type. In the absence of any synaptic input, the activity of cortical pyramidal cells is in the range of 5-10 Hz in monkeys while, in the absence of granule cell excitation the activity of Purkinje cells is far from zero; it is in the range of 50-100 Hz. By virtue of such spontaneous activity, the Purkinje cell does a pretty good job of signaling both net increases and net decreases of its summed excitatory and inhibitory inputs. (An even better job might be accomplished via complementary coding (i.e., opposing Purkinje cells that would correspond to opposing behavioral reactions which works with the idea of opponent muscle pairs), but this will not be part of the regular exercises.)

In terms of solving a linear regression problem, the data and states of a categorization problem is somewhat simpler than the picture used to illustrate linear regression. That is, for didactic purposes and for the purposes of the classical conditioning paradigm that is actually used in psychology experiments, we restrict data to two extremes and ignore the range in between. (One can guess that Nature, in contrast to the experimental psychologist, uses the full range in day-to-day operations that are on-going computations in the cerebellum and in the striatum.)

Fig. X.a illustrates the kind of correlated data that might occur for a neuron that communicates category membership via excitation

Fig. Legend: Correlating net postsynaptic excitation with categorization depends on the sign of a neuron's output. In both figures, the postsynaptic neuron is presumed to have a spontaneous activity level, but we shift the sign as we shift the output values by subtraction of this spontaneous firing rate. As a result of this shifted perspective, when there is zero net excitation, both the excitatory postsynaptic neuron (A) and the inhibitory postsynaptic neuron (B) produce a zero output (but what is it actually?). When a member of the category is present so that it activates the appropriate input, the excitatory neuron's firing speeds up while the inhibitory neuron slows down. On the other hand, when a non-category member excites the system, the excitatory neuron becomes quiescent because the inhibitory neuron speeds up its firing and the excitatory inputs to the primary cell are too weak to excite firing. (end of figure legend)

ERROR

prediction \ category	0	1
0	0	+1
1	-1	0

Δw by postsynaptic type

excitatory

prediction \ category	0	1
0	0	$+x_i$
1	$-x_i$	0

inhibitory

prediction \ category	0	1
0	0	$-x_i$
1	$+x_i$	0

the synapse is excitatory for either postsynaptic type

The sign (plus or minus) of synaptic modification depends on both 1) the sign of the error and 2) the role played by the postsynaptic neuron in the behavior (excitatory vs. inhibitory for the behavior).
In all cases, define error as

error:= *desired – observed*.

Excitatory case. Neuron *j* is excitatory for the behavior:
$\Delta w_{ij}(t+1) = \varepsilon x_i(t-\Delta) error_j(t,t+1), \quad \varepsilon > 0$

For a positive error, the response was too small, but the response will be improved if j fires more.
For a negative error, the response was too big, but the response will be improved if *j* fires less.

Inhibitory case. Neuron *j* is inhibitory for the behavior so everything must be reversed:
$\Delta w_{ij}(t+1) = -\varepsilon x_i(t-\Delta) error_j(t,t+1), \quad \varepsilon > 0$

For a positive error, the response was too small, but the response will be improved if *j* fires less.
For a negative error, the response was too big, but the response will be improved if *j* fires more.

For the neuron whose eventual net effect on prediction is excitatory (e.g., contract the "blink muscles" to block the US is a prediction of the US), we posit a straightforward relationship: excitation of this neuron's inputs leads to faster firing, and this faster-firing signals that the category is present. On the other hand, the inhibitory neuron whose effect on prediction is inhibitory (e.g., relax blink muscles), we require a net inhibitory signal relative to its resting input when the category that this neuron looks for is present; that is, the neuron that is inhibitory for the behavior signals "category present" by slowing down. The presence of another related but different and distinguishable category is signaled by the opposite responses. For example, in the case of the neuron that is inhibitory for the behavior, the presence of another category excites the postsynaptic neuron with the net result of inhibiting the associated behaviorally-mediated decision.

In terms of the synaptic modification equations, we again concern ourselves with the neurobiologically dominant excitatory inputs to each such postsynaptic neuron. As always, synapses are constructed as a pair (*i*,*j*) where $x_i(t-\Delta)$ is the presynaptic activity of input i to postsynaptic neuron *j* at trial *t* and updating is generically $w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t)$. The error on trial *t* for postsynaptic neuron *j* is $error_j(t)$ where error is defined as (*desired – actual*). (Note that we have explicitly offset the presynaptic activity, Δ is positive), but often we will implicitly absorbed the biophysical delay that must occur because of the delay between presynaptic activation x_i and the arrival of the error signal.)

For the postsynaptic neuron *j* that is excitatory for the response, the synaptic modification is $\Delta w_{ij}(t) = \varepsilon x_i(t-\Delta)\left(error_j(t)\right)$. That is, if the error value is positive (where a positive error is the failure to identify a category that was actually present), then the synapse (*i*,*j*) strengthens in

proportion to its presynaptic activity x_i. By implication then, the more the presynaptic input would have improved the output response for a positive error, the more the synaptic modification.

Note how we define the sign of the error in the categorical situation: a positively valued error corresponds to the association of the retrodictive signal of *event-absent* and the prediction of *event-present*. Likewise, if an event is absent but the postsynaptic neuron predicts event-present, the error signal is negatively valued and an excitatory synapse that contributed to the incorrect decision is downgraded in synaptic strength.

For the case of a neuron whose excitation inhibits the behavioral decision prediction, the appropriate modification equation must multiply the previous equation by -1, thus yielding $\Delta w_{ij}(t) = -\varepsilon x_i(t-\Delta)\big(error_j(t)\big)$ where we maintain the same error convention but must insert a negative sign into the Δw equation. Thus, if excitatory input *i* was active and the postsynaptic *j* did not slow down its firing, then synapse (*i*,*j*) will be weakened. Importantly, in both cases, no modification occurs at synapse (*i*,*j*) when the presynaptic input *i* is inactive (i.e., zero). Thus, we describe this rule as presynaptically amplified (or presynaptically controlled in the binary, zero-one case); note that the two earlier synaptic modification rules of this chapter are postsynaptically amplified (or postsynaptically controlled in the binary, zero-one case).

Suppose there is some event in the environment, or even part of an organism, that we want a neuron to predict. For example, suppose a neuron in the cerebellum must predict the position of a leg given a particular movement is commanded. A *feedback-supervised network*, or error-corrected network, calculates the difference between the activation of a neuron and a signal that comes back carrying the error of positioning. This error is then used to adjust particular synaptic weights. Only synapses that were active are adjusted. The neuron is then able to modify its synapses to produce the best fit, in the sense of least mean square error, for producing the correct output. This is equivalent to the neuron calculating the linear regression line for the input set that best predicts the stimulus in the environment being predicted.

Let's call the output of excitatory postsynaptic neuron *j*, o_j, and the desired (or correct) answer d_j. Then the feedback-supervised rule is

$$\Delta w_{ij}(t+1) = \varepsilon x_i(t-\Delta)\big(d_j(t) - o_j(t+1)\big),$$

and where the error signal must return to the postsynaptic neuron and its dendrites before the memory of recent synaptic activation decays away. Rewriting with time implicit,

$$\Delta w_{ij} = \varepsilon x_i\big(d_j - o_j\big).$$

We call this rule feedback supervised because c_j must await the output o_j.

There are two major similarities between the feedback-supervised modification rule and the first two you have studied. These similarities are: 1) the inclusion of a modification rate constant ε and (2) the presence of the product of presynaptic multiplied by postsynaptic.

There is, however, one important difference. Specifically, the parenthetical term differs from the running averager-like process $\left(x_i - w_{ij}\right)$ that is part of the category supervised and self-supervised modification rules. Here the term $\left(d_j - o_j\right)$ is the error in the output of the postsynaptic neuron *j*. Given this difference, we can think of this rule as

$\Delta w_{ij} = \varepsilon \cdot presynaptic_i \left(error\ of\ output_j\right)$.

Implementing this rule with a McCulloch-Pitts neuron

There are two cases of immediate computational interest. The first is a McCulloch-Pitts neuron where $y_j = \sum_i x_i w_{ij}$ and $o_j \in \{0,1\}$ after specifying a threshold value of y_j that fires neuron *j*.

$$o_j = \begin{cases} 0 \text{ if } y_j < \text{ threshold otherwise} \\ 1 \text{ (that is when } y_j \geq \text{ threshold)} \end{cases}$$

This binary output makes the neuron a categorizer (absent/present).

The second, and better case is the continuum output version where, in the simplest scenario, $o_j = y_j = \sum_i x_i w_{ij}$. Of course, we could make o_j equal to any increasing monotonic function, $f\left(y_j\right) = o_j$. But when $o_j = y_j$, the neuron *j* computes the linear regression of the variable x_i against *o*.

Suppose the presynaptic input takes on a positive value, and then consider the following three cases:

$o_j = c_j$
o_j is too big, i.e. $c_j - o_j < 0$
o_j is too small, i.e. $c_j - o_j > 0$

When $o_j = d_j$, the error is zero, implying that $\Delta w_{ij} = 0$ so no change occurs.

When $d_j > o_j$, the error is positive (o_j was too small) and $\Delta w_{ij} > 0$. The next time this situation occurs, the input x_i will tend to make a larger contribution.

When $d_j < o_j$, the error term is negative and $\Delta w_{ij} < 0$. The next time this situation occurs, the input x_i will tend to make a smaller contribution.

When the error-corrector is best:

Suppose the scalar event $y(t)$ (think of this as postsynaptic excitation) is related to the vector event $x(t)$ and noise $\varepsilon(t)$ in a linear fashion with no offset. That is

$y(t) = \sum a_i x_i(t) + \varepsilon(t)$

where ε is independent of the x_i and can be positive or negative but on average $E[\varepsilon] = 0$ with variance

$Var(\varepsilon) = E[\varepsilon^2] > 0.$

Suppose $y(t)$ is predicted by

$\hat{y}(t) = \sum b_i x_i$

then we may define the error of such predictions as

$error(t) = y(t) - \hat{y}(t)$ and

It would be good to minimize $E[(y - \hat{y})^2]$

That is, choose the vector *b* that minimizes

$$E[(y-\hat{y})^2] = E\left[\sum (a_i x_i + \varepsilon - \sum (b_i x_i))^2\right] = E\left[\left(\sum (a_i - b_i) x_i\right)^2\right] + E[\varepsilon^2] + 2E\left[\varepsilon \sum (a_i - b_i) x_i\right]$$

But this last term is zero because noise is independent of the x_i and average value of zero

That is,

$E[\varepsilon x_i] = E[\varepsilon] E[x_i] = 0 \cdot E[x_i] = 0.$

Moreover $E[\varepsilon^2]$ is fixed, so the only term of interest is

$E\left[\left(\sum (a_i - b_i) x_i\right)^2\right]$ which is minimized to zero when each and every $b_i = a_i$.

Thus the error-correction synaptic modification rule will solve the minimization problem if it matches the b_i to the a_i. That is, we recognize b_i as the optimal value of synaptic weight a_i.

Homework 4.3.1.

Compare the error correction to the category-supervised and the unsupervised synaptic modification rules. Using your data sets from Lab 2, compare the learned synaptic weights for the three different synaptic modification rules: category supervised, self-supervised, error-corrected—use the net excitation as the output and convertthe {-1,+1} output to a {0,1} output in zero and one (don't worry if the error correction sends some weights negative). The data sets are:

2 patterns with no overlap
2 patterns with overlap
3 patterns with no overlap

Hint: *Set* $C_j = 1$ *when category present and* $C_j = 0$ *when category absent. Sample as in the category-supervised problem.*

Comment on what is learned and how the neurons perform as a pattern recognition device for each rule. Be sure to present evidence that synaptic weights have nearly converged. In addition, be sure to address the following points:

- Explain your selection of modification rate constant(s).
- How did you decide where to set threshold?
- In what sense is a neuron with synaptic modification capable of abstraction?
- In what sense does it generalize?
- What are the relative advantages and disadvantages of the different synaptic modification rules?

Section 4.3.2. The Pavlovian delay-conditioning paradigm

In this section we will study a network that learns a time interval. This learning problem implies that we take into account the delays between neuronal communications. We will first describe the learning problem then discuss timing.

Conditioning is what psychologists and physiologists decided to call animal training methods when they brought such methods into the laboratory to study learning and memory. As is the case for most scientific ventures, simpler versions of conditioning have always been attractive targets to study because they are often the easiest to understand. Early on, there was also the hope that once a simple conditioning paradigm was understood, the knowledge would generalize across all forms of associative conditioning. Unfortunately, the hope that all associative conditioning paradigms would have the same underlying neuroscience is no longer viable.

Different brain regions have different circuitry. Thus, one must discover the brain region(s) that stores the memory and then discover the underlying neuroscience of that region. This includes

the brain region, cell types, and synapses involved. Of particular interest is the altered physiology, which inevitably, although not exclusively, involves synapses, where synaptic modification can be a change in the strength of an existing synapse, the shedding of a synapse, or even the acquisition of a new synapse. These alterations critically mediate the learned behavior.

Perhaps the simplest and best understood form of associative learning is the Pavlovian paradigm called 'classical conditioning.' In his experiment, Pavlov studied the salivation of a dog upon feeding. Each day, the same man would enter the dog's room, walk over to his cage, and present him with his food. At first, the dog began to salivate only after smelling and tasting the food, a normal and instinctive physiological response. However, after several days of this procedure, the dog would begin to salivate the moment the man entered the room, even before being presented with food. Thus it can be presumed that somehow the dog was able to associate the entrance of the man into the room with the presentation of food moments later. This "prediction", so to speak, lies at the heart of classical conditioning.

In this paradigm, we shall define two types of stimuli. The conditioned stimulus (CS) is the man walking through the door, whereas the unconditioned stimulus (US) is the dog smelling and tasting its food. What's important to understand is that the most important difference between the US (meat) and the CS (the human) is their effect on the chosen behavioral response before any conditioning has occurred. This behavioral response, e.g., salivation, is said to have two forms: (1) the reflexive, unconditioned response (UR), i.e., normal salivation in response to food which occurs before any training, and (2) the learned conditioned response (CR), i.e., learned salivation in response to seeing the man in charge of feeding. It is also important to note that there are two key time scales that must be examined. The first describes the successive trials, or days, over which the dog is fed and slowly conditioned, while the other describes the moment-by-moment events as the man enters the room, walks to the cage, and feeds the dog, over the course of each trial. Let's consider a popular laboratory example, the eyeblink reflex.

Eyeblink Conditioning

In 'eyeblink conditioning,' a gentle air-puff to the eye acts as the US that results in activation of the blinking reflex, the UR. Typically, the CS is a brief tone that in 'delay conditioning' (the version of classical conditioning that we will be examining), will be played before the US and remain on until its onset. In most cases, both the CS and the US will co-terminate (a visual representation of this sequence is shown in Figure 4.12). By specifying the timing between the tone and the airpuff, we can create a paradigm in which the CS can be used to predict the US. In a typical example, the CS might turn on 250 msec before the US, which then lasts for 100 msec. Training trials could be administered randomly with an average time between trials of 30 sec. After enough of these associative training trials, the eyeblink evoked by the CS occurs before the eyeblink evoked by the US. If the animal has been successfully conditioned, this early response can be even obtained by activation of the CS alone. Thus, it is fair to say that the CS can and is being used to predict the US. That is, the temporal correlation between the CS and US allows the organism to anticipate the US; this anticipation takes the form of the CR blink, which reduces the impact of the airpuff when properly timed.

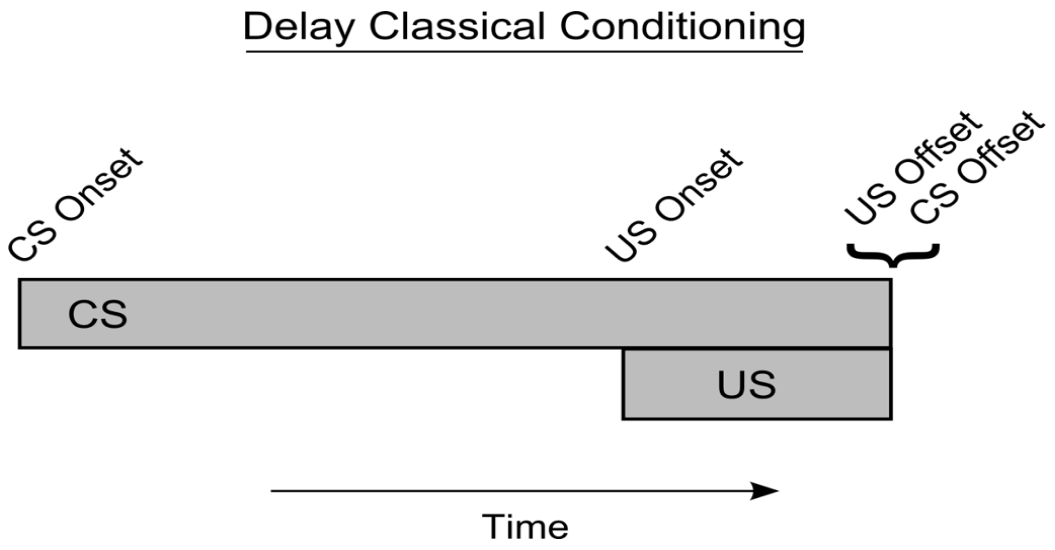


Fig 4.12. A typical delay paradigm. The CS onset precedes and is continuously on before the US onset. Often the CS remains on until US termination.

In every associative learning paradigm, we must run a variety of control tests to measure the effects of (1) repetition of either stimulus alone, (2) unreliable pairing of stimuli (random, temporally uncoupled CS and US presentations, called pseudoconditioning), or (3) temporally inappropriate pairings (e.g., US onset preceding CS onset). These tests should not result in the same degree of conditioning that we expect to see in properly timed eyeblink models. There is also a stability requirement once the learned behavior is established. That is, once associative learning is established, the mere passing of time should not lead to erosion of the CS-evoked CR. However, presentations of the CS alone will eventually lead to deterioration over time, called extinction. These requirements are enumerated in Table 4.1 below.

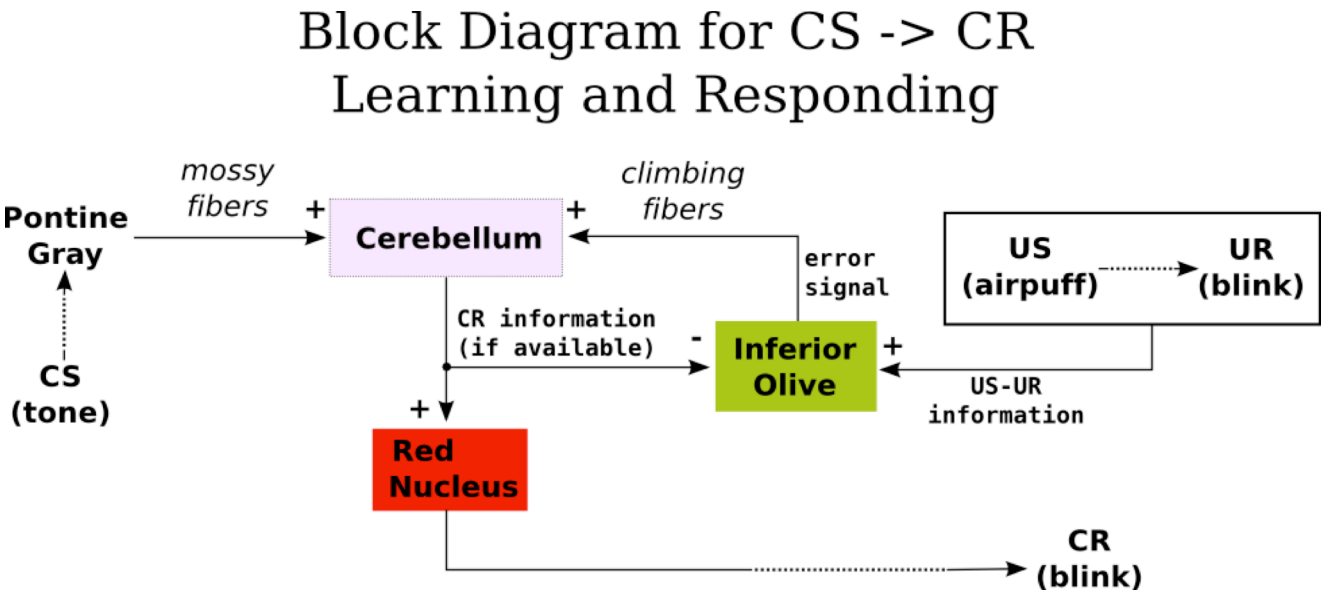
Table 4.1 THE COGNITIVE PARADIGM

The complexity of this cerebellar model is in part due to the vast number of requirements that must be fulfilled. These include:

0. A naïve network fails to produce a properly timed blink (PTB).
1. A naïve network produces a PTB after experiencing enough train trials in which the CS and US are properly ordered and timed.
2. Additional training must not destroy a learned PTB.
3. The mere passage of time does not destroy the learned PTB.
4. Extinction (CS-alone training trials) removes a learned PTB.
5. Non-conditioning controls for the naïve network:
   (a) Training trials of CS-alone does not produce a PTB.
   (b) Training trials of US-alone does not produce a PTB.
   (c) Reversed order or improperly timed (long separation) CS-US trials do not produce a PTB.

The brainstem circuitry of our delay-conditioning model

By recording the activity of neurons before, during, and after conditioning when certain brain regions have been removed, it is possible to define the neural circuitry needed for this learning paradigm. Because the circuitry for blinking in response to an airpuff (US to UR) exists entirely within the brainstem, separating the forebrain from the brainstem leaves the airpuff-blink reflex intact. In addition, the ability to learn and remember the delay-conditioning paradigm also survives removal of the forebrain. This means that the circuitry that learns and the circuitry that mediates the learned response do not include the neocortex, the limbic system, or the basal ganglia. However one critical region needed for learning is the cerebellum. Learning does not occur if the cerebellum is removed or its inputs are compromised. Furthermore, one can identify a change in the firing patterns of the Purkinje cells of the cerebellum as a function of conditioning. We summarize the circuitry on a basic level in Figure 4.13 below.



Note: Pathway activation occurs over the course of a single trial.

Fig 4.13. Presentation of the unconditioned stimulus activates a basic reflex pathway, resulting in the UR eyeblink. Information from this pathway is along the way sent to the Inferior Olivary Nucleus (Olive). On the CS side, presentation of a tone activates pontine gray neurons by travelling through general auditory pathways. The mossy fibers arising from these pontine neurons innervate the cerebellum, which activates neurons of the red nucleus, eventually resulting in an eyeblink. Information from this pathway is also sent to the inferior olive, which computes a difference between signals from the two different pathways. This difference is its output, which we conceptualize as an error signal. This error signal is used to enhance the cerebellum activation by the CS, which in turn enhances the likelihood of red nucleus activation resulting in the CR. It is important to note that red nucleus activation via the CS will not occur until the error signals from the inferior olive have modified cerebellar synapses enough over many trials to allow this. Plus (+) and minus (-) indicate excitatory (glutamate) and inhibitory (GABA) synaptic transmission respectively.

A little more detail

The UR arises from the activation of sensory pain fibers that innervate the eyeball surface. These neurons activate brainstem sensory neurons that in turn activate motoneurons that produce the defensive blink. Such motoneurons are localized in the nuclei of the sixth and seventh cranial nerves. By comparison, once learned, the CS-CR pathway goes through the cerebellum. Activation of excitatory neurons within the cerebellum but located under the cerebellar cortex (the so-called deep cerebellar nuclei, DCN) activate the red nucleus which in turn activates motoneurons (and possibly the sensory neurons) of the blink reflex.

Note that the cerebellum receives two distinct excitatory input classes: (1) the mossy fibers from the pontine gray that carry simple CS sensory information and (2) the climbing fibers from the inferior olive, which carry an error signal. In the case of eyeblink conditioning, error is a failure of the cerebellum to command a blink in a way that blocks an irritating US. To produce a blink normally, the US-UR pathway can only react after the initial sensation occurs. However, the cerebellum can use the CS-sensory information to anticipate the airpuff and to block the US irritation at its onset. When this happens, the olive is inhibited to reduce the error signal, as properly timed firing of the CR blink does not indicate error.

In the next couple figures, we will expand our understanding of the more specific components of the cerebellum involved in the conditioning pathway. The first components we will examine are the deep cerebellar nuclei (DCN), which contain the primary output neurons of the cerebellum. In the DCN, there are two cell-types of interest: (1) an excitatory type, which can excite a CR, and (2) an inhibitory type, which sends its output to the olivary error generator. The excitatory output activates neurons in the red nucleus to produce the CR. This DCN input must be normally inactive, as when there is no US or CS, the eyes should remain open.

Next we will look at purkinje cells of the cerebellar cortex, which inhibit both types of DCN neurons. Purkinje cells are known to be spontaneously active. Because Purkinje cells inhibit DCN cells, whose purpose is to excite the CR, the purkinje cells must themselves be inhibited to allow a learned CR blink to occur. Thus we can expect disinhibition of these cells to be an important part of the conditioning model. Examine Figure 4.14 to better understand the logic of this model, paying particular attention to the sign of the connections.

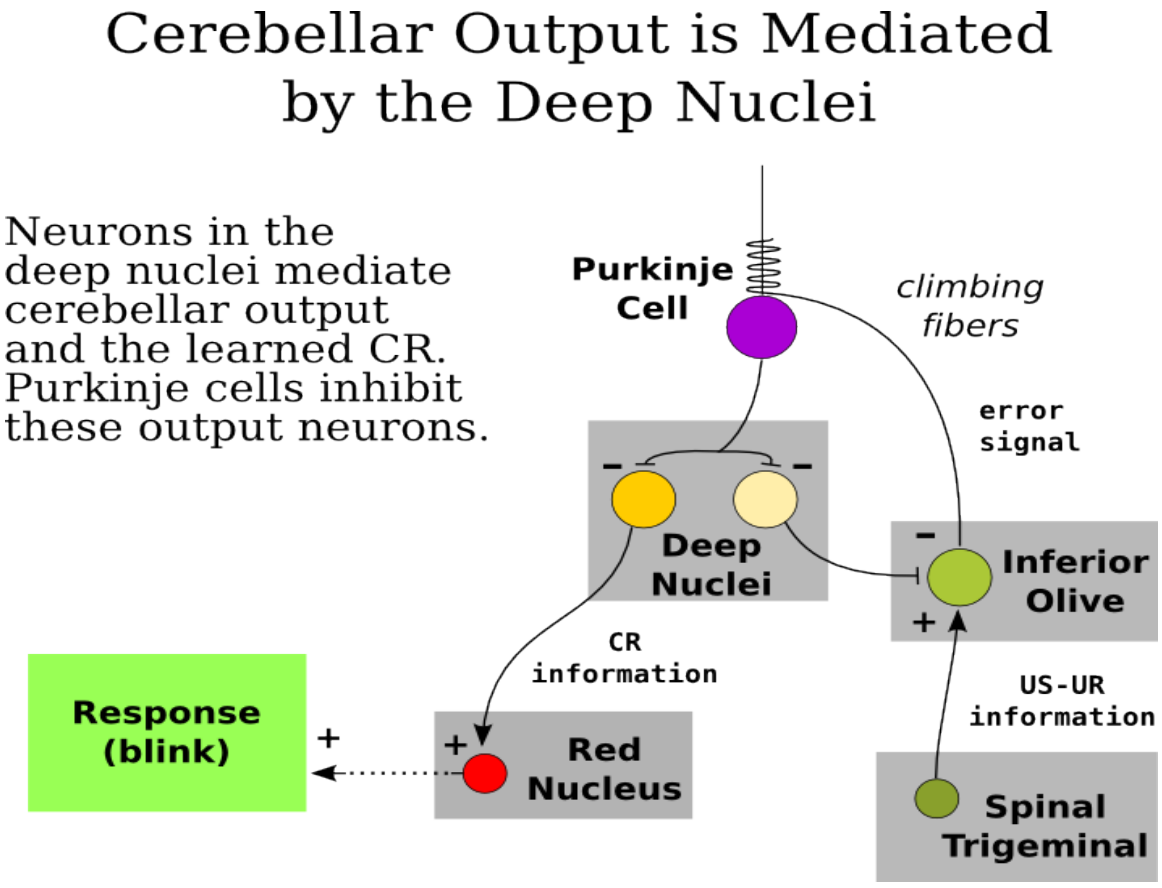


Fig 4.14. The cerebellum is divided into a cortical region and the cells of the deep nuclei. Spontaneously active purkinje cells of the cerebellar cortex inhibit cerebellar output cells located in the deep nuclei, thus inhibiting the CR response. Depression of this spontaneous activity is called disinhibition and leads to a net excitation of the deep nuclei neurons, allowing red nucleus excitation to produce a blink. This is accomplished by the left DCN, while the right lowers the error signal sent by the inferior olive.

The next figure shows a little more detail of the relevant connections within the cerebellar cortex. The sensory input provided through specific pontine neurons (and their axons, called mossy fibers) activates the granule cells of the cerebellar cortex. The role of granule cells is two fold. On the one hand, these neurons directly excite Purkinje cells via the parallel fibers. However, they also indirectly produce a feedforward inhibition of the same Purkinje cells by activating the inhibitory stellate and basket cells (S.C/B.C.), which in turn inhibit Purkinje cells. This inhibition pathway is very important, as its level of activation will determine the level of disinhibition of the purkinje cell and thus ultimately affect the likelihood of a CR blink.

Cerebellar Cortex Model

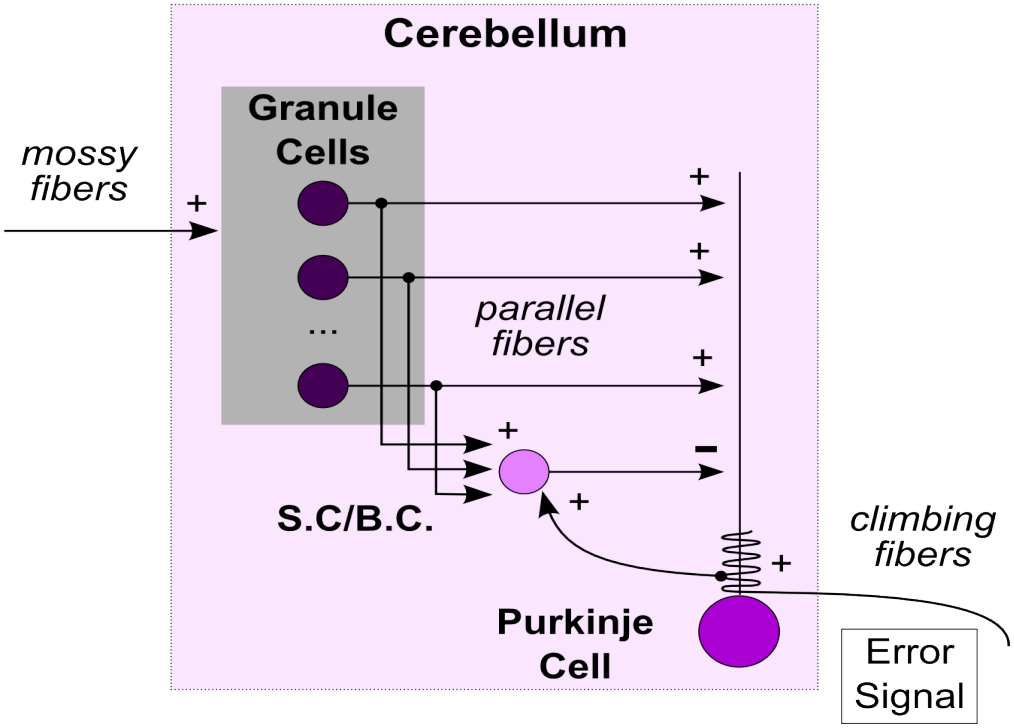


Fig 4.15.
The encoding of the CS-CR learning occurs at synapses in the cerebellar cortex. Synaptic modification occurs at the synapses formed by the granule cells. The parallel fibers of the granule cells are presynaptic to both Purkinje cells and the inhibitory S.C/B.C. Climbing fibers carry the error signal, which controls modification of the granule cells synapses.

Homework 4.3.2.
Let's build the first part of our cerebellar model. Consider just the Purkinje cell, its DCN neurons, and its granule cell inputs. Ignore the inferior olive and S.C/B.C. for now. Assume that all neurons are binary, i.e. outputs are {0,1} and that firing of a given group of neurons occurs when a positively valued threshold is exceeded. (Although it is not necessary for the model to work, please assign a threshold value of 0.5 here and throughout these assignments.) Model inhibition as subtractive and excitation as additive and assume that a Purkinje cell is normally active (output equals one). Start with each grouping (including the CS) having its own time vector showing activation {0,1} at each timestep, and proceed to loop through each timestep to determine sequential activation of the difference regions. See Appendix 4.4 for an outline of how to code your model. The goals of this model are as follows:

1. Granule cells are directly activated by CS (1 timestep apart)
2. Purkinje cells are directly activated by granule cells
3. DCN neurons directly inhibited by purkinje cells. (when Purkinje = 1, DCN = 0) and vice versa.

Homework 4.3.3.

Next, add the SC/BC inhibiting neuron to the cerebellar cortex network. Assume feedforward inhibition by this interneuron is fast and thus reaches the Purkinje cell at the same time as the excitation from the granule cell (it is a fast feedforward subsystem). Also, assume that the Purkinje cell has a continuously present, internal excitation. Define and adjust your parameters (synaptic weights, internal excitation, and a positively valued threshold) so that (i) the Purkinje cell is in the one state (firing) when there is no granule cell activation and (ii) is in the zero state (not firing) when granule cells are active. Explain why you do or do not need a constantly active, internal excitation within the Purkinje cell. Will a simulation work if the absolute values of the excitatory and inhibitory synapses are the same? Explain your reasoning.

Homework 4.3.4.

Add to your program/model a binary threshold neuron for the red nucleus; this neuron is excited to fire when the excitatory DCN neuron fires. (A) Draw this system and include the sign of the synaptic effects as shown in the figures. Label your diagram. Your picture/model should include a granule cell, an interneuron, a Purkinje cell, at least one DCN neuron and a red nucleus neuron. (B) Produce a table showing the state of all of these neurons when (i) the granule cells and the red nucleus cell are in the zero state. Produce a table of neuronal states for the case (ii) that the granule is firing (state equals one) which leads to the red nucleus neuron also firing. (C) Parameterize a model which satisfies both (i) and (ii) of (B). Remember every neuron is {0,1} binary and has a positive threshold that must be reached or exceeded in order to fire. Excitatory weights are positive and inhibitory weights are negative. Although there exist other successful parameterizations, please construct your model with weights whose absolute values do not exceed one (for inhibition this means that the inhibitory weight is not smaller than minus one).

Homework 4.3.5.

List all the individual and qualitatively different parametric alterations you could make to the model so that activation of the granule cells does NOT result in the DCN neuron firing. Restrict your answers to parameters such as a synaptic weight, an internal excitation, or a threshold that exist within the cerebellar cortex. Be sure to give an example of each new parameter value. Hints: the central issue is to prevent granule cell activity from inhibiting Purkinje cell firing. Be sure your list contains the six parameters that can be changed.

Learning by Synaptic Modification

Instead of manually adjusting parameters to force a CR, we will now consider how successive training trials can adjust synaptic weights to invoke learning. Initially, granule cell activity does not excite a CR, but after repeated pairings of CS and US, granule cell activity leads to a CR by reduction of Purkinje cell firing. With regard to synaptic modification, the key synapses are

those of the granule cells. As the granule cells activate both Purkinje Cells as well as S.C/B.C, these are thus the two sites where critical synaptic modification can occur.

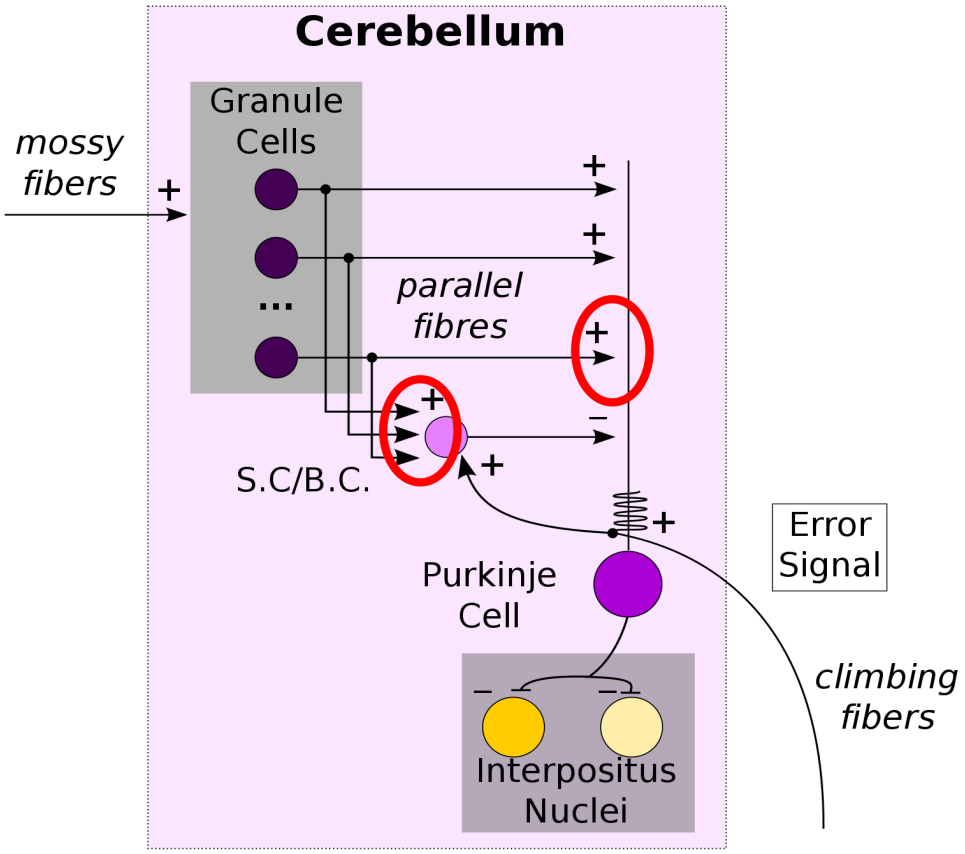


Fig 4.16. Outline of cerebellar synaptic modification. The red ovals indicate sites of critical synaptic modification necessary for learning. Information for this modification is received as an error signal by the climbing fibers projected from the inferior olive. Modification also depends on activity of the granule cell parallel fibers, which when inactive, disallow any weight change at specified synapses.

The two, initial synaptic modification rules for the parallel fiber (PF) synapses defined by truth tables:

If the purkinje cell is postsynaptic:

CF \ PF	0	1
0	0	0
1	0	+

If the interneurons are postsynaptic:

CF \ PF	0	1
0	0	0
1	0	–

Note: CF represents Climbing Fiber input

In a nutshell, these two tables show that synaptic modification at either interneuron (S.C/B.C.) or Purkinje Cells depends both on input from the parallel fibers of the granule cells, as well as an error signal from the climbing fiber of the inferior olive. The two synapses differ only by the sign of their change. As learning occurs, the weight of the presynaptic, excitatory purkinje cell synapse is reduced, while that of the inhibitory interneuron synapse is increased. Note that both modifications effectively reduce Purkinje cell activity and thus increase the likelihood of a CR blink through DCN activation.

You might program synaptic modification something like this:

$w(t+1) = w(t) + const*pf*cf$

pf and *cf* are the binary {0,1} parallel and climbing fibers and *const* is some value chosen so that learning takes about thirty trials. Note that the climbing fiber is active when a US-UR occurs. In the beginning, we will start with a rule built on the assumption that the *cf* input is binary.

In addition, note that Purkinje cell firing slows down due to interneuron firing only so long as this is not offset by granule cell excitation. Thus the two outputs of the granule cells are in a constant balancing state where weight modification at one or both synapses can eventually tip the scales towards or away from DCN firing.

Homework 4.3.6.

Using your computational model that you have built up so far, find parameter values that demonstrate that either modification rule alone can lead to a learned CR. That is, create a program that learns to produce a CR in response to a CS. Describe simulations that use suitable parameter settings (probably different settings for each case) in which (A) weakening of the PF-to-Purkinje cell synapse or (B) in which strengthening of the PF-to-interneuron cell synapse leads to a CR. In both cases, before paired conditioning, your simulations need to satisfy two initial conditions: (i) no CS input implies that the red nucleus neuron is inactive (zero) and (ii) a CS input does not cause a CR (i.e., the red nucleus neuron remains zero). Also, to maintain biological plausibility, make sure your granule cell to Purkinje cell synapse cannot reach a value that is less than zero (the mechanism need not be elegant; you might use an if-then routine). Be sure to report the value of your synapses initially and the values when the first CR occurs. Comparing to the trial just before the first CR, explain why a CR occurs in terms of the Purkinje cell net excitation and the altered synaptic strength. (See text for hints.)

There is a temporal delay built into the synaptic modification equation. That is, the sensory signal of the parallel fibers precedes the error signal by 50-300 msec. We are implicitly presuming that this delay occurs and that the value of synaptic modification depends upon this delay.

Homework 4.3.7.

1. What happens when the model includes modification at both the SC/BC cell and the Purkinje cell? That is, what happens when both types of synapses are allowed to undergo modification? Describe one set of parameters which requires both forms of modification for successful learning; that is, modification of either synapse alone falls short of bringing the Purkinje cell below threshold. Recall that the weight of the excitatory synapse(s) cannot fall below zero and that of the inhibitory synapse cannot go below minus one.
2. Two other questions should be addressed in your homework: Is the system stable when no inputs are active? Explain what this means in terms of memory. Is the system stable when over-trained (i.e., given many more trials of paired conditioning than are needed for the initial CS-CR result). For the interneuron synapse, will its synaptic strength grow without limit if there is no DCN to inferior olive system? Explain how this model can prevent such unlimited growth by taking advantage of the inhibitory output of the DCN.

At this point your model should demonstrate the following characteristics:
1. Before training, the CS-alone does not elicit a CR.
2. Repetition of just the CS does not lead to a CR.
3. With sufficient CS-US pairing, the CS produces a CR.

Extinction by Reversible Synaptic Modification

Now let's enhance our model's functionality. Both types of granule cell synaptic modifications are known to be reversible, which we can use to model extinction. After associative training of the CS with the US produces a learned CS-evoked CR, CS-alone trials can be used to decouple the CS from the learned CR, a.k.a. extinction.

Reversible synaptic modification expressed in a binary contingency table.

If the purkinje cell is postsynaptic:

CF \ PF	0	1
0	0	–
1	0	+

If the interneurons are postsynaptic:

CF \ PF	0	1
0	0	+
1	0	–

As we mentioned before, the activity of the climbing fiber (*cf*) and the parallel fibers (*pf*) can be represented as a binary set {0, 1}. When there is no CS activation or CS-like pontine activity, *pf* = 0, and there is no synaptic modification of either kind. However suppose that the CS is active so that *pf* =1. In this case, the activity of the climbing fiber is what determines the sign/direction of synaptic modification. Remember that the climbing fiber is active (*cf* = 1) as it signals a failure to produce a CR that blocks the US. Thus for a Purkinje cell, given that *pf* =1, the

granule cell synaptic weight decreases when *cf* =1 and increases when *cf* = 0. The reverse is true for the granule cell to interneuron synapse. Just like before, these binary interpretations can be written in a more continuous form as a synaptic modification rule.

For Granule-Purkinje Synapses:

$$\Delta w_p = -const \cdot pf \cdot (-error) = const \cdot pf \cdot (1 - 2 \cdot cf)$$

where const is some positive value which helps determine the rate of learning. Note that the parenthetical term is -1 when *cf* = 1 and +1 when *cf* = 0.

For Granule-S.C/B.C. synapses:

$$\Delta w_{\text{int}} = const \cdot pf \cdot (error) = const \cdot pf \cdot (2 \cdot cf - 1),$$

Once again we use the convention that a climbing fiber that signals 'no error' is represented by an activity level of zero while an error (failure to blink just before the UR is evoked) is signaled by a *cf* value of one. This model will produce an example of extinction, but the model itself suffers in other aspects of its performance.

Be sure that you have a functioning inhibitory projection from the DCN to the inferior olive; define the output of the inferior olive as zero when the DCN projection is active regardless of the input excitation to the inferior olive. When no inferior olive inputs are active, the output will be defined as zero here.

Homework 4.3.8.

Show that the model learns to blink with CS-US pairings (training), but the model fails if training continues for too many trials. Incorporate the above synaptic modification rules into your program. Show that your model can learn the CS-CR response from CS-US pairings and then extinguish the response with CS-alone training trials. Show that overtraining (too much training) leads to an oscillation between a CS-evoked CR and a CS that does not produce a CR. Describe the oscillation and explain, at the level of cells and synapses, why it happens. Explain why your results will generalize across models (i.e., across your selection of modifying synapses, either the interneuron, Purkinje cell, or both).

Final Changes

In order to reduce the oscillation described above, we must consider a roughly continuous three-state model for climbing fiber activity. The key change from the previous, binary model is that there is now an in-between inferior olive/climbing fiber state that signals no error, while the two extreme states signal a positive or a negative error. As before, the positive error corresponds to *cf*=1; now, a negative error corresponds to *cf*=0, and there is a designated level of spontaneous activity, call it $\overline{cf}$, which falls somewhere in between the two extreme types of error states. The three-state *cf* model presumes that spontaneously active cells in the inferior olive have an

average firing rate of $\overline{cf}$ in the absence of any active inputs or when both the excitatory and inhibitory inputs to the inferior olive are simultaneously active. Thus, the three possible cf states are $\left\{0, \overline{cf}, 1\right\}$, where $0 < \overline{cf} < 1$. Taking this into account, the updated synaptic modification rules can be written as follows:

$$\Delta w_{\text{int}} = const \cdot pf \cdot (error\ value) = const \cdot pf \cdot \left(cf - \overline{cf} \right),$$

$$\Delta w_{p} = const \cdot pf \cdot (error\ value) = const \cdot pf \cdot \left(\overline{cf} - cf \right).$$

Homework 4.3.9.

Now use the newest (above) modification rule. Demonstrate that this helps cure the oscillation problem. Explain how, and in what sense, your inferior olive model produces an error signal. Consider four cases (i) no input, (ii) US input alone, (iii) US input together with a CS, and (iv) a CS-CR result without a US input.

Homework 4.3.10.

Parameterize your model so that it demonstrates learning. Looking at your new model versus the one you designed in assignment 4.3.8, explain why this new model does not oscillate during overtraining. What would happen if there were no excitatory input to the inferior olive when a successful blink was delivered during a CS-US trial? Show that the model demonstrates extinction and that after extinction your model shows savings on retraining (i.e., fewer trials to learn the CS-evoked CR than during the original training). Explain the synaptic and cellular basis of this savings. Warning: a poor selection of initial values will cause a simulation to not show savings.

CHIP TO WRITE- The truth about co-activity and association

Putting it All Together

Now that we have discussed the main components of our cerebellar model, it's time to examine more closely how the timing of interneural firing coordinates to actually produce a properly timed blink (PTB), i.e., one that accurately predicts and blocks the US. First, let's summarize what we have discussed thus far in the figure below.

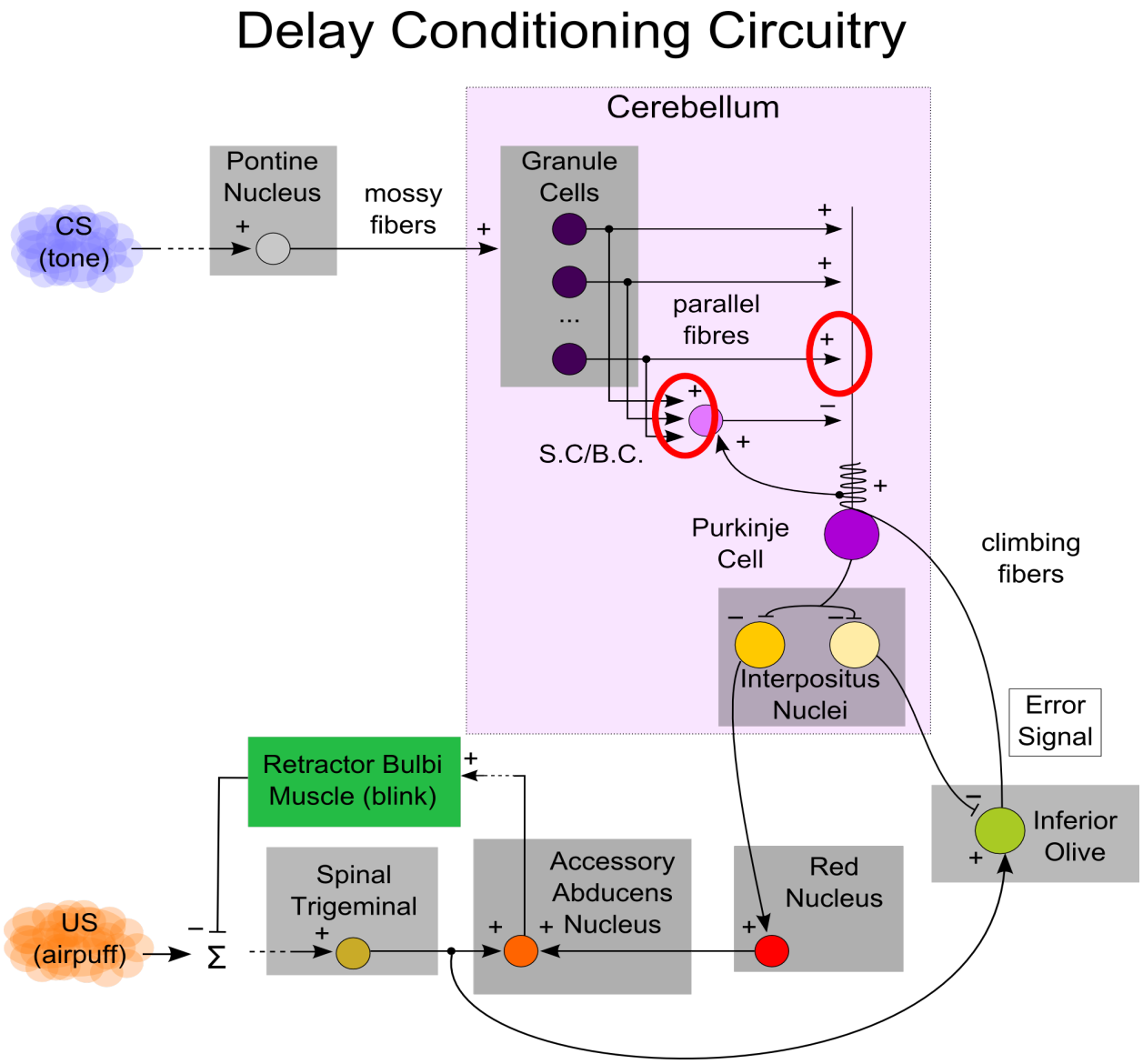
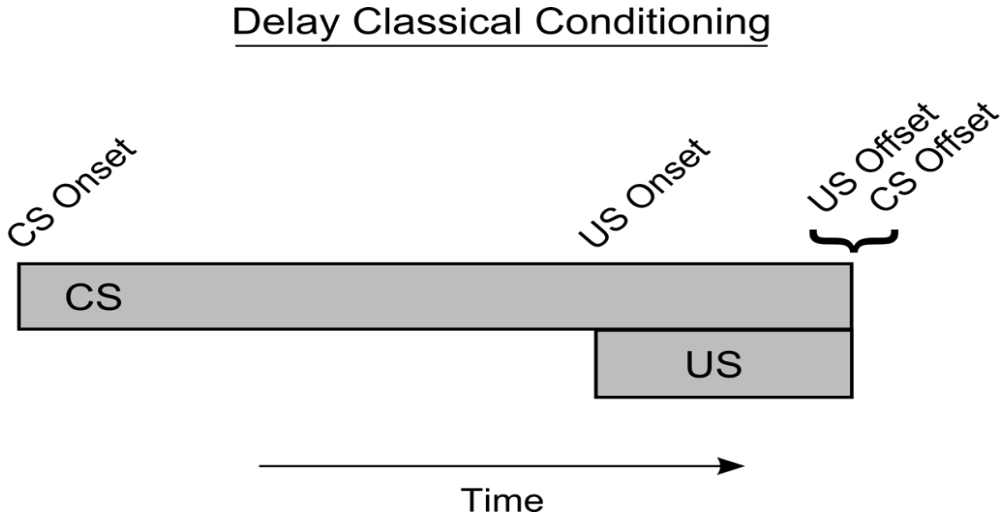


Fig 4.17. A full outline of the delay conditioning circuitry. The ellipsis (four dots) next to the CS replaces the sequence of auditory neurons in the brainstem that leads to innervation of the pontine nucleus. The red ovals indicate sites of critical synaptic modification necessary for learning. In the US-UR circuit, the spinal trigeminal nucleus excites both the Accessory Abducens Nucleus and the inferior olive simultaneously. Information from both the CS-CR and US-UR systems is sent to the accessory abducens nucleus, which is a collection of neurons that innervates the Retractor Bulbi, the muscle that causes a blink. When activated by the CS-CR pathway, this muscle will also inhibit the US-UR pathway. This makes sense, as the main goal of the CR blink is to predict and mitigate the effect of the US. All components lie inside the brain except for the CS, US, and the retractor bulbi.

There is some new information to take note of, chiefly, the retractor bulbi muscle and the accessory abducens nucleus. As the main muscle responsible for blinking, the retractor bulbi relies on innervation from the accessory abducens nerve cluster, which in turn is activated by either the spinal trigeminal in the US-UR pathway, or the red nucleus in the CS-CR pathway. Effective learning can be demonstrated by the activation of the abducens and the bulbi by the CS in such a way that the act of blinking successfully blocks the US and thus inhibits spinal trigeminal activation. Also of note is the dual excitation of the abducens and the inferior olive by the spinal trigeminal. This occurs simultaneously and is the way in which the olive is able to receive information pertaining to the US-UR pathway to determine if the CR was properly timed. CS-CR information is carried to the olive by the DCN as before.

Now let's examine one of the most important parts of delay conditioning, timing. Suppose we decided to condition a rabbit to blink in response to a brief tone played before and during the application of an airpuff to the eye. This situation would resemble the figure below, which you might remember from our earlier introduction to eyeblink conditioning.



The main difference, however, between what you learned previously and our example is that for simplicity's sake, we divide the onset, duration, and latency times into individual timesteps (as in your model). We will also be breaking the circuit into two parts, the US initiated pathway, and the CS initiated pathway.

The US pathway contains three major elements. These are the critical regions required for transduction of the sensory information caused by the airpuff to elicit the UR. Initially, the airpuff (through sensory pain fibers) excites the (i) spinal trigeminal, which then excites the (ii) accessory abducens nucleus (also the inferior olive), which in turn innervates the (iii) retractor bulbi muscle. Each of these successive activations occurs over the course of one timestep. Meaning a US active for three timesteps would in turn produce a spinal trigeminal active for three timesteps, offset by one.

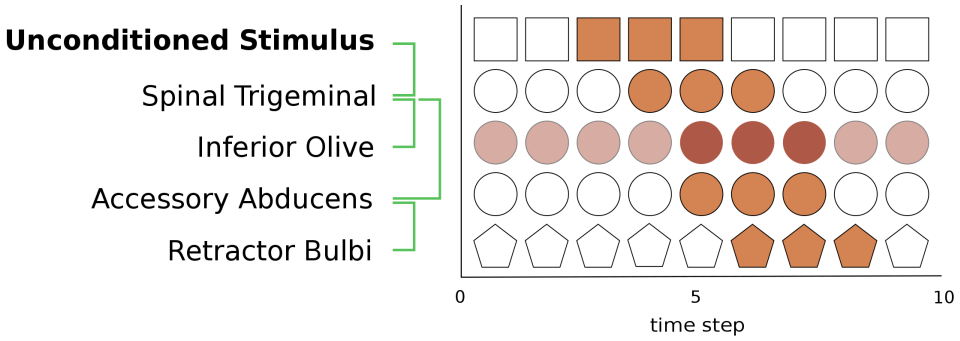


Figure 2. The qualitative sequence of activations that produces the unconditioned response (UR). Sensory activation by the US activates neurons in a sequence of brain regions to elicit the UR. The first neurons in the brain responding to the US are in the spinal trigeminal nucleus. In turn, these neurons directly excite the inferior olive and motor neurons of the accessory abducens nucleus. The motor neurons directly innervate the muscles responsible for blinking. The latency of the UR in response to a US decreases as the intensity of the US increases; typically latencies range from 40-80ms (Schindler CW, 1984). A filled square indicates an external stimulus at a particular time; a filled circle indicates an activation of a brain region at a particular time; a filled pentagon indicates a muscle response at a particular time; empty regions indicate very low or nonexistent activity, while a shaded region indicates a moderate level of spontaneous activity.

This pathway occurs without any learning. Unfortunately, that means it is impossible for the retractor bulbi to anticipate the US and block it. In fact, it takes a full two timesteps just for the membrane to react. Thus we must find some other pathway that can activate the muscle quicker.

On the CS end, there are 8 components, two of which overlap with the US-UR pathway, namely those required for blinking, the accessory abducens and the retractor bulbi. From the top, the CS transmits through the auditory pathway to activate neurons in the (i) pontine nucleus. From its mossy fibers, the nucleus activates the (ii) granule cells, which then activates both the (iii) S.C./B.C and the (iv) purkinje cells. As we mentioned earlier, the S.C/B.C. to purkinje connection is very fast; thus the granule cells can be said to both inhibit through this pathway and excite purkinje cells directly within a singe timestep.

The next half of the pathway is dependent entirely upon the level of inhibition/excitation of the purkinje cell by the granule cell based on the weights of the modifiable synapses. In untrained subjects, the excitation and spontaneous activity of the purkinje cell is high enough to more or less completely inhibit (v) DCN activity, as can be seen in the figure below.

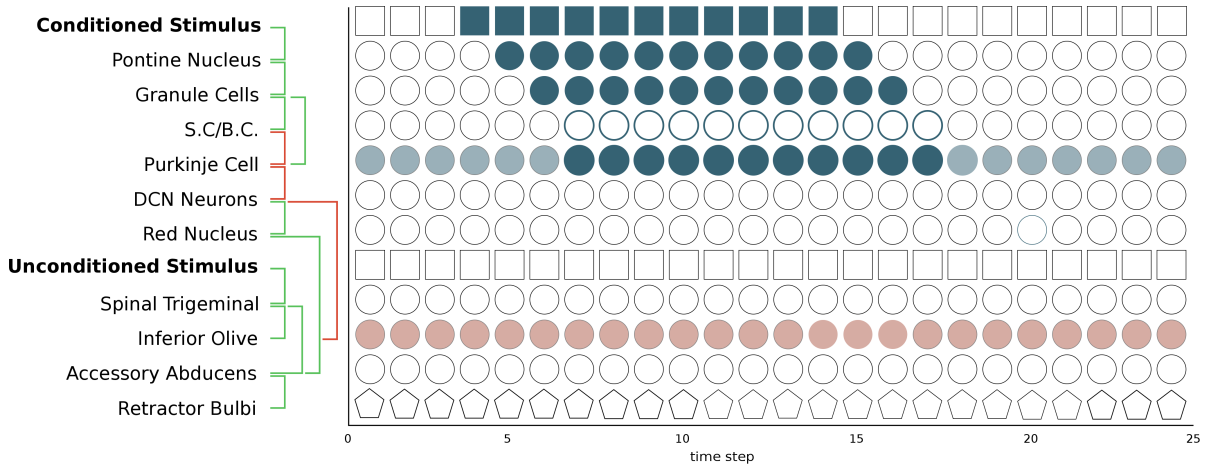


Figure 3: CS alone training. Before conditioning, the CS does not evoke a response. The cerebellar model is tested with only a CS-input, and no blink is produced. Note that the CS produces a sequence of cell firing culminating in enhanced activity in the Purkinje cell. Because the Purkinje cell is inhibitory, no neuron of the output circuit is activated. Inferior olive activity remains at spontaneous level, thus produces no error signal.

Unfortunately, this lack of DCN excitation prevents the excitation of the (vi) red nucleus and eventually the (vii) accessory abducens nucleus and (viii) retractor bulbi activation. This is all that occurs when a CS by itself is presented. However when the US pathway is activated a certain number of timesteps after the CS pathway, we can see a gradual modification of synapses. How does this occur?

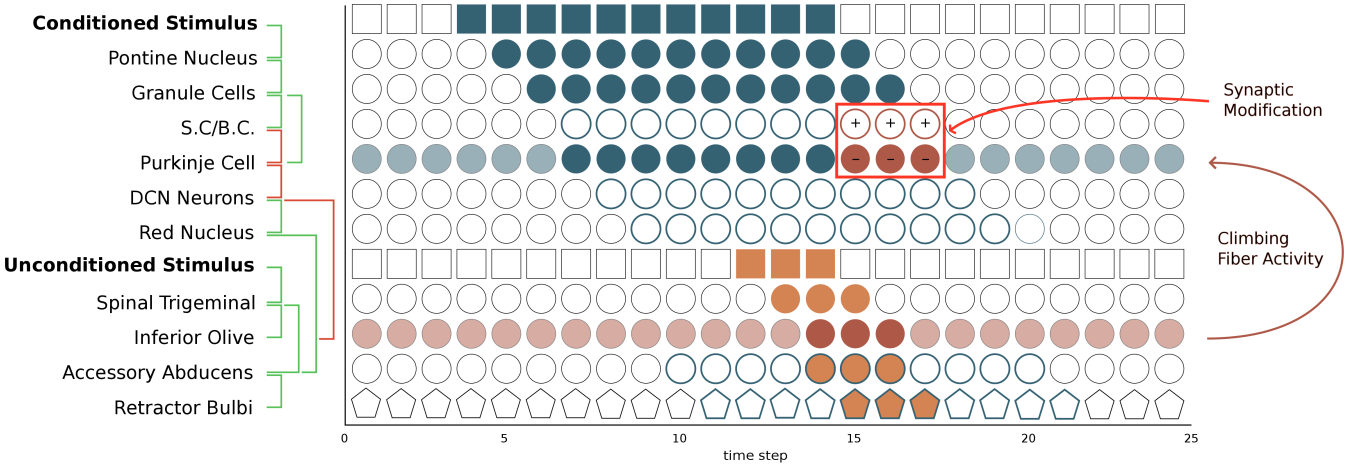
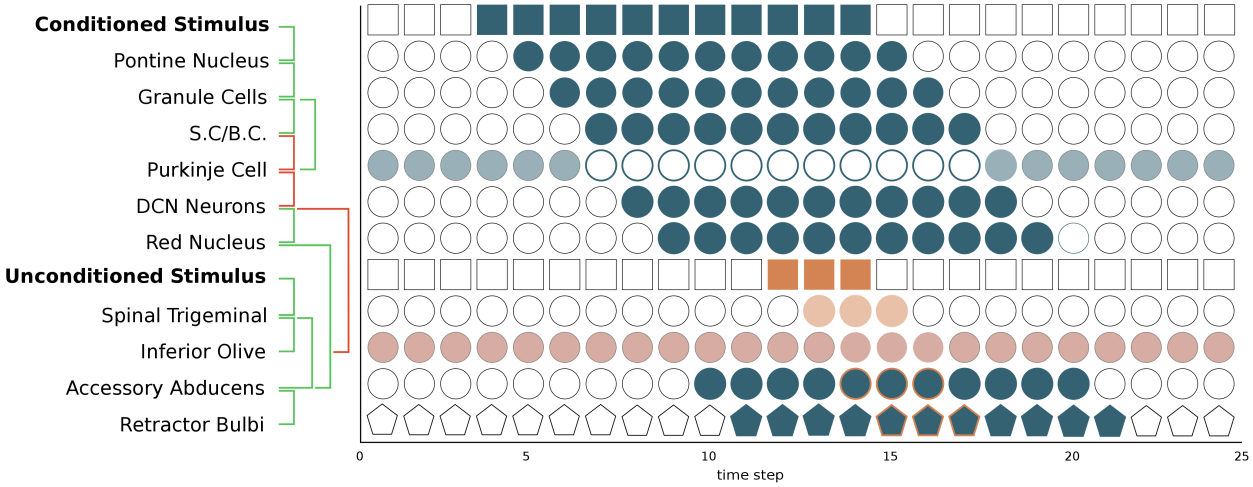


Figure 3: Early Training Trial. Spinal trigeminal activation by the US allows for inferior olive excitation, producing an error signal that directly affects synaptic weights of both the purkinje cell and S.C/B.C. neurons. Circles in the red box show new weights after modification; + indicates increased weight, - indicates decreased. This modification relies on both Granule Cell activity as well as climbing fiber input. Gradual modification in this way over successive trials eventually leads to S.C./B.C. activation and sufficient inhibition of the purkinje cells to allow a CR blink to occur. Blue outlined regions indicate areas that will become excited after sufficient synaptic modification.

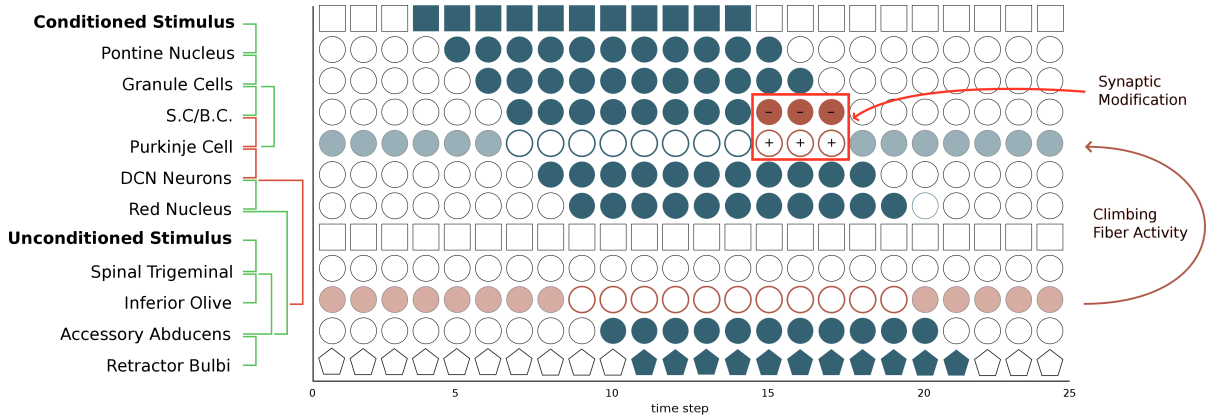
As we mentioned earlier, synaptic modification of the granule cell synapses can eventually lead to disinhibiton of the DCN neurons via inhibition of purkinje cell activity below some threshold. This then activates the pathway for a blink to occur. These synaptic modifications are determined by the error signal sent by the climbing fibers from the inferior olive, which receives input from both the spinal trigeminal and the DCN (from an earlier timestep). Thus, in thinking from the perspective of our simplified timestep model, the sequence goes something like this: The CS, taking one timestep per transmission, activates the granule cells two timesteps after CS onset. However, synaptic modification depends on climbing fiber activity, which would occur two timesteps after US onset; US→Trigeminal→Olive (climbing fiber).

Thus, synaptic modification takes place continuously over the interval starting three timesteps after US onset to three timesteps after co-termination of both stimuli. The final weights after this interval will be used as the starting weights for the next trial. Over a multitude of training trials, this weight may change enough to allow DCN firing and a CR blink to occur.



Late Training Trial after CR Establishment: After many training trials, weights have been modified enough to allow inhibition of purkinje cell activity (in part due to S.C/B.C. activation). This allows DCN firing which eventually leads to the CR. Note that utilizing the CS pathway allows the signal to reach the retractor bulbi faster than could be obtained through the US-UR pathway, causing the muscle to contract and blink a whole timestep before the US even occurs. Shaded trigeminal region indicates lessened activity due to partial blocking of the US by the blink reflex. (see full circuitry diagram)

After sufficient training, the PTB response will occur with or without a US. However each time the CS is presented alone without an accompanying US, the DCN inhibition of the inferior olive is not balanced by spinal trigeminal excitation. Thus, the inferior olive activity is dropped below resting level, causing a negative error signal to be sent to the granule cell synapses as before. This time however, the change is reversed, and the purkinje cell will gradually increase in activity enough to fully inhibit DCN activity, reverting the system to its normal, untrained state.



Extinction due to CS only trials: A sustained inhibition of the inferior olive by the DCN will cause a reverse in synaptic modification, causing unlearning of the trained behavior.

Section 4.3.3. Comments for Teachers

When the Purkinje cell goes from one to zero, the cells of the deep cerebellar nuclei go from zero to one, at least for the two-state model. This is what is needed to get a CR out of the deep cerebellar nuclei. It is also what is needed to cancel the sensory excitation of the inferior olive if we assume that the CR itself does not sufficiently lower inferior olive excitation (in this regard experimental evidence says that deep cerebellar nuclei inhibition of the inferior olive is needed).

Now consider the extinction condition. Presumably the climbing fiber goes to zero because of the deep cerebellar nuclei inhibition. Thus looking at the synaptic modification equation, we see that there will be second quadrant synaptic modification, but there seems to be a problem, which may depend on parameter settings and assumptions (e.g., randomizations/noise felt by the cells of the inferior olive). (Very vague, unsure of what to keep)

Concerning the oscillation, each time a correct CR is delivered during overtraining, there will be second quadrant modification of the cortical synapses of interest. This modification weakens the learned response and can lead to a disappearance of the CS-CR. But then, when the CR is lost, fourth quadrant modification occurs and eventually the CR returns. As this is a threshold-based model (remember the P-cell model here), then we should see the system oscillating between giving and not giving a CR. The oscillation would seem to depend on the size of the synaptic modification and any randomization/noise that is used in the model.

Section 4.3.4. Extra Stuff

(Although it was once frowned upon and controversial at best in the eyes of western psychologists, today we can give ecological interpretation to the reflex and other learned behaviors: the eyeblink reflex is to protect the eye, and the CS provides a valid means for the brain to predict the upcoming US to further enhance this protective, or as eastern European scientists referred to it, defensive reaction.)

The temporal relationships between CS and US for a prototypical delay airpuff paradigm are illustrated in fig X (for more precise details see Thompson and X 2003). Along with the conditioning paradigm, there are control paradigms. These control paradigms are necessary to establish that the CS-CR response is due to associative learning. (Digression #1. motor vs. sensory, it all depends.

The functional circuitry of the inferior olive and cerebellum are replicated in fish that have an electric field sensing ability. In some cases this function is outside of the cerebellum, i.e., in the medulla while in other cases much of the circuitry is in the cerebellum itself. In either case we must suspect a sensory motor system that responds very quickly because so little processing is required. But why is a cerebellar system incorporated with a reflex arc? One possible answer is that the learning abilities of the cerebellum are used to compensate for growth. Fish grow throughout their life and this leads to two complications for sensory-motor reflex arcs. First, there is the change in how much motor flexion produces how much movement. Second, there is the addition of surface area which will likely be accompanied by more sensing organs. In either case, synaptic modification in the cerebellar circuitry will allow the organism to make movements that are both quick and accurate.

It is now known that a forebrain style LTP/LTD rule exists for excitation of inhibitory interneurons. Jorntell and Ekerot showed that paired activation of excitatory parallel fiber input and excitatory climbing fiber input leads to Long Term Potentiation (LTP) of the pf input on interneurons in the cerebellum. In addition, they showed that Long Term Depression (LTD) is caused by pf activation alone, conducive to extinction.

Section 4.3.5 Summary

(Something should go here)

Section 4.3.6 Possible quiz questions

1. What is a category-supervised neuron? How does a neuron learn to recognize a category based on its features?
2. Give the general formula for synaptic modification.
3. What is associative synaptic modification? Explain what it means for a rule to be Hebbian in nature.
4. To what value do weights converge based on a given input set? What happens if a linear self-supervised neuron is trained on a symmetric mixture of two data sets? How is the result different if such a neuron is trained by alternating sampling between two such non-overlapping pattern sets, instead of being trained first on all samples of one of these two data sets and then on the other?
5. What advantage does a non-linear self-supervised neuron have over a linear one?
6. What can you learn about weight convergence by looking at eigenvectors for a given network?
7. Briefly explain what kinds of modifications can be made to synaptic weights and threshold in an error-correction system and what these modifications are based on.

8. What is delay conditioning? Briefly explain how the Purkinje cell network in the cerebellum can exhibit delay conditioning.

Glossary Terms

associative synaptic modification	depends on coordinated changes in the activities of the presynaptic and postsynaptic neurons
category	a set of defined inputs that are treated equivalently; e.g., member or not member in the case of the category supervised modification.
category-supervised neuron	a neuron with at least two classes of input, a category input and a defined set of feature inputs. The synaptic modification of the features is dependent on category absence or presence.
Classical conditioning	see Pavlovian conditioning
Coactivation	a misnomer for the microscopic association of neuronal activity that leads to synaptic modification if these neurons _____ _____ _____activation of two neurons at , or nearly at, the same time, strengthening the synapse between them; in fact, there is a slight temporal asymmetry in the connectivity requirement. (section unwritten)
complementary coding	uses code words that come in pairs where each item in the pair is the opposite of the other. (e.g., black vs. white, move to the left vs. move to the right)
conditional average	an average that depends on the context of the input, or the average of a subpopulation with additional characteristics to distinguish itself from its original parent population
conditioned response (CR)	a response which becomes associated with a conditioned stimulus (e.g., eye blink in response to a tone, salivation in response to a bell)
conditioned stimulus (CS)	a neutral stimulus (e.g., tone, bell) which, when paired with an unconditioned stimulus, reproduces an instinctive response (blinking to an airpuff to the eye; salivation at the sight of food when hungry)
cotermination	ending at the same time

delay conditioning	a form of Pavlovian conditioning in which the CS is turned on and then the UCS is turned on after a short delay, and the two stimuli co-terminate
dominant eigenvector	the eigenvector associated with the largest eigenvalue
eigen system or eigen equation	matrix gvector = scalar gvector where the vector on both sides is the same. For a given matrix, there is a limit set of {vector, scalar} pairs that solve the equation.
eigenvalue	the (a) scalar of the eigen equation
eigenvector	the (a) vector that solves the eigen equation
experimental paradigm	Describes how a psychologist treats each animal in an experiment. An associative learning experiment often consists of many individual trials for each animal. Typically each trial incorporates a single association between the designated stimuli (e.g. the CS is associated with the US in Pavlovian Learning).
extinction	a psychological paradigm particularly effective in reducing the learned responding to the CS; the removal of the conditioned response resulting from the presentation of the conditioned stimulus without the unconditioned stimulus (e.g., a tone without an air puff)
extrapolation	estimation of a value from values within a known range by assuming that the estimated value follows logically from the known values
features	a particular scalar that is part of a pattern (vector)
globally asymptotically convergent	a vector that moves toward another vector, regardless of the starting value. The original vector reaches the final vector, or a defined vicinity of the final vector, in the limit as time approaches infinity
grandmother code	the opposite of a distributive code; one and only one neuron turns on to represent the existence of a particular category

Hebbian	the co-activity characteristic of a synaptic modification rule. Historically pre- and post-activity lead to enhancement of excitatory synaptic strength
interpolation	estimation of a value of (a function or series) between two known values
linear regression	a type of analysis of data which is used to make predictions about a single value that involves discovering the equation for a line that most nearly fits the given data
local	as in 'local information,' refers to information that is biophysically available to a synapse or a neuron
non-local	needs definition
pattern	a valued vector; the dimensions of the vector are a specified set of features
Pavlovian conditioning	Is a learning paradigm that explicitly associates two stimuli, the CS and US. The animal has no direct control over the US, but in an aversive paradigm the animal is given the option of *attempting* to avoid the US. The animal may then use the CS information to attempt to avoid the US, whether or not the US is actually present.
self-supervised neuron	also called an unsupervised neuron. Synaptic modification is controlled by the post-synaptic excitation arising from the same set of synapses that are being modified
synaptic modification	alteration of synaptic strength in either direction making a new synapse or breaking (discarding) an old synapse
trial	This is a term used by psychologists. One trial in a learning paradigm corresponds to a single presentation of the CS-US pair. Many trials may be required for learning. One trial corresponds to one iteration of your learning algorithm.

unconditioned response (UR)	an instinctive response to stimuli from the environment (e.g., eye blink in response to an air puff in the eye, salivation in response to food)
unconditioned stimulus (US)	a stimulus that causes reflexive, unlearned behavior (e.g., an air puff in the eye stimulates a blink, seeing food stimulates salivation)

Appendix 4.1.

Eigenvalues and eigenvectors solve the eigenequation.
Consider the following two equations in three unknowns (w_1, w_2 and λ)

$w_1 x_{1,1} + w_2 x_{2,1} = \lambda w_1$ and
$w_1 x_{1,2} + w_2 x_{2,2} = \lambda w_2$

Using the notation of linear algebra where $w = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$, we can rewrite these two equations as

$w^T \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} = \lambda w^T$, or more compact still using $X = \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix}$
$w^T X = \lambda w^T$ or equivalently
$X^T w = \lambda w$

Because there are only two equations and three unknowns, we need another condition for a full solution to exist. By a consensus that all fields of research have agreed upon is a particular normalization condition. Specifically, the third equation specifies that *w* is of unit (Euclidean) length—that is, $\sqrt{w_1^2 + w_2^2} = 1$. (However, because normalization uses the squares of the elements, the sign of the answer is arbitrary. There is a convention, however, which MatLab does not always follow. This convention is that the vector of all of one sign—it always exists—will be written as nonnegative for all its elements.)

First, we will solve for λ.

$$w_1 x_{1,1} + w_2 x_{2,1} = w_1 \lambda \qquad \Rightarrow \qquad \lambda = x_{1,1} + \frac{w_2}{w_1} x_{2,1} \qquad \text{(A)}$$

$$w_1 x_{1,2} + w_2 x_{2,2} = w_2 \lambda \qquad \Rightarrow \qquad w_1 = \frac{w_2 (\lambda - x_{2,2})}{x_{1,2}} \qquad \text{(B)}$$

Substituting w_1 into equation (A) reveals

$$\lambda = x_{1,1} + \frac{x_{2,1} x_{1,2}}{\lambda - x_{2,2}}$$
$$\lambda(\lambda - x_{2,2}) = \lambda x_{1,1} - x_{1,1} x_{2,2} + x_{2,1} x_{1,2}$$
$$\lambda^2 - \lambda(x_{1,1} + x_{2,2}) + x_{1,1} x_{2,2} - x_{2,1} x_{1,2} = 0$$

$$\lambda = \frac{x_{1,1} + x_{2,2} \pm \sqrt{(x_{1,1} + x_{2,2})^2 - 4(x_{1,1} x_{2,2} - x_{2,1} x_{1,2})}}{2} \qquad \text{(C)}$$

There are two solutions in this case because the vector *x* has two dimensions.

In general, there are n solutions when there are n equations in n unknowns and for each of these n solutions there will be a different *w* that satisfies all of the equations. That is, for $X^T w(k) = \lambda_k w(k)$, where *k* indicates one of the *n* eigenvectors, there are n values of λ_k and n different *w*(*k*)'s each λ_k having an associated *w*(*k*) (note that if by chance there are two or more λ_k's that are equal then there will be less than n distinct solutions).

Now suppose we solve *w* and get λ_1, λ_2 for solutions. It is now possible to solve for *x*. There will be a different solution to *x* for each value of λ.

To solve for *w* use the normalization condition to get

$$w_2^2 = 1 - w_1^2$$

Then substitute this result and (C) into the square of equation (B) to get

$$w_1^2 = \frac{\left(1 - w_1^2\right) t^2}{x_{1,2}^2}$$

where $t = \frac{x_{1,1} + x_{2,2} \pm \sqrt{\left(x_{1,1} + x_{2.2}\right)^2 - 4\left(x_{1,1}x_{2,2} - x_{2,1}x_{1,2}\right)}}{2} - x_{2,2}$. Then

$$w_1^2 x_{1,2}^2 \left(1 + \frac{t^2}{x_{1,2}^2}\right) = t^2$$

$w_1 = \frac{t}{\sqrt{x_{1,2}^2 + t^2}}$ from which we use (B) to get w_2.

$$w_2 = \sqrt{1 - w_1^2}$$

$$= \sqrt{1 - \frac{t^2}{x_{1,2}^2 + t^2}}$$

$$= \sqrt{\frac{x_{1,2}^2}{x_{1,2}^2 + t^2}}$$

The word 'eigen' means 'self' in German, and one can see that, in the so-called eigenequation, the eigenvector, *w*, reproduces itself. Suppose *m* is a square n-dimensional, i.e., {1, …, n} matrix

$$m^T e_i = \lambda_i e_i .$$

One reason why the eigen relationship is so important is the decomposition of any point *x* in the n-dimensional square of *m*

$$mx = \sum_k \lambda$$

and the fact that the inner product of two different eigenvectors of such a system is zero, $w(r)^T * w(s) = 0$. That is, the eigenvectors of a matrix are mutually orthogonal, or equivalent by, at right angles to each other. Because conventional axes in a multivariate Euclidean space are all mutually orthogonal, an eigenvector system can be pictured as a rotated version of the original axes.

Each eigenvector is a potential solution to $Ave\left[XX^T\right]w = wE\left[y\right]$. Therefore each eigenvector is a potential stable point. In fact, the convergence proof must show that only a single eigenvector wins out, and this is indeed what happens. Specifically, the eigenvector with the largest eigenvalue wins.

Appendix 4.2. Table of modification rules

Process	Equation	Variable Legend
Binary threshold modification by excitation	$\theta(t+1) = \theta(t) + \varepsilon(\theta(t) - y(t))$	θ - threshold t - timestep ε - rate constant y - excitation (1 if input*weights ≥ 0, else 0)
Continuous threshold modification by excitation	$\theta(t+1) = \theta(t) + \varepsilon(\theta(t) - y(t))$	As above, except y = input*weights
Threshold modification by desired firing rate	$\theta(t+1) = \theta(t) + \varepsilon(\bar{z} - zpd)$ $\bar{z}(t+1) = \bar{z}(t) + \varepsilon(z - \bar{z}(t))$	As above, and $\bar{z}$ - average firing rate zpd - desired average z - current fire signal
Blinking response conditioning	$x_{cs} = CS * w$ $w_b(t+1) = w_b(t) + \varepsilon * e * x_{cs}(1 - w_b)$ $w_o(t+1) = w_o(t) - \varepsilon * e * x_{cs}(w_o)$	w_b - blinking neuron weight w_o - opening neuron weight x_{cs} - excitation of cs (0 or 1) multiplied by weight e - error signal

Appendix 4.3.

Recall the abstracted paradigm that defines learning and memory.

A system has distinguishable input states and distinguishable output states. Initially, a particular input, call it $x(0)$, produces a particular output, $y(0)$. Assume this input-to-output relationship is relatively stable and reproducible. That is, it does not change merely by time passing with no inputs.

Let the system experience some sequence of conditioning inputs, say $x(1), x(2), x(3)$, and we again probe the system with input $x(0)$. This time, however, the output is not the original $y(0)$ but some other output, $y(0)'$, for example.

Because the system was stable over time, we conclude something in the system has changed due to the input experience, $[x(1), x(2), x(3)]$. A change in the input function-output function is called learning. We label the change 'memory,' and it implies that something physical inside the system has also changed. That is, we usually assume that there is some physical change inside the system.

This general definition can be sharpened in several ways. Note that the demonstration of learning and memory are an inseparable pair. For example, we are generally interested in altered input and output functions that have some longevity. In the brain, long-term changes mediating a cognitive function are presumed to be some form of synaptic modification. That is, either quantitatively (synaptic strength) or qualitatively (forming new synapses or removing old synapses), there is a change in the effects neurons in the network have on each other.

Other less stable forms of memory, i.e., those with abbreviated longevity on a time scale of less than a second, occur and have their importance, but these will not be of interest here.

With this motivation and our credo, we study microscopic input-output changes between a pair of connected neurons with the conjecture that such changes can mediate a system wide, meaningful change of input-output function.

Here we study several types of synaptic modification that mediate associative encoding. Our interest in associative encoding arises from the nature of the learning that dominates our interpretation and responses to the world. Specifically, we are interested in learning that encodes associations between features, objects, and events.

Appendix 4.4.

```
function mainHW6()

      theta = 0.5; %threshold

      %set input
      CS = [0,0,1,1,1,0,0,0]; %represents the tone being on or off
                         %at each timestep of a given trial

      %set initial values
      granule = [0];    %granule cell state at each timestep, vector gets
      purkinje = [1];    %added to as we move through each timestep
      DCN = [1];

      %set initial weights
      w_input_granule  =  1;      %Weight for CS activation of granule cells
      w_preNeuron_postNeuron = 1; %This weight used for all other interneural
                                     connections in our example.  Adjust so
                                     that

      %here's an example of the first trial:
      for i = 2:8 %run from timestep 2 to timestep 8

            %Calculate yj, the excitation
            %Excitation = internal excitation + weights x inputs(on last
            previous step)
            granule(i) = 0 + w_input_granule * CS(i-1);
            purkinje(i) = 1 + w_preNeuron_postNeuron * granule(i-1);
            DCN(i) = 1 - w_preNeuron_postNeuron * purkinje(i-1);

            %Determine if neurons fire
            %Granule
            if (granule(i) >= theta)
                  granule(i) = 1; %excitation above threshold, fire
            else
                  granule(i) = 0; %don't fire
            end
            %Purkinje
            if (purkinje(i) >= theta)
                  purkinje(i) = 1; %excitation above threshold, fire
            else
                  purkinje(i) = 0; %don't fire
            end
            %DCN
            if (DCN(i) >= theta)
                  DCN(i) = 1; %excitation above threshold, fire
            else
                  DCN(i) = 0; %don't fire
            end
      end
end
```

Appendix 4.5.

Historical Notes

Such notation about the presynaptic and postsynaptic neurons is not universal, but it is easy to remember. It is also historical—this notation was used in the 1960's and 1970's by Anderson, Grossberg, Kohonen, Wigstrom, among others.