

# Appendix A

## Some Mathematical Results

### A.1 Discrete Versions of BCM Equations

In the previous work we used the following equations:

$$y = \mathbf{w} \cdot \mathbf{x} \tag{1.24}$$

$$\dot{\mathbf{w}} = \eta y(y - \theta)\mathbf{x} \tag{1.25}$$

$$\theta = E_\tau [y^2] \tag{1.26}$$

where we will define  $E_\tau [y^2] \equiv \frac{1}{\tau} \int_{-\infty}^t y^2(t') e^{-(t-t')/\tau} dt'$ .

We make one change from these equations in order to make them more easily solved with numerical methods, by replacing the equation for  $\theta$  with an equation for  $\dot{\theta}$ :

$$\dot{\theta} = \frac{1}{\tau}(y^2 - \theta) \tag{A.1}$$

Though one can easily see that this equation has the same *qualitative* behavior as Equation 1.26, one can show that it is functionally *identical* to Equation 1.26. To show this we use a Green's function method. We define a function  $G(t, t')$  as a solution to the equation

$$\frac{dG(t, t')}{dt} + \frac{1}{\tau}G(t, t') = \delta(t - t') \tag{A.2}$$

which allows us to write  $\theta$  in the form

$$\theta = \int_{-\infty}^{\infty} G(t, t') \left( \frac{1}{\tau} y^2(t') \right) dt' \tag{A.3}$$

Now all we need to do is solve for  $G(t, t')$ . We note that for  $t \neq t'$  the equation has the solution

$$G(t, t') = A(t') \exp(-t/\tau) \quad \text{if } t < t' \tag{A.4}$$

$$G(t, t') = B(t') \exp(-t/\tau) \quad \text{if } t > t' \tag{A.5}$$

Next we integrate Equation A.2 from  $t = t' + \epsilon$  to  $t = t' - \epsilon$ , which gives the jump condition

$$\begin{aligned} G(t' + \epsilon) - G(t' - \epsilon) &= 1 \\ &= (B(t') - A(t')) \exp(-t'/\tau) \end{aligned} \tag{A.6}$$

Finally we impose causality, which forces  $A(t') = 0$ , and thus arrive at a final form for  $G(t, t')$ .

$$G = \begin{cases} 0 & \text{if } t < t' \\ \exp(-(t - t')/\tau) & \text{if } t > t' \end{cases} \quad (\text{A.7})$$

and then, through Equation A.3, gives us

$$\begin{aligned} \theta &= \int_{-\infty}^{\infty} G(t, t') \left( \frac{1}{\tau} y^2(t') \right) dt' \\ &= \frac{1}{\tau} \int_{-\infty}^t y^2(t') e^{-(t-t')/\tau} dt' \end{aligned} \quad (\text{A.8})$$

which is identical to Equation 1.26!

It then follows that an Euler's method approximation would yield discrete equations

$$\begin{aligned} y_n &= \mathbf{w}_n \cdot \mathbf{x}_n \\ &= \sum_i (\mathbf{w}_n)_i (\mathbf{x}_n)_i \end{aligned} \quad (\text{A.9})$$

$$\theta_{n+1} = \theta_n + \frac{1}{\tau} (y_n^2 - \theta_n) \quad (\text{1.12})$$

$$(\mathbf{w}_i)_{n+1} = (\mathbf{w}_i)_n + \eta y_n (y_n - \theta_{n+1}) (\mathbf{x}_i)_n$$

## A.2 Calculation and Self Consistency Check for BCM Oscillations

We use

$$\begin{aligned} y &= wx \\ \dot{w} &= \eta y (y - \theta) x \\ \theta &= \frac{1}{\tau} \int_0^t y^2(t') e^{-(t-t')/\tau} dt' + e^{-t/\tau} \theta_o \end{aligned}$$

and then assume

$$w = w_o + w_1 e^{i\omega t} e^{-gt}$$

where  $w_1 \ll 1$ ,  $w_o = 1/x$  and  $g$  is a positive damping constant.

$$\begin{aligned} \theta &= \frac{1}{\tau} \int_0^t (w_o + w_1 e^{i\omega t'} e^{-gt'})^2 x^2 e^{-(t-t')/\tau} dt' + e^{-t/\tau} \theta_o \\ &\stackrel{w_1 \ll 1}{\approx} \frac{1}{\tau^2 \omega^2 + g^2 \tau^2 - 2g\tau + 1} \left\{ [2w_1 x \tau \omega] e^{-gt} \sin(\omega t) + [2w_1 x (1 - g\tau)] e^{-gt} \cos(\omega t) \right. \\ &\quad \left. + [2w_1 x (g\tau - 1)] e^{-t/\tau} \right\} + [(\theta_o - 1)] e^{-t/\tau} \\ &\quad + \frac{i}{\tau^2 \omega^2 + g^2 \tau^2 - 2g\tau + 1} \left\{ [2w_1 x \tau \omega] e^{-gt} \cos(\omega t) + [2w_1 x (1 - g\tau)] e^{-gt} \sin(\omega t) \right. \\ &\quad \left. + [2w_1 x \tau \omega] e^{-t/\tau} \right\} \end{aligned}$$

$$\dot{w} - \eta y (y - \theta) x = 0$$

$$w_1 \approx \left\{ \frac{A_1}{B} e^{-gt} \sin(\omega t) + \frac{A_2}{B} e^{-gt} \cos(\omega t) + A_3 e^{-t/\tau} e^{-gt} \cos(\omega t) + \frac{A_4 + A_5}{B} e^{-t/\tau} \right\} \\ + i \left\{ \frac{-A_1}{B} e^{-gt} \cos(\omega t) + \frac{A_2}{B} e^{-gt} \sin(\omega t) + A_3 e^{-t/\tau} e^{-gt} \sin(\omega t) + \frac{A_6}{B} e^{-t/\tau} \right\}$$

$$A_1 = \omega w_1 (-\tau^2 \omega^2 + g^2 \tau^2 - 2g\tau + 1) + 2\eta \tau x^2 = 0 \quad (\text{A.10})$$

$$A_2 = w_1 (-g \cdot (\tau^2 \omega^2 + g^2 \tau^2 - 2g\tau + 1) - x^2 \eta (\tau^2 \omega^2 + g^2 \tau^2 + 1)) = 0 \quad (\text{A.11})$$

$$A_3 = x^2 \eta w_1 (\theta_o - 1) = 0 \quad (\text{A.12})$$

$$A_4 = B \cdot (\theta_o - 1) \eta x$$

$$A_5 = 2w_1 \eta x^2 (g\tau - 1)$$

$$A_4 + A_5 = 0 \quad (\text{A.13})$$

$$A_6 = 2\eta x^2 w_1 \tau \omega \quad (\text{A.14})$$

$$B = \tau^2 \omega^2 + g^2 \tau^2 - 2g\tau + 1 \neq 0 \quad (\text{A.15})$$

Yielding, if we drop Equations A.13 and A.14 for consistency,

$$\alpha \equiv \tau \eta x^2$$

$$\omega = \pm \frac{1}{2} \frac{\sqrt{6\alpha - 1 - \alpha^2}}{\tau} \quad (\text{1.15})$$

$$g = \frac{1}{2} \frac{(1 - \alpha)}{\tau} \quad (\text{1.16})$$

A self-consistency check quickly shows that the approximation we used to drop the terms in Equations A.13 and A.14 to obtain the results of Equations 1.13-1.14 is indeed valid. We look at size of each exponential decay over one period:

$$t = \frac{2\pi}{\omega} = \frac{4\pi\tau}{\sqrt{6\alpha - 1 - \alpha^2}} \equiv \tau K$$

$$.2 < \alpha < 1$$

$$\infty > K > 2\pi$$

$$e^{-gt} = e^{-\frac{1}{2}(1-\alpha)K}$$

$$e^{-gt} e^{-t/\tau} = e^{-\frac{1}{2}(1-\alpha)K} e^{-K}$$

$$e^{-t/\tau} = e^{-K}$$

$$2 \cdot 10^{-3} > e^{-K} > e^{-K} e^{+\frac{1}{2}(1-\alpha)K} > 0$$

So we expect the approximation to fail when  $\alpha$  is small, but that's when we expect it to fail anyway!

### A.3 Full Solution for 2D BCM fixed points

We start with the BCM equations

$$y = \mathbf{w} \cdot \mathbf{x} \quad (1.24)$$

$$\dot{\mathbf{w}} = \eta y (y - \theta) \mathbf{x} \quad (1.25)$$

$$\theta = E_\tau [y^2] \quad (1.26)$$

and assume we have two input patterns,  $\mathbf{x}_1$  and  $\mathbf{x}_2$ , presented to the neuron with probabilities  $p_1$  and  $p_2$ , respectively. As shown earlier, the stable fixed points occur when

- $(\mathbf{w} \cdot \mathbf{x}_1) = 0$  and  $(\mathbf{w} \cdot \mathbf{x}_2) = \theta \neq 0$ : weight orthogonal to  $\mathbf{x}_1$  and has projection  $\theta$  on  $\mathbf{x}_2$
- $(\mathbf{w} \cdot \mathbf{x}_1) = \theta \neq 0$  and  $(\mathbf{w} \cdot \mathbf{x}_2) = 0$ : weight orthogonal to  $\mathbf{x}_2$  and has projection  $\theta$  on  $\mathbf{x}_1$

Without loss of generality, we will assume the latter case, giving us the following equations.

$$y_1 = \mathbf{w} \cdot \mathbf{x}_1 = \theta \quad (A.16)$$

$$y_2 = \mathbf{w} \cdot \mathbf{x}_2 = 0 \quad (A.17)$$

it follows

$$\theta = E[y^2] = p_1 y_1^2 + p_2 y_2^2 \quad (A.18)$$

$$= p_1 y_1^2 \quad (A.19)$$

and from Equation A.16 we get

$$\begin{aligned} y_1 &= p_1 y_1^2 \\ \Rightarrow y_1 &= \frac{1}{p_1} \end{aligned} \quad (A.20)$$

If we now define

$$\mathbf{w} \equiv \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \quad (A.21)$$

$$\mathbf{x}_1 \equiv \begin{pmatrix} x_{11} \\ x_{12} \end{pmatrix} \quad (A.22)$$

$$\mathbf{x}_2 \equiv \begin{pmatrix} x_{21} \\ x_{22} \end{pmatrix} \quad (A.23)$$

then we can solve for  $\mathbf{w}$ .

$$w_1 x_{11} + w_2 x_{12} = \frac{1}{p_1} \quad (A.24)$$

$$w_1 x_{21} + w_2 x_{22} = 0 \quad (A.25)$$

which yields

$$\mathbf{w} = \left( \begin{array}{cc} \frac{\frac{1}{p_1} x_{22}}{x_{11} x_{22} - x_{12} x_{21}} & \frac{-\frac{1}{p_1} x_{21}}{x_{11} x_{22} - x_{12} x_{21}} \end{array} \right)^T$$

as one fixed point. The other can be easily found.

$$\mathbf{w} = \left( \frac{-\frac{1}{p_2}x_{12}}{x_{11}x_{22} - x_{12}x_{21}} \quad \frac{\frac{1}{p_2}x_{11}}{x_{11}x_{22} - x_{12}x_{21}} \right)^T$$

## A.4 Full Solution for 2D PCA fixed points

It was shown earlier that the fixed points for PCA are the eigenvectors of the correlation matrix. It is then straightforward to calculate these eigenvectors in the two dimensional case.

If we have the two input patterns defined as

$$\mathbf{x}_1 \equiv \begin{pmatrix} x_{11} \\ x_{12} \end{pmatrix} \tag{A.26}$$

$$\mathbf{x}_2 \equiv \begin{pmatrix} x_{21} \\ x_{22} \end{pmatrix} \tag{A.27}$$

then the correlation matrix is

$$\begin{aligned} \mathbf{C} &= E[\mathbf{xx}^T] \tag{A.28} \\ &= \frac{1}{2} \left[ \begin{pmatrix} x_{11}^2 & x_{11}x_{12} \\ x_{11}x_{12} & x_{12}^2 \end{pmatrix} + \begin{pmatrix} x_{21}^2 & x_{21}x_{22} \\ x_{21}x_{22} & x_{22}^2 \end{pmatrix} \right] \\ &= \begin{pmatrix} \frac{1}{2}(x_{11}^2 + x_{21}^2) & \frac{1}{2}(x_{11}x_{12} + x_{21}x_{22}) \\ \frac{1}{2}(x_{11}x_{12} + x_{21}x_{22}) & \frac{1}{2}(x_{12}^2 + x_{22}^2) \end{pmatrix} \\ &\equiv \begin{pmatrix} C_1 & C_2 \\ C_3 & C_4 \end{pmatrix} \tag{A.29} \end{aligned}$$

where

$$C_1 \equiv \frac{1}{2}(x_{11}^2 + x_{21}^2) \tag{A.30}$$

$$C_2 = C_3 \equiv \frac{1}{2}(x_{11}x_{12} + x_{21}x_{22}) \tag{A.31}$$

$$C_4 \equiv \frac{1}{2}(x_{12}^2 + x_{22}^2) \tag{A.32}$$

We now want to find the eigenvector of  $\mathbf{C}$  with the maximum eigenvalue.

$$\begin{aligned} \begin{vmatrix} C_1 - \lambda & C_2 \\ C_2 & C_4 - \lambda \end{vmatrix} &= 0 \tag{A.33} \\ &= (C_1 - \lambda)(C_4 - \lambda) - C_2^2 \\ &= \lambda^2 - (C_1 + C_4)\lambda + C_1C_4 - C_2^2 \\ \lambda^\pm &= \frac{(C_1 + C_4) \pm \sqrt{(C_1 + C_4)^2 - 4(C_1C_4 - C_2^2)}}{2} \\ &= \frac{(C_1 + C_4) \pm \sqrt{(C_1 - C_4)^2 + 4C_2^2}}{2} \tag{A.34} \end{aligned}$$

For the maximum eigenvalue,  $\lambda^+$ , we obtain the eigenvector,  $\mathbf{v} \equiv (v_1 \ v_2)^T$ .

$$C_1 v_1 + C_2 v_2 = \lambda^+ v_1 \tag{A.35}$$

$$C_2 v_1 + C_4 v_2 = \lambda^+ v_2 \tag{A.36}$$

$$\Rightarrow v_1 = \frac{\lambda^+ - C_4}{C_2} v_2 \tag{A.37}$$

So we obtain the final solution

$$\mathbf{v} = \frac{1}{\xi} \begin{pmatrix} (\lambda^+ - C_4)/C_2 \\ 1 \end{pmatrix} \tag{A.38}$$

where  $\xi$  is defined to normalize the vector, and  $\lambda^+, C_2$  and  $C_4$  were defined earlier.

## A.5 Stability of BCM fixed points

### A.5.1 Some useful formulae and theorems

•

$$g(y) = E_x[f(x, y)] \Rightarrow \frac{\partial}{\partial y} g(y) = E_x \left[ \frac{\partial}{\partial y} f(x, y) \right]$$

•

$$\begin{aligned} \frac{\partial z}{\partial \mathbf{x}} \equiv \nabla_{\mathbf{x}} z &= \left( \frac{\partial z}{\partial x_1} \hat{\mathbf{x}}_1 + \frac{\partial z}{\partial x_2} \hat{\mathbf{x}}_2 + \dots \right) \\ &= \left( \frac{\partial z}{\partial t} \frac{\partial t}{\partial x_1} \hat{\mathbf{x}}_1 + \frac{\partial z}{\partial t} \frac{\partial t}{\partial x_2} \hat{\mathbf{x}}_2 + \dots \right) \\ &= \frac{\partial z}{\partial t} \left( \frac{\partial}{\partial x_1} \hat{\mathbf{x}}_1 + \frac{\partial}{\partial x_2} \hat{\mathbf{x}}_2 + \dots \right) t \end{aligned}$$

$$\frac{\partial z}{\partial \mathbf{x}} = \frac{\partial z}{\partial t} \frac{\partial t}{\partial \mathbf{x}}$$

•

$$\begin{aligned} \frac{\partial f(\mathbf{x} \cdot \mathbf{a})}{\partial \mathbf{x}} \equiv \nabla_{\mathbf{x}} f(\mathbf{x} \cdot \mathbf{a}) &= \left( \frac{\partial}{\partial x_1} \hat{\mathbf{x}}_1 + \frac{\partial}{\partial x_2} \hat{\mathbf{x}}_2 + \dots \right) f(\mathbf{x} \cdot \mathbf{a}) \\ &= \hat{\mathbf{x}}_1 (a_1 f'(c)|_{\mathbf{x} \cdot \mathbf{a}}) + \hat{\mathbf{x}}_2 (a_2 f'(c)|_{\mathbf{x} \cdot \mathbf{a}}) + \dots \\ \{\hat{\mathbf{a}}_i = \hat{\mathbf{x}}_i\} \Rightarrow \nabla_{\mathbf{x}} f(\mathbf{x} \cdot \mathbf{a}) &= f'(c)|_{\mathbf{x} \cdot \mathbf{a}} \mathbf{a} \end{aligned}$$

$$\frac{\partial f(\mathbf{x} \cdot \mathbf{a})}{\partial \mathbf{x}} = f'(c)|_{\mathbf{x} \cdot \mathbf{a}} \mathbf{a}$$

•

$$\begin{aligned} \frac{\partial z}{\partial t} &= \frac{\partial z}{\partial x_1} \frac{\partial x_1}{\partial t} + \frac{\partial z}{\partial x_2} \frac{\partial x_2}{\partial t} + \dots \\ &= (\nabla_{\mathbf{x}} z) \cdot \frac{\partial \mathbf{x}}{\partial t} \end{aligned}$$

$$\boxed{\frac{\partial z}{\partial t} = \frac{\partial z}{\partial \mathbf{x}} \cdot \frac{\partial \mathbf{x}}{\partial t}}$$

- *Theorem:* if  $\mathbf{x}_o$  extremizes  $f(\mathbf{x})$  on curve  $\mathbf{r}(t)$ , then  $\nabla f(\mathbf{x}_o) \perp \mathbf{r}'(t_o)$  where  $t_o$  is defined such that  $\mathbf{r}(t_o) = \mathbf{x}_o$ .

*Proof:*

$$\begin{aligned} \frac{d}{dt}f(\mathbf{r}(t)) &= \nabla f(\mathbf{r}(t)) \cdot \mathbf{r}'(t) \\ \left. \frac{d}{dt}f(\mathbf{r}(t)) \right|_{t_o} = 0 &= \nabla f(\mathbf{r}(t_o)) \cdot \mathbf{r}'(t_o) \\ &\Rightarrow \nabla f(\mathbf{r}(t_o)) \perp \mathbf{r}'(t_o) \\ \frac{d}{dt}\mathbf{r}(t) &\equiv \mathbf{r}'(t) \text{ is tangent to } \mathbf{r}(t) \\ &\Rightarrow \nabla f(\mathbf{r}(t_o)) \perp \mathbf{r}(t_o) \end{aligned}$$

- *Theorem:* if  $\mathbf{x}_o$  extremizes  $f(\mathbf{x})$  such that constraint  $g(\mathbf{x}) = 0$ , then  $\nabla f(\mathbf{x}_o) \parallel \nabla g(\mathbf{x}_o)$ .

*Proof:* **define** tangent vector  $\mathbf{t}(\mathbf{x})$  such that  $\nabla g(\mathbf{x}) \cdot \mathbf{t}(\mathbf{x}) = 0$ .

$$\begin{aligned} (\max / \min) [f(\mathbf{x})] \text{ with } g(\mathbf{x}) = 0 &\Rightarrow (\max / \min) [f(\mathbf{x})] \text{ on curve } \mathbf{t}(\mathbf{x}) \\ &\Rightarrow \nabla f(\mathbf{x}_o) \cdot \mathbf{t}(\mathbf{x}_o) = 0 = \nabla g(\mathbf{x}_o) \cdot \mathbf{t}(\mathbf{x}_o) \\ &\Rightarrow \nabla f(\mathbf{x}_o) = \lambda \nabla g(\mathbf{x}_o) \end{aligned}$$

- To obtain the stability of the stationary points of the function  $f(\mathbf{x})$ :

- Form the matrix of second derivatives

$$\mathbf{H}_{ij} = \frac{d^2 f(\mathbf{x})}{dx_i dx_j} = (\nabla_{\mathbf{x}} \otimes \nabla_{\mathbf{x}}) f(\mathbf{x})$$

- Find its eigenvalues

- If they are all positive  $\Rightarrow$  minimum. all negative  $\Rightarrow$  maximum. some of each  $\Rightarrow$  saddle point.

- Outer product

$$\begin{aligned} \mathbf{x} \otimes \mathbf{y} &\equiv \begin{pmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_N \\ x_2 y_1 & & \ddots & \\ \vdots & & & \ddots \\ x_N y_1 & & & x_N y_N \end{pmatrix} \\ &= \mathbf{xy}^T \end{aligned}$$

- Multiplication with outer product

$$\begin{aligned}
 (\mathbf{x} \otimes \mathbf{y})\mathbf{w} &= \begin{pmatrix} x_1y_1 & x_1y_2 & \cdots & x_1y_N \\ x_2y_1 & & \ddots & \\ \vdots & & & \ddots \\ x_Ny_1 & & & x_Ny_N \end{pmatrix} \begin{pmatrix} w_1 \\ \vdots \\ w_N \end{pmatrix} \\
 &= \mathbf{x}(\mathbf{y} \cdot \mathbf{w}) \\
 \mathbf{w}^T(\mathbf{x} \otimes \mathbf{y}) &= \begin{pmatrix} w_1, & \cdots, & w_N \end{pmatrix} \begin{pmatrix} x_1y_1 & x_1y_2 & \cdots & x_1y_N \\ x_2y_1 & & \ddots & \\ \vdots & & & \ddots \\ x_Ny_1 & & & x_Ny_N \end{pmatrix} \\
 &= (\mathbf{x} \cdot \mathbf{w})\mathbf{y}^T
 \end{aligned}$$

- Outer Second Derivative

$$\begin{aligned}
 (\nabla_{\mathbf{x}} \otimes \nabla_{\mathbf{x}})f(\mathbf{x} \cdot \mathbf{a}) &= \frac{d^2f(\mathbf{x})}{dx_i dx_j} \\
 &= \begin{pmatrix} (\cdots \frac{\partial}{\partial x_1} \nabla_{\mathbf{x}} f(\mathbf{x} \cdot \mathbf{a}) \cdots) \\ (\cdots \frac{\partial}{\partial x_2} \nabla_{\mathbf{x}} f(\mathbf{x} \cdot \mathbf{a}) \cdots) \\ \vdots \\ (\cdots \frac{\partial}{\partial x_N} \nabla_{\mathbf{x}} f(\mathbf{x} \cdot \mathbf{a}) \cdots) \end{pmatrix} \\
 &= \begin{pmatrix} (\cdots \frac{\partial}{\partial x_1} f'(c)|_{\mathbf{x} \cdot \mathbf{a}} \mathbf{a} \cdots) \\ (\cdots \frac{\partial}{\partial x_2} f'(c)|_{\mathbf{x} \cdot \mathbf{a}} \mathbf{a} \cdots) \\ \vdots \\ (\cdots \frac{\partial}{\partial x_N} f'(c)|_{\mathbf{x} \cdot \mathbf{a}} \mathbf{a} \cdots) \end{pmatrix} \\
 &= f''(c)|_{\mathbf{x} \cdot \mathbf{a}} \begin{pmatrix} a_1 \\ \vdots \\ a_N \end{pmatrix} \begin{pmatrix} a_1, & \cdots, & a_N \end{pmatrix} \\
 &= f''(c)|_{\mathbf{x} \cdot \mathbf{a}} (\mathbf{a} \otimes \mathbf{a})
 \end{aligned}$$

- *Theorem:* if the  $N$  vectors  $\mathbf{x}_i$  are **linearly independent** and **span the space**, then the matrix  $\mathbf{B} \equiv E[\mathbf{x} \otimes \mathbf{x}]$  is **positive definite**.

*Proof:*  $\mathbf{B}$  is **positive definite** if

$$\mathbf{q}^T \mathbf{B} \mathbf{q} > 0 \quad \forall \mathbf{q} \neq \mathbf{0}$$

$$\begin{aligned}
 (\mathbf{B})_{ij} &= \sum_k p_k(\mathbf{x}_k)_i(\mathbf{x}_k)_j \\
 (\mathbf{q}^T \mathbf{B} \mathbf{q}) &= \sum_{ijk} p_k q_i q_j (\mathbf{x}_k)_i (\mathbf{x}_k)_j \\
 &= \sum_k p_k (\mathbf{x}_k \cdot \mathbf{q})(\mathbf{x}_k \cdot \mathbf{q}) \\
 &= \sum_k p_k (\mathbf{x}_k \cdot \mathbf{q})^2 > 0
 \end{aligned}$$

### A.5.2 Defining the Cost Function

In this section we outline the objective function formulation of BCM, as set forth in Intrator and Cooper, 1992. I include it here for completeness. We can consider a neuron as capable of deciding whether to fire or not for a given input and synaptic weight vector. A loss function is attached to each decision, and it is the neuron's task to minimize the loss over the entire set of inputs by modifying the weight vector. If the expected loss, also known as the *risk*, is a smooth function then the minimization can be accomplished by gradient descent. This will lead to at least a local minimum of the risk, which may or may not be close to optimum.

In the discussion so far, we have not defined the loss function. It is possible to obtain different learning rules from the minimization of different forms of the loss function. For example, if the loss function is related to the inverse of the projection variance, then we would obtain a rule which finds directions maximizing the projection variance, ie. principal components. Because we are interested in selectivity, we seek a loss function which emphasizes multi-modal projections or, in other words, seeks directions where the neuron responds strongly to a subset of patterns, and weakly to all others.

We define

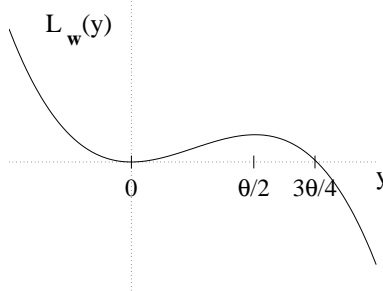
$$\begin{aligned}\theta &\equiv E[(\mathbf{x} \cdot \mathbf{w})^2] \\ \hat{\phi}(y, \theta) &\equiv y^2 - \frac{1}{2}y\theta \\ \phi(y, \theta) &\equiv y^2 - y\theta\end{aligned}$$

Consider the family of loss functions

$$\begin{aligned}L_{\mathbf{w}}(\mathbf{x}) &\equiv -\eta \int_0^{\mathbf{x} \cdot \mathbf{w}} \hat{\phi}(s, \theta) ds \\ &= -\eta \int_0^{\mathbf{x} \cdot \mathbf{w}} \left( s^2 - \frac{1}{2}sE[(\mathbf{x} \cdot \mathbf{w})^2] \right) ds \\ &= -\eta \left. \frac{s^3}{3} \right|_0^{\mathbf{x} \cdot \mathbf{w}} + \eta \left. \frac{s^2}{4} E[(\mathbf{x} \cdot \mathbf{w})^2] \right|_0^{\mathbf{x} \cdot \mathbf{w}}\end{aligned}$$

$$\boxed{L_{\mathbf{w}}(\mathbf{x}) = -\eta \left( \frac{1}{3}(\mathbf{x} \cdot \mathbf{w})^3 - \frac{1}{4}(\mathbf{x} \cdot \mathbf{w})^2 E[(\mathbf{x} \cdot \mathbf{w})^2] \right) = -\eta \left( \frac{1}{3}y^3 - \frac{1}{4}y^2\theta \right)} \quad (\text{A.39})$$

To motivate this choice, below we show the loss function for the case where  $\theta$  is constant.



In this case, one can clearly see that the loss is small either for  $y = 0$  or for  $y > \theta$ . Therefore the neuron prefers either projections which are zero or above  $\theta$ . The loss is negative for all values of  $y > \theta$ , which would be unstable if  $\theta$  were constant: the output distribution could fall completely on one side of the threshold. The value of  $\theta$  depends on  $y$  in a non-linear way, so it positions itself so that the distribution will never be concentrated on only one of its sides. This is due to the fact that  $y^2 < y$  for  $y < 1$  and  $y^2 > y$  for  $y > 1$ .

Now that we have a definition of the loss, we state that the neuron wishes to minimize the *expected loss*, or risk, defined as

$$R_{\mathbf{w}}(\mathbf{x}) = -\eta \left( \frac{1}{3} E[(\mathbf{x} \cdot \mathbf{w})^3] - \frac{1}{4} E^2[(\mathbf{x} \cdot \mathbf{w})^2] \right) \quad (3.1)$$

Minimization of this risk via gradient descent with respect to the weights gives us

$$\begin{aligned} \frac{d\mathbf{w}}{dt} = -\frac{\partial}{\partial \mathbf{w}} R_{\mathbf{w}} &= \eta \frac{\partial}{\partial \mathbf{w}} \left( \frac{1}{3} E[(\mathbf{x} \cdot \mathbf{w})^3] - \frac{1}{4} E^2[(\mathbf{x} \cdot \mathbf{w})^2] \right) \\ &= \eta E[(\mathbf{x} \cdot \mathbf{w})^2 \mathbf{x}] - \underbrace{E[(\mathbf{x} \cdot \mathbf{w})^2]}_{\theta} E[(\mathbf{x} \cdot \mathbf{w}) \mathbf{x}] \\ &= \eta E[(y^2 - y\theta) \mathbf{x}] \end{aligned}$$

$$\frac{d\mathbf{w}}{dt} = \eta E[\phi(y, \theta) \mathbf{x}] \quad (3.2)$$

which is the almost same as the BCM equation (Equation 1.4). The difference is that Equation 3.2 is not longer stochastic: the random variable  $\mathbf{x}$  is averaged over. This equation has the same fixed points as Equation 1.4, though possibly not the same dynamics. Most of the time we are interested in the fixed points, and the deterministic equation can make analysis a lot easier in those cases.

### A.5.3 Single Nonlinear Neuron

The cost function we have found can be made more robust if we can somehow make it less sensitive to outliers and if we allow the output distribution to be shifted so that the part of the distribution  $y < \theta$  is centered at zero. The over-sensitivity to outliers can be handled by defining the output as  $y = \sigma(\mathbf{w} \cdot \mathbf{x})$  where  $\sigma$  is a squashing sigmoid function (see also Section 1.1.2). The ability to shift the distribution so that one of its modes is at zero is achieved by simply introducing an offset to the output:  $y = \sigma(\mathbf{w} \cdot \mathbf{x} + \beta)$ . From a biological viewpoint,  $\beta$  can be considered the spontaneous activity. The value of this  $\beta$  can be obtained by noting that it merely adds a dimension to the input and weight vectors,  $\mathbf{x} = (x_1, \dots, x_n, 1)$  and  $\mathbf{w} = (w_1, \dots, w_n, \beta)$ , so that  $\beta$  can be found using the same modification equations. From now on we will assume that this offset is added to the output when necessary, without explicitly writing it. The modification equation for the non-linear neuron is obtained in the same straightforward way as for the linear neuron.

The risk becomes

$$R_{\mathbf{w}}(\mathbf{x}) = -\eta \left( \frac{1}{3} E[\sigma^3(\mathbf{x} \cdot \mathbf{w})] - \frac{1}{4} E^2[\sigma^2(\mathbf{x} \cdot \mathbf{w})] \right) \quad (A.40)$$

and, as before, the modification function is found via gradient descent.

$$\begin{aligned} \frac{d\mathbf{w}}{dt} &= -\frac{\partial}{\partial \mathbf{w}} R_{\mathbf{w}} \\ &= \eta \left( E[\sigma^2(\mathbf{x} \cdot \mathbf{w}) \sigma'(\mathbf{x} \cdot \mathbf{w}) \mathbf{x}] - \frac{1}{4} 2E[\sigma^2(\mathbf{x} \cdot \mathbf{w})] E[2\sigma(\mathbf{x} \cdot \mathbf{w}) \sigma'(\mathbf{x} \cdot \mathbf{w}) \mathbf{x}] \right) \end{aligned}$$

$$\frac{d\mathbf{w}}{dt} = \eta E[\phi(y, \theta) \sigma'(\mathbf{x} \cdot \mathbf{w}) \mathbf{x}] \quad (A.41)$$

### A.5.4 Fixed points for $N$ linearly independent inputs

In this section we use the objective function formulation of BCM to determine the fixed points and their stability in an environment consisting of  $N$  linearly independent input (not necessarily orthogonal) vectors in a space of dimensionality  $N$ . This section shows that the only stable fixed points achieved by the BCM neuron in this environment are those where the neuron responds to only *one* of the inputs, and has zero response to all others. Furthermore, the response to the one input the neuron is selective to has a magnitude of  $1/p$  where  $p$  is the probability of that vector appearing in the input set. Thus the neuron responds most strongly to those patterns which occur less often.

We define  $N$  linearly independent inputs  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$  each occurring in the input set with the probability  $P(\mathbf{x}_i) \equiv p_i$  with  $\sum_i p_i = 1$ . The BCM weight modification is defined

$$\frac{d\mathbf{w}}{dt} = \eta E[(\mathbf{x} \cdot \mathbf{w})(\mathbf{x} \cdot \mathbf{w} - E[(\mathbf{x} \cdot \mathbf{w})^2])] \quad (\text{A.42})$$

We define a response vector,  $\mathbf{v}$ , whose elements,  $v_i$ , are equal to the response of the neuron to each input vector  $\mathbf{x}_i$ .

$$\begin{pmatrix} (\cdots & \mathbf{x}_1 & \cdots) \\ (\cdots & \mathbf{x}_2 & \cdots) \\ & \vdots & \\ (\cdots & \mathbf{x}_N & \cdots) \end{pmatrix} \mathbf{w} = \mathbf{v}$$

We note, then, that weight stationary points occur when, for all  $i$ ,  $\mathbf{x}_i \cdot \mathbf{w} = 0$  or  $\mathbf{x}_i \cdot \mathbf{w} = \theta \neq 0$ . This is the same as saying the values of the vector  $\mathbf{v}$  are either zero or  $\theta$ . There are  $2^N$  ways of achieving this, so there are  $2^N$  stationary points. These stationary points are of the form:

$$\mathbf{v}_{(0)} = (0, \dots, 0) \quad (\text{A.43})$$

$$\mathbf{v}_{(1)} = \left( \frac{1}{p_1}, 0, \dots, 0 \right) \quad (\text{A.44})$$

$$\mathbf{v}_{(2)} = \left( 0, \frac{1}{p_2}, 0, \dots, 0 \right) \quad (\text{A.45})$$

$$\mathbf{v}_{(3)} = \left( \frac{1}{p_1 + p_2}, \frac{1}{p_1 + p_2}, 0, \dots, 0 \right) \quad (\text{A.46})$$

$\vdots$

$$\mathbf{v}_{(2^N-1)} = (1, \dots, 1) \quad (\text{A.47})$$

All of the stationary points can be derived in a straightforward fashion. The following is an outline of the procedure for a few of them.

- **Stationary Point 0:**

$$\mathbf{x}_i \cdot \mathbf{w} = 0 \quad \forall i \Rightarrow \mathbf{v}_{(0)} = (0, \dots, 0)$$

- **Stationary Point 1:**

$$\begin{aligned}
 \mathbf{x}_1 \cdot \mathbf{w} &= \theta \neq 0 \\
 \mathbf{x}_i \cdot \mathbf{w} &= 0 \quad \forall (i > 1) \\
 \theta = E[(\mathbf{x} \cdot \mathbf{w})^2] &= \sum_{i=1}^N p_i (\mathbf{x}_i \cdot \mathbf{w})^2 \\
 &= p_1 (\mathbf{x}_1 \cdot \mathbf{w})^2 \\
 \mathbf{x}_1 \cdot \mathbf{w} = \theta &= p_1 (\mathbf{x}_1 \cdot \mathbf{w})^2 \\
 \Rightarrow \mathbf{v}_{(1)} &= \left( \frac{1}{p_1}, 0, \dots, 0 \right)
 \end{aligned}$$

- **Stationary Point 2:**

$$\begin{aligned}
 \mathbf{x}_2 \cdot \mathbf{w} &= \theta \neq 0 \\
 \mathbf{x}_i \cdot \mathbf{w} &= 0 \quad \forall (i \neq 2) \\
 \theta = E[(\mathbf{x} \cdot \mathbf{w})^2] &= \sum_{i=1}^N p_i (\mathbf{x}_i \cdot \mathbf{w})^2 \\
 &= p_2 (\mathbf{x}_2 \cdot \mathbf{w})^2 \\
 \mathbf{x}_2 \cdot \mathbf{w} = \theta &= p_2 (\mathbf{x}_2 \cdot \mathbf{w})^2 \\
 \Rightarrow \mathbf{v}_{(2)} &= \left( 0, \frac{1}{p_2}, 0, \dots, 0 \right)
 \end{aligned}$$

- **Stationary Point 3:**

$$\begin{aligned}
 \mathbf{x}_1 \cdot \mathbf{w} = \mathbf{x}_2 \cdot \mathbf{w} &= \theta \neq 0 \\
 \mathbf{x}_i \cdot \mathbf{w} &= 0 \quad \forall (i > 2) \\
 \theta = E[(\mathbf{x} \cdot \mathbf{w})^2] &= \sum_{i=1}^N p_i (\mathbf{x}_i \cdot \mathbf{w})^2 \\
 &= p_1 (\mathbf{x}_1 \cdot \mathbf{w})^2 + p_2 (\mathbf{x}_2 \cdot \mathbf{w})^2 \\
 \mathbf{x}_1 \cdot \mathbf{w} = \theta &= p_1 (\mathbf{x}_1 \cdot \mathbf{w})^2 + p_2 (\mathbf{x}_2 \cdot \mathbf{w})^2 \\
 \Rightarrow \mathbf{v}_{(3)} &= \left( \frac{1}{p_1 + p_2}, \frac{1}{p_1 + p_2}, 0, \dots, 0 \right)
 \end{aligned}$$

- **Stationary Point  $2^N - 1$ :**

$$\mathbf{v}_{(2^N-1)} = (1, \dots, 1)$$

### A.5.5 Calculating the Stability

- Find the gradient of the Risk with respect to the weight vector  $\mathbf{w}$

$$\begin{aligned}
 R_{\mathbf{w}}(\mathbf{x}) &= -\frac{1}{3}E[(\mathbf{x} \cdot \mathbf{w})^3] + \frac{1}{4}E^2[(\mathbf{x} \cdot \mathbf{w})^2] \\
 \nabla_{\mathbf{w}} R_{\mathbf{w}}(\mathbf{x}) &= -E[(\mathbf{x} \cdot \mathbf{w})^2 \mathbf{x}] + E[(\mathbf{x} \cdot \mathbf{w})^2] E[(\mathbf{x} \cdot \mathbf{w}) \mathbf{x}]
 \end{aligned}$$

- Form the matrix of second derivatives of the Risk function with respect to the weight vector  $\mathbf{w}$

$$\begin{aligned}
 (\mathbf{H})_{ij} &\equiv \frac{d^2 R_{\mathbf{w}}(\mathbf{x})}{dm_i dm_j} \\
 \mathbf{H} &= (\nabla_{\mathbf{w}} \otimes \nabla_{\mathbf{w}}) R_{\mathbf{w}}(\mathbf{x}) \\
 &= -2E[(\mathbf{x} \cdot \mathbf{w})(\mathbf{x} \otimes \mathbf{x})] + E[(\mathbf{x} \cdot \mathbf{w})^2] E[(\mathbf{x} \otimes \mathbf{x})] + 2E[(\mathbf{x} \cdot \mathbf{w})\mathbf{x}] \otimes E[(\mathbf{x} \cdot \mathbf{w})\mathbf{x}]
 \end{aligned}$$

- If  $\mathbf{H}$  is **positive definite** for  $\mathbf{w}$  equal to a stationary weight vector, then it will have only **positive eigenvalues** which corresponds to a minimum. Otherwise the stationary point is **unstable**.
- Notational reminder:

$$\text{pth fixed point: } \mathbf{v}_{(p)} = \left( \underbrace{v_1}_{(\mathbf{x}_1 \cdot \mathbf{w})}, \underbrace{v_2}_{(\mathbf{x}_2 \cdot \mathbf{w})}, \underbrace{v_3}_{(\mathbf{x}_3 \cdot \mathbf{w})}, \dots, \underbrace{v_N}_{(\mathbf{x}_N \cdot \mathbf{w})} \right)$$

- **Stationary Point 0:**  $\mathbf{v}_{(0)} = (0, \dots, 0)$

$$\mathbf{H}|_{\mathbf{v}_{(0)}} = 0 \Rightarrow \text{unstable}$$

- **Stationary Point 1:**  $\mathbf{v}_{(1)} = \left( \frac{1}{p_1}, 0, \dots, 0 \right)$

• Look at each term in  $\mathbf{H}$ .

$$\begin{aligned}
 E[(\mathbf{x} \cdot \mathbf{w})(\mathbf{x} \otimes \mathbf{x})] &= \sum_{i=1}^N p_i (\mathbf{x}_i \cdot \mathbf{w})(\mathbf{x}_i \otimes \mathbf{x}_i) \\
 &= \mathbf{x}_1 \otimes \mathbf{x}_1 \\
 E[(\mathbf{x} \cdot \mathbf{w})^2] &= \frac{1}{p_1} \\
 E[(\mathbf{x} \otimes \mathbf{x})] &\equiv \mathbf{B} \text{ (which is positive definite)} \\
 E[(\mathbf{x} \cdot \mathbf{w})\mathbf{x}] &= \sum_{i=1}^N p_i (\mathbf{x}_i \cdot \mathbf{w})\mathbf{x}_i \\
 &= \mathbf{x}_1 \\
 \mathbf{H}|_{\mathbf{v}_{(1)}} &= -2(\mathbf{x}_1 \otimes \mathbf{x}_1) + \frac{1}{p_1} \mathbf{B} + 2(\mathbf{x}_1 \otimes \mathbf{x}_1) \\
 &= \frac{1}{p_1} \mathbf{B} \Rightarrow \text{stable}
 \end{aligned}$$

- **Stationary Point 3:**  $\mathbf{v}_{(3)} = \left( \frac{1}{p_1+p_2}, \frac{1}{p_1+p_2}, 0, \dots, 0 \right)$

- Look at each term in  $\mathbf{H}$ .

$$\begin{aligned}
 E[(\mathbf{x} \cdot \mathbf{w})(\mathbf{x} \otimes \mathbf{x})] &= \sum_{i=1}^N p_i (\mathbf{x}_i \cdot \mathbf{w})(\mathbf{x}_i \otimes \mathbf{x}_i) \\
 &= p_1 (\mathbf{x}_1 \cdot \mathbf{w})(\mathbf{x}_1 \otimes \mathbf{x}_1) + p_2 (\mathbf{x}_2 \cdot \mathbf{w})(\mathbf{x}_2 \otimes \mathbf{x}_2) \\
 &= \frac{p_1}{p_1 + p_2} \mathbf{x}_1 \otimes \mathbf{x}_1 + \frac{p_2}{p_1 + p_2} \mathbf{x}_2 \otimes \mathbf{x}_2 \\
 E[(\mathbf{x} \cdot \mathbf{w})^2] &= \frac{1}{p_1 + p_2} \\
 E[(\mathbf{x} \otimes \mathbf{x})] &\equiv \mathbf{B} \\
 E[(\mathbf{x} \cdot \mathbf{w})\mathbf{x}] &= \sum_{i=1}^N p_i (\mathbf{x}_i \cdot \mathbf{w})\mathbf{x}_i \\
 &= \frac{p_1}{p_1 + p_2} \mathbf{x}_1 + \frac{p_2}{p_1 + p_2} \mathbf{x}_2 \\
 \mathbf{H}|_{\mathbf{v}_{(3)}} &= \left( \frac{-2p_1}{p_1 + p_2} + \frac{2p_1^2}{(p_1 + p_2)^2} \right) (\mathbf{x}_1 \otimes \mathbf{x}_1) + \\
 &\quad \left( \frac{-2p_2}{p_1 + p_2} + \frac{2p_2^2}{(p_1 + p_2)^2} \right) (\mathbf{x}_1 \otimes \mathbf{x}_1) + \frac{1}{p_1 + p_2} \mathbf{B} + \\
 &\quad \frac{2p_1 p_2}{(p_1 + p_2)^2} (\mathbf{x}_1 \otimes \mathbf{x}_2 + \mathbf{x}_2 \otimes \mathbf{x}_1)
 \end{aligned}$$

- Define a direction orthogonal to all but either  $\mathbf{x}_1$  or  $\mathbf{x}_2$ , whichever is least likely. This direction we will show to be unstable. We will choose  $p_1 > p_2$ , with no loss of generality and define the direction vector  $\mathbf{y}$  such that

$$\mathbf{y} \cdot \mathbf{x}_i = 0 \quad \forall i \neq 2$$

Note that this does *not* imply that  $\mathbf{y}$  is parallel to  $\mathbf{x}_2$ , just orthogonal to all others.

- Look at each term of  $\mathbf{y}^T \mathbf{H} \mathbf{y}$ , which is equal to  $\mathbf{H}$  in the  $\mathbf{y}$  direction

$$\begin{aligned}
 \mathbf{y}^T (\mathbf{x}_i \otimes \mathbf{x}_i) \mathbf{y} &= 0 \quad \forall i \neq 2 \\
 \mathbf{y}^T (\mathbf{x}_i \otimes \mathbf{x}_2) \mathbf{y} = \mathbf{y}^T (\mathbf{x}_2 \otimes \mathbf{x}_i) \mathbf{y} &= 0 \quad \forall i \neq 2 \\
 \mathbf{y}^T (\mathbf{x}_2 \otimes \mathbf{x}_2) \mathbf{y} &= (\mathbf{x}_2 \cdot \mathbf{y})^2 \\
 \mathbf{y}^T \mathbf{B} \mathbf{y} &= p_2 (\mathbf{x}_2 \cdot \mathbf{y})^2 \\
 \mathbf{y}^T \mathbf{H} \mathbf{y} &= \left( \frac{-2p_2}{p_1 + p_2} + \frac{2p_2^2}{(p_1 + p_2)^2} \right) (\mathbf{x}_2 \cdot \mathbf{y})^2 + \\
 &\quad \frac{p_2}{p_1 + p_2} (\mathbf{x}_2 \cdot \mathbf{y})^2 \\
 &= \frac{(\mathbf{x}_2 \cdot \mathbf{y})^2}{(p_1 + p_2)^2} (p_2(p_2 - p_1)) < 0 \Rightarrow \text{unstable}
 \end{aligned}$$

## A.6 Stability of PCA fixed points

This section is motivated by a derivation in (Hertz et al., 1991). It is included here for completeness.

If we define the correlation matrix as

$$\mathbf{C} = E[\mathbf{x}\mathbf{x}^T] \tag{A.48}$$

with normalized eigenvectors  $\mathbf{v}_{(\alpha)}$  and eigenvalues  $\lambda_{(\alpha)}$ , then the fixed point solution for the PCA equation is given simply by

$$\mathbf{w} = \mathbf{v}_{(\alpha)} \quad (\text{A.49})$$

For stability, we expand around this solution, and prove that the only stable fixed point is the *eigenvector with the maximum eigenvalue*.

$$\mathbf{w} = \mathbf{v}_{(\alpha)} + \boldsymbol{\epsilon} \quad (\text{A.50})$$

$$E[\delta\boldsymbol{\epsilon}] = E[\delta\mathbf{w}] = \mathbf{C}\mathbf{w} - (\mathbf{w}^T\mathbf{C}\mathbf{w})\mathbf{w} \quad (\text{A.51})$$

$$\begin{aligned} &= \mathbf{C}(\mathbf{v}_{(\alpha)} + \boldsymbol{\epsilon}) - \left( (\mathbf{v}_{(\alpha)}^T + \boldsymbol{\epsilon}^T)\mathbf{C}(\mathbf{v}_{(\alpha)} + \boldsymbol{\epsilon}) \right) (\mathbf{v}_{(\alpha)} + \boldsymbol{\epsilon}) \\ &= \lambda_{(\alpha)}\mathbf{v}_{(\alpha)} + \mathbf{C}\boldsymbol{\epsilon} - (\mathbf{v}_{(\alpha)}^T\mathbf{C}\mathbf{v}_{(\alpha)})\mathbf{v}_{(\alpha)} - (\boldsymbol{\epsilon}^T\mathbf{C}\mathbf{v}_{(\alpha)})\mathbf{v}_{(\alpha)} - \\ &\quad (\mathbf{v}_{(\alpha)}^T\mathbf{C}\boldsymbol{\epsilon})\mathbf{v}_{(\alpha)} - (\mathbf{v}_{(\alpha)}^T\mathbf{C}\mathbf{v}_{(\alpha)})\boldsymbol{\epsilon} + O(\boldsymbol{\epsilon}^2) \\ &= \mathbf{C}\boldsymbol{\epsilon} - 2(\lambda_{(\alpha)}\boldsymbol{\epsilon}^T\mathbf{v}_{(\alpha)})\mathbf{v}_{(\alpha)} - \lambda_{(\alpha)}\boldsymbol{\epsilon} + O(\boldsymbol{\epsilon}^2) \end{aligned} \quad (\text{A.52})$$

We now look at this change projected along a possibly different eigenvector  $\mathbf{v}_{(\beta)}$ .

$$E[\mathbf{v}_{(\beta)}^T \cdot \delta\boldsymbol{\epsilon}] = (\lambda_{(\beta)} - \lambda_{(\alpha)} - 2\lambda_{(\alpha)}\delta_{\alpha\beta})\mathbf{v}_{(\beta)}^T\boldsymbol{\epsilon} \quad (\text{A.53})$$

which is unstable (greater than zero) if  $\lambda_{(\beta)} > \lambda_{(\alpha)}$ , which implies that the solution  $\mathbf{w} = \mathbf{v}_{(\alpha)}$  is only stable if  $\mathbf{v}_{(\alpha)}$  is the eigenvector with the maximum eigenvalue.

It follows that the variance of the output of the neuron

$$E[y^2] = \mathbf{w}^T\mathbf{x}\mathbf{x}^T\mathbf{w} = \mathbf{w}^T\mathbf{C}\mathbf{w} = \lambda_{(\alpha)} \quad (\text{A.54})$$

is maximized along this direction.

## A.7 Deriving the Wyatt Equation

This section is just a rewritten form of the first part, and Appendix 1, of Wyatt's original paper (Wyatt and Elfadel, 1995). It outlines the full dynamical solution for the PCA learning rule. It is here for completeness.

Oja's algorithm can be stated quite simply, with two equations.

$$\begin{aligned} y_k &= \mathbf{w}_k^T \mathbf{x}_k \quad (\text{activation rule}) \\ \mathbf{w}_{k+1} &= \mathbf{w}_k + \eta y_k (\mathbf{x}_k - y_k \mathbf{w}_k) \quad (\text{learning rule}) \end{aligned}$$

where  $\mathbf{x}_k$ ,  $y_k$  and  $\mathbf{w}_k$ , are the input vector, output, and weight vector, respectively, at time  $k$ .

Taking the limit as  $\eta \rightarrow 0$ , averaging over all input space, assuming that  $\langle \mathbf{x} \rangle = 0$ , and defining the correlation matrix  $\mathbf{C} = \langle \mathbf{x}\mathbf{x}^T \rangle$ , one arrives at the differential equation for the weight vector:

$$\dot{\mathbf{w}}(t) = \mathbf{C}\mathbf{w}(t) - \mathbf{w}(t)\mathbf{w}^T(t)\mathbf{C}\mathbf{w}(t) \quad (\text{A.55})$$

We now try to simplify the form of Equation A.55.

$$\begin{aligned}
 \dot{\mathbf{w}}(t) &= \mathbf{C}\mathbf{w}(t) - \mathbf{w}(t)\mathbf{w}^T(t)\mathbf{C}\mathbf{w}(t) \\
 &= (\mathbf{C} - \mathbf{w}^T\mathbf{C}\mathbf{w}\mathbf{1})\mathbf{w} \\
 &\equiv \mathbf{A}(t)\mathbf{w}
 \end{aligned}$$

where

$$\begin{aligned}
 \mathbf{A}(t) &\equiv (\mathbf{C} - \mathbf{w}^T(t)\mathbf{C}\mathbf{w}(t)\mathbf{1}) \equiv (\mathbf{C} - a(t)\mathbf{1}) \\
 a(t) &\equiv \mathbf{w}^T(t)\mathbf{C}\mathbf{w}(t) \\
 \mathbf{1} &\equiv \begin{pmatrix} 1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & 1 \end{pmatrix}
 \end{aligned}$$

Because the quantity  $a(t)$  is a scalar, and the correlation matrix  $\mathbf{C}$  is time independent,  $\mathbf{A}(t)$  has the property  $[\mathbf{A}(t_1), \mathbf{A}(t_2)] = 0$ . This lets us immediately write down the solution for  $\mathbf{w}(t)$  given the initial condition  $\mathbf{w}_0$ :

$$\begin{aligned}
 \mathbf{w}(t) &= \exp\left[\int_0^t \mathbf{A}(t')dt'\right] \mathbf{w}_0 \\
 &= e^{\mathbf{C}t} e^{-\int_0^t a(t')dt'} \mathbf{w}_0
 \end{aligned} \tag{A.56}$$

Plugging back into the definition of  $a(t)$  we get

$$\begin{aligned}
 a(t) &\equiv \mathbf{w}^T(t)\mathbf{C}\mathbf{w}(t) \\
 &= e^{-2\int_0^t a(t')dt'} \mathbf{w}_0^T \mathbf{C} e^{2\mathbf{C}t} \mathbf{w}_0
 \end{aligned}$$

Therefore

$$a(t)e^{2\int_0^t a(t')dt'} = \mathbf{w}_0^T \mathbf{C} e^{2\mathbf{C}t} \mathbf{w}_0$$

which can be written

$$\frac{d}{dt} \frac{1}{2} e^{2\int_0^t a(t')dt'} = \mathbf{w}_0^T \frac{1}{2} \frac{d}{dt} [e^{2\mathbf{C}t}] \mathbf{w}_0$$

Integrating, and substituting the initial condition, we get

$$e^{2\int_0^t a(t')dt'} = \mathbf{w}_0^T e^{2\mathbf{C}t} \mathbf{w}_0 + 1 - \mathbf{w}_0^T \mathbf{w}_0$$

Substituting this into Equation A.56 we get the final Wyatt solution:

$$\boxed{\mathbf{w}(t) = \frac{e^{\mathbf{C}t} \mathbf{w}_0}{\left(\|e^{\mathbf{C}t} \mathbf{w}_0\|^2 + 1 - \|\mathbf{w}_0\|^2\right)^{\frac{1}{2}}} } \tag{A.57}$$

## A.8 Classical Rearing Conditions: PCA Analysis

In this section we use the Wyatt solution to determine the time dynamics of a neuron following Oja's learning rule in an environment modeling the classical visual cortical plasticity experiments.

Some notation:

single eye input vectors:	$\mathbf{x}^l \equiv$ left eye inputs, $\mathbf{x}^r \equiv$ right eye inputs
single eye correlation matrix:	$\mathbf{C} \equiv \langle \mathbf{x}\mathbf{x}^T \rangle$
full input vector:	$\begin{pmatrix} \mathbf{x}^l \\ \mathbf{x}^r \end{pmatrix}$
full correlation matrix:	$\mathcal{C} \equiv \left\langle \begin{pmatrix} \mathbf{x}^l \\ \mathbf{x}^r \end{pmatrix} \begin{pmatrix} (\mathbf{x}^l)^T & (\mathbf{x}^r)^T \end{pmatrix} \right\rangle$ $= \begin{pmatrix} \mathbf{C}^{ll} & \mathbf{C}^{lr} \\ \mathbf{C}^{rl} & \mathbf{C}^{rr} \end{pmatrix}$
natural scene (single eye) correlation matrix:	$\mathbf{C}$
eigenvectors/values of correlation matrix:	$\mathbf{C}\mathbf{v}_j = \lambda_j\mathbf{v}_j \quad (j = 1, \dots, n)$ $\mathbf{v}_i^T \mathbf{v}_j = \delta_{ij}$ $\lambda_1 > \lambda_2 > \dots > \lambda_n$
(assumption) noise (single eye) correlation matrix:	$\sigma^2 = \begin{pmatrix} \sigma^2 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \sigma^2 \end{pmatrix}$

The rearing experiments in this new notation are as follows:

Experiment	initial weight vector	full correlation matrix
Normal Rearing (NR)	$\mathbf{w}_0^{\text{NR}}$	$\begin{pmatrix} \mathbf{C} & \mathbf{C} \\ \mathbf{C} & \mathbf{C} \end{pmatrix}$
Monocular Deprivation (MD)	$\mathbf{w}_0^{\text{MD}} = \mathbf{w}^{\text{NR}}(t \rightarrow \infty)$	$\begin{pmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \sigma^2 \end{pmatrix}$
Binocular Deprivation (BD)	$\mathbf{w}_0^{\text{BD}} = \mathbf{w}^{\text{NR}}(t \rightarrow \infty)$	$\begin{pmatrix} \sigma^2 & \mathbf{0} \\ \mathbf{0} & \sigma^2 \end{pmatrix}$
Reverse Suture (RS)	$\mathbf{w}_0^{\text{RS}} = \mathbf{w}^{\text{MD}}(t \rightarrow \infty)$	$\begin{pmatrix} \sigma^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{C} \end{pmatrix}$

The general procedure will be the following:

- expand the initial weight vector  $\mathbf{w}_0$  in terms of the eigenvectors of the correlation matrix ( $\mathbf{v}_j$ ).
- plug into the Wyatt solution (Equation A.57) to obtain  $\mathbf{w}(t)$ . We will have to use the full correlation matrix  $\mathcal{C}$  instead of the single eye correlation matrix  $\mathbf{C}$ .
- add approximations to simplify the expression

We present each rearing condition currently.

### A.8.1 Normal Rearing (NR)

- expand the initial weight vector  $\mathbf{w}_0$  in terms of the eigenvectors of the correlation matrix ( $\mathbf{v}_j$ ).

$$\mathbf{w}_0 = \begin{pmatrix} \mathbf{w}_0^l \\ \mathbf{w}_0^r \end{pmatrix} = \sum_j \begin{pmatrix} a_j^l \mathbf{v}_j \\ a_j^r \mathbf{v}_j \end{pmatrix}$$

- plug into the Wyatt solution (Equation A.57) to obtain  $\mathbf{w}(t)$

We need to do this part in steps, calculating  $\mathcal{C}\mathbf{w}_0$  first, then calculating each term in Equation A.57.

$$\begin{aligned} \mathcal{C}\mathbf{w}_0 &= \begin{pmatrix} \mathbf{C} & \mathbf{C} \\ \mathbf{C} & \mathbf{C} \end{pmatrix} \sum_j \begin{pmatrix} a_j^l \mathbf{v}_j \\ a_j^r \mathbf{v}_j \end{pmatrix} = \sum_j \begin{pmatrix} \lambda_j (a_j^l + a_j^r) \mathbf{v}_j \\ \lambda_j (a_j^l + a_j^r) \mathbf{v}_j \end{pmatrix} \\ &= \sum_j \lambda_j (a_j^l + a_j^r) \begin{pmatrix} \mathbf{v}_j \\ \mathbf{v}_j \end{pmatrix} \\ \mathcal{C}^2\mathbf{w}_0 &= \begin{pmatrix} \mathbf{C} & \mathbf{C} \\ \mathbf{C} & \mathbf{C} \end{pmatrix} \sum_j \lambda_j (a_j^l + a_j^r) \begin{pmatrix} \mathbf{v}_j \\ \mathbf{v}_j \end{pmatrix} \\ &= \sum_j 2\lambda_j^2 (a_j^l + a_j^r) \begin{pmatrix} \mathbf{v}_j \\ \mathbf{v}_j \end{pmatrix} \\ &\vdots \\ \mathcal{C}^i\mathbf{w}_0 &= \sum_j 2^{i-1} \lambda_j^i (a_j^l + a_j^r) \begin{pmatrix} \mathbf{v}_j \\ \mathbf{v}_j \end{pmatrix} \\ &\vdots \end{aligned}$$

$$\begin{aligned} e^{\mathcal{C}t}\mathbf{w}_0 &= \left( 1 + (\mathcal{C}t) + \frac{(\mathcal{C}t)^2}{2!} + \dots \right) \mathbf{w}_0 \\ &= \sum_j \left\{ (a_j^l + a_j^r) \begin{pmatrix} \mathbf{v}_j \\ \mathbf{v}_j \end{pmatrix} \left( 1 + \lambda_j t + \frac{2\lambda_j^2 t^2}{2!} + \dots + \frac{2^{i-1} \lambda_j^i t^i}{i!} + \dots \right) - \begin{pmatrix} a_j^r \mathbf{v}_j \\ a_j^l \mathbf{v}_j \end{pmatrix} \right\} \\ &= \sum_j \frac{1}{2} \begin{pmatrix} \mathbf{v}_j [(a_j^l + a_j^r) e^{2\lambda_j t} + (a_j^l - a_j^r)] \\ \mathbf{v}_j [(a_j^l + a_j^r) e^{2\lambda_j t} + (a_j^r - a_j^l)] \end{pmatrix} \end{aligned}$$

$$\|\mathbf{w}_0\|^2 = \mathbf{w}_0^T \mathbf{w}_0 = \sum_j [(a_j^l)^2 + (a_j^r)^2]$$

$$a_{j+} \equiv (a_j^l + a_j^r)$$

$$a_{j-} \equiv (a_j^l - a_j^r)$$

$$\begin{aligned} \|e^{\mathcal{C}t}\mathbf{w}_0\|^2 &= \sum_j \frac{1}{4} [a_{j+}^2 e^{4\lambda_j t} + a_{j-}^2 + 2a_{j-} a_{j+} e^{2\lambda_j t} + a_{j+}^2 e^{4\lambda_j t} + a_{j-}^2 - 2a_{j-} a_{j+} e^{2\lambda_j t}] \\ &= \frac{1}{2} \sum_j [(a_j^l + a_j^r)^2 e^{4\lambda_j t} + (a_j^l - a_j^r)^2] \end{aligned}$$

which brings us to our solution for normal rearing

$$\mathbf{w}^{\text{NR}}(t) = \frac{\sum_j \frac{1}{2} \begin{pmatrix} \mathbf{v}_j [(a_j^l + a_j^r)e^{2\lambda_j t} + (a_j^l - a_j^r)] \\ \mathbf{v}_j [(a_j^l + a_j^r)e^{2\lambda_j t} + (a_j^r - a_j^l)] \end{pmatrix}}{\left(\frac{1}{2} \sum_j [(a_j^l + a_j^r)^2 e^{4\lambda_j t} + (a_j^l - a_j^r)^2] + 1 - \sum_j [(a_j^l)^2 + (a_j^r)^2]\right)^{1/2}} \quad (\text{A.58})$$

- add approximations to make the equations simpler

The  $t \rightarrow \infty$  limiting case, needed for the calculations in the next few sections, is straightforward to calculate. If the largest eigenvalue is non-degenerate, then  $\mathbf{w}$  will become

$$\mathbf{w}^{\text{NR}}(t \rightarrow \infty) = \sqrt{\frac{1}{2}} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_1 \end{pmatrix}$$

### A.8.2 Monocular Deprivation (MD)

- expand the initial weight vector  $\mathbf{w}_0$  in terms of the eigenvectors of the correlation matrix ( $\mathbf{v}_j$ ).

Since we are starting from the  $\mathbf{w}^{\text{NR}}(t \rightarrow \infty)$  state, the initial weight vector for monocular deprivation is already in terms of the eigenvectors of the correlation matrix.

$$\mathbf{w}_0 = \sqrt{\frac{1}{2}} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_1 \end{pmatrix}$$

- plug into the Wyatt solution (Equation A.57) to obtain  $\mathbf{w}(t)$ .

$$\begin{aligned} \mathcal{C}\mathbf{w}_0 &= \begin{pmatrix} \mathbf{C} & \mathbf{0} \\ \mathbf{0} & \sigma^2 \end{pmatrix} \sqrt{\frac{1}{2}} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_1 \end{pmatrix} = \sqrt{\frac{1}{2}} \begin{pmatrix} \lambda_1 \mathbf{v}_1 \\ \sigma^2 \mathbf{v}_1 \end{pmatrix} \\ e^{\mathcal{C}t}\mathbf{w}_0 &= \sqrt{\frac{1}{2}} \begin{pmatrix} e^{\lambda_1 t} \mathbf{v}_1 \\ e^{\sigma^2 t} \mathbf{v}_1 \end{pmatrix} \\ \|\mathbf{w}_0\|^2 &= 1 \\ \|e^{\mathcal{C}t}\mathbf{w}_0\|^2 &= \frac{1}{2}(e^{2\lambda_1 t} + e^{2\sigma^2 t}) \end{aligned}$$

which yields

$$\mathbf{w}^{\text{MD}}(t) = \frac{\begin{pmatrix} e^{\lambda_1 t} \mathbf{v}_1 \\ e^{\sigma^2 t} \mathbf{v}_1 \end{pmatrix}}{(e^{2\lambda_1 t} + e^{2\sigma^2 t})^{1/2}} \quad (\text{A.59})$$

- add approximations to make the equations simpler

The assumption we are going to make is that  $\lambda_1 > \sigma^2$ . This is reasonable, because the noise to the closed eye should be smaller than the eigenvalue from the natural scene correlation. With this assumption, the  $t \rightarrow \infty$  limiting case becomes

$$\mathbf{w}^{\text{MD}}(t \rightarrow \infty) = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{0} \end{pmatrix}$$

### A.8.3 Binocular Deprivation (BD)

- expand the initial weight vector  $\mathbf{w}_0$  in terms of the eigenvectors of the correlation matrix ( $\mathbf{v}_j$ ).

As in MD, the initial weight vector is

$$\mathbf{w}_0 = \sqrt{\frac{1}{2}} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_1 \end{pmatrix}$$

- plug into the Wyatt solution (Equation A.57) to obtain  $\mathbf{w}(t)$ .

$$\begin{aligned} \mathcal{C}\mathbf{w}_0 &= \begin{pmatrix} \sigma^2 & \mathbf{0} \\ \mathbf{0} & \sigma^2 \end{pmatrix} \sqrt{\frac{1}{2}} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_1 \end{pmatrix} = \sqrt{\frac{1}{2}} \begin{pmatrix} \sigma^2 \mathbf{v}_1 \\ \sigma^2 \mathbf{v}_1 \end{pmatrix} \\ e^{Ct}\mathbf{w}_0 &= \sqrt{\frac{1}{2}} \begin{pmatrix} e^{\sigma^2 t} \mathbf{v}_1 \\ e^{\sigma^2 t} \mathbf{v}_1 \end{pmatrix} \\ \|\mathbf{w}_0\|^2 &= 1 \\ \|e^{Ct}\mathbf{w}_0\|^2 &= e^{2\sigma^2 t} \end{aligned}$$

which yields

$$\mathbf{w}^{\text{BD}}(t) = \sqrt{\frac{1}{2}} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_1 \end{pmatrix} \tag{A.60}$$

Equation 2.4 implies that a neuron following Oja's rule, experiencing binocular deprivation following normal rearing, performs a random walk about the normal reared state.

### A.8.4 Reverse Suture (RS)

- expand the initial weight vector  $\mathbf{w}_0$  in terms of the eigenvectors of the correlation matrix ( $\mathbf{v}_j$ ).

We run into an immediate problem if we try to use  $\mathbf{w}^{\text{MD}}(t \rightarrow \infty)$  as  $\mathbf{w}_0$ : the newly opened eye *never* recovers. To alleviate this, we assume that the monocular deprivation experiment did not achieve  $t = \infty$ , but just some large number  $T$ . In that case the initial weight vector for RS is

$$\mathbf{w}_0 = \begin{pmatrix} \mathbf{v}_1 \\ \epsilon \mathbf{v}_1 \end{pmatrix}$$

where  $\epsilon \sim e^{(\sigma^2 - \lambda_1)T} \ll 1$

- plug into the Wyatt solution (Equation A.57) to obtain  $\mathbf{w}(t)$ .

$$\begin{aligned} \mathcal{C}\mathbf{w}_0 &= \begin{pmatrix} \sigma^2 & \mathbf{0} \\ \mathbf{0} & \mathbf{C} \end{pmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \epsilon \mathbf{v}_1 \end{pmatrix} = \begin{pmatrix} \sigma^2 \mathbf{v}_1 \\ \lambda_1 \epsilon \mathbf{v}_1 \end{pmatrix} \\ e^{Ct}\mathbf{w}_0 &= \begin{pmatrix} e^{\sigma^2 t} \mathbf{v}_1 \\ e^{\lambda_1 t} \epsilon \mathbf{v}_1 \end{pmatrix} \\ \|\mathbf{w}_0\|^2 &= 1 \\ \|e^{Ct}\mathbf{w}_0\|^2 &= (e^{2\sigma^2 t} + \epsilon e^{2\lambda_1 t}) \end{aligned}$$

which yields

$$\mathbf{w}^{\text{RS}}(t) = \frac{\begin{pmatrix} e^{\sigma^2 t} \mathbf{v}_1 \\ e^{\lambda_1 t} \epsilon \mathbf{v}_1 \end{pmatrix}}{(e^{2\sigma^2 t} + \epsilon e^{2\lambda_1 t})^{1/2}} \quad (\text{A.61})$$

## A.9 Some Useful Results From the Simple Environment

### A.9.1 From Papoulis (1984)

$$y = g(x) \quad (\text{A.62})$$

$$\text{(all possible solutions) } x_i \equiv g^{-1}(y) \quad (\text{A.63})$$

$$f_y(y) = \frac{1}{|g'(x_1)|} f_x(x_1) + \frac{1}{|g'(x_2)|} f_x(x_2) + \dots \quad (\text{A.64})$$

$$y = ax + b \quad (\text{A.65})$$

$$f_y(y) = \frac{1}{|a|} f_x\left(\frac{y-b}{a}\right) \quad (\text{A.66})$$

$$z = x + y \quad (\text{A.67})$$

$$f_z(z) = \int_{-\infty}^{-\infty} f(z-y, y) dy \quad (\text{A.68})$$

$$\text{if independent: } f(x, y) = f_x(x) f_y(y) \quad (\text{A.69})$$

$$f_z(z) = \int_{-\infty}^{-\infty} f_x(z-y) f_y(y) dy \quad (\text{A.70})$$

## Appendix B

# Some Computational Issues

### B.1 Code Segments for Section 2.7

#### B.1.1 MATLAB

##### Toy Correlation-based Model (Section 2.7.1)

```
coopernotation=1;
if (coopernotation)
    w_str='m'; y_str='c'; x_str='d';
else
    w_str='w'; y_str='y'; x_str='x';
end

bigfonts;
figpos([380 129 672 619]);
my_subplot(2,2,1);
x=-1:.1:1;
l=length(x);
q=1;

d=-2:.1:2;
C=(1+q*cos(pi*d))*(x(2)-x(1));
plot(d,C); xrange([-2 2]);
yr=yrange;
if ( (yr(1)>0) & (yr(2)>0) )
    yrange([0 yr(2)]);
else
    yrange([-max(abs(yr)) max(abs(yr))]);
```

```
end
xlabel('x-x''');
ylabel('C(x-x''')');
adjustaxes(20);

my_subplot(2,2,2);
d=tony(x(:),1)-tony(x(:)',1);
C=(1+q*cos(pi*d))*(x(2)-x(1));

imagesc(x,x,C); colormap(gray); axis image;
colorbar; drawnow;
xlabel('x'''); ylabel('x');

[V,D]=eig(C);

[D,idx]=sort(diag(-D)); D=-D;
V=V(:,idx);

for i=1:3
    my_subplot(2,3,2,i);
    plot(x,V(:,i)); yr=yrange;
    yrange([-0.5 0.5]);
    title(sprintf('\lambda=%.1f',D(i)));
    xlabel('x');
    if (i==1)
        ylabel(w_str);
    end
    adjustaxes(20);
    drawnow;
end
overtitle(sprintf('q=%.1f',q));
```

### Correlation-based Model of Orientation Selectivity (Section 2.7.2)

```
bigfonts;
figpos([178 124 891 687]);

subtractive_constraint=1;

l=13;

[x,y]=meshgrid(1:l,1:l);
```

---

```

r=-1:1;

ra=6.5;
arbor=(r.^2<=ra^2);
  ra=6.5;   rc=0.24;

C_s=exp(-r.^2/(ra^2*rc^2))-(1/9)*exp(-r.^2/(9*ra^2*rc^2));
C_o=-.8*C_s;

C_sum=C_s+C_o;
C_diff=C_s-C_o;

my_subplot(2,3,1,1); plot(r,C_sum,'-'); title('C^{S}(r)');
zeroaxes;
adjustaxes(20);
my_subplot(2,3,2,1); plot(r,C_diff,'-'); xlabel('r'); title('C^{D}(r)');
zeroaxes;
adjustaxes(20);

r=sqrt((tony(x(:),l^2)-tony(x(:)',l^2)).^2+(tony(y(:),l^2)-tony(y(:)',l^2)).^2);

ra=6.5;
arbor=(r.^2<=ra^2);
  ra=6.5;   rc=0.24;

C_s=exp(-r.^2/(ra^2*rc^2))-(1/9)*exp(-r.^2/(9*ra^2*rc^2));
C_o=-.8*C_s;

C_sum=C_s+C_o;
C_diff=C_s-C_o;

if (subtractive_constraint)
  n=ones([l^2 1]);
  P=eye(l^2)-2*n*n'/(n'*n); % only for C_sum
else
  P=eye(l^2);
end

C_sum=P*C_sum; % no subtractive constraint for C_diff

```

---

```

C_sum=C_sum.*arbor; C_diff=C_diff.*arbor;
    my_subplot(2,3,1,2);
C_plot=C_sum;
idx=find(C_plot>0); C_plot(idx)=setminmax(C_plot(idx),32,64);
idx=find(C_plot<0); C_plot(idx)=setminmax(C_plot(idx),0,32);
image(C_plot); colormap(gray); axis image; title('C^{S}');
adjustaxes(20);
drawnow;

my_subplot(2,3,2,2);
C_plot=C_diff;
idx=find(C_plot>0); C_plot(idx)=setminmax(C_plot(idx),32,64);
idx=find(C_plot<0); C_plot(idx)=setminmax(C_plot(idx),0,32);
image(C_plot); colormap(gray); axis image; title('C^{D}');
adjustaxes(20);
drawnow;
[V,D]=eig(C_sum);
D=diag(D); idx=find(~withineps(imag(D),0)); % find imaginary parts
D(idx)=[]; V(:,idx)=[];
if (~isempty(idx))
    printf('%d imaginary eigenvals from C_sum\n',length(idx));
end
[D,idx]=sort(-D); D=-D;
V=V(:,idx);
for i=1:4

    ii=floor( (i-1)/2); jj=rem((i-1),2);
    my_subplot(4,6,ii+1,jj+5);

    vec=shape(V(:,i)).*shape(arbor(85,:));
    idx=find(vec>0); vec(idx)=setminmax(vec(idx),32,64);
    idx=find(vec<0); vec(idx)=setminmax(vec(idx),0,32);

    % myimage(shape(V(:,i)));
    image(vec); colormap(gray); axis image;
    set(gca,'XTicklabel',[],'YTickLabel',[]);
    title(sprintf('\lambda=%.3f',D(i)));
    adjustaxes(20);
    drawnow;
end
[V,D]=eig(C_diff);
D=diag(D); idx=find(~withineps(imag(D),0)); % find imaginary parts

```

---

```

D(idx)=[]; V(:,idx)=[];
if (~isempty(idx))
    printf('%d imaginary eigenvals from C_diff\n',length(idx));
end
[D,idx]=sort(-D); D=-D;
V=V(:,idx);
for i=1:4
    ii=floor( (i-1)/2); jj=rem((i-1),2);
    my_subplot(4,6,ii+3,jj+5);

    vec=shape(V(:,i)).*shape(arbor(85,:));
    idx=find(vec>0); vec(idx)=setminmax(vec(idx),32,64);
    idx=find(vec<0); vec(idx)=setminmax(vec(idx),0,32);

    % myimage(shape(V(:,i)));
    image(vec); colormap(gray); axis image;
    set(gca,'XTicklabel',[],'YTickLabel',[]);
    title(sprintf('\lambda=%.2f',D(i)));
    adjustaxes(20);
    drawnow;
end

```

## B.2 Code Segments for Section 3.6

### B.2.1 Maple

#### Integral for 2D Laplace (Equation 3.111)

```

assume(m1,positive);
assume(m2,positive);
assume(lambda,positive);

# y>0
simplify((1/(4*lambda^2*m1*m2))*(
int(exp((y-y2)/(m1*lambda))*exp(-y2/(m2*lambda)),y2=y..infinity) +
int(exp(-(y-y2)/(m1*lambda))*exp(-y2/(m2*lambda)),y2=0..y) +
int(exp(-(y-y2)/(m1*lambda))*exp(y2/(m2*lambda)),y2=-infinity..0)));

# y<0
simplify((1/(4*lambda^2*m1*m2))*(
int(exp((y-y2)/(m1*lambda))*exp(-y2/(m2*lambda)),y2=0..infinity) +

```

---

```
int(exp(-(y-y2)/(m1*lambda))*exp(y2/(m2*lambda)),y2=-infinity..y) +
int(exp((y-y2)/(m1*lambda))*exp(y2/(m2*lambda)),y2=y..0));
```

## 2D Laplace Fixed Points (Equation 3.120)

```
assume(lambda,positive);
assume(m1,positive);
assume(m2,positive);

s1:=diff(lambda^3*(m1^5-m2^5)/(m1^2-m2^2)-(lambda^4/4)*(m1^2+m2^2)^2,m1);
s2:=diff(lambda^3*(m1^5-m2^5)/(m1^2-m2^2)-(lambda^4/4)*(m1^2+m2^2)^2,m2);

solve({s1=0,s2=0},{m1,m2});

K2:=9*lambda^4*m1^4+6*lambda^4*m1^2*m2^2+9*lambda^4*m2^4;
K2R:=simplify(subs(m1=R*cos(theta),m2=R*sin(theta),K2));

eq:=diff(K2R,theta);
eq2:=diff(eq,theta);

eval(subs(theta=0,eq2));
eval(subs(theta=Pi/2,eq2));
eval(subs(theta=Pi/4,eq2));
eval(subs(theta=3*Pi/4,eq2));
```

## 2D Gaussian Fixed Points (Equation 3.95)

```
assume(sigma,positive);
assume(m1,positive);
assume(m2,positive);

(1/(m1*m2))*(1/(2*Pi*sigma^2))*int(exp(-(y-y2)^2/(2*sigma^2*m1^2))*
exp(-y2^2/(2*sigma^2*m2^2)),y2=-infinity..infinity);

II:= y-> 1/(sqrt(2*Pi*sigma^2))*1/(sqrt(m1^2+m2^2))*
exp(-y^2/(2*sigma^2*(m1^2+m2^2)));

z2:=simplify(int(z^2*II(z),z=0..infinity));
z3:=factor(simplify(int(z^3*II(z),z=0..infinity)));
```

---

```

R:=(1/3)*z3-(1/4)*z2^2;

s1:=simplify(diff(R,m1));
s2:=simplify(diff(R,m2));

solve({s1,s2},{m1,m2});

H:=hessian(R,[m1,m2]);
ev:=eigenvals(H); # must both be negative

simplify(subs({m1=0,m2=0},ev[1]));
simplify(subs({m1=0,m2=0},ev[2]));

simplify(subs({m1=4*sqrt(2)/(sqrt(Pi)*sigma),m2=0},ev[1]));
simplify(subs({m1=4*sqrt(2)/(sqrt(Pi)*sigma),m2=0},ev[2]));

simplify(subs({m2=4*sqrt(2)/(sqrt(Pi)*sigma),m1=0},ev[1]));
simplify(subs({m2=4*sqrt(2)/(sqrt(Pi)*sigma),m1=0},ev[2]));

simplify(subs({m1=0,m2=((Pi*sigma^2*m1-32)/Pi)^(1/2)/sigma},ev[1]));
simplify(subs({m1=4*sqrt(2)/(sqrt(Pi)*sigma),m2=0},ev[2]));

```

## 2D Uniform Fixed Points (Equation 3.104)

```

with(linalg);
assume(a,positive);
assume(m1,positive);
assume(m2,positive);

E2:=simplify(int(z^2*(1/(2*a*m2)),z=0..a*(m2-m1))+
  int(z^2*(a*(m1+m2)-z)/(4*a^2*m1*m2),z=a*(m2-m1)..a*(m2+m1)));
E3:=simplify(int(z^3*(1/(2*a*m2)),z=0..a*(m2-m1))+
  int(z^3*(a*(m1+m2)-z)/(4*a^2*m1*m2),z=a*(m2-m1)..a*(m2+m1)));

R:=(simplify((1/3)*E3-(1/4)*E2^2));
s1:=simplify(diff(R,m1));
s2:=simplify(diff(R,m2));

solve({s1,s2},{m1,m2});

H:=hessian(R,[m1,m2]);
ev:=eigenvals(H); # must both be negative

```

```

simplify(subs({m1=0,m2=9/(2*a)},ev[1]));
simplify(subs({m1=0,m2=9/(2*a)},ev[2]));

simplify(subs({m1=2*sqrt(5)/a,m2=2/a},ev[1]));
simplify(subs({m1=2*sqrt(5)/a,m2=2/a},ev[2]));

simplify(subs({m1=18/(5*a),m2=18/(5*a)},ev[1]));
simplify(subs({m1=18/(5*a),m2=18/(5*a)},ev[2]));

```

### Integral for 2D Laplace-Uniform Mixture (Equation 3.64)

```

assume(m1,positive);
assume(m2,positive);
assume(lambda,positive);
assume(a,positive);

simplify((1/(2*a*m2))*(1/(2*lambda*m1))*(int(exp(-y/(m1*lambda)),y=0..a*m2)+
    int(exp(y/(m1*lambda)),y=-a*m2..0)));

f1:=y->factor((1/(4*lambda*m1*a*m2))*simplify(int(exp(u/(m1*lambda)),
    u=(-a*m2-y)..0)+int(exp(-u/(m1*lambda)),u=0..(a*m2-y))));
f2:=y->factor((1/(4*lambda*m1*a*m2))*simplify(int(exp(u/(m1*lambda)),
    u=(-a*m2-y)..(a*m2-y))));

# should be 1/2
simplify(int(f1(y),y=0..a*m2)+int(f2(y),y=a*m2..infinity));

```

## B.2.2 MATLAB

### Visualizing 1D Laplace (Equation 3.34)

```

lambda=1;
y=lambda*randlaplace([1 10000]);
[n,x]=hist(y,100); n=n/(x(2)-x(1))/sum(n);
f=1/(2*lambda)*exp(-abs(x)/lambda);
plot(x,n,'yo',x,f,'y-'); yr=yrange; yrange([0 yr(2)]);

```

### Visualizing 2D Mixture of Laplace-Laplace (Equation 3.112)

```

m1=1; m2=2; lambda=1;
y=[m1 m2]*lambda*randlaplace([2 10000]);
[n,x]=hist(y,100); n=n/(x(2)-x(1))/sum(n);

```

```
f=1/(2*lambda*(m1-m2)*(m1+m2))*(m1*exp(-abs(x)/(m1*lambda))- ...
    m2*exp(-abs(x)/(m2*lambda)));

plot(x,n,'yo',x,f,'y-'); yr=yrange; yrange([0 yr(2)]);
```

### Visualizing 2D Mixture of Gaussian-Gaussian (Equation 3.87)

```
m1=1; m2=2; sigma=2;
y=[m1*sigma m2*sigma]*randn([2 10000]);
[n,x]=hist(y,100); n=n/(x(2)-x(1))/sum(n);

f=1/sqrt(2*pi*sigma^2)*1/sqrt(m2^2+m1^2)*exp(-x.^2/(2*sigma^2*(m1^2+m2^2)));

plot(x,n,'yo',x,f,'y-'); yr=yrange; yrange([0 yr(2)]);
```

### Visualizing 2D Mixture of Uniform-Uniform

```
a=1; m1=1; m2=2;
y=2*a*[m1 m2]*(rand([2 10000])-.5);
[n,x]=hist(y,100); n=n/(x(2)-x(1))/sum(n);

f= (1/(2*a*max(m1,m2)))*(abs(x)<a*abs(m2-m1))+ ...
    ((m2-abs(x)+m1)/(4*a*m1*m2)).*((abs(x)>a*abs(m2-m1)) & (abs(x)<a*(m2+m1)));

plot(x,n,'o',x,f,'y-'); yr=yrange; yrange([0 yr(2)]);
```

### Visualizing 2D Mixture of Laplace-Uniform (Equation 3.66)

```
a=5; m1=1; m2=2;lambda=1;
y=sum([m2*a*2*(rand([1 10000])-.5);m1*lambda*randlaplace([1 10000])]);

[n,x]=hist(y,100); dx=(x(2)-x(1)); n=n/(x(2)-x(1))/sum(n);

f1=(1/(4*a*m2))*(2-exp(-(a*m2+abs(x))/(m1*lambda))- ...
    exp(-(a*m2-abs(x))/(m1*lambda))).*(abs(x)<(a*m2));
f2=(1/(4*a*m2))*(exp((a*m2-abs(x))/(m1*lambda))- ...
    exp(-(a*m2+abs(x))/(m1*lambda))).*(abs(x)>=(a*m2));
f=f1+f2;

c1=(1/(4*a*m2))*exp((a*m2)/(m1*lambda));
c2=(1/(4*a*m2))*exp(-(a*m2)/(m1*lambda));
c3=(2/(4*a*m2));
```

```
f1=(c3-c2*(exp(-abs(x)/(m1*lambda))+exp(abs(x)/(m1*lambda)))).*(abs(x)<(a*m2));  
f2=(c1-c2)*exp(-abs(x)/(m1*lambda)).*(abs(x)>=(a*m2));  
  
f=f1+f2;  
  
plot(x,n,'yo',x,f,'y-'); yr=yrange; yrange([0 yr(2)]);
```

# Index

- A**
- anatomy ..... 24
- arbor function  
 definition ..... 50
- B**
- BCM  
 assumptions ..... 3  
 averaged variables ..... 3  
 cost function ..... 58  
 dependence on  $\tau$  ..... 43  
 dependence on noise ..... 44, 65  
 deprivation ..... 42  
 direction selectivity ..... 92  
 learning rule ..... *see* modification equation  
 loss ..... 116  
 measuring  $\tau$  ..... 13  
 modification equation ..... 8  
   general properties ..... 62  
 multi-modality ..... 58, 116  
 N linearly independent inputs ..... 118  
   fixed points ..... 118  
 negative activity ..... *see* negative activity  
 negative weights ..... *see* negative weights  
 one dimension, constant input ..... 9  
   fixed point ..... 9  
   oscillations ..... 12  
   oscillatory perturbation ..... 12  
   simulations ..... 10  
   upper bound on oscillation frequency ..... 13  
 parameters ..... 42  
 parameters (valid) ..... 45  
 projection pursuit ..... 58  
 quadratic form ..... 8  
 risk ..... 117  
 role of  $\tau$  ..... 12  
 role of preprocessing ..... 88  
 sensitivity to outliers ..... 92, 117  
 sigmoid ..... *see* sigmoid  
 sliding threshold ..... 8  
   experimental measurement ..... 19  
   homeostasis ..... 23  
   priming ..... 23  
 time course of deprivation ..... 43  
 time scales ..... 3, 13  
 two dimensions ..... 16, 59  
   binocular deprivation ..... 79  
   deprivation ..... 75  
   example cost maximization ..... 59  
   fixed points ..... 17, 111  
   Laplace and uniform distributions ..... 75  
   monocular deprivation ..... 75  
   reverse suture ..... 83  
   role of sigmoid ..... 82  
   strabismus ..... 83  
   two eyes ..... 72
- binocular deprivation ..... 31  
 correlation-based rules ..... 55  
 PCA ..... 41  
 simple environment ..... 79
- C**
- chickens ..... *see* penguins
- correlation-based rules ..... 48  
 arbor function ..... 50  
 assumptions ..... 48  
 constraints ..... 49  
 monocular deprivation ..... 53  
 multiplicative constraint ..... 49  
 non-local constraints ..... 55  
 orientation selectivity ..... 51  
 PCA ..... 49

- 
- problems ..... 53
- binocular deprivation ..... 55
  - non-robust orientation selectivity ..... 53, 55
- subtractive constraint ..... 49
- toy model ..... 50
- weight bounds ..... 49
- critical period ..... 30
- D**
- dark rearing ..... 31
- deprivation
- binocular ..... 31
  - dark rearing ..... 31
  - experimental results ..... 39
  - model of ..... *see* model
  - monocular ..... 31
  - noise ..... *see* model
  - relative timing ..... 37
  - reverse suture ..... 31
  - strabismus ..... 32
  - stripe rearing ..... 32
  - strobe rearing ..... 33
- difference of Gaussians
- ON center cells ..... 33
  - X and Y cells ..... 87
- direction selectivity
- definition ..... 28
  - effects of binocular deprivation ..... 31
  - effects of strobe rearing ..... 33
  - index measure ..... 92
  - lagged and non-lagged cells ..... 89
  - model of ..... 89
    - BCM ..... 92
    - kurtosis ..... 92
    - PCA ..... 92
    - skewness ..... 92  - parallel with cortical misalignment ..... 92
  - spatiotemporal receptive fields ..... 89
  - strobe environment ..... 92
- DOG ..... *see* difference of Gaussians
- F**
- functional disconnection ..... 26
- G**
- Gabor filters ..... 88
- H**
- Hebb rule ..... 7
- assumptions ..... 3
  - averaged variables ..... 3
  - correlation-based rules ..... 49
  - instability ..... 7
  - negative activity ..... *see* negative activity
  - negative weights ..... *see* negative weights
  - PCA ..... *see* PCA
  - time scales ..... 3
- homeostasis ..... 10
- I**
- ICA ..... 48, 86
- K**
- kurtosis ..... 62
- dependence on noise ..... 65
  - direction selectivity ..... 92
  - Gaussian distribution ..... 69
  - Laplace distribution ..... 69
  - learning rule ..... 63, 64
  - sensitivity to outliers ..... 92
  - stability ..... 63
  - two dimensions
    - binocular deprivation ..... 79
    - Laplace and uniform distributions ..... 75
    - monocular deprivation ..... 75
    - reverse suture ..... 83
    - strabismus ..... 83
    - uniform distribution ..... 69
- L**
- lateral geniculate nucleus ..... 26
- function of ..... 28
  - receptive field ..... 27
  - response properties ..... 26
- LGN ..... *see* lateral geniculate nucleus
- locality ..... 55
- definition of ..... 6
  - quasi-local ..... 6
-

- 
- long term depression ..... 19
    - enhanced by priming ..... 23
    - heterosynaptic ..... 21
    - homosynaptic ..... 21
  - long term potentiation ..... 19
  - LTD ..... *see* long term depression
  - LTP ..... *see* long term potentiation
- M**
- model
    - architecture ..... 33
    - assumptions about activation ..... 3
    - assumptions about modification ..... 6
    - binocular deprivation ..... 36
    - deprivation ..... 35
    - locality ..... 6
    - low dimensional ..... 69
    - monocular deprivation ..... 36
    - natural scene environment ..... 33
    - noise ..... 36
    - reverse suture ..... 36
    - simplicity ..... 1
  - monocular deprivation ..... 31
    - correlation-based rules ..... 53
    - noise dependence of models ..... 65
    - simple environment ..... 75
    - TTX ..... 66
- N**
- negative activity ..... 4
  - negative weights ..... 4, 49
  - neuron
    - action potential ..... 3
    - anatomy ..... 3
    - selectivity ..... 6
    - sparse coding ..... 92
    - synapse ..... *see* synapse
- O**
- ocular dominance
    - definition ..... 28
    - effects of binocular deprivation ..... 31
    - effects of monocular deprivation ..... 31
    - effects of reverse suture ..... 32
    - effects of strabismus ..... 32
    - modeling ..... 33
    - shift ..... 31
  - Oja
    - stabilized Hebb rule ..... *see* PCA
  - optics ..... 30
  - orientation selectivity
    - alignment of LGN cells ..... 28, 30
    - correlation-based rules ..... 51
    - definition ..... 28
    - discovery ..... 29
    - effects of binocular deprivation ..... 31
    - modeling ..... 33, 48
    - present at birth ..... 30
  - output distribution ..... 60
    - definition ..... 18
    - Laplace ..... 61
    - natural scenes ..... 60
- P**
- PCA
    - cost function ..... 59
    - dependence on noise ..... 65
    - direction selectivity ..... 92
    - fixed points ..... 15, 121
    - maximum variance ..... 122
    - non linear ..... 64
    - one dimension ..... 16
    - projection pursuit ..... 59
    - properties ..... 15
    - solution ..... 122
    - stabilized Hebb rule ..... 7
    - time course of deprivation ..... 41
    - two dimensions ..... 18, 59
      - example cost maximization ..... 59
      - fixed points ..... 112
    - weight normalization ..... 15, 63
    - wyatt solution ..... 122
  - penguins ..... *see* rutabaga
  - projection pursuit
    - BCM ..... 58
    - cost function ..... 58
    - definition ..... 57
-

- 
- kurtosis ..... 61  
 modification function ..... 58  
 PCA ..... 59  
 skewness ..... 61  
 two dimensional example ..... 59
- R**
- receptive field  
   changes with structure removal ..... 93  
   definition ..... 25  
   LGN ..... *see* LGN  
   measuring ..... 27  
   retina ..... *see* retina  
   reverse correlation ..... 27  
   visual cortex ..... *see* visual cortex  
 retina  
   receptive field ..... 27  
   response properties ..... 26  
   X cells ..... 28, 87  
   Y cells ..... 28, 87  
 reverse correlation ..... 27  
 reverse suture ..... 31  
   simple environment ..... 83  
 rutabaga ..... *see* chickens
- S**
- selectivity ..... 6  
   orientation ..... 6  
 sigmoid ..... 4, 117  
   necessary for BCM ..... 62  
   projection pursuit ..... 62  
 simplicity ..... 1  
 skewness ..... 62  
   dependence on noise ..... 65  
   direction selectivity ..... 92  
   learning rule ..... 63, 64  
   sensitivity to outliers ..... 92  
   stability ..... 63  
 sparse coding algorithm ..... 48  
 spontaneous activity ..... 4, 117  
 strabismus ..... 32  
   simple environment ..... 83  
 stripe rearing ..... 32
- strobe rearing ..... 33  
 structure removal ..... 92  
 synapse ..... 3  
   effective ..... 4  
   efficacy ..... 3  
   modification ..... 3, 6
- T**
- TTX ..... 66  
   monocular deprivation ..... 66
- V**
- visual cortex  
   binocular cell ..... 28  
   direction selectivity .... *see* direction selectivity  
   monocular cell ..... 28  
   ocular dominance ..... *see* ocular dominance  
   orientation selectivity *see* orientation selectivity  
   receptive field ..... 28  
 visual pathway ..... 24  
 visual system ..... 24
- Y**
- Yaus ..... v
-