Physics 178/278 - David Kleinfeld - Winter 2022

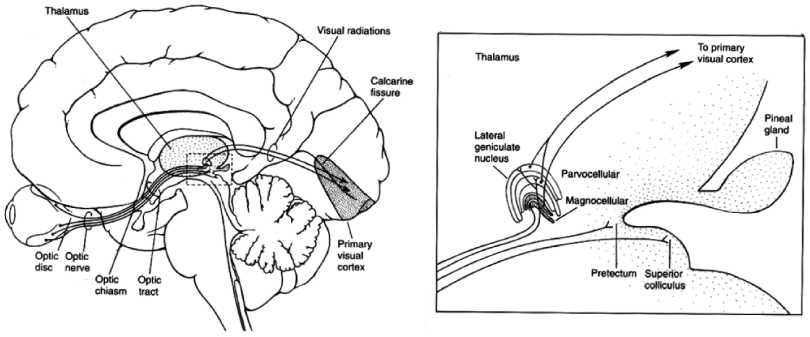
Lesson 18
Revised 9 March 2022 22:12

18 Synaptic Weights from Receptive Fields

18.1 General description of receptive fields

We consider a phenomenological description of the stimulus that causes a neuron to fire. Our description will be general. As a matter of practice, it is convenient to think in terms of the visual system (Figure 1) and visual objects (Figure 2), *i.e.*, a pattern of illumination that evolves over time and space. The receptive field forms a kernel, of filter, such that the spike rate of the cell is the temporal convolution of the stimulus with the receptive field and the spatial overlap of the stimulus with the receptive field. The way to think of this is that the inputs to cell comprise a set of photoreceptors, and each receptors has an accompanying synaptic weight and time dependence. This is a lot of information to specify. We shall see that in proactive there is typically only one or two time dependences, each with an accompanying weight matrix.

Figure 1: Overview of visual processing

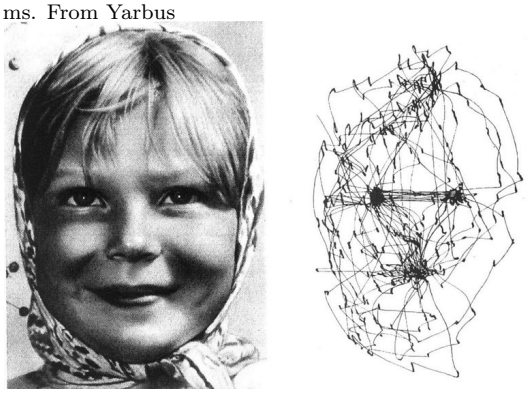


.

We define the inhomogeneous spike rate as $r(t)$. This is the rate that goes into a Poisson rate expression where, for example, the probability of no spikes in the interval $[0, t]$ and one spike in the interval;$(t, t + dt]$ is $r(t) \cdot exp\left(-\int_0^t dt' r(t')\right)$. Then

$$r(t) = f\left[I_o + \int_{-\infty}^{\infty} d^2\vec{x} \int_{-\infty}^{t} dt' I(\vec{x}, t') R(\vec{x}, t - t')\right] \qquad (18.1)$$

Figure 2: Focal attention on faces causes the visual gaze to be maintained at key locations for 100 ms. From Yarbus



where $f[\cdot]$ is the nonlinear input-output relation, $I(\vec{r},t)$ is the stimulus or input, $R(\vec{x},t)$ is the receptive field with $\vec{x}$ the two-dimensional spatial vector, and I_o is the baseline input.

When the stimuli driven part of the input is small compared to I_o, we can expand $g[\cdot]$ in a Taylor series and write

$$r(t) \simeq r_o + f' \int_{-\infty}^{\infty} d^2\vec{x} \int_{-\infty}^{t} dt' I(\vec{x},t') R(\vec{x},t-t') \tag{18.2}$$

where $r_o = f[I_o]$ and

$$f' = \left.\frac{df}{dI}\right|_{I=I_o} \tag{18.3}$$

so that the firing rate is a linear function of the stimulus. This allows us to focus on the receptive field without worrying about the nonlinearity $f[\cdot]$. Reviews by Chichilnisky (2001) and by Aljadeff, Lansdell, Fairhall and Kleinfeld (2017) addresses the assignment of both $R(\vec{x},t)$ and $f[\cdot]$ when the stimulus driven part of the input is not small compared to I_o. Aljadeff et al. (2016) also address high-order statistical descriptions of neuronal data.

The simplest procedure to define a receptive field is to compute the spike triggered average, which corresponds to the cross-correlation between the stimulus and the time of an action potential (Figure 3). This defines the space-time receptive field, as the averaging occurs for all lag times. It is illustrated for two classes of neurons in visual thalamus (Figure 4), magnocellular (fast, luminance) versus parvocellular (slow, chromatic plus luminance sensitive). Note that the receptive field does not have to be simple nor understandable in simple terms!

As a technical issue, any measure of activity can be used to define a receptive field - and that includes calcium signals from cells in cortex (Figure 5). In fact, this method permits the mapping

Figure 3: Receptive field mapping. From Chichilnisky, 2001

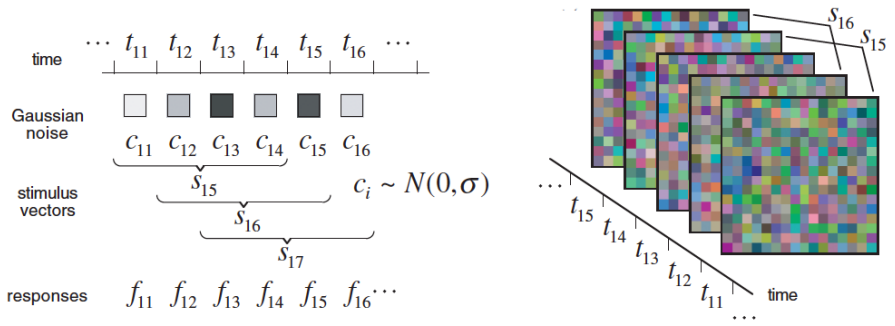
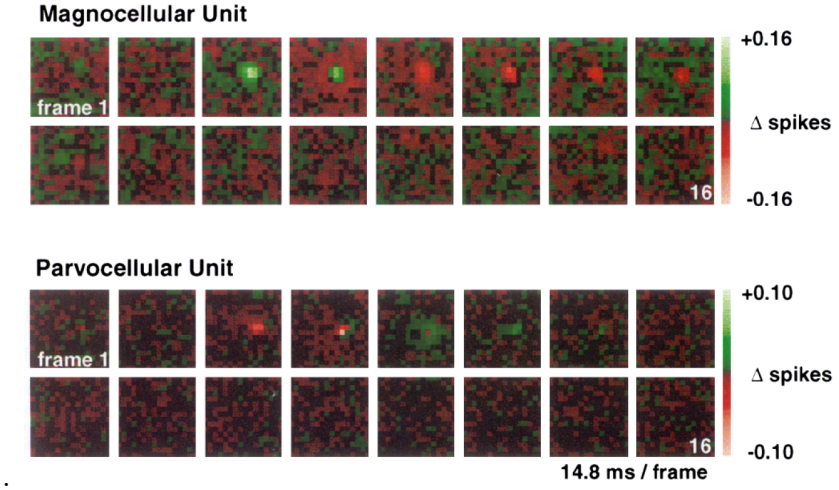


Figure 4: Spacetime receptive fields for thalamic (LGN) neurons in cat. From Golomb, Kleinfeld, Reid, Shapley and Shraiman, 1994



of receptive fields from very many neurons in cortex in the same imaging field (Figure 6).

To gain some insight into the general response properties of neurons, we recall that a matrix can always be expanded in terms of its eigenvectors by a singular valued decomposition. In terms of the receptive field, we have

$$R(\vec{x},t) \equiv \sum_{n=1}^{rank(R)} \lambda_n u_n(\vec{x}) v_n(t) \tag{18.4}$$

where the functions $u_n(\vec{x})$ form an orthonormal basis set in space and $v_n(t)$ for an orthonormal basis set in time. The eigenvalues for these basis sets are given by λ_n^2 and. of course, are ordered so that $\lambda_1 > \lambda_2 > \lambda_3 \cdots$. When λ_1 is the only significant term the receptive

Figure 5: Spatial receptive field for a L2/L3 neuron in mouse visual cortex measured from the neuronal Ca^{2+} response. The temporal dimension has been collapsed. From Cossell, Iacaruso, Muir, Houlton, Sader, Ko, Hofer and Mrsic-Flogel, 1994

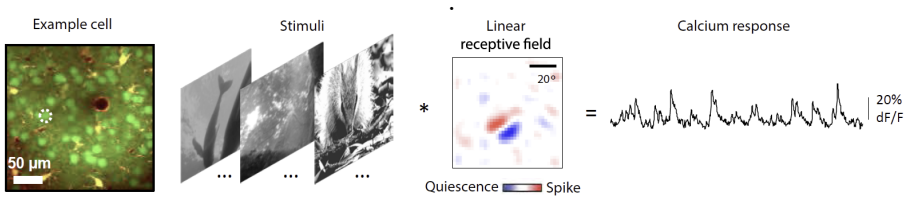
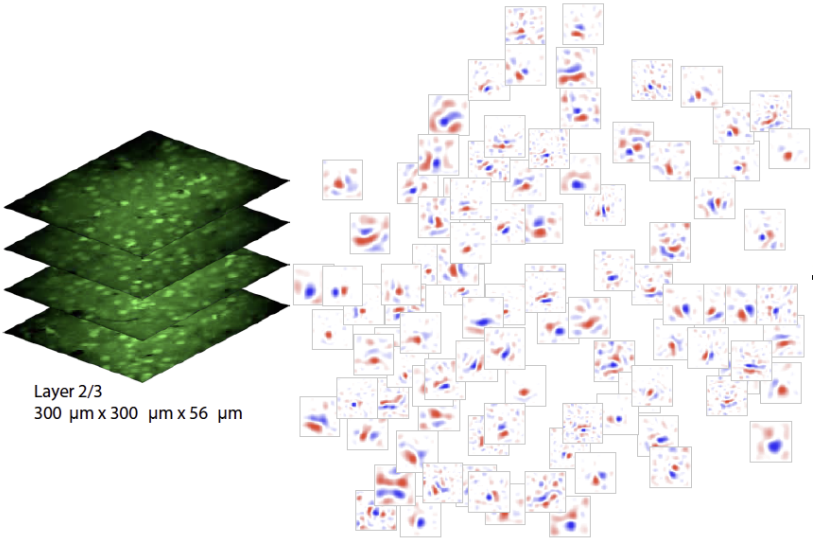


Figure 6: Spacetime receptive fields for multiple L2/L3 neurons. From Cossell, Iacaruso, Muir, Houlton, Sader, Ko, Hofer and Mrsic-Flogel, 1994



field is said to be separable, as the spatial and temporal functions factor (Figure 7).

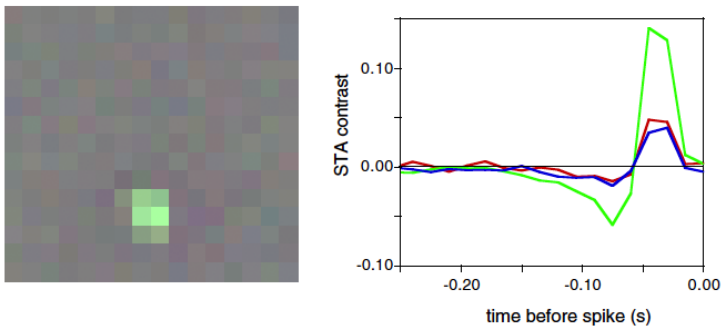
In general the receptive field is not separable, as first discussed by the work of McClean and Palmer (1989) (Figure 8) and analyzed in some detail by Golomb, Kleinfeld, Reid, Shapley and Shraiman (1994)(Figure 9; this is an analysis of the data in Figure 4). Then

$$r(t) \simeq r_o + f' \sum_{n=1}^{rank(R)} \lambda_n \int_{-\infty}^{\infty} d^2\vec{x} u_n(\vec{x}) \int_{-\infty}^{t} dt' I(\vec{x}, t') v_n(t - t'). \quad (18.5)$$

Now suppose that the stimulus is separable, as is often the case in primary sensory areas. For example, in vision our eyes shift from position to position about five times a second. In this case we may write

$$I(\vec{x}, t) \equiv X(\vec{x})T(t). \quad (18.6)$$

Figure 7: Separable visual receptive field. From Chichilnisky, 2001



.

so that

$$r(t) \simeq r_o + f' \sum_{n=1}^{rank(R)} \lambda_n \int_{-\infty}^{\infty} d^2\vec{x} X(\vec{x}) u_n(\vec{x}) \int_{-\infty}^{t} dt' T(t') v_n(t-t'). \tag{18.7}$$

The spatial part of the stimulus that each mode "sees" is given by the overlap integral of the spatial pattern of the stimulus with the spatial pattern of each mode, *i.e.*,

$$U_n = \int_{-\infty}^{\infty} d^2\vec{x}\ X(\vec{x})\ u_n(\vec{x}). \tag{18.8}$$

where the U_n are scalars. In this case the $u_n(\vec{x})$ act as the weights and the U_n are the output of say a dendritic branch as opposed to the entire cell.

The time dependence of the stimulus is convoluted with each of the associated temporal modes to form the temporal evolution for that mode, *i.e.*,

$$V_n(t) = \int_{-\infty}^{t} dt'\ T(t')\ v_n(t-t'). \tag{18.9}$$

where the $V_n(t)$ are functions. We thus find

$$r(t) = r_o + f' \sum_{n=1}^{rank(R)} \lambda_n\ U_n\ V_n(t). \tag{18.10}$$

so that each temporal waveform is weighted by the expansion coefficient for the receptive field and the spatial overlap of the mode with the stimulus. The point is that the temporal response of the neuron, given by $r(t)$, depends on the spatial pattern of the input as well as the temporal evolution of the stimulus. This is what some call a "temporal code", *i.e.*, the coding of different stimuli,

Figure 8: Spacetime receptive fields. From McClean and Palmer, 1989

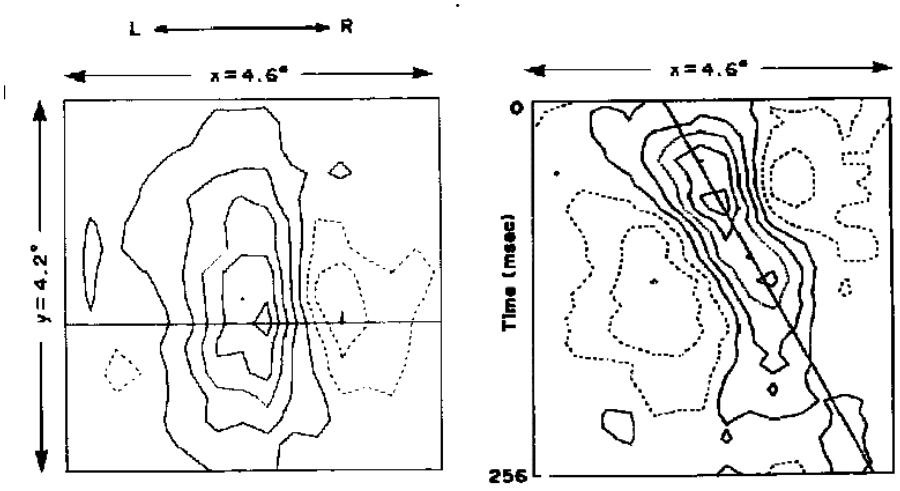
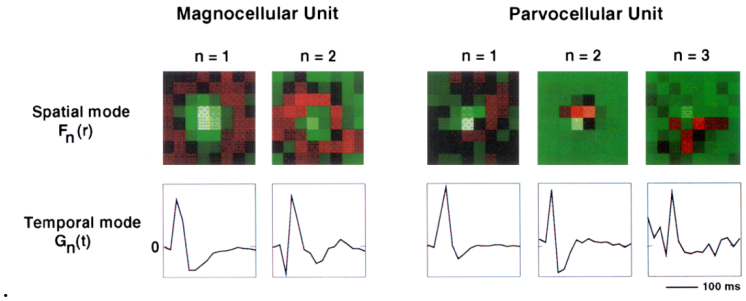


Figure 9: SVD modes of receptive fields from thalamus. From Golomb, Kleinfeld, Reid, Shapley and Shraiman, 1994



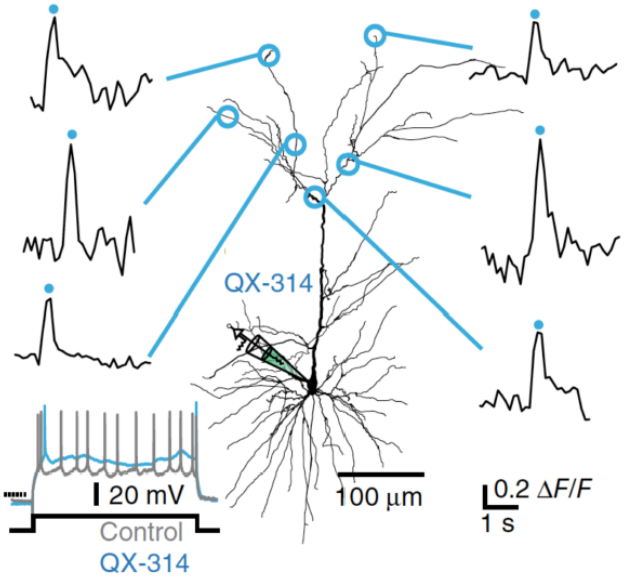
even quasi-static stimuli, by different temporal patterns of spike rates. The inhomogeneous rate $r(t)$ may evolve in time as fast as the response of the sensory cells, such at retinal ganglion cells for the case of vision.

A final point is that the summation over modes rarely contains more than a few terms, not the full rank of the matrix R. The spatial coefficient U_n has a signal-to-noise ratio that varies in proportion to λ_n for the $n-th$ mode. Thus the above series is cut off after two or three terms as the signal dives below the noise. The SVD expansion can be used as a data compression scheme in the description of the receptive field. For magnocellular cells,

$$r(t) \approx r_o + [f'\lambda_1 U_1]\, V_1(t) + [f'\lambda_2 U_2]\, V_2(t). \qquad (18.11)$$

Another interpretation is that each mode corresponds to the input to a different dendrite of a spatially extended neuron (Figure 10).

Figure 10: Near independent integration of inputs along different dendritic branches. From Palmer, Shai, Reeve, Anderson, Paulsen and Larkum, 2014



.

18.2 Singular value decomposition

In the expansion

$$R(\vec{x},t) \equiv \sum_{n=1}^{rank(R)} \lambda_n u_n(\vec{x}) v_n(t) \tag{18.12}$$

the functions satisfy the orthonormality constraints

$$\int_{-\infty}^{\infty} d^2\vec{x}\; u_n(\vec{x}) u_m(\vec{x}) = \delta_{nm} \tag{18.13}$$

and

$$\int_{-\infty}^{\infty} dt'\; v_n(t') v_m(t') = \delta_{nm}. \tag{18.14}$$

We now consider the contraction of the receptive field matrices to form a symmetric correlation matrix, *i.e.*,

$$\begin{aligned} C(t,t') &\equiv \int_{-\infty}^{\infty} d^2\vec{x} R(\vec{x},t) R(\vec{x},t') \qquad (18.15)\\ &= \sum_{n=1}^{rank(R)} \sum_{m=1}^{rank(R)} \lambda_n \lambda_m \int_{-\infty}^{\infty} d^2\vec{x}\; u_n(\vec{x}) u_m(\vec{x})\, v_n(t) v_m(t') \\ &= \sum_{n=1}^{rank(R)} \sum_{m=1}^{rank(R)} \lambda_n \lambda_m\, \delta_{nm}\, v_n(t) v_m(t') \\ &= \sum_{n=1}^{rank(R)} \lambda_n^2\, v_n(t) v_n(t'). \end{aligned}$$

Then $v_n(t)$ solves the eigenvalue equation

$$\begin{aligned}\int_{-\infty}^{\infty} dt' C(t,t') v_n(t') &= \sum_{m=1}^{rank(R)} \lambda_m^2 v_m(t) \int_{-\infty}^{\infty} dt' v_n(t') v_m(t') \\ &= \lambda_n^2 \, v_n(t)\end{aligned} \tag{18.16}$$

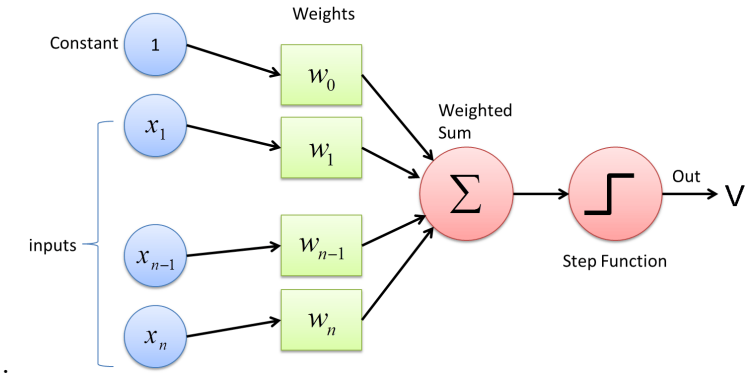
and the $u_n(\vec{x})$ are found from

$$\begin{aligned}\int_{-\infty}^{\infty} dt' R(\vec{x},t') v_n(t') &= \sum_{m=1}^{rank(R)} u_m(\vec{x}) \int_{-\infty}^{\infty} dt' \, v_m(t') v_n(t') \\ &= u_n(\vec{x}).\end{aligned} \tag{18.17}$$

18.3 Perceptrons

We interpret the receptive field as a feedforward network (Figure 11). These may be considered as the "front end" of nervous system, such as retinal ganglia cells that feed into neurons in the thalamus for the case of vision.

Figure 11: Classic perceptron.



We write the calculated output of the Perceptron, denoted $\hat{S}$, as

$$\hat{S} = f\left[\sum_{j=1}^{N} W_j x_j \; - \; \theta\right] \; = \; f\left[\vec{W} \cdot \vec{x} - \; \theta\right] \tag{18.18}$$

where the x's are the inputs in the form of an N-dimensional vector $\vec{x}$, "θ" is a threshold level, $f[\cdot]$ is a sigmoidal input-output function, and the W_j's are the weights to each of the inputs. We follow the scaling used throughout the feed forward network literature of choosing of $W \propto O\left(\frac{1}{N}\right)$, so that the sum is $O(1)$. One model for

f is

$$f[x] = tanh{-}\beta x \tag{18.19}$$

where β is the gain.

Consider the case of an AND gate with two inputs and one true output, y. We restrict ourselves to $N = 2$ inputs solely as a means to be able to draw pictures; a model for integration of inputs by a neuron may consist of $N \approx 10,000$ inputs!

x_1	x_2	S
-1	-1	-1
1	-1	-1
-1	1	-1
1	1	1

(18.20)

If we look at the input and require that it is positive for $W_1x_1 + W_2x_2 - \theta > 0$ and negative for $W_1x_1 + W_2x_2 - \theta < 0$, we will have

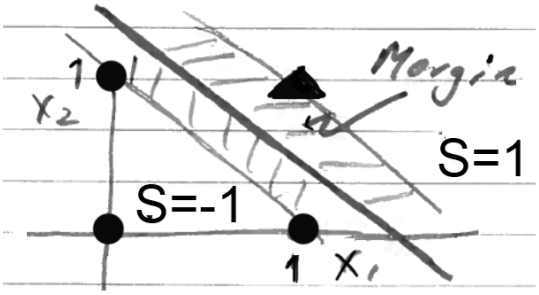
$$\begin{aligned} -W_1 - W_2 &< \theta \\ W_1 - W_2 &< \theta \\ W_2 - W_2 &< \theta \\ W_1 + W_2 &> \theta \end{aligned} \tag{18.21}$$

There are an infinite number of set of values of W_1, W_2, and θ that will work. One that gives the largest margin (Figure 12), *i.e.*, is least susceptibility to variations in the input, is the choice

$$\begin{aligned} W_1 &= 1 \\ W_2 &= 1 \\ \theta &= 1 \end{aligned} \tag{18.22}$$

.

Figure 12: Separation line with greatest margin.



.

This defines a line that divides the "1" output from the "-1" output. The "OR" function is similarly defined, except that $\theta = 1/2$.

So far so good. But we observe that "XOR" cannot be described by this formalism, as there are now two regions with a positive output, not one. So we cannot split the space of inputs with a single line.

x_1	x_2	V
-1	-1	-1
1	-1	1
-1	1	1
1	1	-1

(18.23)

The "XOR" can be solved by introducing an additional dimension so that we can split the space in 3-dimensions by a plane.

18.4 Perceptron learning - Binary gain function

For a binary input-output function, we write a Hebb-like construction rule for the values of $\vec{W}$ starting from $\vec{W} = (0,0)$ and using the known pairs of inputs and outputs. We write

$$\vec{W} \leftarrow \vec{W} + S\vec{x}. \tag{18.24}$$

when $\hat{S} \neq S$ and is unchanged for $\hat{S} = S$.

18.4.1 Convergence of perceptron learning

We now consider a proof that the Perceptron can aways learn a rule when there is a plane that divides the input space. Consider "n" sets of Boolean functions with

$$\text{Input(n)} \equiv \vec{x}(n) = \{x_1(n), x_2(n), \cdots, x_N(n)\} \tag{18.25}$$

and

$$\text{True output(n)} \equiv S(n) = \pm 1. \tag{18.26}$$

Training consists of learning the $\vec{W}$s from the different sets of $\vec{x}(n)$ and $S(n)$, denoted $\{S(k), \vec{x}(k)\}$. We calculate the predicted output above from each $\vec{x}$ using the old values of $\vec{W}$s and compare it with the true of $S(n)$. Specifically

$$\text{Calculated output} \equiv \tilde{S}(n) = f[\vec{W}(n) \cdot \vec{x}(n)]. \tag{18.27}$$

The update rule to the $\vec{W}$s is in terms of the True output and the Calculated output, *i.e.*,

$$\begin{aligned} \vec{W}(m+1) &= \vec{W}(m) + \frac{1}{2}\left[S(n) - \tilde{S}(n)\right]\vec{x}(n) \\ &= \vec{W}(m) + \frac{1}{2}\left[S(n) - f[\vec{W}(n) \cdot \vec{x}(n)]\right]\vec{x}(n) \end{aligned} \tag{18.28}$$

where we use the notation $\vec{W}(m)$ for the set of weights after m iterations of learning. Clearly we have used $m \geq n$. Correct categorization will lead to

$$\tilde{S}(n) = S(n) \quad \text{and; implies} \quad \vec{W}(m+1) = \vec{W}(m) \tag{18.29}$$

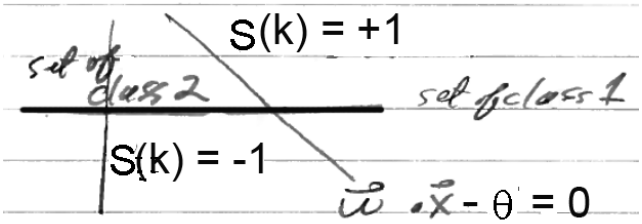
while incorrect categorization leads to a change in weights

$$\tilde{S}(n) \neq S(n) \quad \text{implies} \quad \vec{W}(m+1) = \begin{cases} \vec{W}(m) + \vec{x}(n) & \text{if } \vec{W}(n) \cdot \vec{x}(n) < 0 \\ \vec{W}(m) - \vec{x}(n) & \text{if } \vec{W}(n) \cdot \vec{x}(n) > 0. \end{cases} \tag{18.30}$$

One way to look at learning is that the examples can be divided into two training sets (Figure 13):

$$\begin{array}{lllll} \text{Set of class 1 examples} & \{S(k), \vec{x}(k)\} & \text{with} & S(k) = +1 & \forall \\ \text{Set of class 2 examples} & \{S(k), \vec{x}(k)\} & \text{with} & S(k) = -1 & \forall \quad k. \end{array} \tag{18.31}$$

Figure 13: Categorization by Perceptron.



An important point about the use of Perceptrons is the existence of a learning rule that can be proved to converge. The idea in the proof is to show that with consideration of more and more examples, *i.e.*, with increasing n, the corrections to the $\vec{W}(n)$ grow faster than the number of errors.

18.4.2 Growth of corrections to the $\vec{W}(n)$ as a function of iteration

Suppose we make m errors, which leads to m updates, among our set of n input-output pairs. That is, $\vec{W}(m) \cdot \vec{x}(m) < 0 \, \forall m$ for the set of class 1 examples, yet $S(m) = +1$. Let us estimate how the corrections to the $\vec{W}(m)$s grow as a function of the number of learning steps, *e.g.*, as m, m^2, m^3, *etc.* The update rule is

$$\begin{aligned} \vec{W}(m+1) &= \vec{W}(m) + \vec{x}(m) \\ &= \vec{W}(m-1) + \vec{x}(m-1) + \vec{x}(m) \\ &= \vec{W}(m-2) + \vec{x}(m-2) + \vec{x}(m-1) + \vec{x}(m) \\ &= \cdots \\ &= \vec{W}(0) + \sum_{k=0}^{m} \vec{x}(k). \end{aligned} \tag{18.32}$$

In the above, all of the m corrections made use of the first m entries of the set of class 1 examples. With no loss of generality, we take the initial value of the weight vector as $\vec{W}(0) = 0$, so that

$$\vec{W}(m+1) = \sum_{k=0}^{m} \vec{x}(k). \tag{18.33}$$

Now consider a solution to the Perceptron, denoted $\vec{W}_1$, that is based on the set of class 1 examples; by definition there is no index to this set of weights. Further, this satisfies $\vec{W}_1 \cdot \vec{x}(m) > 0 \, \forall m$ for any set of $\vec{x}$. We use the overlap of $\vec{W}_1$ as a means to form bounds on the corrections with increasing iterations of learning. We have

$$\begin{aligned} \vec{W}_1 \cdot \vec{W}(m+1) &= \sum_{k=1}^{m} \vec{W}_1 \cdot \vec{x}(k) \\ &\geq m \times \text{minimum} \left\{ \vec{W}_1 \cdot \vec{x}(k) \right\} \end{aligned} \tag{18.34}$$

where $\vec{x}(k) \in$ set 1 examples. Then

$$\|\vec{W}_1 \cdot \vec{W}(m+1)\| \geq m \times \min \left\{ \vec{W}_1 \cdot \vec{x}(k) \right\}. \tag{18.35}$$

But by the Cauchy-Schwartz inequality,

$$\|\vec{W}_1\| \|\vec{W}(m+1)\| \; \geq \; \|\vec{W}_1 \cdot \vec{W}(m+1)\| \tag{18.36}$$

so

$$\|\vec{W}_1\| \|\vec{W}(m+1)\| \; \geq \; m \times \min \left\{ \vec{W}_1 \cdot \vec{x}(k) \right\} \tag{18.37}$$

or

$$\|\vec{W}(m+1)\| \; \geq \; m \times \frac{\min \left\{ \vec{W}_1 \cdot \vec{x}(k) \right\}}{\|\vec{W}_1\|} \tag{18.38}$$

and we find that the correction to the weight vector $\vec{W}$ after m steps of learning scales as m.

18.4.3 Growth of errors in the $\vec{W}(n)$ as a function of learning

We now estimate how the error to the weight vector $\vec{W}(m)$ grows as a function as the number of learning steps. The error can grow as each learning step can add noise as well as corrects for errors in the output $\hat{y}(m)$. The key for convergence is that the error grows more slowly than the correction, *i.e.*, as most as $m^{2-\epsilon}$. We start with the change in the weight vector as as function of the update step. After m updates, we have

$$\vec{W}(m+1) = \vec{W}(m) \; + \; \vec{x}(m). \tag{18.39}$$

But

$$\begin{aligned} \|\vec{W}(m+1)\|^2 &= \|\vec{W}(m) + \vec{x}(m))\|^2 \\ &= \|\vec{W}(m)\|^2 + \|\vec{x}(m)\|^2 + 2\vec{W}(m)\cdot\vec{x}(m) \\ &\leq \|\vec{W}(m)\|^2 + \|\vec{x}(m)\|^2 \end{aligned} \tag{18.40}$$

so

$$\|\vec{W}(m+1)\|^2 - \|\vec{W}(m)\|^2 \leq \|\vec{x}(m)\|^2. \tag{18.41}$$

Now we can iterate:

$$\begin{aligned} \|\vec{W}(m+1)\|^2 - \|\vec{W}(m)\|^2 &\leq \|\vec{x}(m)\|^2 \\ \|\vec{W}(m)\|^2 - \|\vec{W}(m-1)\|^2 &\leq \|\vec{x}(m-1)\|^2 \\ &\cdots \\ \|\vec{W}(1)\|^2 - \|\vec{W}(0)\|^2 &\leq \|\vec{x}(0)\|^2. \end{aligned} \tag{18.42}$$

We sum the right and left sides separately, and again take $\vec{W}(0) = 0$, to get

$$\begin{aligned} \|\vec{W}(m+1)\|^2 &\leq \sum_{k=0}^{m} \|\vec{x}(k)\|^2 \\ &\leq (m+1) \times \text{maximum}\left\{\|\vec{x}(k)\|^2\right\}. \end{aligned} \tag{18.43}$$

Thus we find that the errors to the weight vector $\vec{W}$ after m corrections scale as $\sqrt{m+1} \approx \sqrt{m}$, i.e.,

$$\|\vec{W}(m+1)\| \leq \sqrt{m \times \max\{\|\vec{x}(k)\|^2\}}. \tag{18.44}$$

or

$$\sqrt{m \times \max\{\|\vec{x}(k)\|^2\}} \geq \|\vec{W}(m+1)\|. \tag{18.45}$$

18.4.4 Proof of convergence

We now have two independent constraints on $\|\vec{W}(m+1)\|$:

$$\sqrt{m \times \max\{\|\vec{x}(k)\|^2\}} \geq \|\vec{W}(m+1)\| \geq m \times \frac{\min\left\{\vec{W}_1 \cdot \vec{x}(k)\right\}}{\|\vec{W}_1\|} \tag{18.46}$$

and thus

$$m \geq \frac{\max\{\|\vec{x}(k)\|^2\}}{\left(\min\left\{\vec{W}_1 \cdot \vec{x}(k)\right\}\right)^2} \|\vec{W}_1\|^2. \tag{18.47}$$