Lecture 2
Feedforward and recurrent networks

Jochen Braun

April 8, 2006

Synpopsis:

1) Derivation of firing-rate models
2) Feedforward networks and coordinate transforms
3) Recurrent networks and selective amplification
4) Recurrent networks and associative memory

Credits:

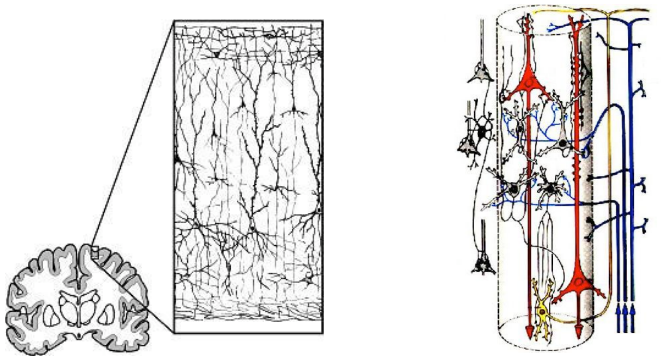
Dayan & Abbott, Chapter 7
Salinas & Abbott (1995) Transfer of coded information from sensory to motor networks. J Neurosci 15: 6461-6474.

1 Derivation of firing-rate models

1.1 The need to simplify

A typical network of cortical neurons involves at a minimum some millions of excitatory and inhibitory neurons (150,000 per mm^3), each with complex dendrites and axons, with many types of membrane channels for Na^+, K^+, Ca^{2+}, etc. and receiving input and emitting output through 8,000 synapses on average. Models of this scale are extremely expensive and unwieldy (many time-scales!).

To simplify, we can consider the firing rate of sub-populations rather than the spiking of neurons. For example, we can consider a cortical column as a single model node.



1.2 Firing-rate model

For one model node, we define the input (pre-synaptic) firing rate vector $\vec{u}(t)$ and the output (post-synaptic) firing rate $v(t)$. We assume a synaptic current with time-course $K_s(t)$, a synaptic weight of w_b and compute the total synaptic current I_s:

$$I_s(t) = \sum_b I_b(t) = \sum_b w_b \sum_i K_s(t-t_i) = w_b \int_{-\infty}^{t} K_s(t-\tau)\,\rho_b(\tau)\,d\tau \qquad \rho_b(\tau) \equiv \sum_i \delta(\tau - t)$$

where b indexes synapses and i indexes spikes at each synapse.

We can ignore "spike train variability" (i.e., replace spike train $\rho_b(t)$ by spike rate $u(t)$) if (i) the synapse current $K_s(t)$ is slow or (ii) the spikes arriving at different synapses are uncorrelated.

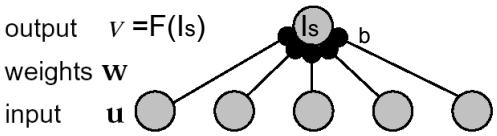


Figure 7.1: Feedforward inputs to a single neuron. Input rates **u** drive a neuron at an output rate v through synaptic weights given by the vector **w**.

1.3 Two time constants

To a first approximation, the synaptic current $I_s(t)$ is a low-pass filtered version of the input firing $u(t)$. This means that we can express their relation in terms of a differential equation:

$$I_s = \sum_b w_b \int_{-\infty}^{t} K_s(t-\tau)\,u_b(\tau)\,d\tau \qquad \Leftrightarrow \qquad \tau_s \frac{dI_s}{dt} = -I_s + \sum_b w_b\,u_b$$

This equation is exact, if the $K_s(t)$ is an exponential decay with time-constant τ_s (try to prove this!):

$$K_s(t) = \frac{1}{\tau_s}\,\exp\left(-\frac{t}{\tau_s}\right)$$

For an electrotonically compact dendrite, τ_s reflects the time-constant of the synaptic conductance, which may be as short as a few milliseconds (e.g. AMPA glutamate receptors). For a synapse on the distal part of a thin dendrite, it may be larger.

Similarly, the output firing rate $v(t)$ approximates a low-pass filtered version of the synaptic current $I_s(t)$, producing a second differential equation:

$$\tau_r \frac{dv}{dt} = -v + F(I_s)$$

where $F(I_s)$ is the "activation function".

The time-constant τ_r is NOT the membrane time-constant. Most network models use considerably value of τ_r that is considerably less than the membrane time-constant. Detailed simulations show that the effective value of τ_r depends on the firing-rate regime. Low-pass filtering by the membrane time-constant can be neglected if the neuron is always firing (i.e., always close to threshold) but must be taken into account otherwise.

1.4 Compact model

We now consider ways of combining our equations for I_s and v into a more compact model:

$$\tau_s \frac{dI_s}{dt} = -I_s + \sum_b w_b u_b = -I_s + \boldsymbol{w} \cdot \boldsymbol{u} \qquad \tau_r \frac{dv}{dt} = -v + F(I_s)$$

If $\tau_r >> \tau_s$, we can replace I_s by its equilibrium value

$$I_s = \boldsymbol{w} \cdot \boldsymbol{u} \qquad \Rightarrow \qquad \tau_r \frac{dv}{dt} = -v + F(\boldsymbol{w} \cdot \boldsymbol{u})$$

Alternatively, if $\tau_r << \tau_s$, we can replace v by its equilibrium value $F(I_s)$ and use

$$\tau_s \frac{dI_s}{dt} = -I_s + \boldsymbol{w} \cdot \boldsymbol{u} \qquad \text{with} \qquad v = F(I_s)$$

In both cases, the steady-state firing rate v_∞ is given by

$$v_\infty = F(\boldsymbol{w} \cdot \boldsymbol{u})$$

These approximations suffice to show the computational potential of network models. More accurate models would explicitly model individual spikes. However, provided that the spikes in a spiking model do not synchronize, the predictions of rate models are typically quite accurate.

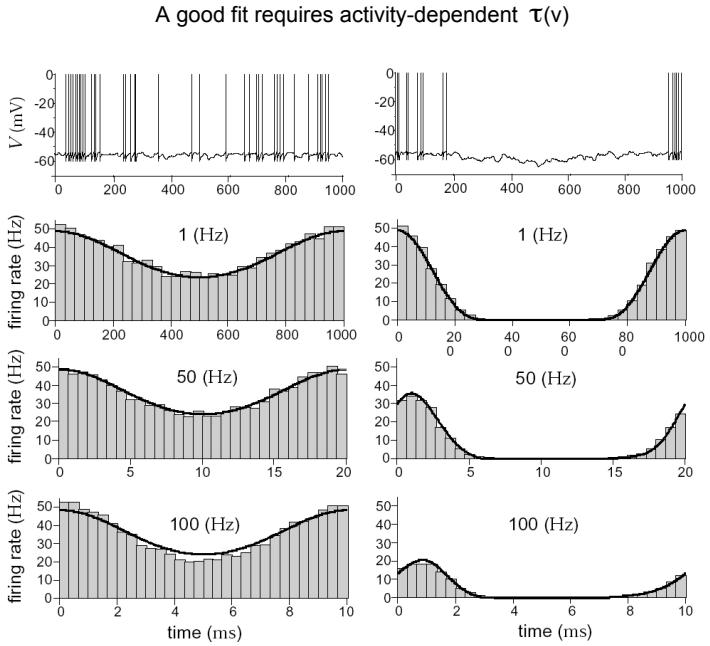


Figure 7.2: Firing rate of an integrate-and-fire neuron receiving balanced excitatory and inhibitory synaptic input and both constant and sinusoidally varying injected current. For the left panels, the constant component of the injected current was adjusted so the firing never stopped during the oscillation of the varying part of the injected current. For the right panel, the constant component was lowered so the firing stopped during part of the cycle. The upper panels show two representative voltage traces of the model cell. The histograms beneath these traces were obtained by binning spikes generated over multiple cycles. They show the firing rate as a function of the time during each cycle of the injected current oscillations. The different rows show 1, 50, and 100 Hz oscillation frequencies for the injected current. The solid curves show the fit of a firing-rate model that involves both instantaneous and low-pass filtered effects of the injected current. (Adapted from Chance *et al*, 2000.)

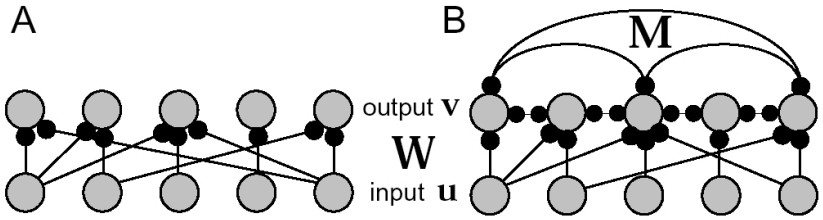


Figure 7.3: Feedforward and recurrent networks. A) A feedforward network with input rates $\mathbf{u}$, output rates $\mathbf{v}$, and a feedforward synaptic weight matrix $\mathbf{W}$. B) A recurrent network with input rates $\mathbf{u}$, output rates $\mathbf{v}$, a feedforward synaptic weight matrix $\mathbf{W}$, and a recurrent synaptic weight matrix $\mathbf{M}$. Although we have drawn the connections between the output neurons as bidirectional, this does not necessarily imply connections of equal strength in both directions.

1.5 Feedforward and recurrent networks

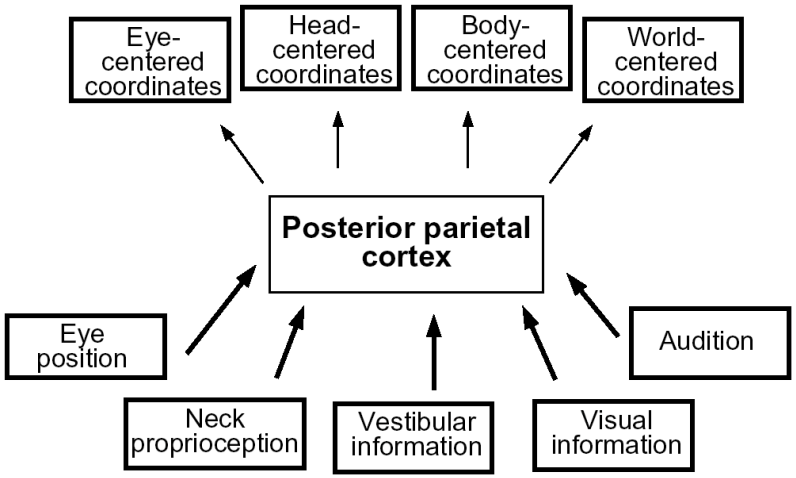
In a feedforward network, N_u input units with rates $\boldsymbol{u}$ are driving N_v output units with rates $\boldsymbol{v}$, it is convenient to use vector notation and assemble the synaptic weights w_{oi} (i.e., from input unit i to output unit o, not the order of indices) into a matrix $\boldsymbol{W}$. The output rates are then determined by a system of equations as follows:

$$\tau_r \frac{d\boldsymbol{v}}{dt} = -\boldsymbol{v} + \boldsymbol{F}(\boldsymbol{W}\cdot\boldsymbol{u}) \qquad \text{or} \qquad \tau_r \frac{dv_o}{dt} = -v_o + F\left(\sum_i W_{oi}\, u_i\right)$$

In a recurrent network, there exist additional interconnections between output neurons, which are described by a synaptic matrix $\boldsymbol{M}$. Matrix element $M_{aa'}$ gives the weight of the connection from output unit a' to output unit a (again note order of indices). The equations governing this recurrent network are:

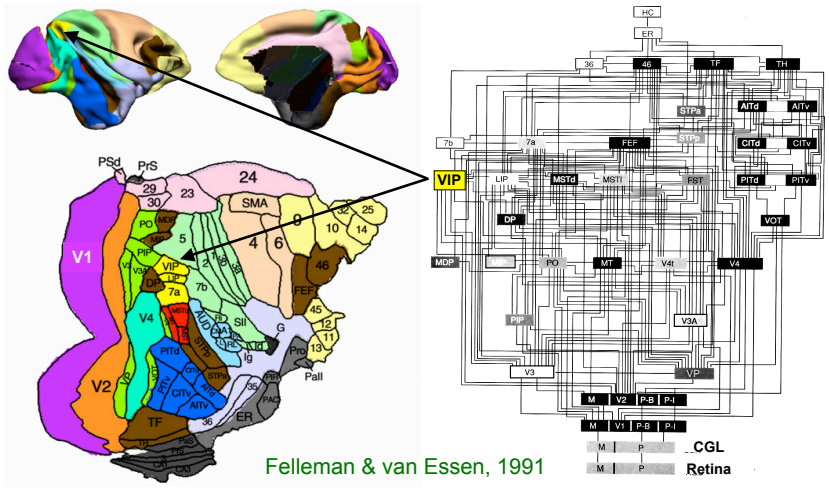
$$\tau_r \frac{d\boldsymbol{v}}{dt} = -\boldsymbol{v} + \boldsymbol{F}(\boldsymbol{M}\cdot\boldsymbol{v} + \boldsymbol{W}\cdot\boldsymbol{u}) \qquad \text{or} \qquad \tau_r \frac{dv_o}{dt} = -v_o + F\left(\sum_{o'} M_{oo'}\, v_{o'} + \sum_i W_{oi}\, u_i\right)$$

In biologically detailed networks, the connectivity matrices $\boldsymbol{W}$ and $\boldsymbol{M}$ exhibits certain patterns, for example, that weights originating from one neuron must all have the same sign, because individual neurons are either excitatory or inhibitory ("Dale's law"). However, in firing-rate models, connections originating from one NODE (i.e., a population of excitatory and inhibitory neurons) can easily have different signs. Accordingly, a rate model does not have to be constrained by Dale's Law.



2 Feedforward networks and coordinate transforms

2.1 Parietal cortex



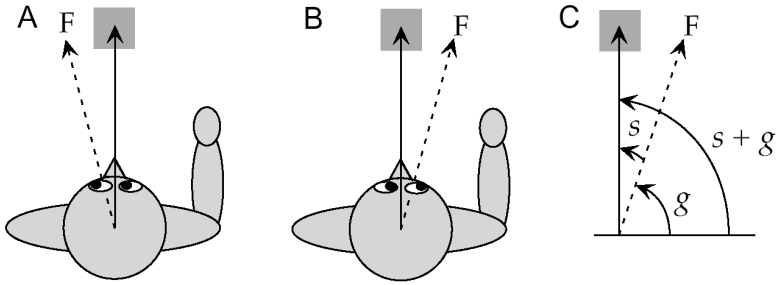
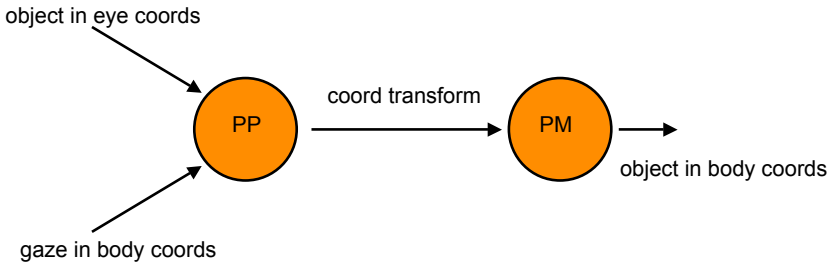


Figure 7.4: Coordinate transformations during a reaching task. A, B) The location of the target (the grey square) relative to the body is the same in A and B, and thus the movements required to reach toward it are identical. However, the image of the object falls on different parts of the retina in A and B due to a shift in the gaze direction produced by an eye rotation that shifts the fixation point F. C) The angles used in the analysis: s is the angle describing the location of the stimulus (the target) in retinal coordinates; g is the gaze direction angle, indicating the orientation of the eyes relative to the body. The direction of the target relative to the body is $s + g$.

Reaching movements necessitate coordinate transforms between retinal and body coordinates. For example, target direction in body coordinates is the sum of target direction in eye coordinates and gaze direction in body coordinates.

Visual neurons in parietal cortex are sensitive to particular retinal locations: their receptive field is in EYE coordinates. Parietal cortex projects to premotor cortex, where some neurons also respond to visual stimuli. However, their receptive field is in BODY coordinates: Changing gaze direction leaves the tuning curve unchanged, but rotating the head shifts the tuning curve shifts by the same angle.

How can visual neurons with receptive fields in EYE coordinates drive pre-motor neurons with receptive fields in BODY coordinates?



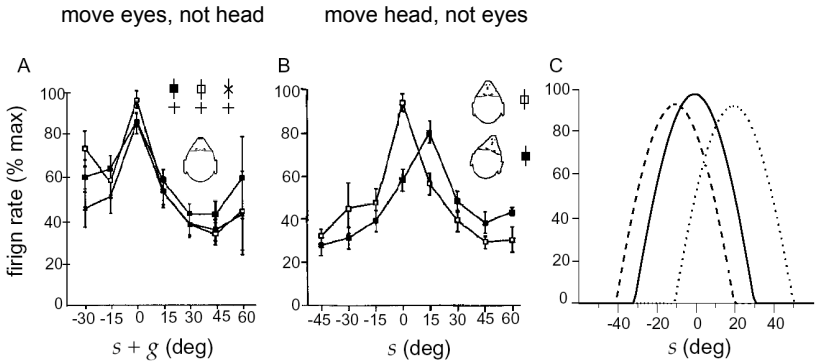
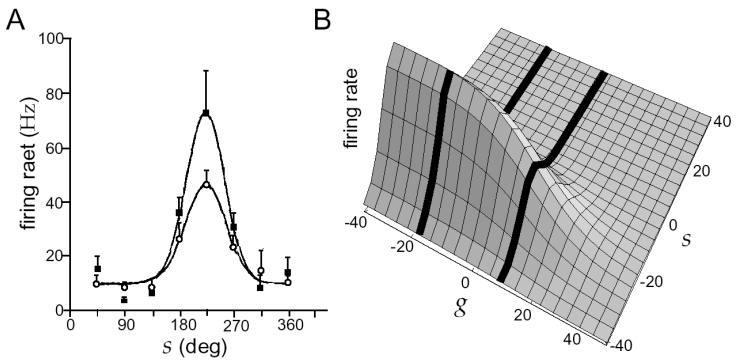


Figure 7.5: Tuning curves of a visually responsive neuron in the premotor cortex of a monkey. Incoming objects approaching at various angles provided the visual stimulation. A) When the monkey fixated on the three points denoted by the cross symbols, the response tuning curve did not shift with the eyes. In this panel, unlike B and C, the horizontal axis refers to the stimulus location in head-based, not retinal, coordinates ($s+g$, not s). B) Turning the monkey's head by 15° produced a 15° shift in the response tuning curve as a function of retinal location, indicating that this neuron encoded the stimulus direction in head-based coordinates. C) Model

The answer was discovered by Richard Andersen and colleagues, who recorded in parietal cortex and found that visual receptive fields are often linearly modulated by eye position (Zipser, Andersen, 1988). Subsequent work (Salinas Abbott, 1995; Pouget, Sejnowski, 1995) proposed the following dependence on stimulus position s (in retinal coordinates) and on gaze position g (in body coordinates) for these neurons:

$$u = f_u(s-\xi, g-\gamma) = \frac{\exp\left[\kappa\,(g-\gamma)\right]}{1+\exp\left[\kappa\,(g-\gamma)\right]}\,\exp\left(-\frac{(s-\xi)^2}{2\sigma^2}\right)$$

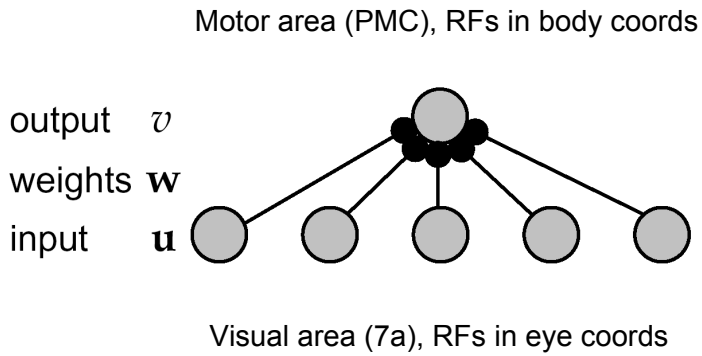
where ξ and γ are the preferred stimulus position and the critical gaze position, respectively. These functions describe a Gaussian tuning for $s-\xi$ and a sigmoidal increase in response gain (or amplitude) for $g-\gamma$. The neurons are thus "stimulus-tuned" and "gaze-modulated".



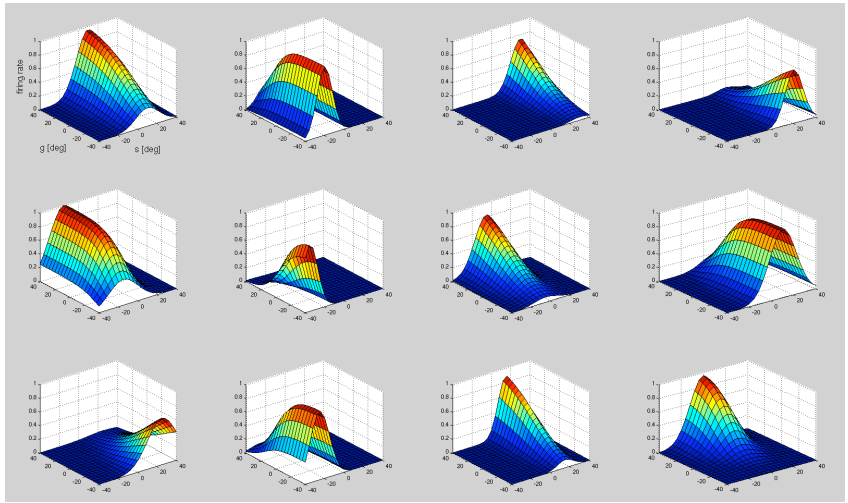
Our input layer u models visual area 7a and consists of a population of such gain-modulated units $u(\xi,\gamma)$, representing all possible combinations of ξ and γ. This population feeds via a connectivity matrix $w(\xi,\gamma)$ into an output unit v, which models a unit in pre-motor cortex. We neglect dynamic effects and consider only the steady-state response of the output unit, which is given by

$$v_\infty = F\left[\rho_\xi\,\rho_\gamma \int w(\xi,\gamma)\, f_u(s-\xi, g-\gamma)\, d\xi\, d\gamma\right]$$

Is there a connectivity matrix $w(\xi,\gamma)$, such that the output response is a function of $s+g$? Yes, there is! In other words, our feed-forward network can respond to stimulus location in BODY coordinates.



Which weighted sum extracts s+g ?



Proof:

Change the integration variables:

$$\xi \to \xi - g$$
$$\gamma \to \gamma + g$$

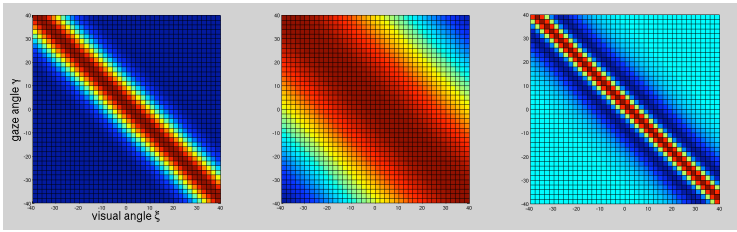
$$v_\infty = F\left[\rho_\xi \rho_\gamma \int w(\xi - g, \gamma + g)\, f_u(s - \xi + g, -\gamma)\, d\xi\, d\gamma\right]$$

We can now see that v_∞ is a function of $s+g$ provided that $w(\xi - g, \gamma + g) = w(\xi, \gamma)$ (i.e., provided that he g-dependencies of w cancel). This, in turn, is the case if w is a function of $\xi + \gamma$:

$$w(\xi, \gamma) = w(\xi + \gamma)$$

$$v_\infty(s + g) = F\left[\rho_\xi \rho_\gamma \int w(\xi + \gamma)\, f_u(s + g - \xi, -\gamma)\, d\xi\, d\gamma\right]$$

QED.

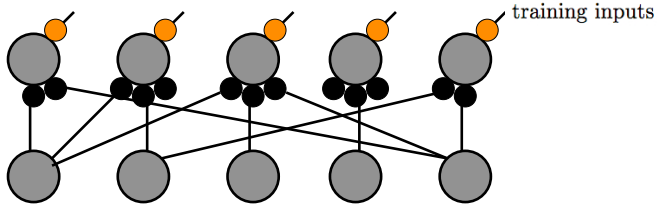


2.2 Feed-forward coordinate transform: an example

We consider a particular example for coordinate transformations on the basis of gain modulated response. We consider the feedforward projection between a sensory network, where activity depends on retinal position x and gaze position y, and a motor network, where activity depends on target position z.

R_j^m motor firing rate — c_j preferred target position z

η_j motor noise



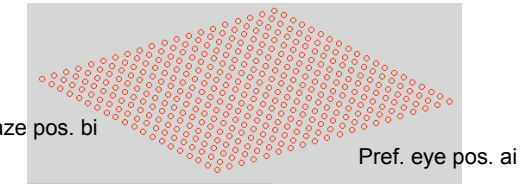
R_i^s sensory firing rate — a_i preferred retinal position x

η_i sensory noise — b_i preferred gaze position y

Sensory activity is given by a Gaussian tuning curve (average activity) for stimulus position and gaze position with additive noise. The noise is Gaussian distributed around a zero mean. Its standard deviation is very large and equal to the average firing rate:

$$R_i^s = R_{max}\, f\,(a_i - x)\; g\,(b_i - y) + \eta_i$$

$$f\,(a_i - x) = \exp\left(-\frac{(a_i - x)^2}{2\,\sigma^2}\right) \qquad g\,(b_i - y) = \exp\left(-\frac{(b_i - y)^2}{2\,\sigma^2}\right)$$



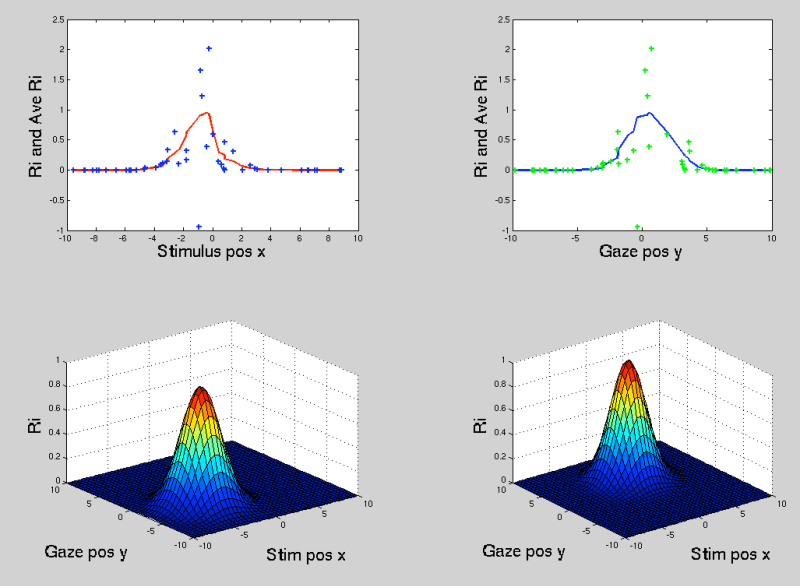
Alternatively, sensory activity might exhibit Gaussian tuning for stimulus position only, with multiplicative gain-modulation by gaze position:

$$R_i^s = R_{max}\, f\,(a_i - x)\; g'\,(b_i - y) + \eta_i$$

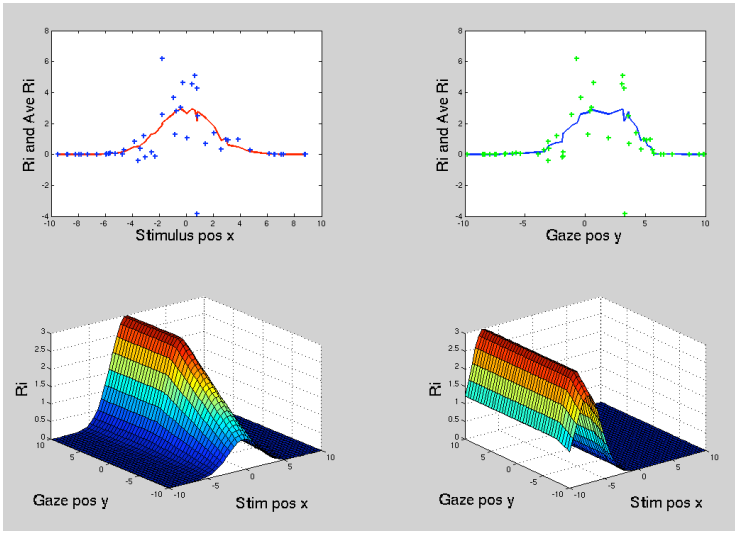
$$f\,(a_i - x) = \exp\left(-\frac{(a_i - x)^2}{2\,\sigma^2}\right) \qquad g'\,(b_i - y) = \frac{\exp\left[\kappa_i\,(b_i - y)\right]}{1 + \exp\left[\kappa_i\,(b_i - y)\right]}$$

where b_i is the 'threshold' gaze position, and s_i is the positive or negative slope of the gaze-dependence.

Gaussian times Gaussian



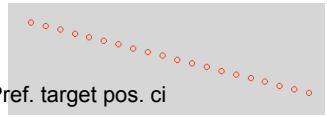
Gaussian times linear ramp

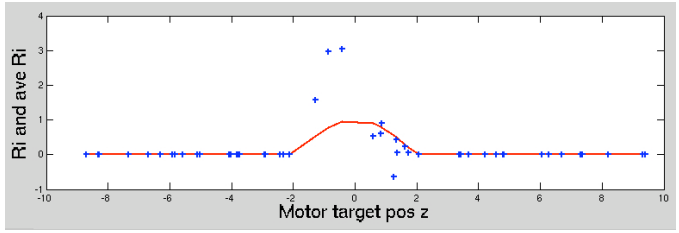


Motor activity DURING THE TRAINING PHASE is modeled in a similar way. Note, that we won't need these formulas after the feedforward input from the sensory network is put in place!

$$R_j^m = h(c_j - z) + \eta_j$$

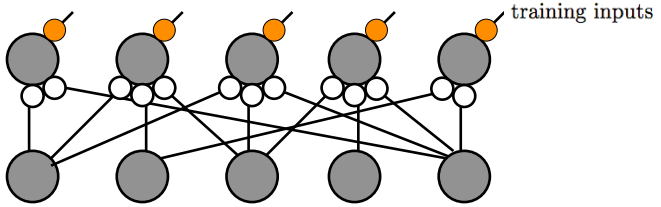
$$h(c_j - z) = R_{max}\left[\cos\left(\frac{\pi(c_j - z)}{2\sigma}\right)\right]_+$$





Training phase with immature synapses ○

R_j^m motor firing rate
η_j motor noise
c_j preferred target position z

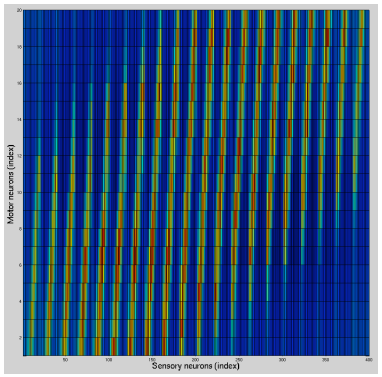


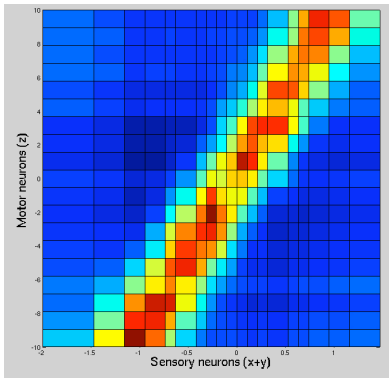
R_i^s sensory firing rate
η_i sensory noise
a_i preferred retinal position x
b_i preferred gaze position y

In the training phase, the system generates random movements and compares correlated sensory inputs. For example, a baby randomly moves its hands and visually follows these movements, giving its brain the opportunity to associate eye position, gaze position, and hand position. Once the association has been learned, the baby can visually coordinate movements.

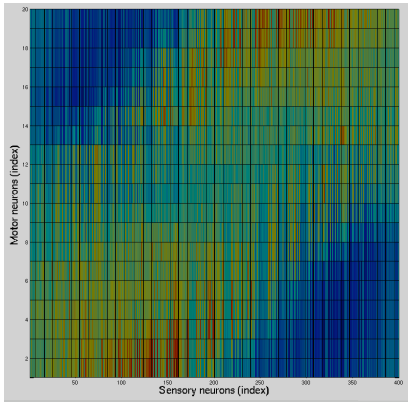
A suitable set of connection weights W_{ji} is given by is the *covariance rule*, which will be discussed later in the course. The covariance rule states that positive and negative connections weights should be proportional to the *covariance* in the activities of each pair of neurons i and j:

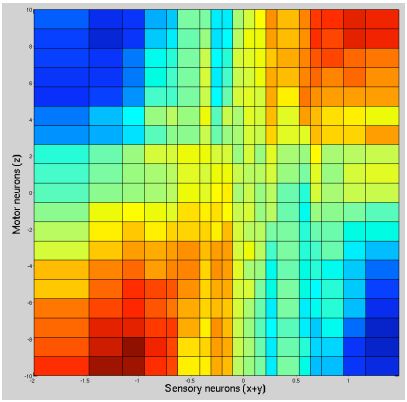
$$W_{ji} = \langle R_j^m R_i^s \rangle - \langle R_i^m \rangle \langle R_j^s \rangle = \langle R_j^m R_i^s \rangle - k$$





Gaussian times linear ramp



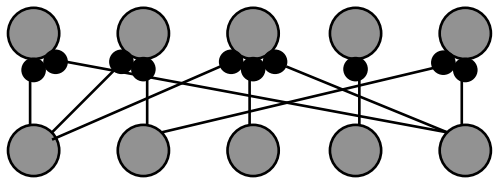


Given the tuning functions of sensory and motor layers, we can compute the desired set of connection weights in an alternative way, by integrating the product of tuning functions over all possible eye and gaze positions:

$$W_{ji} = \int dz\, g\left(|c_i - z|\right)\, f\left(|a_i - x|\,, |b_i - y|\right) - k$$

Operating phase with mature synapses

R_j^m	motor firing rate	c_j	preferred target position z
η_j	motor noise		

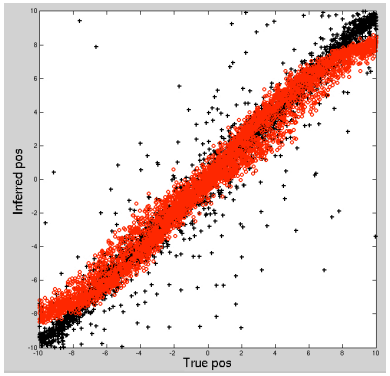


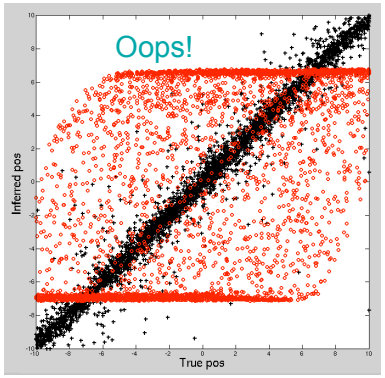
R_i^s	sensory firing rate	a_i	preferred retinal position x
η_i	sensory noise	b_i	preferred gaze position y

In the mature system, activity in the sensory network drives activity in the motor network through synaptic couplings W_{ji}. The last step is to *decode* the motor activity, that is, to translate activity in the motor population into a motor position z. We use an activity-weighted average over the preferred motor positions c_j of all neurons j:

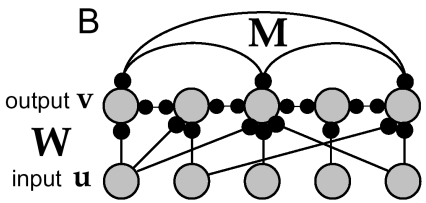
$$z_{inferred} = \frac{\sum_j R_j^m c_j}{\sum_j R_j^m}$$

Here we compare true position with position inferred during the training phase (activity given by tuning curves) and during the operating phase (activity given by feedforward input):





3 Recurrent networks



3.1 Linear networks and selective amplification

While recurrent networks offer a far wider range of behaviours than feedforward models, they are also significantly more difficult to analyse. To develop some insight and intuition into recurrent dynamics, we first consider a model with a linear activation function $F(I_s)$. Note that this has some highly non-biological consequences (e.g., firing rates can become negative). The dynamic equation of a linear recurrent model with $\tau_r >> \tau_s$ is

$$\tau_r \frac{d\boldsymbol{v}}{dt} = -\boldsymbol{v} + \boldsymbol{h} + \boldsymbol{M} \cdot \boldsymbol{v}$$

where τ_r is the time-constant with which firing follows soma current, $\boldsymbol{v}$ is the output activity vector, $\boldsymbol{h}$ is the feedforward input, and $\boldsymbol{M}$ is the recurrent connectivity matrix. If $\boldsymbol{M}$ is symmetric, the eigenvectors form an orthogonal basis and we can solve for the activity in terms of linear combinations of eigenvectors:

$$\boldsymbol{M} \cdot \boldsymbol{e}_\mu = \lambda_\mu \boldsymbol{e}_\mu \qquad \boldsymbol{e}_\mu \cdot \boldsymbol{e}_\nu = \delta_{\mu\nu}$$

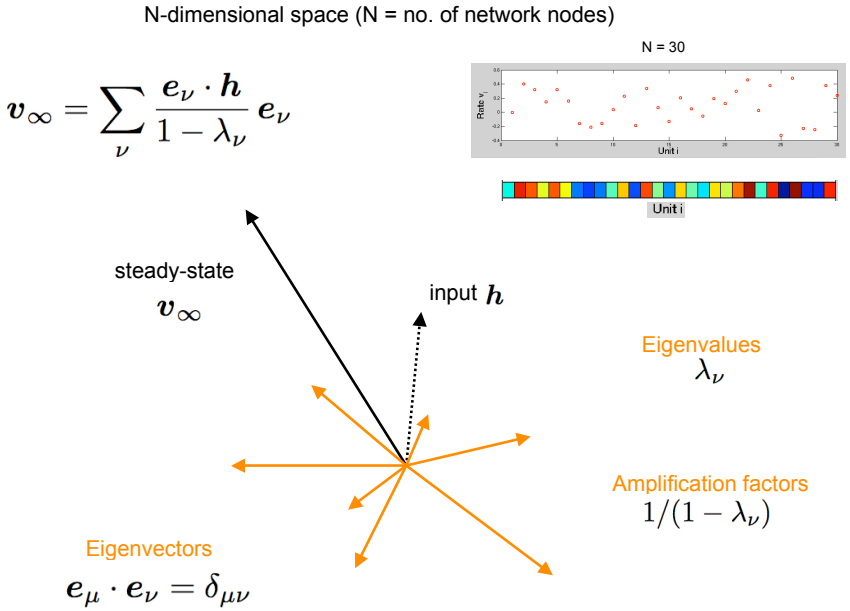
$$\boldsymbol{v}(t) = \sum_\nu c_\nu(t)\, \boldsymbol{e}_\nu \qquad \boldsymbol{v}_\infty = \sum_\nu \frac{\boldsymbol{e}_\nu \cdot \boldsymbol{h}}{1-\lambda_\nu} \boldsymbol{e}_\nu$$

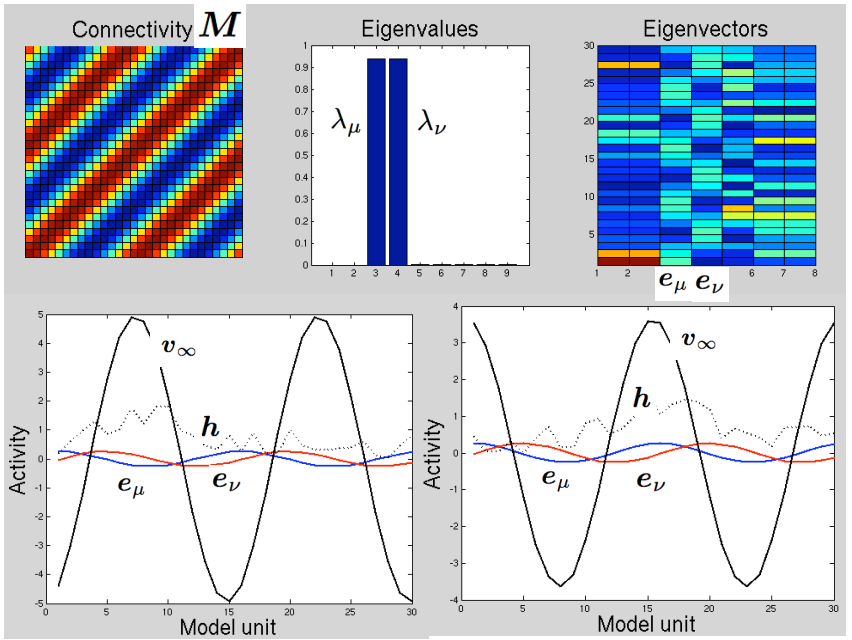
For $\lambda_\nu > 1$, the exponential functions grow without bound and the network is unstable. For $\lambda_\nu < 1$, the network approaches its steady-state value $\boldsymbol{v}_\infty$ with time constants $\tau_r/(1-\lambda_n u)$. In this steady-state, the projection of input vector $\boldsymbol{h}$ onto eigenvector $\boldsymbol{e}_\nu$ is amplified by a factor $1/(1-\lambda_\nu)$.

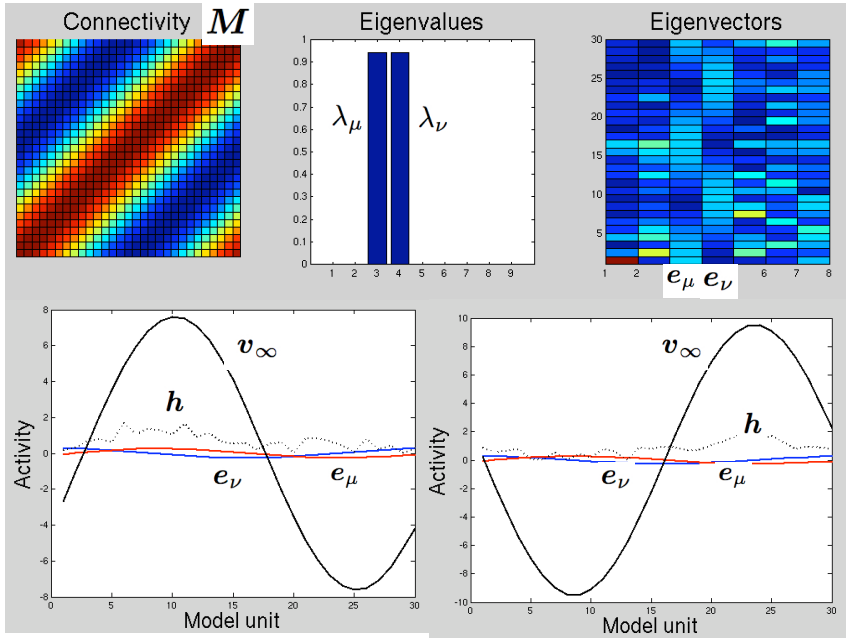
Suppose that two eigenvectors $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ have identical eigenvalues $\lambda_1 = \lambda_2 < 1$ and that the other eigenvalues are much smaller than unity. In this case, the response is dominated by the projection of the input vector onto the plane defined by $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. The amplitude of the response is multiplied by $1/(1-\lambda_1)$:

$$\boldsymbol{v}_\infty \approx \frac{(\boldsymbol{e}_1 \cdot \boldsymbol{h})\,\boldsymbol{e}_1 + (\boldsymbol{e}_2 \cdot \boldsymbol{h})\,\boldsymbol{e}_2}{1-\lambda_1}$$

Accordingly, the network selectively amplifies input fluctuations that happen to coincide with the two degenerate eigenvectors. A recurrent network with n degenerate eigenvalues can amplify the projection of the input into the n-dimensional subspace spanned by the degenerate eigenvectors.







3.2 Non-linear networks and selective amplification

A linear network is of limited value in describing neural networks, because it allows negative firing rates. To fix this problem, we need to introduce a non-linear activation function, for example a threshold rectification:

$$\tau_r \frac{d\boldsymbol{v}}{dt} = -\boldsymbol{v} + \boldsymbol{F}(\boldsymbol{h} + \boldsymbol{M}\cdot\boldsymbol{v}) \qquad \boldsymbol{F}(\boldsymbol{h}+\boldsymbol{M}\cdot\boldsymbol{v}) = [\boldsymbol{h}+\boldsymbol{M}\cdot\boldsymbol{v}-\boldsymbol{\gamma}]_+$$

This modification retains some of the features of linear models, but also introduces some new features. In particular, consider a continuous model with recurrent couplings, which converges to a steady-staty for any constant input (provided that λ_1 is not too large):

$$\tau_r \frac{dv(\theta)}{dt} = -v(\theta) + \left[h(\theta) + \int_{-\pi}^{\pi} M(\theta,\theta')\, v(\theta')\, d\theta'\right]_+ \qquad M(\theta-\theta') = \frac{\lambda_1}{\pi\,\rho_\theta}\cos(\theta-\theta')$$

$$\rho_\theta \int_{-\pi}^{\pi} M(\theta-\theta')\, e_\mu(\theta')\, d\theta' = \lambda_\mu\, e_\mu(\theta) \qquad \lambda_\mu = \rho_\theta \int_{-\pi}^{\pi} M(\theta-\theta')\cos(\mu\theta')\, d\theta'$$

$$\lambda_1 = 1 \qquad \lambda_{\mu>1} = 0$$

This network selectively amplifies a noisy input. The steady-state activity profile resembles the positive part of a sinus function (rectification!). Due to the suppression of negative input, the network remains stable for larger values of λ_1. Compared to the linear network, more Fourier components are amplified. However, the amplification does remain selective.

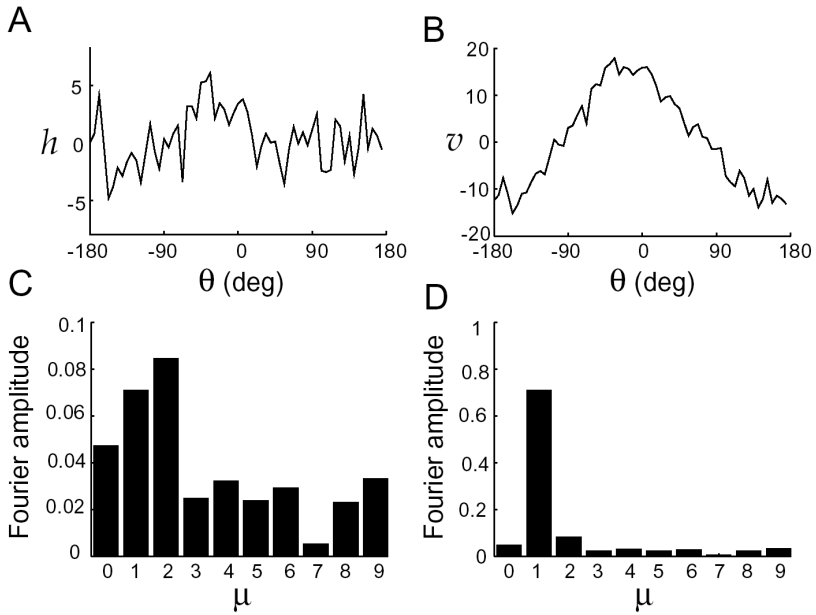
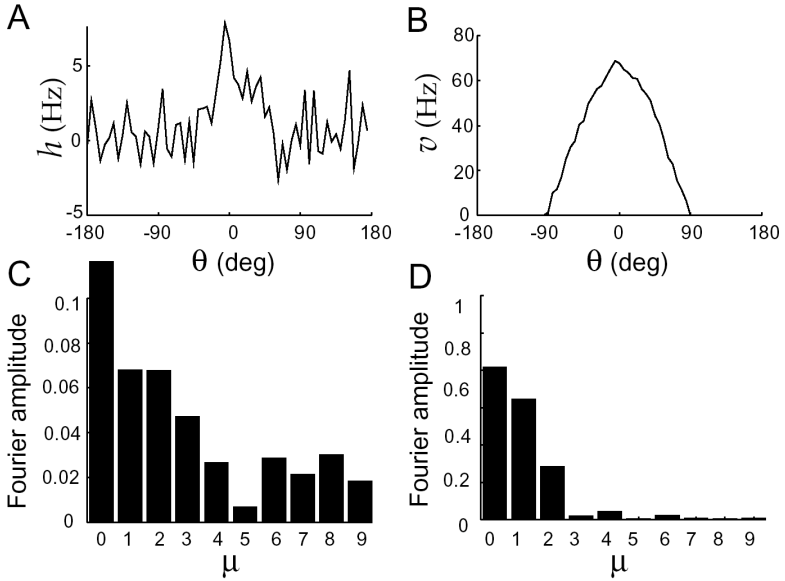
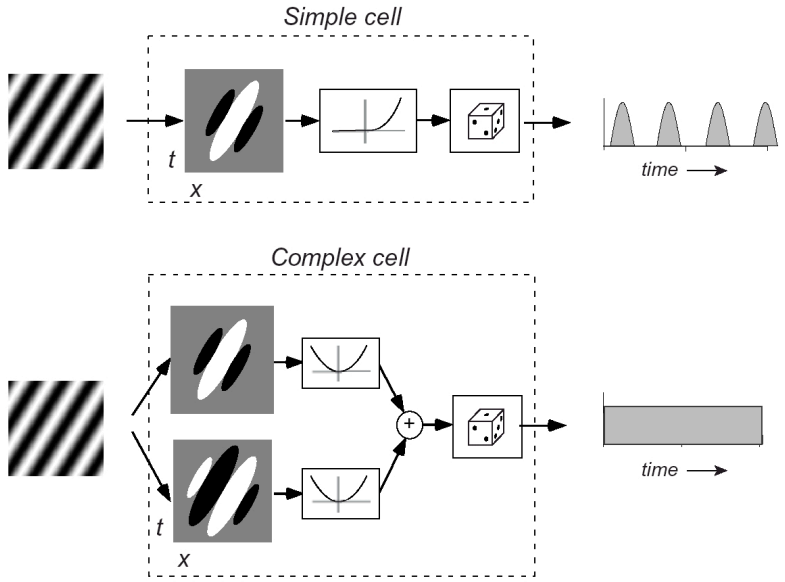


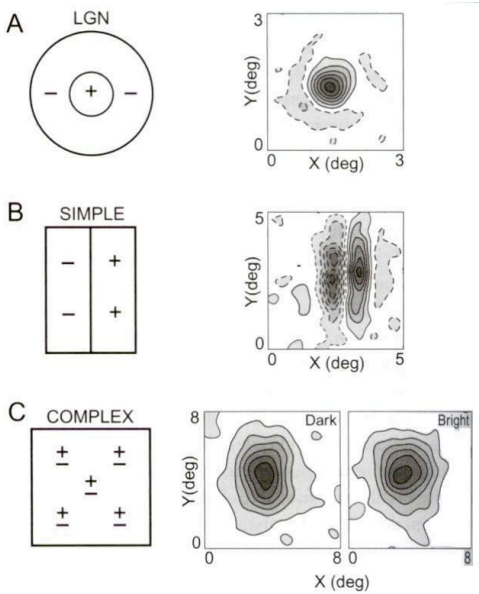
Figure 7.8: Selective amplification in a linear network. A) The input to the neurons of the network as a function of their preferred stimulus angle. B) The activity of the network neurons plotted as a function of their preferred stimulus angle in response to the input of panel A. C) The Fourier transform amplitudes of the input shown in panel A. D) The Fourier transform amplitudes of the output shown in panel B. The recurrent coupling of this network model took the form of equation 7.34 with $\lambda_1 = 0.9$. (This figure, and figures 7.9, 7.12, 7.13, and 7.14, were generated using software from Carandini and Ringach, 1998.)

Figure 7.9: Selective amplification in a recurrent network with rectification. A) The input $h(\theta)$ of the network plotted as a function of preferred angle. B) The steady-state output $v(\theta)$ as a function of preferred angle. C) Fourier transform amplitudes of the input $h(\theta)$. D) Fourier transform amplitudes of the output $v(\theta)$. The recurrent coupling took the form 7.34 with $\lambda_1 = 1.9$.





3.3 Simple and complex cells



3.3.1 Recurrent model of simple cells

Cortical neurons receive far more recurrent inputs (i.e., from other neurons in the same area) than feed-forward inputs (e.g., from thalamic relay neurons). This suggests that recurrent interactions may play an important role in shaping response properties. To model the possible effect of such interactions, we add a global inhibition to our favorite connectivity matrix and assume that the input is only weakly orientation-dependent:

$$\tau_r \frac{dv(\theta)}{dt} = -v(\theta) + \left[h(\theta) + \int_{-\pi/2}^{\pi/2} \left[-\lambda_0 + \lambda_1 \cos(2(\theta - \theta')) \right] v(\theta')\, d\theta' \right]_+$$

$$h(\theta) = A\,[1 - \epsilon + \epsilon \cos(2\theta)]$$

where A is the input strength and ϵ parameterizes the orientation dependence. For $\epsilon = 0$, all units receive identical input, regardless of orientation. While the steady-state response $v(\theta)$ represents an activity distribution across the network, it can also be read as the 'tuning function' of a particular network unit. In this model, recurrent interactions amplify tuning curves but do not broaden them, consistent with experimental observations.

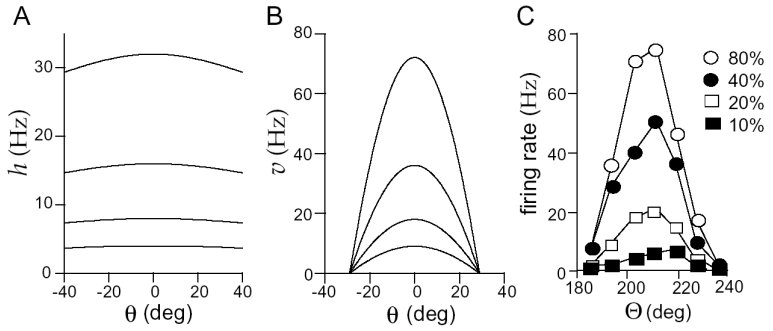


Figure 7.10: The effect of contrast on orientation tuning. A) The feedforward input as a function of preferred orientation. The four curves, from top to bottom, correspond to contrasts of 80%, 40%, 20%, and 10%. B) The output firing rates in response to different levels of contrast as a function of orientation preference. These are also the response tuning curves of a single neuron with preferred orientation zero. As in A, the four curves, from top to bottom, correspond to contrasts of 80%, 40%, 20%, and 10%. The recurrent model had $\lambda_0 = 7.3$, $\lambda_1 = 11$, $A = 40$ Hz, and $\epsilon = 0.1$. C) Tuning curves measure experimentally at four contrast levels as indicated in the legend. (C adapted from Sompolinsky and Shapley, 1997; based on data from Sclar and Freeman, 1982.)

3.3.2 Recurrent model of complex cells

Recall that complex cells (in contrast to simple cells) do not have distinct ON and OFF regions in their receptive fields and are therefore insensitive to stimulus phase. This model to be discussed shows that recurrent connections between simple cells can almost eliminate their dependence on phase (if the coupling is sufficiently strong). We consider a population of neurons tuned to different stimulus phases ϕ, with ubiquitous connections of constant strength $M(\phi, \phi') = \lambda_1/(2\pi\rho_\phi)$:

$$\tau_r \frac{dv(\phi)}{dt} = -v(\phi) + \left[h(\phi) + \frac{\lambda_1}{2\pi} \int_{-\pi}^{\pi} v(\phi')\, d\phi' \right]_+ \qquad h(\phi) = A\,[\cos\phi]_+$$

For $\lambda_1 = 0$, the output exhibits the same phase-dependence as the input, $v(\phi) = h(\phi)$. As the value of λ_1 grows close to one, the output looses its phase-dependence almost completely. In other words, the coupling turns simple cells into complex cells. The mathematical reason is that the eigenfunction associated with λ_1 is flat, that is, independent of spatial phase. As a result, the network amplifies input activity independent of spatial phase.

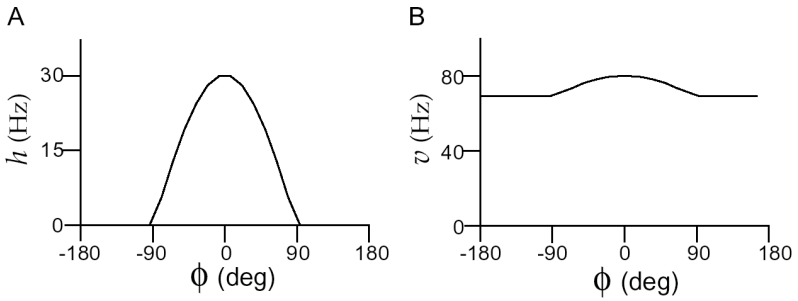
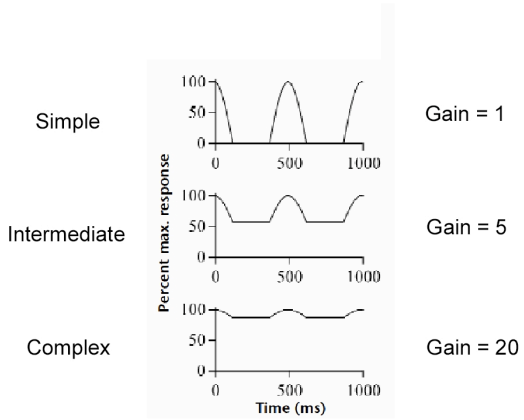
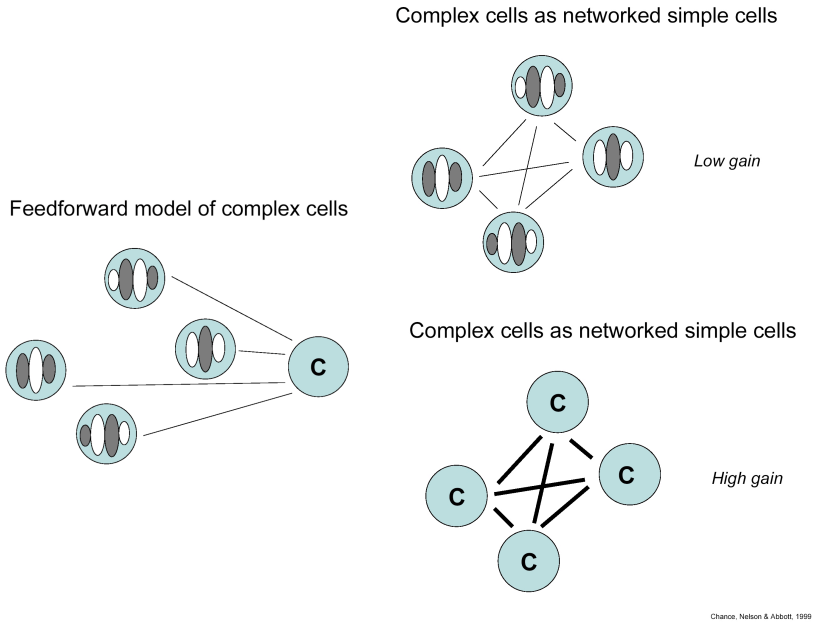


Figure 7.11: A recurrent model of complex cells. A) The input to the network as a function of spatial phase preference. The input $h(\phi)$ is equivalent to that of a simple cell with spatial phase preference ϕ responding to a grating of zero spatial phase. B) Network response, which can also be interpreted as the spatial phase tuning curve of a network neuron. The network was given by equation 7.39 with $\lambda_1 = 0.95$. (Adapted from Chance *et al*, 1999.)

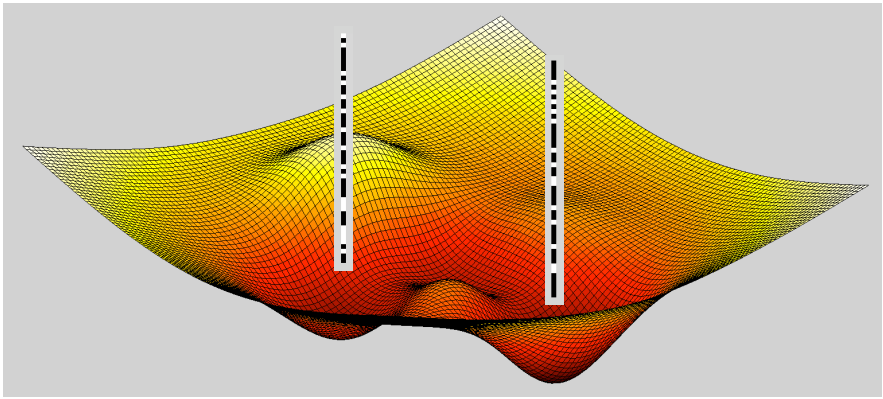
Complex cells as networked simple cells



Chance, Nelson & Abbott, 1999



From yet another point of view, the network can be considered to traverse an "energy surface", where each stored memory pattern has a "basin of attraction", which is defined as the set of initial states for which the memory relaxes. The networks with which we are concerned always relax to a fixed point.



1 Associative memory

The principle of "associative" or "content-addressable" memory is that a partial or corrupted exemplar is used to recall the full item. This type of memory device is thought to provide a model for mammalian memory systems that are characterised by recurrent connections, such as area CA3 of hippocampus or prefrontal cortex.

We already encountered networks with one stereotypical activity profile. We now consider networks with a broader set of characteristic activity profiles, which we call memory patterns. The recall of a particular memory item is thus modelled as a particular activity profile. Note that memory patterns are stored in the synaptic weights, not in persistent activity. Persistent activity serves to signal the most recently recalled memory item.

From another point of view, the network performs a kind of pattern matching, that is, it finds the memory pattern that most closely matches an initial pattern. Specifically, the network is initialised to an input pattern and allowed to relax to a fixed point. The fixed point activity is treated as the desired output, namely, the best match among the memory pattern. Recall the maximum likelihood decoding (curve fitting) example from earlier lecture, in which an approximate or distorted input is turned into a stereoptyped output.

5 Example network

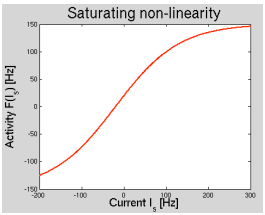
We consider the following example network with a sigmoidal non-linearity in the activity function $F(I_s)$ and time-constant τ_r, maximal activity $r_{max} = 150Hz$, and background activity (negative threshold) $\gamma = -20Hz$:

$$\tau_r \frac{d\boldsymbol{v}}{dt} = -\boldsymbol{v} + \boldsymbol{F}(\boldsymbol{M} \cdot \boldsymbol{v}) \qquad F(I_s) = r_{max}\left[\tanh\left(\frac{I_s - \gamma}{r_{max}}\right)\right]_+$$

With this network has N_v units we will store N_{mem} activity patterns v^m, where $N_{mem} << N_v$. For simplicity, we consider patterns containing αN_v components of value c and $(1-\alpha)N_v$ components of value 0. Parameter α represents the "sparseness" of the memory patterns. Following the example of the book, we use $N_v = 50$, $N_{mem} = 4$, and $\alpha = 0.25$.

When the network starts in an initial state approximately proportional to one of the memory patterns, we would like it to evolve towards a fixed point of the corresponding memory pattern.

$$\boldsymbol{v}(t) \to \boldsymbol{v}^m \qquad \boldsymbol{v}^m = \boldsymbol{F}(\boldsymbol{M} \cdot \boldsymbol{v}^m)$$



To find a suitable connection matrix $\boldsymbol{M}$, we first consider a matrix K that has the to-be-stored patterns $\boldsymbol{v}_m$ as eigenvectors:

$$\boldsymbol{K} \cdot \boldsymbol{v}^m = \lambda \boldsymbol{v}^m$$

$$\boldsymbol{M} = \boldsymbol{K} - \frac{\boldsymbol{n}\,\boldsymbol{n}}{\alpha\, N_v} \qquad \text{or} \qquad M_{aa'} = K_{aa'} - \frac{1}{\alpha\, N_v}$$

The vector $\boldsymbol{n}$ is defined with N_v components equal to one. Note that we are using the "vector outer product" $\boldsymbol{x}\,\boldsymbol{y}$ which produces a matrix.

$$\boldsymbol{n} \cdot \boldsymbol{v}^m = c\,\alpha\, N_v \qquad \boldsymbol{v}^n \cdot \boldsymbol{v}^m \approx c^2\,\alpha^2\, N_v \quad n \neq m \qquad \boldsymbol{v}^m \cdot \boldsymbol{v}^m = c^2\,\alpha\, N_v$$

The fixed point condition can be rewritten as

$$\lambda\, \boldsymbol{v}^m = \boldsymbol{F}\,(\lambda\, \boldsymbol{v}^m - c\,\boldsymbol{n}) \qquad \text{since} \qquad \boldsymbol{M} \cdot \boldsymbol{v}^m = \lambda\, \boldsymbol{v}^m - c\,\boldsymbol{n}$$

Taken component by component, this yields two conditions, one for components c and another for components 0:

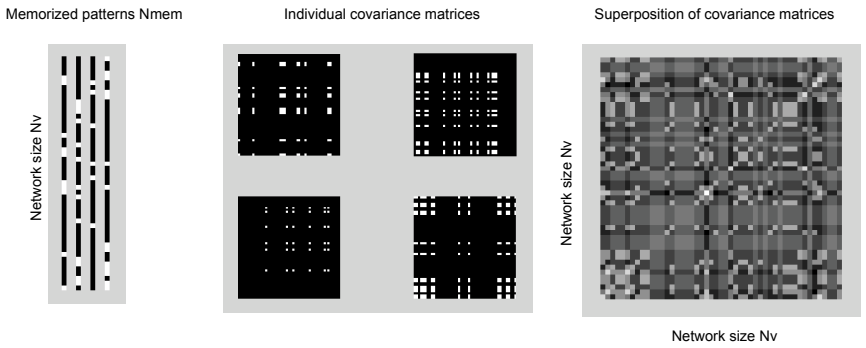
$$0 = F[-c] \qquad \text{if} \qquad v^m = 0$$

$$c = F[c\,(\lambda - 1)] \qquad \text{if} \qquad v^m = 0$$

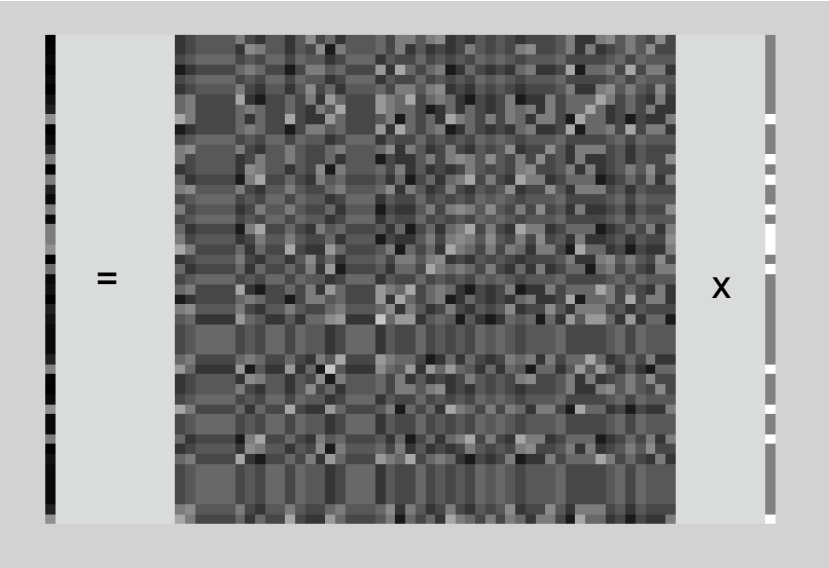
It's not hard to find non-linearities $F()$ which satisfy these conditions. The problem thus reduces to finding a suitable matrix K.

A simple way of constructing a symmetric connection matrix $\boldsymbol{M}$ with the desired properties is to sum the covariance matrices of the stored memory patterns plus a general inhibitory term:

$$\boldsymbol{M} = \frac{1.25}{(1-\alpha)\,\alpha\, N_v} \sum_{m=1}^{N_{mem}} (\boldsymbol{v}^m - \alpha\,\boldsymbol{n})\;(\boldsymbol{v}^m - \alpha\,\boldsymbol{n}) - \frac{1}{\alpha\, N_v}\,\boldsymbol{n}\,\boldsymbol{n}$$



Memory patterns are fixed points



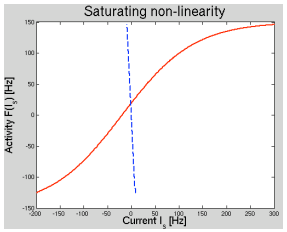
A simple way of constructing a symmetric connection matrix $\boldsymbol{M}$ with the desired properties is to sum the covariance matrices of the stored memory patterns plus a general inhibitory term:

$$\boldsymbol{M} = \frac{1.25}{(1-\alpha)\,\alpha\, N_v} \sum_{m=1}^{N_{mem}} (\boldsymbol{v}^m - \alpha\,\boldsymbol{n})\;(\boldsymbol{v}^m - \alpha\,\boldsymbol{n}) - \frac{1}{\alpha\, N_v}\,\boldsymbol{n}\,\boldsymbol{n}$$

In this case, the conditions to be satisfied are

$$F\,[-c(1+\alpha\,\lambda)] = 0 \qquad c = F\,[c(\lambda - 1 - \alpha\,\lambda)]$$

If furthermore the neglected residual terms are small, the network will perform as an associative memory. Otherwise, if $N_{mem} \approx N_v$, the residual becomes large enough to destabilize the memory states as fixed points. Note that large networks can store large numbers of patterns only if these patterns are sparse. The covariance prescription used here is far from optimal, more efficient methods can be devised.



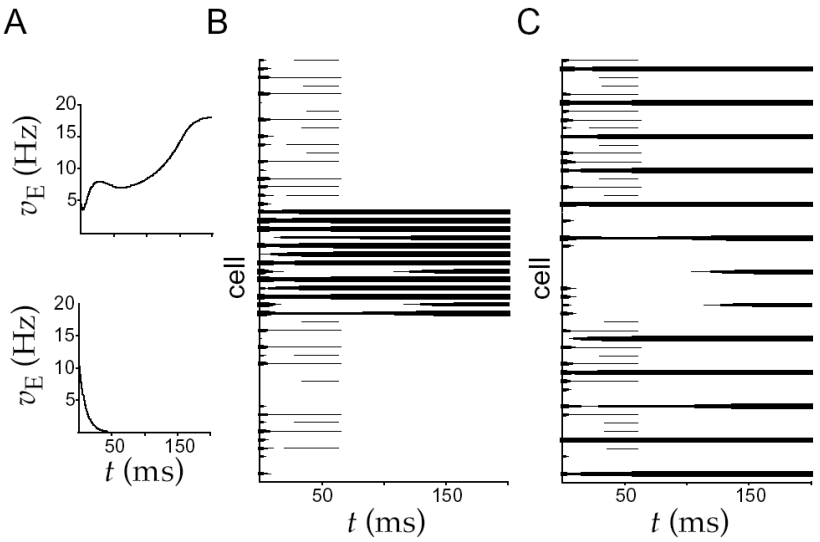


Figure 7.16: Associative recall of memory patterns in a network model. Panel A shows two representative model neurons, while panels B and C show the firing rates of all 50 cells plotted against time. The thickness of the horizontal lines in these plots is proportional to the firing rate of the corresponding neuron. A) Firing rates of representative neurons. The upper panel shows the firing rate of one of the excitatory neurons corresponding to a nonzero component of the recalled memory pattern. The firing rate achieves a nonzero steady-state value. The lower panel shows the firing rate of another excitatory neuron corresponding to a zero component of the recalled memory pattern. This firing rate goes to zero. B) Recall of one of the stored memory patterns. The stored pattern had nonzero values only for cells 18 through 31. The initial state of the network was random but with a bias toward this particular pattern. The final state is similar to the memory pattern. C) Recall of another of the stored memory patterns. The stored pattern had nonzero values only for every fourth cell. The initial state of the network was again random but biased toward this pattern. The final state is similar to the memory pattern.