Attractor Dynamics and Recurrent Networks

October 9, 2026

Roadmap. We first reduce spiking synaptic input to a firing rate description, then analyse the linear recurrent network and its eigenmodes (selective amplification, integration, line attractors). We then turn to discrete point attractors in the Hopfield network, generalise the energy function to continuous-valued rate networks, and finally construct a continuous attractor in the ring model.

A question to carry through both lectures. Everything below is governed by one object, the recurrent weight matrix $\mathbf{J}$, and the question is always what its spectrum does. Here we meet two structured choices: a low-rank, outer-product $\mathbf{J}$ built from stored patterns (Hopfield), and a translation-invariant $\mathbf{J}$ whose eigenvalues are simply its Fourier coefficients J_0 and J_1 (the ring). In the next lecture we take $\mathbf{J}$ with independent random entries, and the same question is answered by the circular law and a transition to chaos.

Modeling synaptic inputs

The synaptic current I_s could be described by the following equation

$$I_s = \bar{g}_s P(V - E_s), \tag{1}$$

where P is the neurotransmitter release probability. If we consider the fact that the rising time has certain delay τ_r, one could use the following equation

$$P = \frac{P_{max} t}{\tau_r} \exp(-t/\tau_s).$$

However, when $\tau_r \ll \tau_s$, we may assume that the release probability approaches some constant value P_{max} immediately after the action potential arrives at time t_k, and then decays with a time constant τ_s: $P = P_{max} e^{-(t-t_k)/\tau_s}$, for $t > t_k$.

The current that *depolarizes* the cell is $-I_s = \bar{g}_s P(E_s - V)$. If the membrane potential stays far from the reversal potential, the driving force $E_s - V$ changes little during a synaptic event and can be treated as constant. We may then define the synaptic weight

$$J = \bar{g}_s P_{max}(E_s - V),$$

which is positive for an excitatory synapse ($E_s > V$) and negative for an inhibitory one. The total synaptic drive is obtained by summing over all spikes in the past,

$$I_s = J \sum_k K(t - t_k),$$

where $K(t) = \Theta(t) \exp(-t/\tau_s)$ and Θ is the Heaviside step function. In the form of a differential equation, this can be rewritten as

$$\tau_s \frac{dI_s}{dt} = -I_s + \tau_s J \sum_k \delta(t - t_k) \tag{2}$$

When we are summing synaptic inputs coming from many neurons, the total synaptic currents become

$$\tau_s \frac{dI_s^i}{dt} = -I_s^i + \tau_s \sum_j J_{ij} \sum_k \delta(t - t_{j,k}) \tag{3}$$

Note that the synaptic conductance $\bar{g}_s$ carries a subscript throughout; in the next lecture g without a subscript will denote the coupling gain of a random network, an unrelated quantity.

Firing rate model with dynamics

In previous lectures, we have discussed the integrate-and-fire model, and how a deterministic spiking pattern would arise given a constant external input. However, each neuron receives time-varying inputs from thousands of other neurons, and experimental data have shown that the spike timing for a cortical network is stochastic, making it difficult to treat Equation 2. Indeed, spiking models impose a great computational and practical challenge. To ameliorate these difficulties, we shall substitute the spiking variable $\rho(t) = \sum_k \delta(t - t_k)$ with the firing rate $r(t)$.

Now in a firing rate model, Equation 2 is changed to

$$\tau_s \frac{dI_s}{dt} = -I_s + \tau_s J r(t)$$

Summing up the synaptic inputs from all the neurons, we shall have

$$\tau_s \frac{dI_s^i}{dt} = -I_s^i + \tau_s \sum_{j=1}^{N} J_{ij} r_j(t), \tag{4}$$

where j denotes presynaptic neuron j.

Equation 4 shows how the synaptic current of a neuron depends on the firing rate of all presynaptic neurons. To finish formulating the firing rate model, we should also specify how the firing rate of the postsynaptic neuron r_i depends on the total synaptic input I_s^i. Recall that the membrane potential of a neuron

could be modeled as a low-pass filter of the applied external input. Motivated by this observation, the firing rate of the postsynaptic neuron may be written as

$$\tau_r \frac{dr_i}{dt} = -r_i + \phi(I_s^i(t)) \tag{5}$$

I must emphasize that this is a phenomenological characterization. It describes how rapidly the postsynaptic firing rate follows the time-varying synaptic current. $\phi(\cdot)$ is a nonlinear function of the synaptic current, say $\phi(I_s) = [I_s - \gamma]_+$. Note that τ_r is not the same as the membrane time constant τ_m, for the latter is used to describe the membrane potential.

Equations 4 and 5 provide a complete description of the firing rate model. In the limit $\tau_s \ll \tau_r$, the synaptic current relaxes to its steady state $I_s^i = \tau_s \sum_{j=1}^N J_{ij} r_j(t)$ much faster, and Equation 5 becomes

$$\tau_r \frac{dr_i}{dt} = -r_i + \phi\left(\sum_{j=1}^N J_{ij} r_j(t)\right) \tag{6}$$

Note that in practice τ_s has been absorbed into the synaptic weight J_{ij}.

In the opposite limit $\tau_r \ll \tau_s$, we shall write

$$\begin{aligned} \tau_s \frac{dI_s^i}{dt} &= -I_s^i + \tau_s \sum_{j=1}^N J_{ij} r_j(t), \\ r_i &= \phi(I_s^i(t)) \end{aligned} \tag{7}$$

Feedforward and Recurrent Networks

There are two basic network topologies, the feedforward network and the recurrent network. The feedforward network has N_v output units with firing rates v_i $(i = 1, 2, \ldots, N_v)$, denoted collectively by the vector $\mathbf{v}$, driven by N_u input units with rates $\mathbf{u}$. The feedforward synaptic weights form a matrix $\mathbf{W}$, whose component W_{ij} represents the strength of the synapse from input unit j to output unit i.

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \phi(\mathbf{W} \cdot \mathbf{u}) \tag{8}$$

The recurrent network also has two layers of neurons with rates $\mathbf{u}$ and $\mathbf{v}$. But now the output layer is also interconnected, with a recurrent weight matrix $\mathbf{J}$. In this case, the output firing rates are determined by

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \phi(\mathbf{W} \cdot \mathbf{u} + \mathbf{J} \cdot \mathbf{v}) \tag{9}$$

From here on, $\mathbf{W}$ always denotes feedforward weights and $\mathbf{J}$ always denotes recurrent weights.

It is often convenient to define the total feedforward input to each neuron as a vector $\mathbf{h} = \mathbf{W} \cdot \mathbf{u}$. Then, the output rates are

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \phi(\mathbf{h} + \mathbf{J} \cdot \mathbf{v}) \tag{10}$$

Neurons are typically classified as either excitatory or inhibitory, meaning that they have either excitatory or inhibitory effects on all of their postsynaptic targets. This property is formulated as Dale's law, which states that a neuron cannot excite some of its postsynaptic cells and inhibit others. This rule imposes a strong constraint on the weight matrix: the sign of J_{ij} is fixed by the *presynaptic* index j, so that every entry in a given column of $\mathbf{J}$ has the same sign.

Remark on two conventions

Equation 10 is *rate-based*: the rate is the dynamical variable and the nonlinearity acts on the summed input. The alternative is not a new postulate. Substituting $r_i = \phi(I_s^i)$ into Equation 7 eliminates the rate and moves the nonlinearity inside the sum; renaming the current $\mathbf{x}$, absorbing τ_s into the weights and restoring the feedforward input,

$$\tau_r \frac{d\mathbf{x}}{dt} = -\mathbf{x} + \mathbf{J}\phi(\mathbf{x}) + \mathbf{h}. \tag{11}$$

The two conventions are thus the opposite orderings of τ_s and τ_r, each with its fast variable eliminated.

They are more closely related than that suggests. Put $\mathbf{x} := \mathbf{J}\mathbf{v} + \mathbf{h}$ and left-multiply Equation 10 by $\mathbf{J}$; using $\mathbf{J}\, d\mathbf{v}/dt = d\mathbf{x}/dt$ and $\mathbf{J}\mathbf{v} = \mathbf{x} - \mathbf{h}$, the result is exactly Equation 11. The two are the same flow in two coordinate systems.

This rests on two assumptions. (i) $\mathbf{h}$ is constant: with $\mathbf{h}(t)$ an extra term $\tau_r\, d\mathbf{h}/dt$ appears, so the conventions filter a time-varying stimulus differently even though their steady states agree. (ii) $\mathbf{J}$ is invertible, otherwise the map is a projection rather than a change of variables — not an exotic case, since the Hopfield matrix below has rank $P \ll N$. Note finally that $\mathbf{v} = \mathbf{J}^{-1}(\mathbf{x} - \mathbf{h})$ equals $\phi(\mathbf{x})$ only at a fixed point, so what is called the firing rate differs between the two conventions during a transient.

Equation 11 is the form used in the next lecture, since it is the one for which dynamic mean-field theory is cleanest: $\sum_j J_{ij}\phi(x_j)$ is a sum of many weakly correlated terms, which for random $\mathbf{J}$ can be replaced by a Gaussian field.

The linear recurrent network and its energy function

Now let's consider some special cases of Equation 10. For example, take away ϕ and consider a linear relationship between firing rate and synaptic current. This is a drastic assumption, as it would allow v to be negative, which is impossible.

Nevertheless, the linear model is extremely useful for exploring properties of recurrent circuits, as we will see below. In this case, the equation becomes

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \mathbf{h} + \mathbf{J} \cdot \mathbf{v} \tag{12}$$

This is the linear recurrent neural network.

Next, let us ask when these dynamics can be written as gradient descent on a scalar function,

$$\tau_r \frac{d\mathbf{v}}{dt} = -\nabla E \tag{13}$$

It is clear that

$$\frac{dE}{dt} = \sum_{i=1}^{N} \frac{\partial E}{\partial v_i} \frac{dv_i}{dt} = \nabla E^T \frac{d\mathbf{v}}{dt} \tag{14}$$

Thus

$$\frac{dE}{dt} = -\frac{1}{\tau_r} \|\nabla E\|^2 \leq 0$$

so E can only decrease along a trajectory. If we can find an energy function satisfying

$$(\mathbf{I} - \mathbf{J})\mathbf{v} - \mathbf{h} = \nabla E,$$

we must have

$$(\mathbf{I} - \mathbf{J})_{ij} = \frac{\partial^2 E}{\partial v_i \partial v_j}$$

Since second partial derivatives commute, this requires the recurrent weight matrix $\mathbf{J}$ to be symmetric, $\mathbf{J} = \mathbf{J}^T$. Indeed, the energy function we can come up with is

$$E = \frac{1}{2}\mathbf{v}^T\mathbf{v} - \frac{1}{2}\mathbf{v}^T\mathbf{J}\mathbf{v} - \mathbf{h}^T\mathbf{v} \tag{15}$$

Note that symmetry is exactly what Dale's law forbids in a network with both excitatory and inhibitory cells; the energy function is therefore a mathematical convenience rather than a biological statement. It is also the property that fails completely for the random $\mathbf{J}$ of the next lecture, which is why chaos becomes possible there.

Eigenmode analysis and selective amplification

Now let's consider the case where $\mathbf{J}$ is symmetric. A real symmetric matrix can be diagonalised by an orthonormal set of eigenvectors $\mathbf{e}_\mu$ for $\mu = 1, 2, \ldots, N_v$,

$$\mathbf{J}\mathbf{e}_\mu = \lambda_\mu \mathbf{e}_\mu, \qquad \mathbf{e}_\mu \cdot \mathbf{e}_\nu = \delta_{\mu\nu}.$$

In particular, we can use this basis to represent

$$\mathbf{v}(t) = \sum_{\mu=1}^{N_v} c_\mu(t)\mathbf{e}_\mu$$

Plugging this back into Equation 12, we have

$$\tau_r \sum_{\mu=1}^{N_v} \frac{dc_\mu(t)}{dt}\mathbf{e}_\mu = -\sum_{\mu=1}^{N_v}(1-\lambda_\mu)c_\mu(t)\mathbf{e}_\mu + \mathbf{h}.$$

The sum over μ can be eliminated by taking the dot product with $\mathbf{e}_\mu$ on both sides, and by using $\mathbf{e}_\mu \cdot \mathbf{e}_\nu = \delta_{\mu\nu}$ we have

$$\tau_r \frac{dc_\mu(t)}{dt} = -(1-\lambda_\mu)c_\mu(t) + \mathbf{e}_\mu \cdot \mathbf{h}.$$

The solution to this equation is given by

$$c_\mu(t) = \frac{\mathbf{e}_\mu \cdot \mathbf{h}}{1-\lambda_\mu}\left[1 - \exp(-t(1-\lambda_\mu)/\tau_r)\right] + c_\mu(0)\exp(-t(1-\lambda_\mu)/\tau_r)$$

Each mode relaxes with its own effective time constant $\tau_r/(1-\lambda_\mu)$. Provided $\lambda_\mu < 1$ for every μ, the network is stable and the steady state firing rate is given by

$$\mathbf{v}_\infty = \sum_{\mu=1}^{N_v} \frac{\mathbf{e}_\mu \cdot \mathbf{h}}{1-\lambda_\mu}\mathbf{e}_\mu$$

Modes with λ_μ close to 1 are both strongly amplified and slow: this is selective amplification, and it is the mechanism behind the ring model at the end of the lecture. Modes with $\lambda_\mu > 1$ grow without bound, and a nonlinearity is needed to stabilise them. The condition $\max_\mu \lambda_\mu < 1$ is the first appearance of a criterion we will meet repeatedly; for a non-symmetric $\mathbf{J}$ it becomes a statement about the rightmost point of the eigenvalue spectrum in the complex plane.

Input integration and short term memory

If the recurrent weight matrix has an eigenvalue exactly equal to 1, $\lambda_1 = 1$, and all the other eigenvalues satisfy $\lambda_\mu < 1$, a linear recurrent network can act as an integrator of its input. In this case, c_1 satisfies the following equation

$$\tau_r \frac{dc_1}{dt} = \mathbf{e}_1 \cdot \mathbf{h}.$$

For arbitrary time-dependent inputs, the solution is given by

$$c_1(t) = c_1(0) + \frac{1}{\tau_r}\int_0^t dt'\, \mathbf{e}_1 \cdot \mathbf{h}(t')$$

Now let's imagine that $\mathbf{h}$ represents a sensory stimulus presented to the network. If this input is transient, that is $\mathbf{h}(t) = \mathbf{s}_0\delta(t-t_0)$, then the stimulus representation can be stored persistently within the network,

$$c_1(t) = c_1(0) + \frac{\mathbf{e}_1 \cdot \mathbf{s}_0}{\tau_r}, \qquad t > t_0.$$

Because c_1 can take any value, the set of steady states forms a continuous one-dimensional manifold of fixed points: a *line attractor*. This is the idealisation of the oculomotor integrator, and it also illustrates the fragility of such models, since $\lambda_1 = 1$ must hold exactly.

In general, during a working memory task, we would like to hold the percept of a past memory in the neural circuit for an indefinite period of time. Conceptually, this means that the representation of the sensory stimulus should be time invariant during that period. Mathematically, this means that

$$\frac{d\mathbf{s}}{dt} = 0 \tag{16}$$

In the linear coding framework, the representation of a sensory stimulus $\mathbf{s}$ is a linear combination of each neuron's receptive field, which can be viewed as a feature vector $\mathbf{d}_i$, weighted by the activity of that neuron

$$\mathbf{s} = \sum_i r_i(t)\mathbf{d}_i \tag{17}$$

Writing everything in matrix form, where $\mathbf{D} = \{\mathbf{d}_1, \mathbf{d}_2 \ldots\}$ and $\mathbf{r} = [r_1, r_2 \ldots]^T$, we have

$$\mathbf{s} = \mathbf{D}\mathbf{r}$$

Equation 16 can now be written as

$$\sum_i \mathbf{d}_i \frac{dr_i(t)}{dt} = 0 \tag{18}$$

If each neuron encodes a different feature vector $\mathbf{d}_i$ and all these feature vectors are linearly independent, then we must have

$$\frac{dr_i(t)}{dt} = 0$$

for all neurons. This leads to the time-invariant persistent activity discussed above. However, if these feature vectors are *not* linearly independent, then even if individual neurons have time-varying activity patterns, the representation as a *whole* can still remain constant. We now decompose the activity vector into two terms, namely

$$\mathbf{r}(t) = \mathbf{D}^T(\mathbf{D}\mathbf{D}^T)^{-1}\mathbf{s} + \mathbf{r}_\perp(t) \tag{19}$$

where

$$\mathbf{D}\mathbf{r}_\perp(t) = 0, \tag{20}$$

so that $\mathbf{r}_\perp(t)$ belongs to the null space of $\mathbf{D}$.

The geometrical meaning of the null space is the following. The row vectors of $\mathbf{D}$ span a subspace of the N-dimensional activity space, and $\mathbf{r}_\perp$ is any vector orthogonal to that subspace. Activity may drift freely along the null space without changing what the population represents.

Now let's consider a recurrent neural network

$$\tau_r \frac{d\mathbf{r}}{dt} = -\mathbf{r} + \mathbf{J}\mathbf{r} \tag{21}$$

Plugging 21 into Equation 16, we have

$$\mathbf{D}(-\mathbf{r} + \mathbf{J}\mathbf{r}) = 0 \tag{22}$$

We impose that this should be true for all activity patterns. As a result, we find that

$$\mathbf{D} = \mathbf{D}\mathbf{J} \tag{23}$$

Again, consider the eigenvectors $\mathbf{e}_\mu$ for $\mu = 1, 2, \ldots, N_v$ satisfying

$$\mathbf{J}\mathbf{e}_\mu = \lambda_\mu \mathbf{e}_\mu$$

Equation 23 thus requires that

$$\mathbf{D}\mathbf{e}_\mu = \lambda_\mu \mathbf{D}\mathbf{e}_\mu \tag{24}$$

This equation has only two possible solutions. First, $\lambda_\mu = 1$: this is the familiar one-dimensional persistent activity. And second,

$$\mathbf{D}\mathbf{e}_\mu = 0,$$

that is, the eigenvector of the recurrent matrix belongs to the null space of the dictionary, and activity along it is invisible to the readout.

Long term memory: the Hopfield network

In a modern computer, memories (in bits) are stored at different physical addresses. However, in our daily life, recall of a memory item may be triggered by a partial or approximate version of the stored item. This is called associative memory, which is addressed by content rather than by address: content-addressable memory.

Above, we discussed short-term memory models in which information is stored by means of persistent activity: the eye position can be represented by a stereotyped population activity profile after stimulus onset (in this case, some bursting motor neuron activity from the brain stem). In that model, the dynamics is effectively one-dimensional. This idea can be extended to a broader set of population profiles, which are called memory patterns. Each of these is a fixed point of the network dynamics. The memory patterns are determined by, and stored within, the recurrent synaptic weights of the network. To remember something (memory retention) does not require persistent activity; however, to recall the memory requires persistent activity of a specific population of neurons.

During recall, an associative memory performs the computational operation of pattern matching, finding the memory pattern that most closely matches a

distorted or partial activity pattern. This is achieved by initializing the network with an activity profile close to one of the memory patterns, letting it relax to a fixed point, and treating the network activity at the fixed point as the best matching pattern. Next, we shall discuss a famous and elegant mathematical model that instantiates the above ideas: the Hopfield network.

Model definition and storage of a single pattern

In the Hopfield network, we make the even more drastic assumption that the activity of a neuron is a binary variable, $s_i \in \{+1, -1\}$. In the deterministic version of this model, the state of a cell is set according to

$$s_i = \text{sgn}\left(\sum_j J_{ij} s_j + h_i\right), \tag{25}$$

where h_i is the external input to a neuron. We consider an asynchronous update rule (choosing either a random or a fixed update order, and updating one s_i at each clock cycle). We also take $J_{ij} = J_{ji}$ and $J_{ii} = 0$; both will matter for the energy argument below.

The basic problem setup is that we want to store a number of patterns $\boldsymbol{\xi}^\mu$ ($\mu = 1, \ldots, P$) in the connection matrix of the network. Therefore, we require that $\boldsymbol{\xi}^\mu$ be a stable fixed point of the network. Let us denote a single pattern by $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_N]$. The condition for $\boldsymbol{\xi}$ to be a fixed point of the network dynamics is

$$\xi_i = \text{sgn}\left(\sum_j J_{ij} \xi_j\right) \tag{26}$$

So we need a matrix $\mathbf{J}$ satisfying the above equation. Let's choose $J_{ij} = c\,\xi_i \xi_j$ with $c > 0$; clearly the equation is then satisfied, since $\sum_j \xi_j^2 = N > 0$. Choosing the constant of proportionality as $c = 1/N$, we have $J_{ij} = \frac{1}{N}\xi_i \xi_j$. This is a Hebbian rule: the weight is positive between pairs of neurons that are co-active in the pattern. Note that $\mathbf{J}$ here is a rank-one matrix, the outer product of the pattern with itself.

Storing many patterns and the crosstalk term

What if we want to store multiple patterns $\boldsymbol{\xi}^\mu$ ($\mu = 1, \ldots, P$) in the connection matrix? A straightforward generalization of the synaptic weight matrix suggests

$$J_{ij} = \frac{1}{N} \sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu \tag{27}$$

which is a matrix of rank P.

Why might this work? Let us plug this into the update rule; the fixed point condition for storing memory pattern $\boldsymbol{\xi}^\nu$ requires that

$$\xi_i^\nu = \text{sgn}\left(\xi_i^\nu + \frac{1}{N}\sum_{j=1}^{N}\sum_{\mu\neq\nu}^{P}\xi_i^\mu\xi_j^\mu\xi_j^\nu\right) \tag{28}$$

Note that the ξ_i^μ are uncorrelated random variables with mean zero and variance 1. Thus the second term on the right hand side (the crosstalk term) has mean 0 and variance $NP/N^2 = P/N$. In the limit $P \ll N$ this term is smaller than 1, and $\boldsymbol{\xi}^\nu$ is a stable fixed point.

Spurious states

Given the synaptic weight matrix defined above, we have shown that the desired patterns $\boldsymbol{\xi}^\mu$ are stable fixed points of the network dynamics in the limit $P \ll N$. However, they are not the only stable fixed points; many others exist. A non-trivial example is any mixture of an odd number of memory patterns, such as

$$\boldsymbol{\xi}^{mix} = \text{sgn}(\boldsymbol{\xi}^{\mu_1} + \boldsymbol{\xi}^{\mu_2} + \boldsymbol{\xi}^{\mu_3}) \tag{29}$$

To see this, let's plug this into the update rule:

$$\sum_j J_{ij}\xi_j^{mix} = \frac{1}{N}\sum_j\sum_{\mu=1}^{P}\xi_i^\mu\xi_j^\mu\xi_j^{mix}$$

$$= \frac{1}{N}\left(\sum_j \xi_i^{\mu_1}\xi_j^{\mu_1}\xi_j^{mix} + \sum_j \xi_i^{\mu_2}\xi_j^{\mu_2}\xi_j^{mix} + \sum_j \xi_i^{\mu_3}\xi_j^{\mu_3}\xi_j^{mix} + \text{crosstalk terms}\right)$$

Looking at this equation, the mean value $\langle\xi_j^{\mu_1}\xi_j^{mix}\rangle$ is $\frac{3}{4}(+1) + \frac{1}{4}(-1) = \frac{1}{2}$, because ξ_j^{mix} agrees with $\xi_j^{\mu_1}$ in three of the four equally likely sign combinations of the other two patterns. Thus

$$\sum_j J_{ij}\xi_j^{mix} = \frac{1}{2}(\xi_i^{\mu_1} + \xi_i^{\mu_2} + \xi_i^{\mu_3}) + \text{crosstalk term}$$

Ignoring the crosstalk term (it is small when $P \ll N$), we confirm that the right hand side has the same sign as ξ_i^{mix}, and this spurious memory pattern is also a stable fixed point.

Storage capacity

The probability of error-free recall of a stored pattern of N neurons is $(1 - P_{error})^N \approx 1 - NP_{error}$. The P_{error} for a single neuron can be calculated as follows. The crosstalk term is approximately Gaussian with zero mean and variance $\sigma^2 = P/N$, and an error occurs when it exceeds the signal, of size 1, and has the opposite sign:

$$P_{error} = \frac{1}{\sqrt{2\pi}\sigma} \int_1^\infty \exp\left(-\frac{x^2}{2\sigma^2}\right) dx = \frac{1}{2}\left[1 - \mathrm{erf}\left(\sqrt{N/2P}\right)\right] \tag{30}$$

Note that in the limit $x \to \infty$, $1 - \mathrm{erf}(x) \approx \frac{\exp(-x^2)}{\sqrt{\pi}x}$. Now if we require that $P_{error} < 0.01/N$, we obtain

$$\log P_{error} \approx -\log 2 - N/2P - \frac{1}{2}\log \pi - \frac{1}{2}\log(N/2P) < \log 0.01 - \log N$$

Taking the leading order in N, we obtain

$$N/2P > \log N, \qquad \text{i.e.} \qquad P_{max} \sim \frac{N}{2\log N} \tag{31}$$

If instead we tolerate a small fixed fraction of incorrect bits in the retrieved pattern, rather than demanding error-free recall, the capacity is proportional to N: the classical result of Amit, Gutfreund and Sompolinsky is $P_{max} \approx 0.138N$, beyond which the retrieval states abruptly disappear.

Energy landscape of the Hopfield model

A more physical way to think about the dynamics in the Hopfield model is to consider an energy landscape, defined by

$$E = -\frac{1}{2}\sum_{i,j} s_i J_{ij} s_j - \sum_i h_i s_i \tag{32}$$

It is easy to show that the update rule does not increase E. Suppose neuron i is updated from s_i to s_i^* while all other units are held fixed. Using $J_{ii} = 0$ and $J_{ij} = J_{ji}$, the energy change is

$$\Delta E = -\left(s_i^* - s_i\right)\left(\sum_j J_{ij} s_j + h_i\right) \leq 0,$$

since $s_i^* = \mathrm{sgn}\left(\sum_j J_{ij} s_j + h_i\right)$ has the same sign as the bracket. The energy is bounded below and decreases at every flip, so the asynchronous dynamics must converge to a local minimum: there are no limit cycles and no chaos.

Now by substituting the expression for the synaptic weight matrix, Equation 27, and setting $\mathbf{h} = 0$, we have

$$E = -\frac{1}{2N}\sum_{ij} s_i \left(\sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu\right) s_j = -\frac{1}{2N}\sum_{\mu=1}^{P}\left(\sum_i \xi_i^\mu s_i\right)\left(\sum_j \xi_j^\mu s_j\right) \tag{33}$$

Writing in vector form, the energy function can be rewritten as

$$E = -\frac{1}{2N}\sum_{\mu=1}^{P} (\boldsymbol{\xi}^\mu \cdot \mathbf{s})^2 \tag{34}$$

The energy is therefore lowest when the state has a large overlap with one of the stored patterns, which is exactly the statement that the memories sit at the bottom of the landscape.

Energy landscape for continuous-valued networks

The Hopfield construction above relies on binary units. We now show that a similar energy function exists for the continuous-valued rate network, under certain conditions. As before, by analogy with mechanics, a ball moves downhill on the landscape and the steady state is a local minimum.

Let's consider

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \phi(\mathbf{h} + \mathbf{J} \cdot \mathbf{v}), \tag{35}$$

where ϕ, the activation function, specifies the relationship between steady state firing rate and total synaptic input current; we assume it is monotonically increasing and invertible, so $\phi' > 0$. To this end, let us define the instantaneous output

$$\mathbf{r} = \phi(\mathbf{h} + \mathbf{J} \cdot \mathbf{v}),$$

with inverse ϕ^{-1}. This plays the role that $\mathbf{s}$ played in the Hopfield network above; we reserve $\mathbf{s}$ for binary spins. The energy function is defined as

$$\begin{aligned} E = -\frac{1}{2}\sum_{ij} r_i J_{ij} r_j - \sum_i h_i r_i + \sum_i G(r_i), \\ G(r_i) = \int_0^{r_i} \phi^{-1}(x)dx \end{aligned} \tag{36}$$

Next we shall prove that for a symmetric and nonsingular weight matrix $\mathbf{J}$, $\frac{dE}{dt} \leq 0$ and E is bounded from below for a biologically plausible activation function ϕ. To show this, note that

$$\frac{dE}{dt} = \sum_i \frac{\partial E}{\partial r_i}\frac{dr_i}{dt}$$

Substituting the expression for E, the quadratic term contributes twice, and

$$\frac{dE}{dt} = -\frac{1}{2}\left(\sum_i\sum_j J_{ij} r_j \frac{dr_i}{dt} + \sum_j\sum_i r_i J_{ij}\frac{dr_j}{dt}\right) - \sum_i h_i \frac{dr_i}{dt} + \sum_i \phi^{-1}(r_i)\frac{dr_i}{dt}$$

If the synaptic weight matrix is symmetric, $J_{ij} = J_{ji}$, the above is further simplified. In matrix form,

$$\frac{dE}{dt} = \left[-\mathbf{J}\mathbf{r} - \mathbf{h} + \phi^{-1}(\mathbf{r})\right] \cdot \frac{d\mathbf{r}}{dt}$$

We next derive the explicit expression for $\frac{d\mathbf{r}}{dt}$. Writing ϕ' for the diagonal matrix $\mathrm{diag}(\phi'(h_i + (\mathbf{J}\mathbf{v})_i))$,

$$\frac{d\mathbf{r}}{dt} = \phi'\mathbf{J}\frac{d\mathbf{v}}{dt}$$

We can further express $\mathbf{v}$ in terms of $\mathbf{r}$ using the inverse function ϕ^{-1}:

$$\mathbf{v} = \mathbf{J}^{-1}[\phi^{-1}(\mathbf{r}) - \mathbf{h}],$$

and

$$\tau_r \frac{d\mathbf{v}}{dt} = -\mathbf{v} + \mathbf{r} = -\mathbf{J}^{-1}[\phi^{-1}(\mathbf{r}) - \mathbf{h}] + \mathbf{r}.$$

Thus,

$$\tau_r \frac{d\mathbf{r}}{dt} = \phi'[-\phi^{-1}(\mathbf{r}) + \mathbf{h} + \mathbf{J}\mathbf{r}]$$

As a result, the rate of change of the energy function is

$$\frac{dE}{dt} = -\frac{1}{\tau_r} \sum_i \phi_i' \left[\mathbf{h} + \mathbf{J}\mathbf{r} - \phi^{-1}(\mathbf{r})\right]_i^2 \leq 0 \tag{37}$$

where the inequality uses $\phi' > 0$. The energy function is bounded from below provided $\mathbf{r}$ does not diverge. Indeed, this is a biologically plausible assumption: no neuron can have an excessively high firing rate. In practice, ϕ is chosen as a sigmoidal function so that its value saturates for high synaptic currents.

The transformer architecture

A transformer processes a sequence of tokens by repeatedly applying two operations: attention mixes information across token positions, and a feedforward network transforms the features at each position [1]. A token can represent a word fragment, an image patch, or another unit of input. Its initial representation is a learned embedding, with information about position supplied by positional embeddings or by position-dependent transformations inside attention.

The original transformer contains an encoder and a decoder, with cross-attention allowing the decoder to read the encoder's representations. For our purposes, we focus on a causal, decoder-only transformer, as used in autoregressive language modeling. At position t, it can access positions $s \leq t$ and predicts the next token.

Let $\mathbf{x}_t^{(\ell)} \in \mathbb{R}^d$ be the representation at layer ℓ. One common modern implementation is a pre-normalized block:

$$\mathbf{a}_t^{(\ell)} = \mathbf{x}_t^{(\ell)} + \mathrm{MHA}_t\Big(\{\mathrm{LN}(\mathbf{x}_s^{(\ell)})\}_{s \leq t}\Big), \tag{38}$$

$$\mathbf{x}_t^{(\ell+1)} = \mathbf{a}_t^{(\ell)} + \mathrm{FFN}\Big(\mathrm{LN}(\mathbf{a}_t^{(\ell)})\Big). \tag{39}$$

Here LN is layer normalization, MHA is multi-head attention, and a simple feedforward sublayer has the form

$$\mathrm{FFN}(\mathbf{y}) = \mathbf{W}_2 f_{\mathrm{ff}}(\mathbf{W}_1 \mathbf{y} + \mathbf{c}_1) + \mathbf{c}_2,$$

where f_{ff} acts componentwise. Each sublayer adds its output to its input through a residual connection. This normalization placement is a common modern choice; the original transformer normalizes after the residual addition. After

the final layer, a linear map and a softmax over the vocabulary produce next-token probabilities. Training adjusts the projections and other parameters by minimizing prediction error across sequences.

To understand attention, suppress the layer index and consider one head. From the input representation it constructs a query, a key, and a value:

$$\mathbf{q}_t = \mathbf{W}_Q \mathbf{x}_t, \qquad \mathbf{k}_t = \mathbf{W}_K \mathbf{x}_t, \qquad \mathbf{b}_t = \mathbf{W}_V \mathbf{x}_t, \tag{40}$$

with $\mathbf{q}_t, \mathbf{k}_t \in \mathbb{R}^{d_k}$ and $\mathbf{b}_t \in \mathbb{R}^{d_v}$. The query asks what information to retrieve; a key identifies a stored item; its value contains the associated information. Causal scaled dot-product attention is

$$\begin{aligned} a_{ts} &= \frac{\exp(\mathbf{q}_t^T \mathbf{k}_s / \sqrt{d_k})}{\sum_{r \le t} \exp(\mathbf{q}_t^T \mathbf{k}_r / \sqrt{d_k})}, \qquad s \le t, \\ \mathbf{o}_t &= \sum_{s \le t} a_{ts} \mathbf{b}_s. \end{aligned} \tag{41}$$

Thus the output is a weighted average of values, with greater weight assigned to keys that match the query. The factor $1/\sqrt{d_k}$ controls the scale of the dot products. Multiple heads perform this operation with different projections; their outputs are concatenated and mapped back to dimension d by a learned output matrix $\mathbf{W}_O$.

This is content-addressable retrieval, but it is not yet the recurrent attractor dynamics of the Hopfield model. Within a head, the query reads the memory once. In standard cached decoding the memory consists of the growing list of keys and values. Computing attention over a sequence involves $O(T^2)$ query–key interactions, while each new token accesses $O(T)$ past items. The feedforward sublayer, in contrast, acts separately at each position.

Linear attention: from a list of items to a memory matrix

Can we summarize the past in a fixed-size state? Replace the exponential similarity by a factorized kernel,

$$\exp(\mathbf{q}^T \mathbf{k} / \sqrt{d_k}) \quad \longrightarrow \quad \kappa(\mathbf{q}, \mathbf{k}) = \varphi(\mathbf{q})^T \varphi(\mathbf{k}),$$

where $\varphi : \mathbb{R}^{d_k} \to \mathbb{R}^m$ is a feature map. We use φ rather than ϕ, which already denotes the neuronal transfer function. For example, a componentwise strictly positive feature map gives positive attention weights and a nonzero normalization denominator. Such a kernel is generally a different model; particular feature constructions can instead approximate the exponential kernel.

Define $\widetilde{\mathbf{q}}_t = \varphi(\mathbf{q}_t)$ and $\widetilde{\mathbf{k}}_t = \varphi(\mathbf{k}_t)$. Associativity of matrix multiplication gives the exact rearrangement

$$\mathbf{o}_t = \frac{\sum_{s \le t} \mathbf{b}_s (\widetilde{\mathbf{k}}_s^T \widetilde{\mathbf{q}}_t)}{\sum_{s \le t} \widetilde{\mathbf{k}}_s^T \widetilde{\mathbf{q}}_t} = \frac{\mathbf{M}_t \widetilde{\mathbf{q}}_t}{\mathbf{z}_t^T \widetilde{\mathbf{q}}_t}, \tag{42}$$

where

$$\mathbf{M}_t = \sum_{s \le t} \mathbf{b}_s \widetilde{\mathbf{k}}_s^T, \qquad \mathbf{z}_t = \sum_{s \le t} \widetilde{\mathbf{k}}_s. \tag{43}$$

Instead of retaining every key and value, we can update

$$\boxed{\begin{aligned} \mathbf{M}_t &= \mathbf{M}_{t-1} + \mathbf{b}_t \widetilde{\mathbf{k}}_t^T, \\ \mathbf{z}_t &= \mathbf{z}_{t-1} + \widetilde{\mathbf{k}}_t, \end{aligned}} \tag{44}$$

starting with zero states. Their dimensions, $d_v \times m$ and m, do not grow with sequence length. For fixed feature dimensions the attention work is linear in T. This recurrent representation is the central observation behind linear transformers [2]. The recurrence is exact for the chosen factorized kernel; replacing the softmax kernel was the architectural change. "Linear attention" refers to sequence-length complexity, not to linearity of the entire network in its inputs.

The connection with Hebbian associative memory

To isolate the storage mechanism, now consider an *unnormalized* variant. Absorb any feature map into the notation for keys and queries, and use $\mathbf{k}_t, \mathbf{q}_t \in \mathbb{R}^{d_k}$:

$$\mathbf{M}_t = \mathbf{M}_{t-1} + \mathbf{b}_t \mathbf{k}_t^T, \qquad \mathbf{o}_t = \mathbf{M}_t \mathbf{q}_t. \tag{45}$$

Dropping the denominator changes the model again. Later recurrent attention architectures often normalize or gate the output separately; these operations do not restore the denominator in Equation 42.

Each outer product writes an association from a key to a value. Indeed,

$$\mathbf{M}_t \mathbf{q} = \sum_{s \le t} \mathbf{b}_s (\mathbf{k}_s^T \mathbf{q}). \tag{46}$$

If the stored keys are orthonormal, querying with $\mathbf{q} = \mathbf{k}_r$ returns $\mathbf{b}_r$ exactly. For unit-length but nonorthogonal keys,

$$\mathbf{M}_t \mathbf{k}_r = \mathbf{b}_r + \sum_{s \ne r} \mathbf{b}_s (\mathbf{k}_s^T \mathbf{k}_r).$$

The second term is crosstalk, directly analogous to the crosstalk term in the Hopfield model. At most d_k keys can be mutually orthogonal, so this simple exact-retrieval construction has a finite dimensional limitation.

The connection becomes explicit if $d_k = d_v = N$ and we tie keys and values to the same binary patterns:

$$\mathbf{k}_\mu = \mathbf{b}_\mu = \frac{\boldsymbol{\xi}^\mu}{\sqrt{N}}, \qquad \mathbf{M}_P = \frac{1}{N} \sum_{\mu=1}^{P} \boldsymbol{\xi}^\mu (\boldsymbol{\xi}^\mu)^T. \tag{47}$$

This is precisely the Hebbian outer-product matrix. To obtain the zero-diagonal Hopfield convention used in the energy argument above, take $\mathbf{J} = \mathbf{M}_P - (P/N)\mathbf{I}$.

Attention with distinct keys and values is a *heteroassociative* memory: one pattern retrieves another. Tying them gives an *autoassociative* memory, where a pattern retrieves itself.

The equality of storage matrices does not establish equality of retrieval dynamics. Attention evaluates $\mathbf{M}_t\mathbf{q}_t$ once. Hopfield retrieval holds $\mathbf{J}$ fixed and repeatedly updates activity, with the asynchronous convention specified above. Token position t, network depth ℓ, and the iteration number of an attractor retrieval are three different indices. In particular, the classical capacity $0.138N$ concerns Hopfield attractor retrieval under its stated assumptions; it is not automatically the capacity of a linear-attention head.

The delta rule: write the error rather than the whole value

The additive Hebbian rule has no mechanism to check whether an association has already been stored. Repeated writes keep accumulating, and a changed value for the same key is added on top of the old value. A natural alternative is to first predict the value already associated with the current key,

$$\widehat{\mathbf{b}}_t = \mathbf{M}_{t-1}\mathbf{k}_t, \qquad \mathbf{e}_t = \mathbf{b}_t - \widehat{\mathbf{b}}_t,$$

and then write only the error:

$$\boxed{\mathbf{M}_t = \mathbf{M}_{t-1} + \beta_t \mathbf{e}_t \mathbf{k}_t^T.} \tag{48}$$

This is the delta rule, or an error-modulated Hebbian update. Its component form is $\Delta M_{ij} = \beta_t e_{t,i} k_{t,j}$. Linear attention and this error-correcting extension can be understood as fast-weight memories [3]; efficient sequence-parallel implementations underlie DeltaNet [4].

The rule follows from the instantaneous squared prediction error

$$\mathcal{L}_t(\mathbf{M}) = \frac{1}{2}\|\mathbf{M}\mathbf{k}_t - \mathbf{b}_t\|^2, \qquad \nabla_{\mathbf{M}}\mathcal{L}_t = (\mathbf{M}\mathbf{k}_t - \mathbf{b}_t)\mathbf{k}_t^T. \tag{49}$$

Equation 48 is one gradient step with step size β_t, evaluated at $\mathbf{M}_{t-1}$. In the normalized-key setting, $\|\mathbf{k}_t\| = 1$, multiply the update by $\mathbf{k}_t$ to obtain

$$\mathbf{M}_t\mathbf{k}_t = (1-\beta_t)\widehat{\mathbf{b}}_t + \beta_t\mathbf{b}_t. \tag{50}$$

For $0 \le \beta_t \le 1$, the retrieved value interpolates between the old prediction and the new target. At $\beta_t = 1$ the current association is replaced exactly. Its error is reduced by the factor $1 - \beta_t$. Without key normalization, the corresponding factor is $1 - \beta_t\|\mathbf{k}_t\|^2$, so the step size must account for key norm.

What happens to another association? For any query $\mathbf{q}$,

$$(\mathbf{M}_t - \mathbf{M}_{t-1})\mathbf{q} = \beta_t \mathbf{e}_t(\mathbf{k}_t^T\mathbf{q}). \tag{51}$$

Thus the write leaves orthogonal cues unchanged and affects overlapping cues in proportion to their similarity. It corrects the current association but does not

generally preserve all earlier ones. For example, with a single unit key and two successive values, $\beta_1 = \beta_2 = 1$ gives the second value after the second write; additive Hebbian storage gives the sum of both values. If that key was already predicted correctly, the delta write is zero.

In what sense is this an RNN?

Rearranging the delta update gives

$$\mathbf{M}_t = \mathbf{M}_{t-1}(\mathbf{I} - \beta_t \mathbf{k}_t \mathbf{k}_t^T) + \beta_t \mathbf{b}_t \mathbf{k}_t^T, \qquad \mathbf{o}_t = \mathbf{M}_t \mathbf{q}_t. \tag{52}$$

The hidden state of this RNN is a matrix. It has a transition determined by the current input and a rank-one input-dependent write. For fixed current keys and gates the recurrence is affine in the previous state, although the full network is nonlinear in its inputs. The readout here uses the state *after* the current write, consistent with attention to $s \leq t$.

There are two levels of adaptation. The projection matrices and the functions producing β_t are slow parameters learned across training sequences. The state $\mathbf{M}_t$ changes within each sequence, including during ordinary inference. It is a fast weight matrix in the associative-memory interpretation, but computationally it is an activation state, not a change to the trained projection parameters.

The loss in Equation 49 is also different from the Hopfield energy: gradient descent changes the memory matrix, and the loss changes with each input pair. Hopfield recall instead changes the activity under fixed couplings. Even if keys and values are tied, delta updates need not preserve matrix symmetry. There is consequently no general classical Hopfield energy argument for this recurrent memory.

Optional extension: active forgetting and KDA

Error correction selectively changes the mapping for the current key, but does not explicitly erase unrelated, obsolete context. To give the network control over retention, first decay the memory and then apply the delta rule:

$$\begin{aligned} \overline{\mathbf{M}}_t &= a_t \mathbf{M}_{t-1}, \qquad 0 < a_t < 1, \\ \mathbf{M}_t &= \overline{\mathbf{M}}_t + \beta_t (\mathbf{b}_t - \overline{\mathbf{M}}_t \mathbf{k}_t) \mathbf{k}_t^T. \end{aligned} \tag{53}$$

This combines the complementary operations of forgetting and targeted writing, as in Gated DeltaNet [5]. The gate can be learned as a function of the current token. In the absence of writes, a constant $a_t = a$ gives $\mathbf{M}_{t+n} = a^n \mathbf{M}_t$, with retention time $\tau_{\text{mem}} = -1/\log a$ measured in tokens. Input-dependent gates allow a context change to trigger rapid erasure.

Kimi Delta Attention extends this scalar gate to a separate retention factor for each key channel [6]. Define

$$\mathbf{\Lambda}_t = \text{Diag}(\alpha_{t,1}, \ldots, \alpha_{t,d_k}), \qquad 0 < \alpha_{t,j} < 1.$$

The recurrence is

$$\boxed{\begin{aligned}\overline{\mathbf{M}}_t &= \mathbf{M}_{t-1}\mathbf{\Lambda}_t,\\ \mathbf{e}_t &= \mathbf{b}_t - \overline{\mathbf{M}}_t\mathbf{k}_t,\\ \mathbf{M}_t &= \overline{\mathbf{M}}_t + \beta_t\mathbf{e}_t\mathbf{k}_t^T,\\ \mathbf{o}_t &= \mathbf{M}_t\mathbf{q}_t.\end{aligned}} \tag{54}$$

Right multiplication by $\mathbf{\Lambda}_t$ decays the columns of $\mathbf{M}_{t-1}$, so different key coordinates can have different retention times. These are coordinate channels, not individually addressable stored items. The scalar-gated case is recovered when $\mathbf{\Lambda}_t = a_t\mathbf{I}$.

Combining the steps,

$$\begin{aligned}\mathbf{M}_t = \mathbf{M}_{t-1}\mathbf{\Lambda}_t(\mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^T)\\ + \beta_t\mathbf{b}_t\mathbf{k}_t^T.\end{aligned} \tag{55}$$

The order of the factors matters: decay comes first, and the prediction error must be computed from the decayed memory. Writing $\mathbf{S}_t = \mathbf{M}_t^T$ gives the convention used in the Kimi K3 report [7]:

$$\begin{aligned}\mathbf{S}_t &= (\mathbf{I} - \beta_t\mathbf{k}_t\mathbf{k}_t^T)\mathbf{\Lambda}_t\mathbf{S}_{t-1} + \beta_t\mathbf{k}_t\mathbf{b}_t^T,\\ \mathbf{o}_t &= \mathbf{S}_t^T\mathbf{q}_t.\end{aligned} \tag{56}$$

In that implementation, short causal convolutions and nonlinearities help construct queries, keys, and values; queries and keys are normalized; and the retrieved output is normalized and gated before an output projection. The retention factors and write strengths are input dependent. These details surround the memory operation derived here. K3 combines KDA layers with global attention, so the fixed-size state description applies to a KDA head, not to all memory in the complete hybrid model.

For our comparison with Hopfield networks, the essential connection is the outer-product associative memory. Linear attention accumulates associations; the delta rule corrects them; KDA additionally controls their retention. Attractor retrieval would require a further dynamical construction with the memory held fixed and activity iterated. Its convergence cannot be inferred from the classical Hopfield proof, since a general KDA memory is neither square nor symmetric.

Questions for discussion. (1) Verify exact retrieval for orthonormal keys and identify the crosstalk for nonorthogonal keys. (2) For repeated presentations of one fixed unit key and target, with constant β and no forgetting, show that the retrieval error decays as $(1-\beta)^n$ after n writes. (3) Starting from a symmetric memory with tied keys and values, determine when one delta update preserves symmetry. Which additional condition is needed for channel-dependent decay to preserve it?

Nonlinear Ring Model

We have talked about working memory in the context of encoding a continuous variable such as eye position in the oculomotor integrator of the goldfish, in which the firing rates of a group of neurons vary monotonically with eye position. The neural activity can be viewed as a projection of the stimulus onto a one-dimensional vector (an eigenvector of the recurrent connectivity matrix, for example), and this is the so-called line attractor. In a second scheme, neural activity may represent selective features of the external world: for example, each neuron has a preferred spatial location at which it fires with the highest rate. Examples are the representation of space in place cells, the coding of head direction in head direction systems, and orientation selectivity in V1. Unlike the Hopfield network, whose attractors are isolated points, we shall now build a network whose attractor is a continuum.

Below, I will discuss how recurrent neural dynamics can implement the second scheme, in the abstract setting of an angular variable. This variable could be the orientation of a bar in the visual field, or the head direction as you move around. The ring network is a very useful and beautiful framework. From here on we measure time in units of the rate time constant and set $\tau_r = 1$, as we shall do throughout the next lecture. Let us introduce the model

$$\frac{dv_i}{dt} = -v_i + \phi\left(\sum_{j=1}^{N} J_{ij} v_j + I_i^0\right), \tag{57}$$

where $\phi(x) = [x]_+$ is a rectified linear function and the weights between neurons are determined by their angular difference and are thus translation invariant,

$$J_{ij} = \frac{1}{N} J(\theta_i - \theta_j), \tag{58}$$

where $\theta_i = -\pi + \frac{2\pi i}{N}$, the kernel $J(\cdot)$ is a 2π-periodic function, and I_i^0 is also 2π-periodic. Note that the kernel and the matrix are the same object: $J(\theta_i - \theta_j)/N$ is just the entry J_{ij} written as a function of the angular separation.

In the large N limit, we can replace $\sum_j \frac{1}{N} J(\theta_i - \theta_j) v_j(t)$ by an integral, and write

$$\frac{\partial v(\theta, t)}{\partial t} = -v(\theta, t) + \left[\frac{1}{2\pi}\int_{-\pi}^{\pi} J(\theta - \theta') v(\theta', t) d\theta' + I^0(\theta)\right]_+, \tag{59}$$

where we have equated $\frac{1}{N} = \frac{\Delta\theta}{2\pi}$ and then taken the limit $\Delta\theta \to 0$.

Here let us consider the Fourier expansion of the weight kernel and the input.

$$J(\theta) = J_0 + J_1 \cos\theta + \ldots \tag{60}$$

$$I^0(\theta) = I_0 + I_1 \cos(\theta - \theta_0), \tag{61}$$

where the external stimulus is centered at θ_0; we consider the simplest model by ignoring the higher order terms in J. Note that $J_0 < 0$ with $J_1 > 0$ is the

familiar "Mexican hat" arrangement: broad inhibition with local excitation. Because J is translation invariant, its eigenvectors are the Fourier modes and its eigenvalues are the coefficients themselves: J_0 for the uniform mode and $J_1/2$ for each of the two first harmonics. Everything below is the eigenmode analysis of the earlier section applied to this particular spectrum.

Now let us consider the steady state solution of the ring model:

$$v(\theta) = \left[\frac{1}{2\pi} \int_{-\pi}^{\pi} J(\theta - \theta') v(\theta') d\theta' + I^0(\theta) \right]_+ \tag{62}$$

Since $\cos(\theta - \theta') = \mathrm{Re} \exp(i(\theta - \theta'))$, we can plug in the expressions for J and I^0 and obtain

$$a_0 = \frac{1}{2\pi} \int_{-\pi}^{\pi} v(\theta') d\theta' \tag{63}$$

$$a_1 \exp(-i\psi) = \frac{1}{2\pi} \int_{-\pi}^{\pi} v(\theta') \exp(-i\theta') d\theta' \tag{64}$$

$$v(\theta) = \left[J_0 a_0 + I_0 + J_1 a_1 \cos(\theta - \psi) + I_1 \cos(\theta - \theta_0) \right]_+ \tag{65}$$

where ψ is the phase of the first Fourier mode, that is, the position of the activity bump on the ring.

Now let us first assume that (1) we are in the linear regime, such that the argument of the rectification is always positive. Let us make an *ansatz* that the solution has the form

$$\psi = \theta_0 \tag{66}$$

$$v(\theta) = J_0 a_0 + I_0 + (J_1 a_1 + I_1) \cos(\theta - \theta_0) \tag{67}$$

This is a solution provided that

$$a_0 = J_0 a_0 + I_0 \tag{68}$$

$$a_1 = \frac{1}{2}(J_1 a_1 + I_1) \tag{69}$$

In other words,

$$a_0 = \frac{I_0}{1 - J_0} \tag{70}$$

$$a_1 = \frac{I_1}{2 - J_1} \tag{71}$$

And the steady state neural activity is given by

$$v(\theta) = \frac{I_0}{1 - J_0} + \frac{2I_1}{2 - J_1} \cos(\theta - \theta_0) \tag{72}$$

Let us now define the selectivity index of the input, $SI_{input} = I_1/I_0$, and assume $I_1 \ll I_0$. The selectivity index of the recurrent network is then

$$SI_{network} = \frac{2(1 - J_0)}{2 - J_1} SI_{input}.$$

There is thus a large amplification of tuning if $J_1 < 2$ but close to 2, even though the input tuning is weak. This is precisely the $\lambda_\mu \to 1$ selective amplification of the eigenmode analysis, since the eigenvalue of the first harmonic is $J_1/2$.

Case 1: what happens when $J_0 > 1$?
Case 2: what happens when $J_1 > 2$? Here, let us consider the special case where there is no tuned external input, so that $I_1 = 0$. Without loss of generality, consider $\psi = 0$. We write $v(\theta) = [v_0 + v_1 \cos\theta]_+ = v_1[\cos\theta - \cos\theta_c]_+$, where v_0 and v_1 are the uniform and first-harmonic components of the argument of the rectification, and $\cos\theta_c = -v_0/v_1$ with $v_1 > 0$; the activity profile is now a bump of half-width θ_c. According to Eq. 64, and using $v_1 = J_1 a_1$, we have

$$v_1 = \frac{J_1}{2\pi}\int_{-\theta_c}^{\theta_c} v_1 \exp(-i\theta)(\cos\theta - \cos\theta_c)d\theta = \frac{J_1}{2\pi}\int_{-\theta_c}^{\theta_c} v_1 \cos\theta(\cos\theta - \cos\theta_c)d\theta \tag{73}$$

where the $\sin\theta$ part of the integral vanishes by symmetry. As a result, we have

$$1 = J_1 \frac{1}{2\pi}\left(\theta_c - \frac{1}{2}\sin 2\theta_c\right) \tag{74}$$

which determines the bump width θ_c from J_1 alone. Likewise, from the zeroth Fourier mode,

$$-v_1 \cos\theta_c = \frac{J_0}{2\pi}\int_{-\theta_c}^{\theta_c} v_1(\cos\theta - \cos\theta_c)d\theta + I_0 \tag{75}$$

Now defining

$$f(\theta_c) = \frac{1}{\pi}(\sin\theta_c - \theta_c \cos\theta_c),$$

we obtain

$$v_1 = \frac{-I_0}{f(\theta_c)J_0 + \cos\theta_c} \tag{76}$$

Now let us consider $\psi \neq 0$ with $I_1 = 0$. It is easy to show that any ψ is also a solution when $J_1 > 2$, because the self-consistency condition becomes

$$v_1 = \frac{J_1}{2\pi}\int_{-\theta_c}^{\theta_c} v_1 e^{-i(\theta-\psi)}[\cos(\theta - \psi) - \cos\theta_c]d\theta \tag{77}$$

which is independent of ψ. The network therefore possesses a *continuous* family of bump states, one for each position on the ring: a ring attractor. This is the rotational symmetry of $J(\theta - \theta')$ expressing itself as a zero mode of the dynamics, and it is what allows the bump to be moved smoothly by a weak tuned input, or to drift when the symmetry is broken by heterogeneity or noise.

In this lecture $\mathbf{J}$ has been structured and, apart from the ring kernel's shift symmetry, symmetric; the dynamics has consequently always settled onto fixed points. In the next lecture we draw J_{ij} independently at random, so that $\mathbf{J}$ is

strongly non-normal, the energy function no longer exists, and the same transfer function ϕ leads instead to sustained chaotic activity.

In the attractor models of this lecture, $\mathbf{J}$ has been structured and symmetric, allowing us to describe fixed-point dynamics and energy landscapes. The evolving, generally asymmetric attention memory $\mathbf{M}_t$ is a separate construction. In the next lecture we draw J_{ij} independently at random, so that $\mathbf{J}$ is strongly non-normal, the energy function no longer exists, and the same transfer function ϕ leads instead to sustained chaotic activity.

References

[1] A. Vaswani et al., *Attention Is All You Need*, NeurIPS (2017). arXiv:1706.03762.

[2] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, *Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention*, ICML (2020). arXiv:2006.16236.

[3] I. Schlag, K. Irie, and J. Schmidhuber, *Linear Transformers Are Secretly Fast Weight Programmers*, ICML (2021). arXiv:2102.11174.

[4] S. Yang, B. Wang, Y. Zhang, Y. Shen, and Y. Kim, *Parallelizing Linear Transformers with the Delta Rule over Sequence Length*, NeurIPS (2024). arXiv:2406.06484.

[5] S. Yang, J. Kautz, and A. Hatamizadeh, *Gated Delta Networks: Improving Mamba2 with Delta Rule*, ICLR (2025). arXiv:2412.06464.

[6] Kimi Team et al., *Kimi Linear: An Expressive, Efficient Attention Architecture* (2025). arXiv:2510.26692.

[7] Kimi Team, *Kimi K3: Open Frontier Intelligence*, technical report (2026), Section 2.1.1. Supplied report, arXiv:2607.24653v2.