# Hopfield network

John J. Hopfield (2007), Scholarpedia, 2(5):1977. doi:10.4249/scholarpedia.1977 revision #196687 [link to/cite this article]


- **Dr. John J. Hopfield**, Princeton University, NJ, USA


A **Hopfield net** is a recurrent neural network having synaptic connection pattern such that there is an underlying Lyapunov function for the activity dynamics. Started in any initial state, the state of the system evolves to a final state that is a (local) minimum of the Lyapunov function.

There are two popular forms of the model:

- Binary neurons with discrete time, updated one at a time

$$V_j(t+1) = \begin{cases} 1, & \text{if } \Sigma_k T_{jk} V_k(t) + I_j > 0 \\ 0, & \text{otherwise} \end{cases}$$

- Graded neurons with continuous time

$$dx_j/dt = -x_j/\tau + \Sigma_k T_{jk} g(x_k) + I_j \ .$$

Here,

- $V_j$ denotes activity of the $j$-th neuron.
- $x_j$ is the mean internal potential of the neuron.
- $I_j$ is direct input (e.g., sensory input or bias current) to the neuron.
- $T_{jk}$ is the strength of synaptic input from neuron $k$ to neuron $j$ .
- $g$ is a monotone function that converts internal potential into firing rate output of the neuron, i.e., $V_j = g(x_j)$ .

Contents

# Computers as dynamical systems

All real computers are dynamical systems that carry out computation through their change of state with time. A simple digital computer can be thought of as having a large number of binary storage registers. The state of the computer at a particular time is a long binary word. A computation is begun by setting the computer in an initial state determined by standard initialization + program + data. At each tick of the computer clock the state changes into another state, following a rule that is built in by the design of the machine. After a while the computer stops changing its state, and the answer to the computation is read out from a subset of the storage registers. A fragment of a two dimensional representation of the motion in state space is shown in Figure 1.

The digital states are represented by small blue dots, the state transitions by the arrow lines, and final states (which have no transitions leading out of them) by large red dots.

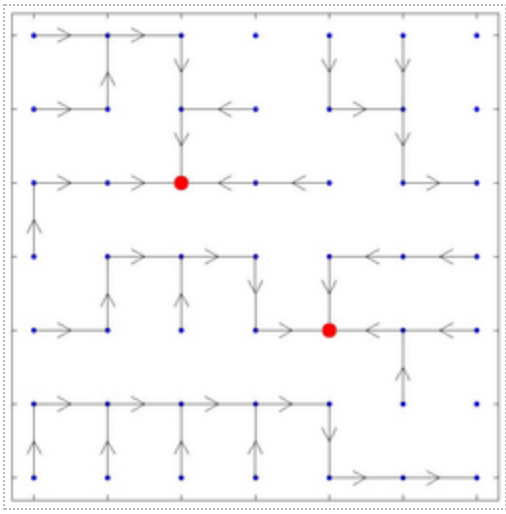

Figure 1: Two-dimensional representation of motion in state space. Transitions to states outside this fragment are not indicated

Neural network computation or analog computation can similarly be described as state space flows but without the discretization of state variables and time. For general neural networks, the only way to understand what future state will evolve after a long time is by following the dynamics in detail.

**Hopfield networks** have a Lyapunov function $E$ (or 'energy function') behind their dynamics which leads to an understanding of possible final states. The Lyapunov function decreases in a monotone fashion under the dynamics, and is bounded below. Computation in this system can now be thought of as being done by starting from an initial state from which a final state evolves by moving downhill on $E$. Because of the existence of an elementary Lyapunov function for the dynamics, the only possible asymptotic result is a state on an attractor. Hopfield networks have been shown to be capable of universal computation in the Turing sense.

# Binary neurons

The original Hopfield net [1982] used model neurons with two values of activity, that can be taken as 0 and 1. The strength of the synaptic connection from neuron $k$ to neuron $j$ is described by $T_{jk}$. The state vector $V$ of the network at a particular time $t$ has components $V_j$ describing the activity of neuron $j$ at time $t$. The dynamics of the system are defined as follows:

- Activity of neuron $j$ is $V_j$
- Strength of synaptic connection from neuron $k$ to neuron $j$ is $T_{jk}$
- Direct input (e.g. sensory input or bias current) to neuron $j$ is $I_j$
- Overall input to neuron $j$ is $x_j = \Sigma_k T_{jk} V_k + I_j$

Dynamical update of state:

- choose a neuron $p$ at random.
- If $x_p > 0$ set $V_j = 1$ else $V_j = 0$.
- Iterate this process.

If there are no self-connections (i.e. $T_{kk} = 0$ for all $k$) and the connections are symmetric ($T_{jk} = T_{kj}$ for all $j$, $k$) then this system has the Lyapunov function

$$E = -\frac{1}{2}\,\Sigma_{jk} T_{jk} V_j V_k - \Sigma_j I_j V_j\ .$$

This $E$ is also a Lyapunov function if the neurons are repetitively updated one at a time in any order.

Positive values of $T$ represent excitatory connections, and negative values inhibitory connections. Minor transformations of variables relate this system to a magnetic Ising system, with $T_{jk}$ equivalent to the exchange $J_{jk}$ between Ising spins $j$ and $k$. A related set of automata networks, which use synchronous update of all neurons at once, has been extensively studied.

The dynamical update described above can be thought of as a Monte Carlo procedure or a Glauber dynamics for the change of state of a thermodynamic system at zero temperature, where the energy of the system is described by $E$. The *Boltzmann Machine* uses a Hopfield net with stochastic update to simulate a non-zero temperature.

## Graded neuron response and continuous variables

Let the stochastic binary neurons be replaced by neurons whose instantaneous activity increases with input, so that $V_j = g(x_j)$ where $g$ is any monotone increasing function, bounded below and bounded above. In this representation, details of action potential timing are suppressed, and action potentials are thought of as being generated in a stochastic fashion with an instantaneous rate $g(x)$. The customary bounds on $g(x)$ are thus often taken as zero and the maximum firing rate. Let synaptic currents lag behind the firing rate with a simple exponential kernel $\exp(-t/\tau)$, corresponding in real neurons to a synaptic current that rises rapidly then decays exponentially. Within this description, the evolution of the state of the network is given by the ordinary differential equations

$$dx_j/dt = -x_j/\tau + \Sigma_{jk} T_{jk} V_k + I_j$$

When $T$ is symmetric, this dynamics [1984] has the Lyapunov function

$$E = -\frac{1}{2}\,\Sigma_{jk} T_{jk} V_j V_k - \Sigma_j I_j V_j + \frac{1}{\tau}\,\Sigma_j \int^{V_j} g^{-1}(Z) dZ$$

where $g^{-1}$ is the inverse of the gain function $g$. There is a significant limiting case of this function when $T$ has no diagonal elements and the input-output relation becomes a step, going from 0 to a maximum firing rate (for convenience, scaled to 1). The third term in this Lyapunov function is then zero or infinite, and this $E$ is then identical to the function for binary stochastic neurons except that now the third term restricts the domain to $0 \le V_j \le 1$ on which the third term is zero, instead of the earlier $V_j = 0$ or 1. With no diagonal elements in $T$, the minima of $E$ are all located at corners of the hypercube $0 \le V_j \le 1$. In this limit, the stable states of the continuous variable system are exactly the same as the stable states of the binary system.

## Illustration: 2 neuron flip-flop

A 2-neuron case with reciprocal inhibition illustrates the behavior. The input-output relation used for the neurons is shown in the first panel of Figure 2. Excitation was provided by setting $I_1 = I_2 > 0$.

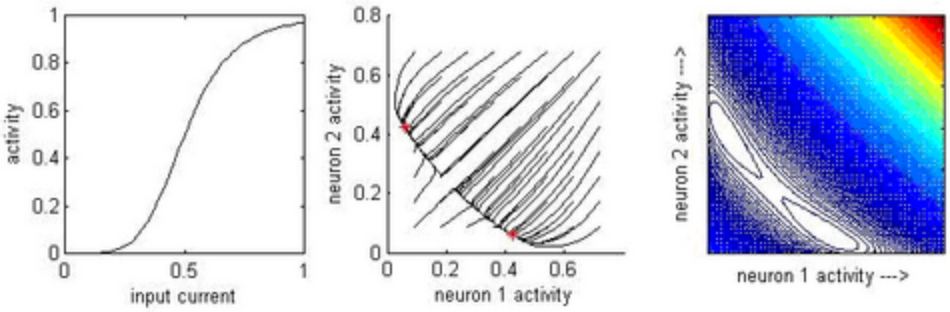


Figure 2: Phase portrait of 2-neuron Hopfield Network.

The second panel shows the trajectories of the system in the $(V_1, V_2)$ phase plane from a variety of starting states. Each trajectory starts at the end of a black line, and the activity moves along that line to ultimately terminate in one of the two point attractors located at the two red symbols "*". The third panel presents contour lines and a color map of $E$. The low $E$ contours are drawn. The color map shows the higher values of $E$, progressively larger from black to blue to yellow to red. The red dots of the central panel are located at the bottoms of the two minima seen in the third panel.

## Associative memory

The phenomenon of associative memory matches the idea of dynamics controlled by a Lyapunov function. Consider a set of states $M^p$ that are desired as memories. The location of each memory in state space describes the attributes of the memory. The idea of associative memory is that when a memory clue is presented, the actual memory that is most like the clue will be recapitulated. Suppose synaptic connections are constructed so that each memory vector $M^p$ is a local minimum of $E$. Starting with partial information about some memory $s$ means starting relatively nearer to $M^s$ than to other memories. This starting state is then likely to be within the 'valley' of the terrain $E$ that has $M^s$ as its lowest point. If so, the dynamics must result in the final state $M^s$, the correct memory reconstruction. Such a reconstruction is illustrated

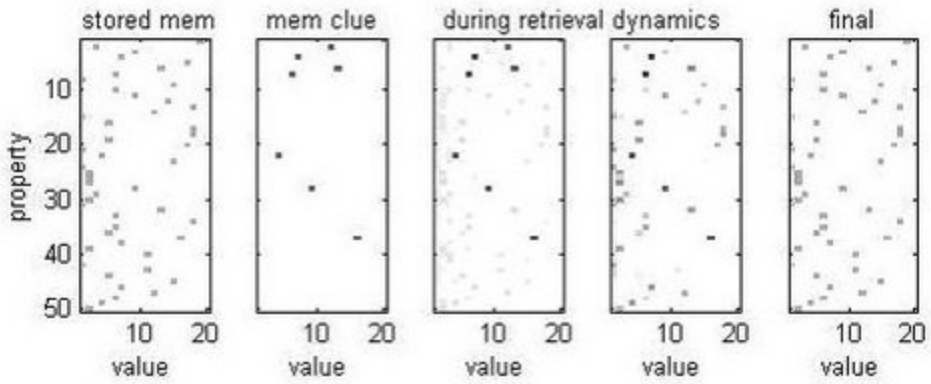


Figure 3: Memory recall from a small clue.

for a memory having 50 properties (e.g. name) and within each property 20 values (e.g., John, Mary). This memory contains 250 'known friends'. A particular 'known friend' can be described by a set of 1's locating the values for each property, and the information about one particular friend is shown in the grid at the left. Memory retrieval begins from a clue, an initial state describing partial information, as typified in the second grid where values are indicated for 7 categories. The next two panels show intermediate states during the retrieval process, and the final stable state is shown in the last panel.

Careful inspection of the figure will verify that the final state is the memory from which the clue was taken, except that the clue itself was slightly erroneous (an incorrect value for property 2) and the memory dynamics properly corrects that error.

When the number of memories to be contained in $T$ is not too large, and the memories not correlated, a new memory $M^{\text{new}}$ can be incorporated in the system through a synapse change rule

$$T_{ij}(\text{including the new memory}) = T_{ij}(\text{without}) + M_i^{\text{new}} M_j^{\text{new}}$$

This is a Hebbian rule in that the synapse change is proportional to the product of the activity of the pre- and post-synaptic neurons. This rule is appropriate to the case of neurons with binary activity represented as $V = \pm 1$, and memories chosen at random, and requires modifications for more general circumstances.

An excitatory-inhibitory generalized Hopfield network can implement an associative memory that is rather more biological in its details. It no longer requires memories to be uncorrelated; synaptic connections obey Dale's Law; neuron activity is limited by inhibitory feedback rather than by the maximum firing rate of a neuron; and memories are learned through changes in excitatory synapses only. In the 250 memory example above, a learning rule of the Hebbian type yielded completely accurate recall of 232 of the memories starting from clues containing 20% of the information in a memory. The number of memories that can be stored without disaster scales with the number of neurons. Neuronal noise will decrease the accuracy of recall, but the effect of the noise decreases with the size of the system, since the height of the 'barriers' in E separating different memories scales with the size. For large systems, the operation thus becomes collective, and the overall behavior is fail-soft to damage.

## Generalized Hopfield networks: other networks equivalent to symmetric networks

Lyapunov functions can be constructed for a variety of other networks that are related to the above networks by mathematical transformation or simple extensions. For example, if $T_{jk}$ is a symmetric matrix, and $\lambda_j$ and $\mu_k$ are vectors with all positive components, a network connected through a matrix $S_{jk} = T_{ij}\lambda_j\mu_k$ also has a Lyapunov function.

A broader class of related networks can be generated through using additional 'fast' neurons whose inputs and outputs are related in a way that produces an equivalent direct pathway that is symmetric. For example, if an additional neuron receives input $\Sigma_k A_k V_v$ and has an instantaneous output $f(\Sigma_k A_k V_v)$, where $f$ is the derivative of a monotone function $F$, and this in turn has connections that result in a current in cell $j$ that is $A_j f(\Sigma_k A_k V_v)$, the Lyapunov function simply requires the additional term $F(\Sigma_k A_k V_v)$. Simulations show that it is not necessary that such additional neurons be infinitely fast, but only that they are fast enough that the overall system does not oscillate.

When this additional neural circuitry is inhibitory, it can implement constraints on the overall behavior of the system. $A_k V_k = B$ describes a plane in activity ($V$) space. When $f$ is a function with a rapid onset, this inhibition can constrain the activity of the network to be on one side of the plane.

Related mathematics has been used to understand simple Hopfield networks with synchronous update, to special time-dependent problems, and to some models of synchronization of action potentials.

## Computation through optimization

Optimizations are common computational problems, and come in many forms. Finding the best straight line through a set of (x, y) data points, linear programming, and finding the shortest Traveling Salesman route to visit a set of cities, are typical examples. Other problems that are not normally thought of as optimizations can often be restated in the language of optimization. For example, a Sudoku puzzle can be described in terms of 'place the largest number of integers 1-9 into the blank spaces in the puzzle that you can, without violating any of the rules'. The correct solution is the unique pattern which has no blanks left at all, and thus has maximized the number of entries the puzzle-solver has put in.

Many optimization problems can be readily represented on Hopfield nets, by transforming the problem into variables such that the desired optimization corresponds to the minimization of the respective Lyapunov function [Hopfield and Tank 1985] In this representation, the dynamics of change in network state with time takes the system to a local energy minimum. If this local minimum is also the global minimum, the solution of the desired optimization task has been carried out by the convergence of the network state. Indeed, the energy function can be thought of as a programming language for transforming optimization problems into a solution method applying network dynamics. The resulting network could be either built in analog hardware or implemented in software on a digital machine.

Linear programming, the worker assignment problem, and decomposing signals into a basis set can all be solved exactly by Hopfield networks because the Lyapunov function for these problems can be constructed with a single (and thus global) minimum. When more computationally difficult problems are programmed using this approach, the Lyapunov function often has multiple local minima, and the dynamics of the network may converge to a local minimum rather than to the global minimum. Finding a good half-tone image from a gray-scale photograph and the n-queens chess problem can be programmed in this way. How effective such a network can be in finding a good solution is strongly dependent on the problem class.

## Dense Associative Memory or Modern Hopfield Network

Hopfield Networks [Hopfield 1982] are recurrent neural networks with dynamical trajectories converging to fixed point attractor states and described by an energy function. The state of each model neuron $i$ is defined by a time-dependent variable $V_i$, which can be chosen to be either discrete or continuous. A complete model describes the mathematics of how the future state of activity of each neuron depends on the known present or previous activity of all the neurons.

In the original Hopfield model of associative memory [Hopfield 1982], the variables were binary, and the dynamics were described by a one-at-a-time update of the state of the neurons. An energy function quadratic in the $V_i$ was defined, and the dynamics consisted of changing the activity of each single neuron $i$ only if doing so would lower the total energy of the system. This same idea was extended to the case of $V_i$ being a continuous variable representing the output of neuron $i$, and $V_i$ being a monotonic function of an input current. The dynamics became expressed as a set of first-order differential equations for which the "energy" of the system always decreased [Hopfield 1984]. The energy in the continuous case has one term which is quadratic in the $V_i$ (as in the binary model), and a second term which depends on the gain function (neuron's activation function). While having many desirable properties of associative memory, both of these classical systems suffer from a small memory storage capacity, which scales linearly with the number of input features [Hopfield 1982].

Dense Associative Memories [Krotov & Hopfield 2016] (also known as the Modern Hopfield Networks [Ramsauer et al. 2020]) are generalizations of the classical Hopfield Networks that break the linear scaling relationship between the number of input features and the number of stored memories. This is achieved by introducing stronger non-linearities (either in the energy function or neurons' activation functions) leading to super-linear [Krotov & Hopfield 2016] (even an exponential [Demircigil et al. 2017]) memory storage capacity as a function of the number of feature neurons. The network still requires a sufficient number of hidden neurons [Krotov & Hopfield 2020].

The key theoretical idea [Krotov & Hopfield 2016] behind the Modern Hopfield Networks is to use an energy function and an update rule that is more sharply peaked around the stored memories in the space of neuron's configurations compared to the classical Hopfield Network.

## Discrete Variables

A simple example [Krotov & Hopfield 2016] of the Modern Hopfield Network can be written in terms of binary variables $V_i$ that represent the active $V_i = +1$ and inactive $V_i = -1$ state of the model neuron $i$.

$$E = - \sum_{\mu=1}^{N_{\text{mem}}} F\Big( \sum_{i=1}^{N_f} \xi_{\mu i} V_i \Big)$$

In this formula the weights $\xi_{\mu i}$ represent the matrix of memory vectors (index $\mu = 1 ... N_{\text{mem}}$ enumerates different memories, and index $i = 1 ... N_f$ enumerates the content of each memory corresponding to the $i$-th feature neuron), and the function $F(x)$ is a rapidly growing non-linear function. The update rule for individual neurons (in the asynchronous case) can be written in the following form

$$V_i^{(t+1)} = Sign\Big[ \sum_{\mu=1}^{N_{\text{mem}}} \Big( F\Big( \xi_{\mu i} + \sum_{j \neq i} \xi_{\mu j} V_j^{(t)} \Big) - F\Big( - \xi_{\mu i} + \sum_{j \neq i} \xi_{\mu j} V_j^{(t)} \Big) \Big) \Big]$$

which states that in order to calculate the updated state of the $i$-th neuron the network compares two energies: the energy of the network with the $i$-th neuron in the ON state and the energy of the network with the $i$-th neuron in the OFF state, given the states of the remaining neuron. The updated state of the $i$-th neuron selects the state that has the lowest of the two energies [Krotov & Hopfield 2016].

In the limiting case when the non-linear energy function is quadratic $F(x) = x^2$ these equations reduce to the familiar energy function and the update rule for the classical binary Hopfield Network [Hopfield 1982].

The memory storage capacity of these networks can be calculated for random binary patterns. For the power energy function $F(x) = x^n$ the maximal number of memories that can be stored and retrieved from this network without errors is given by [Krotov & Hopfield 2016]

$$N_{\text{mem}}^{max} \approx \frac{1}{2(2n-3)!!} \frac{N_f^{n-1}}{\ln(N_f)}$$

For an exponential energy function $F(x) = e^x$ the memory storage capacity is exponential in the number of feature neurons [Demircigil et al. 2017]

$$N_{\text{mem}}^{max} \approx 2^{N_f/2}$$

## Continuous Variables

Modern Hopfield Networks or Dense Associative Memories can be best understood in continuous variables and continuous time [Ramsauer et al. 2020], [Krotov & Hopfield 2020] . Consider the network architecture, shown in Fig.4, and the equations for neuron's states evolution [Krotov & Hopfield 2020]

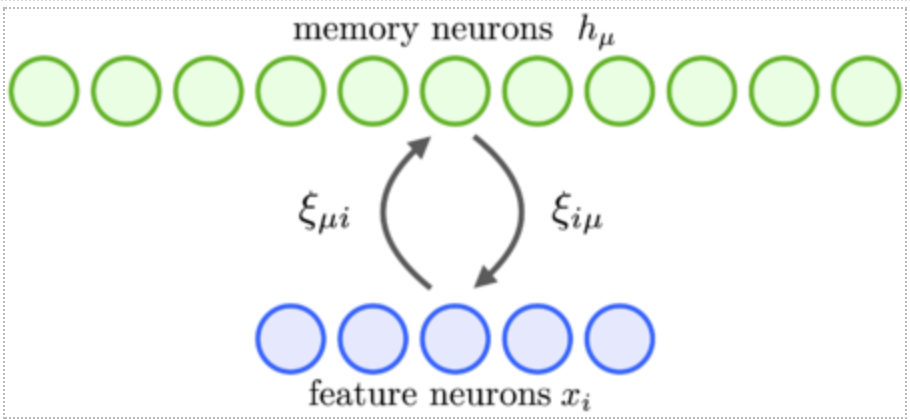


Figure 4: An example of a continuous Modern Hopfield Network with $N_f = 5$ feature neurons and $N_{\text{mem}} = 11$ memory (hidden) neurons with symmetric synaptic connections between them.

$$\begin{cases} \tau_f \frac{dx_i}{dt} = \sum\limits_{\mu=1}^{N_h} \xi_{i\mu} f_\mu - x_i + I_i \\ \tau_h \frac{dh_\mu}{dt} = \sum\limits_{i=1}^{N_f} \xi_{\mu i} g_i - h_\mu \end{cases} \qquad (1)$$

where the currents of the feature neurons are denoted by $x_i$, and the currents of the memory neurons are denoted by $h_\mu$ ($h$ stands for hidden neurons). There are no synaptic connections among the feature neurons or the memory neurons. A matrix $\xi_{\mu i}$ denotes the strength of synapses from a feature neuron $i$ to the memory neuron $\mu$. The synapses are assumed to be symmetric, so that the same value characterizes a different physical synapse from the memory neuron $\mu$ to the feature neuron $i$. The outputs of the memory neurons and the feature neurons are denoted by $f_\mu$ and $g_i$, which are non-linear functions of the corresponding currents. In general these outputs can depend on the currents of all the neurons in that layer so that $f_\mu = f(\{h_\mu\})$ and $g_i = g(\{x_i\})$. It is convenient to define these activation function as derivatives of the Lagrangian functions for the two groups of neurons [Krotov & Hopfield 2020]

$$f_\mu = \frac{\partial L_h}{\partial h_\mu}, \quad \text{and} \quad g_i = \frac{\partial L_v}{\partial x_i} \qquad (2)$$

This way the specific form of the equations for neuron's states is completely defined once the Lagrangian functions are specified. Finally, the time constants for the two groups of neurons are denoted by $\tau_f$ and $\tau_h$, $I_i$ is the input current to the network that can be driven by the presented data.

General systems of non-linear differential equations can have many complicated behaviors that can depend on the choice of the non-linearities and the initial conditions. For Hopfield Networks, however, this is not the case - the dynamical trajectories always converge to a fixed point attractor state. This property is achieved because these equations are specifically engineered so that they have an underlying energy function [Krotov & Hopfield 2020]

$$E(t) = \Big[\sum_{i=1}^{N_f}(x_i - I_i)g_i - L_x\Big] + \Big[\sum_{\mu=1}^{N_h} h_\mu f_\mu - L_h\Big] - \sum_{\mu,i} f_\mu \xi_{\mu i} g_i \qquad (3)$$

| | Model A | Model B | Model C |
|---|---|---|---|
| gian ns | $L_h = \sum_{\mu=1}^{N_h} F(h_\mu)$ <br> $L_x = \sum_{i=1}^{N_f} \lvert x_i \rvert$ | $L_h = \log\Big(\sum_{\mu=1}^{N_h} e^{h_\mu}\Big)$ <br> $L_x = \frac{1}{2}\sum_{i=1}^{N_f} x_i^2$ | $L_h = \sum_{\mu=1}^{N_h} F(h_\mu)$ <br> $L_x = \sqrt{\sum_{i=1}^{N_f} x_i^2}$ |
| y | $E = -\sum_{\mu=1}^{N_h} F\Big(\sum_{i=1}^{N_f} \xi_{\mu i} V_i\Big)$ | $E = \frac{1}{2}\sum_{i=1}^{N_f} x_i^2 - \log\Big(\sum_{\mu=1}^{N_h} \exp(\sum_{i=1}^{N_f} \xi_{\mu i} x_i)\Big)$ | $E = -\sum_{\mu=1}^{N_h} F\Big(\sum_{i=1}^{N_f} \xi_{\mu i} \frac{x_i}{\sqrt{\sum_j x_j^2}}\Big)$ |
| ve te | $\tau \frac{dx_i}{dt} = \sum_{\mu=1}^{N_h} \xi_{i\mu} f\Big(\sum_{j=1}^{N_f} \xi_{\mu j} V_j\Big) - x_i$ | $\tau \frac{dx_i}{dt} = \sum_{\mu=1}^{N_h} \xi_{i\mu} \mathrm{softmax}\Big(\sum_{j=1}^{N_f} \xi_{\mu j} x_j\Big) - x_i$ | $\tau \frac{dx_i}{dt} = \sum_{\mu=1}^{N_h} \xi_{i\mu} f\Big[\sum_{j=1}^{N_f} \xi_{\mu j} \frac{x_j}{\sqrt{\sum_k x_k^2}}\Big] - x_i$ |

ffective theory on the feature neurons for various common choices of the Lagrangian functions. Model A reduces to the models Krotov & Hopfield 2016] or [Demircigil et al. 2017] depending on the choice of the activation function, model B reduces to the ied in [Ramsauer et al. 2020], model C reduces to the model of [Krotov & Hopfield 2020].

The terms grouped into square brackets represent a Legendre transform of the Lagrangian function with respect to the states of the neurons. If the Hessian matrices of the Lagrangian functions are positive semi-definite, the energy function is guaranteed to decrease on the dynamical trajectory [Krotov & Hopfield 2020]

$$\frac{dE(t)}{dt} = -\tau_f \sum_{i,j=1}^{N_f} \frac{dx_i}{dt} \frac{\partial^2 L_x}{\partial x_i \partial x_j} \frac{dx_j}{dt} - \tau_h \sum_{\mu,\nu=1}^{N_h} \frac{dh_\mu}{dt} \frac{\partial^2 L_h}{\partial h_\mu \partial h_\nu} \frac{dh_\nu}{dt} \leq 0 \qquad (4)$$

This property makes it possible to prove that the system of dynamical equations describing temporal evolution of neurons' activities will eventually reach a fixed point attractor state.

In certain situations one can assume that the dynamics of hidden neurons equilibrates at a much faster time scale compared to the feature neurons, $\tau_h \ll \tau_f$. In this case the steady state solution of the second equation in the system (1) can be used to express the currents of the hidden units through the outputs of the feature neurons. This makes it possible to reduce the general theory (1) to an effective theory for feature neurons only. The resulting effective update rules and the energies for various common choices of the Lagrangian functions are shown in Fig.5. In the case of log-sum-exponential Lagrangian function the update rule (if applied once) for the states of the feature neurons is the attention mechanism [Ramsauer et al. 2020] commonly used in many modern AI systems (see also Ref.[Krotov & Hopfield 2020] for the derivation of this result from the continuous time formulation).

## Relationship to Classical Hopfield Network with Continuous Variables

Classical formulation of continuous Hopfield Networks [Hopfield 1984] can be understood [Krotov & Hopfield 2020] as a special limiting case of the Modern Hopfield Networks with one hidden layer. Continuous Hopfield Networks for neurons with graded response are typically described [Hopfield 1984] by the dynamical equations

$$\tau_f \frac{dx_i}{dt} = \sum_{j=1}^{N_f} T_{ij} V_j - x_i + I_i \qquad (5)$$

and the energy function

$$E = -\frac{1}{2}\sum_{i,j=1}^{N_f} T_{ij} V_i V_j - \sum_{i=1}^{N_f} V_i I_i + \sum_{i=1}^{N_f} \int^{V_i} g^{-1}(z) dz \qquad (6)$$

where $V_i = g(x_i)$, and $g^{-1}(z)$ is the inverse of the activation function $g(x)$. This model is a special limit of the class of models that is called models A [Krotov & Hopfield 2020], with the following choice of the Lagrangian functions

$$L_v = \sum_{i=1}^{N_f} \int^{x_i} g(x) dx, \quad \text{and} \quad L_h = \frac{1}{2}\sum_{\mu=1}^{N_h} h_\mu^2 \qquad (7)$$

that, according to the definition (2), leads to the activation functions

$$V_i = g(x_i), \quad \text{and} \quad f_\mu = h_\mu \qquad (8)$$

If we integrate out the hidden neurons the system of equations (1) reduces to the equations on the feature neurons (5) with $T_{ij} = \sum\limits_{\mu=1}^{N_h} \xi_{\mu i}\xi_{\mu j}$, and the general expression for the energy (3) reduces to the effective energy

$$E = -\frac{1}{2}\sum_{i,j=1}^{N_f} T_{ij} V_i V_j - \sum_{i=1}^{N_f} V_i I_i + \sum_{i=1}^{N_f} \left( x_i V_i - \int^{x_i} g(x) dx \right) \qquad (9)$$

While the first two terms in equation (6) are the same as those in equation (9), the third terms look superficially different. In equation (9) it is a Legendre transform of the Lagrangian for the feature neurons, while in (6) the third term is an integral of the inverse activation function. Nevertheless, these two expressions are in fact equivalent, since the derivatives of a function and its Legendre transform are inverse functions of each other. The easiest way to see that these two terms are equal explicitly is to differentiate each one with respect to $x_i$. The results of these differentiations for both expressions are equal to $x_i g(x_i)'$. Thus, the two expressions are equal up to an additive constant. This completes the proof [Krotov & Hopfield 2020] that the classical Hopfield Network with continuous states [Hopfield 1984] is a special limiting case of the Modern Hopfield Network (1) with energy (3).

## General Formulation of the Modern Hopfield Network

Biological neural networks have a large degree of heterogeneity in terms of different cell types. This section describes a mathematical model of a fully connected Modern Hopfield Network assuming the extreme degree of heterogeneity: every single neuron is different [Krotov 2021]. Specifically, an energy function and the corresponding dynamical equations are described assuming that each neuron has its own activation function and kinetic time scale. The network is assumed to be fully connected, so that every neuron is connected to every other neuron using a symmetric matrix of weights $W_{IJ}$, indices $I$ and $J$ enumerate different neurons in the network, see Fig.6. The easiest way to mathematically formulate this problem is to define the architecture through a Lagrangian function $L(\{x_I\})$ that depends on the activities of all the neurons in the network. The activation function for each neuron is defined as a partial derivative of the Lagrangian with respect to that neuron's activity

$$g_I = \frac{\partial L}{\partial x_I} \qquad (10)$$

From the biological perspective one can think about $g_I$ as an axonal output of the neuron $I$. In the simplest case, when the Lagrangian is additive for different neurons, this definition results in the activation that is a non-linear function of that neuron's activity. For non-additive Lagrangians this activation function can depend on the activities

of a group of neurons. For instance, it can contain contrastive (softmax) or divisive normalization. The dynamical equations describing temporal evolution of a given neuron are given by [Krotov 2021]

$$\tau_I \frac{dx_I}{dt} = \sum_{J=1}^{N} W_{IJ} g_J - x_I \qquad (11)$$

This equation belongs to the class of models called firing rate models in neuroscience. Each neuron $I$ collects the axonal outputs $g_J$ from all the neurons, weights them with the synaptic coefficients $W_{IJ}$ and produces its own time-dependent activity $x_I$. The temporal evolution has a time constant $\tau_I$, which in general can be different for every neuron. This network has a global energy function [Krotov 2021]

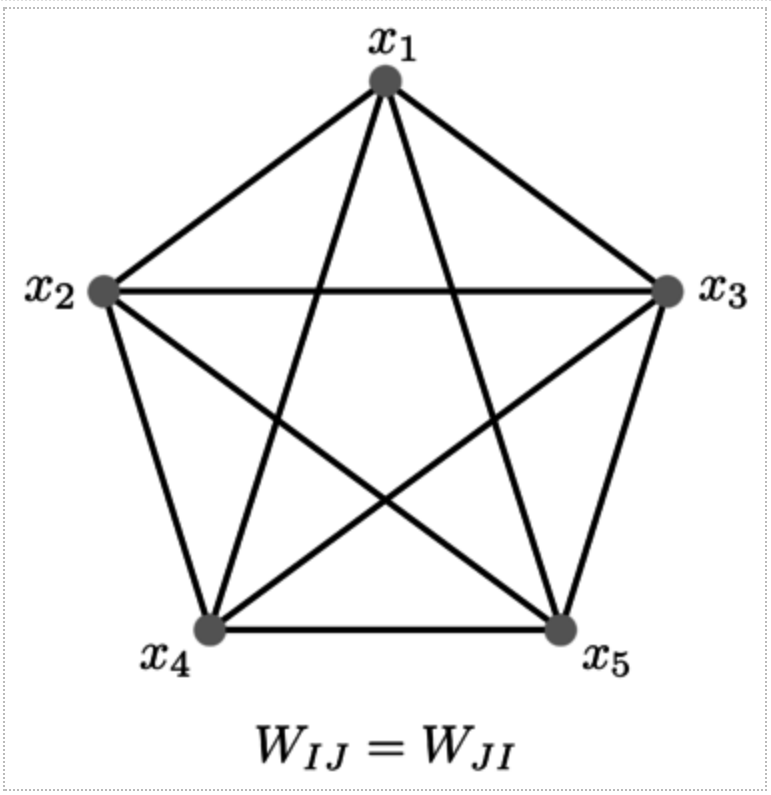


Figure 6: The connectivity diagram of the fully-connected Modern Hopfield Network consisting of five neurons. The synaptic weights are described by a symmetric matrix $W_{IJ}$.

$$E = \sum_{I=1}^{N} x_I g_I - L - \frac{1}{2} \sum_{I,J=1}^{N} g_I W_{IJ} g_J \qquad (12)$$

where the first two terms represent the Legendre transform of the Lagrangian function with respect to the neurons' currents $x_I$. The temporal derivative of this energy function can be computed on the dynamical trajectories leading to (see [Krotov 2021] for details)

$$\frac{dE}{dt} = -\sum_{I,K=1}^{N} \frac{dx_I}{dt} M_{IK} \frac{dx_K}{dt} \leq 0, \quad \text{where} \quad M_{IK} = \tau_I \frac{\partial^2 L}{\partial x_I \partial x_K} \qquad (13)$$

The last inequality sign holds provided that the matrix $M_{IK}$ (or its symmetric part) is positive semi-definite. If, in addition to this, the energy function is bounded from below the non-linear dynamical equations are guaranteed to converge to a fixed point attractor state. The advantage of formulating this network in terms of the Lagrangian functions is that it makes it possible to easily experiment with different choices of the activation functions and different architectural arrangements of neurons. For all those flexible choices the conditions of convergence are determined by the properties of the matrix $M_{IK}$ and the existence of the lower bound on the energy function.

## Hierarchical Associative Memory Network

Neurons can be organized in layers so that every neuron in a given layer has the same activation function and the same dynamic time scale. If we assume that there are no horizontal connections between the neurons within the layer (lateral connections) and there are no skip-layer connections, the general fully connected network (11), (12) reduces to the architecture shown in Fig.7. It has $N_{\text{layer}}$ layers of recurrently connected neurons with the states

described by continuous variables $x_i^A$ and the activation functions $g_i^A$, index $A$ enumerates the layers of the network, and index $i$ enumerates individual neurons in that layer. The activation functions can depend on the activities of all the neurons in the layer. Every layer can have a different number of neurons $N_A$. These neurons are recurrently connected with the neurons in the preceding and the subsequent layers. The matrices of weights that connect neurons in layers $A$ and $B$ are denoted by $\xi_{ij}^{(A,B)}$ (the order of the upper indices for weights is the same as the order of the lower indices, in the example above this means that the index $i$ enumerates neurons in the layer $A$, and index $j$ enumerates neurons in the layer $B$). The feedforward weights and the feedback weights are equal. The dynamical equations for the neurons' states can be written as [Krotov 2021]

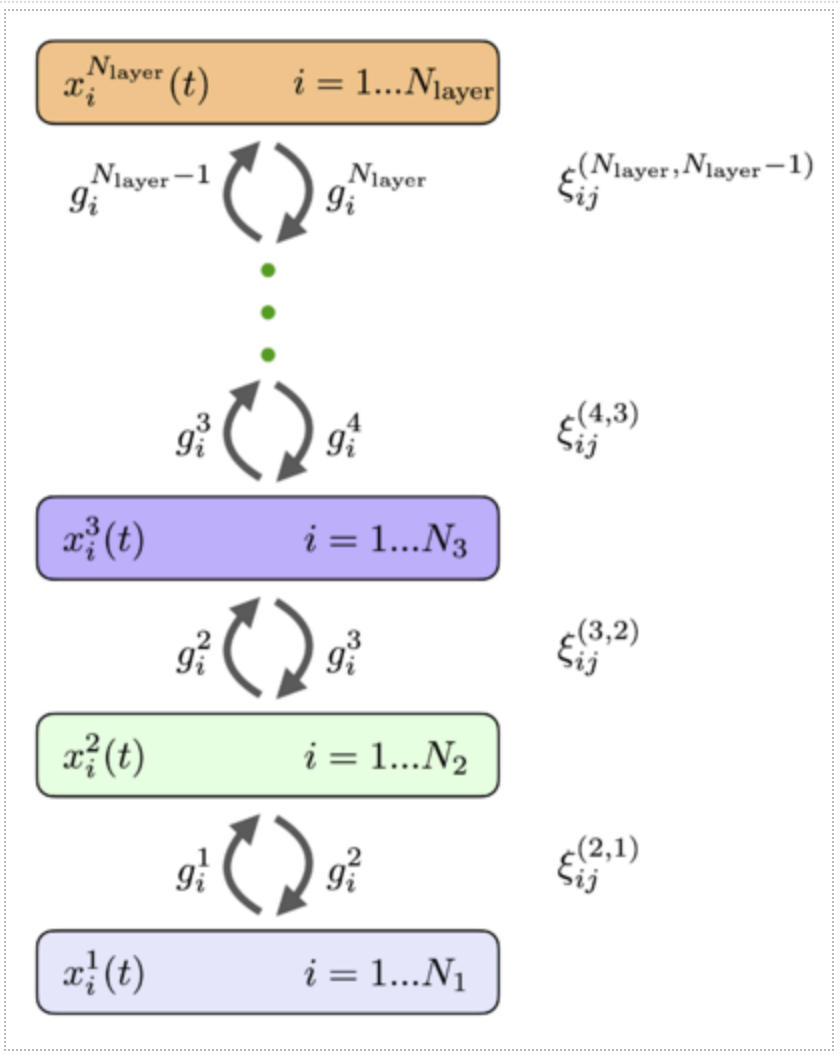


Figure 7: The connectivity diagram of the layered Hierarchical Associative Memory network [Krotov 2021]. Each layer can have different number of neurons, different activation function, and different time scales. The feedforward weights and feedback weights are equal.

$$\tau_A \frac{dx_i^A}{dt} = \sum_{j=1}^{N_{A-1}} \xi_{ij}^{(A,A-1)} g_j^{A-1} + \sum_{j=1}^{N_{A+1}} \xi_{ij}^{(A,A+1)} g_j^{A+1} - x_i^A \qquad (14)$$

with boundary conditions

$$g_i^0 = 0, \quad \text{and} \quad g_i^{N_{\text{layer}}+1} = 0 \qquad (15)$$

The main difference of these equations from the conventional feedforward networks is the presence of the second term, which is responsible for the feedback from higher layers. These top-down signals help neurons in lower layers to decide on their response to the presented stimuli. Following the general recipe it is convenient to introduce a Lagrangian function $L^A(\{x_i^A\})$ for the $A$-th layer, which depends on the activities of all the neurons in that layer [Krotov 2021]. The activation functions in that layer can be defined as partial derivatives of the Lagrangian

$$g_i^A = \frac{\partial L^A}{\partial x_i^A} \qquad (16)$$

With these definitions the energy (Lyapunov) function is given by [Krotov 2021]

$$E = \sum_{A=1}^{N_{\text{layer}}} \Big[ \sum_{i=1}^{N_A} x_i^A g_i^A - L^A \Big] - \sum_{A=1}^{N_{\text{layer}}-1} \sum_{i=1}^{N_{A+1}} \sum_{j=1}^{N_A} g_i^{A+1} \xi_{ij}^{(A+1,A)} g_j^A \qquad (17)$$

If the Lagrangian functions, or equivalently the activation functions, are chosen in such a way that the Hessians for each layer are positive semi-definite and the overall energy is bounded from below, this system is guaranteed to converge to a fixed point attractor state. The temporal derivative of this energy function is given by [Krotov 2021]

$$\frac{dE}{dt} = - \sum_{A=1}^{N_{\text{layer}}} \tau_A \sum_{i,j=1}^{N_A} \frac{dx_j^A}{dt} \frac{\partial^2 L^A}{\partial x_j^A \partial x_i^A} \frac{dx_i^A}{dt} \leq 0 \qquad (18)$$

Thus, the hierarchical layered network is indeed an attractor network with the global energy function. This network is described by a hierarchical set of synaptic weights that can be learned for each specific problem.

## References


Golez, E. and Servet, M. (1990) Neural and Automata Networks, Kluwer Academic Publisher, Boston MA.

Hertz, J., Krogh A., and Palmer, R.G. (1991) Introduction to the theory of neural computation, Addison Wesley, Redwood City CA.

Hopfield, J. J. (1982) Neural networks and physical systems with emergent collective computational properties. Proc. Nat. Acad. Sci. (USA) 79, 2554-2558.

Hopfield, J. J. (1984) Neurons with graded response have collective computational properties like those of two-sate neurons. Proc. Nat. Acad. Sci. (USA) 81, 3088-3092.

Hopfield, J. J. and Tank, D. W. “Neural” computation of decisions in optimization problems. (1985) Biological Cybernetics 55, 141-146.

Hopfield, J. J. (2006) Searching for memories, Sudoku, implicit check-bits, and the iterative use of not-always-correct rapid neural computation. http://arxiv.org/abs/q-bio.NC/0609006

Seung, H. S. (1998) Continuous attractors and oculomotor control, Neural Networks 11, 1253-1258.

Sima, J. and Orponen, P, (2003) Continuous-time symmetric Hopfield nets are computationally universal, Network Computation 15, 693-733.

Krotov, D., and Hopfield, J. J. (2016). Dense associative memory for pattern recognition. Advances in neural information processing systems, 29, 1172-1180 full text (https://arxiv.org/abs/1606.01164) .

Demircigil, M., et al. (2017). On a model of associative memory with huge storage capacity. Journal of Statistical Physics, 168(2), 288-299 full text (https://link.springer.com/article/10.1007/s10955-017-1806-y) .

Ramsauer, H., et al. (2020). Hopfield networks is all you need. International Conference on Learning Representations 2021, arXiv:2008.02217 full text (https://arxiv.org/abs/2008.02217) .

Krotov, D., and Hopfield, J. J. (2020). Large associative memory problem in neurobiology and machine learning. International Conference on Learning Representations 2021, arXiv:2008.06996 full text (https://arxiv.org/abs/2008.06996) .

Krotov, D. (2021). Hierarchical Associative Memory. arXiv:2107.06446 full text (https://arxiv.org/abs/2107.06446)
.

**Internal references**

- John W. Milnor (2006) Attractor. Scholarpedia, 1(11):1815.
- Chris Eliasmith (2007) Attractor network. Scholarpedia, 2(10):1380.
- Geoffrey E. Hinton (2007) Boltzmann machine. Scholarpedia, 2(5):1668.
- James Meiss (2007) Dynamical systems. Scholarpedia, 2(2):1629.
- Mark Aronoff (2007) Language. Scholarpedia, 2(5):3175.
- Peter Jonas and Gyorgy Buzsaki (2007) Neural inhibition. Scholarpedia, 2(9):3286.
- Jeff Moehlis, Kresimir Josic, Eric T. Shea-Brown (2006) Periodic orbit. Scholarpedia, 1(7):1358.
- Philip Holmes and Eric T. Shea-Brown (2006) Stability. Scholarpedia, 1(10):1838.
- Arkady Pikovsky and Michael Rosenblum (2007) Synchronization. Scholarpedia, 2(12):1459.

## External Links

Author's webpage (http://www.genomics.princeton.edu/hopfield/)

## See Also

Associative Memory, Attractor Network, Boltzmann Machine, Brain-State-in-a-Box, Dynamical Systems, Oscillatory Associative Memory, Recurrent Neural Networks, Simulated Annealing

Sponsored by: Eugene M. Izhikevich, Editor-in-Chief of Scholarpedia, the peer-reviewed open-access encyclopedia
Reviewed by (http://www.scholarpedia.org/w/index.php?title=Hopfield_network&oldid=11216) : Anonymous
Accepted on: 2007-04-05 15:23:19 GMT (http://www.scholarpedia.org/w/index.php?title=Hopfield_network&oldid=11216)

Categories: Computational Intelligence | Computational Neuroscience | Recurrent Neural Networks | Neural Networks | Eponymous