21 Hopfield Network

Objectives

This chapter will discuss the Hopfield recurrent neural network — a network that was highly influential in bringing about the resurgence of neural network research in the early 1980s. We will begin with a description of the network, and then we will show how Lyapunov stability theory can be used to analyze the network operation. Finally, we will demonstrate how the network can be designed to behave as an associative memory.

This chapter brings together many topics discussed in previous chapters: the discrete-time Hopfield network (Chapter 3), eigenvalues and eigenvectors (Chapter 6); associative memory and the Hebb rule (Chapter 7); Hessian matrices, conditions for optimality, quadratic functions and surface and contour plots (Chapter 8); steepest descent and phase plane trajectories (Chapter 9); continuous-time recurrent networks (Chapter 18); and Lyapunov's Stability Theorem and LaSalle's Invariance Theorem (Chapter 20). This chapter is, in some ways, a culmination of all our previous efforts.

Theory and Examples

Much of the resurgence of interest in neural networks during the early 1980s can be attributed to the work of John Hopfield. As a well-known Cal. Tech. physicist, Hopfield's visibility and scientific credentials lent renewed credibility to the neural network field, which had been tarnished by the hype of the mid-1960s. Early in his career he studied the interaction between light and solids. Later he focused on the mechanism of electron transfer between biological molecules. One can imagine that his academic study in physics and mathematics, combined with his later experiences in biology, prepared him uniquely for the conception and presentation of his neural network contribution.

Hopfield wrote two highly influential papers in 1982 [Hopf82] and 1984 [Hopf84]. Many of the ideas in these papers were based on the previous work of other researchers, such as the neuron model of McCulloch and Pitts [McPi43], the additive model of Grossberg [Gros67], the linear associator of Anderson [Ande72] and Kohonen [Koho72] and the Brain-State-in-a-Box network of Anderson, Silverstein, Ritz and Jones [AnSi77]. However, Hopfield's papers are very readable, and they bring together a number of important ideas and present them with a clear mathematical analysis (including the application of Lyapunov stability theory).

There are several other reasons why Hopfield's papers have had such an impact. First, he identified a close analogy between his neural network and the Ising model of magnetic materials, which is used in statistical physics. This brought a significant amount of existing theory to bear on the analysis of neural networks, and it encouraged many physicists, as well as other scientists and engineers, to turn their attention to neural network research.

Hopfield also had close contacts with VLSI chip designers, because of his long association with AT&T Bell Laboratories. As early as 1987, Bell Labs had successfully developed neural network chips based on the Hopfield network. One of the main promises of neural networks is their suitability for parallel implementation in VLSI and optical devices. The fact that Hopfield addressed the implementation issues of his networks distinguished him from most previous neural network researchers.

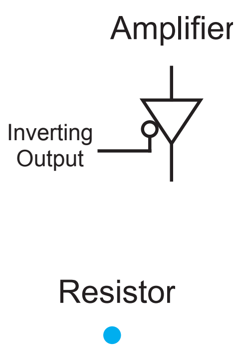
Hopfield emphasized practicality, both in the implementation of his networks and in the types of problems they solved. Some of the applications that he described in his early papers include content-addressable memory (which we will discuss later in this chapter), analog-to-digital conversion [TaHo86], and optimization [HoTa85] (as in the traveling salesman problem).

In the next section we will present the Hopfield model. We will use the continuous-time model from the 1984 paper [Hopf84]. Then we will apply Lyapunov stability theory and LaSalle's Invariance Theorem to the analysis of the Hopfield model. In the final section we will demonstrate how the Hebb rule can be used to design Hopfield networks as content-addressable memories.

Hopfield Model

Hopfield Model

In keeping with his practical viewpoint, Hopfield presented his model as an electrical circuit. The basic *Hopfield model* (see [Hopf84]) is shown in Figure 21.1.



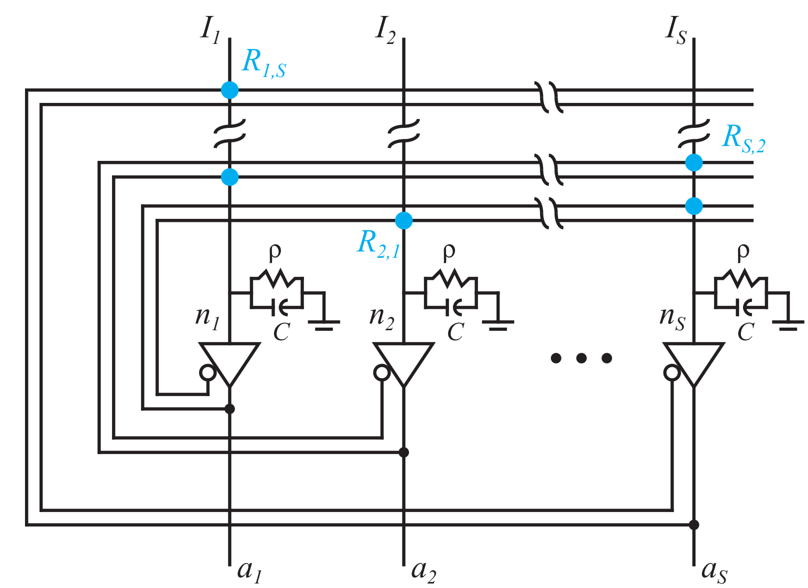


Figure 21.1 Hopfield Model

Each neuron is represented by an operational amplifier and its associated resistor/capacitor network. There are two sets of inputs to the neurons. The first set, represented by the currents $I_1, I_2, \ldots$, are constant external inputs. The other set consists of feedback connections from other op-amps. For instance, the second output, a_2, is fed to resistor $R_{S,2}$, which is connected, in turn, to the input of amplifier S. Resistors are, of course, only positive, but a negative input to a neuron can be obtained by selecting the inverted output of a particular amplifier. (In Figure 21.1, the inverting output of the first amplifier is connected to the input of the second amplifier through resistor $R_{2,1}$.)

The equation of operation for the Hopfield model, derived using Kirchhoff's current law, is

$$C\frac{dn_i(t)}{dt} = \sum_{j=1}^{S} T_{i,j}a_j(t) - \frac{n_i(t)}{R_i} + I_i, \tag{21.1}$$

where n_i is the input voltage to the ith amplifier, a_i is the output voltage of the ith amplifier, C is the amplifier input capacitance and I_i is a fixed input current to the ith amplifier. Also,

$$\left|T_{i,j}\right| = \frac{1}{R_{i,j}},\ \frac{1}{R_i} = \frac{1}{\rho} + \sum_{j=1}^{S} \frac{1}{R_{i,j}},\ n_i = f^{-1}(a_i)\ (\text{or } a_i = f(n_i)), \tag{21.2}$$

where $f(n)$ is the amplifier characteristic. Here and in what follows we will assume that the circuit is symmetric, so that $T_{i,j} = T_{j,i}$.

The amplifier transfer function, $a_i = f(n_i)$, is ordinarily a sigmoid function. Both this sigmoid function and its inverse are assumed to be increasing functions. We will provide a specific example of a suitable transfer function later in this chapter.

If we multiply both sides of Eq. (21.1) by R_i, we obtain

$$R_i C \frac{dn_i(t)}{dt} = \sum_{j=1}^{S} R_i T_{i,j} a_j(t) - n_i(t) + R_i I_i . \tag{21.3}$$

This can be transformed into our standard neural network notation if we define

$$\varepsilon = R_i C,\ w_{i,j} = R_i T_{i,j} \text{ and } b_i = R_i I_i . \tag{21.4}$$

Now Eq. (21.3) can be rewritten as

$$\varepsilon \frac{dn_i(t)}{dt} = -n_i(t) + \sum_{j=1}^{S} w_{i,j} a_j(t) + b_i . \tag{21.5}$$

In vector form we have

$$\varepsilon \frac{d\mathbf{n}(t)}{dt} = -\mathbf{n}(t) + \mathbf{W}\mathbf{a}(t) + \mathbf{b} . \tag{21.6}$$

and

$$\mathbf{a}(t) = \mathbf{f}(\mathbf{n}(t)) . \tag{21.7}$$

The resulting Hopfield network is displayed in Figure 21.2.

Thus, Hopfield's original network of S operational amplifier circuits can be represented conveniently in our standard network notation. Note that the input vector $\mathbf{p}$ determines the initial network output. This form of the Hopfield network is used for associative memory networks, as will be discussed at the end of this chapter.

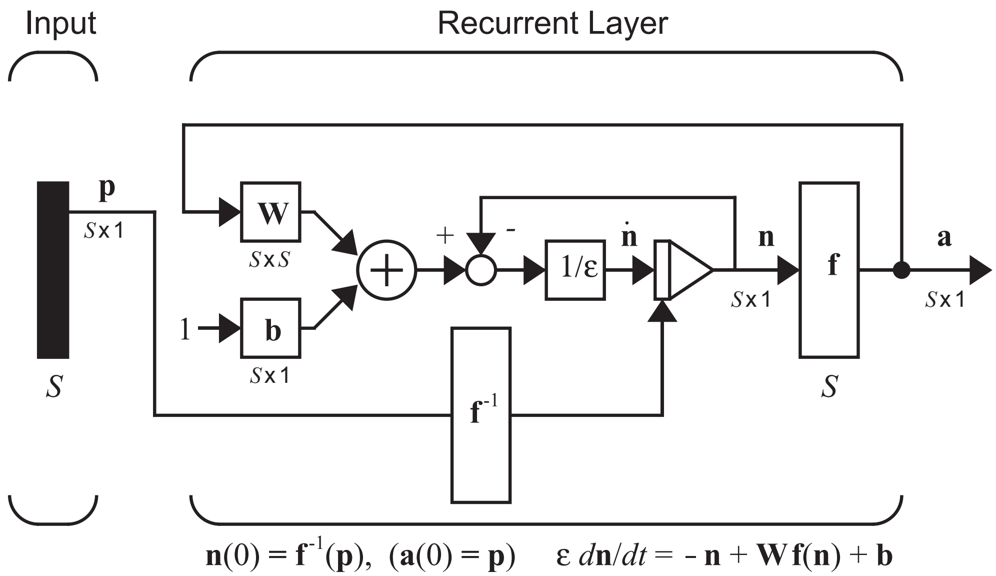


Figure 21.2 Hopfield Network

Lyapunov Function

The application of Lyapunov stability theory to the analysis of recurrent networks was one of the key contributions of Hopfield. (Cohen and Grossberg also used Lyapunov theory for the analysis of competitive networks at about the same time [CoGr83].) In this section we will demonstrate how LaSalle's Invariance Theorem, which was presented in Chapter 20, can be used with the Hopfield network. The first step in using LaSalle's theorem is to choose a Lyapunov function. Hopfield suggested the following function:

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} + \sum_{i=1}^{S}\left\{\int_0^{a_i} f^{-1}(u)\,du\right\} - \mathbf{b}^T\mathbf{a} . \tag{21.8}$$

Hopfield's choice of this particular Lyapunov candidate is one of his key contributions. Notice that the first and third terms make up a quadratic function. In a later section of this chapter we will use our previous results on quadratic functions to help develop some insight into this Lyapunov function.

To use LaSalle's theorem, we will need to evaluate the derivative of $V(\mathbf{a})$. For clarity, we will consider each of the three terms of $V(\mathbf{a})$ separately. Using Eq. (8.37), the derivative of the first term is

$$\frac{d}{dt}\left\{-\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a}\right\} = -\frac{1}{2}\nabla[\mathbf{a}^T\mathbf{W}\mathbf{a}]^T\frac{d\mathbf{a}}{dt} = -[\mathbf{W}\mathbf{a}]^T\frac{d\mathbf{a}}{dt} = -\mathbf{a}^T\mathbf{W}\frac{d\mathbf{a}}{dt}. \quad (21.9)$$

The second term in $V(\mathbf{a})$ consists of a sum of integrals. If we consider one of these integrals, we find

$$\frac{d}{dt}\left\{\int_0^{a_i} f^{-1}(u)du\right\} = \frac{d}{da_i}\left\{\int_0^{a_i} f^{-1}(u)du\right\}\frac{da_i}{dt} = f^{-1}(a_i)\frac{da_i}{dt} = n_i\frac{da_i}{dt}. \quad (21.10)$$

The total derivative of the second term in $V(\mathbf{a})$ is then

$$\frac{d}{dt}\left[\sum_{i=1}^{S}\left\{\int_0^{a_i} f^{-1}(u)du\right\}\right] = \mathbf{n}^T\frac{d\mathbf{a}}{dt}. \quad (21.11)$$

Using Eq. (8.36), we can find the derivative of the third term in $V(\mathbf{a})$.

$$\frac{d}{dt}\{-\mathbf{b}^T\mathbf{a}\} = -\nabla[\mathbf{b}^T\mathbf{a}]^T\frac{d\mathbf{a}}{dt} = -\mathbf{b}^T\frac{d\mathbf{a}}{dt} \quad (21.12)$$

Therefore, the total derivative of $V(\mathbf{a})$ is

$$\frac{d}{dt}V(\mathbf{a}) = -\mathbf{a}^T\mathbf{W}\frac{d\mathbf{a}}{dt} + \mathbf{n}^T\frac{d\mathbf{a}}{dt} - \mathbf{b}^T\frac{d\mathbf{a}}{dt} = [-\mathbf{a}^T\mathbf{W} + \mathbf{n}^T - \mathbf{b}^T]\frac{d\mathbf{a}}{dt}. \quad (21.13)$$

From Eq. (21.6) we know that

$$[-\mathbf{a}^T\mathbf{W} + \mathbf{n}^T - \mathbf{b}^T] = -\varepsilon\left[\frac{d\mathbf{n}(t)}{dt}\right]^T. \quad (21.14)$$

This allows us to rewrite Eq. (21.13) as

$$\frac{d}{dt}V(\mathbf{a}) = -\varepsilon\left[\frac{d\mathbf{n}(t)}{dt}\right]^T\frac{d\mathbf{a}}{dt} = -\varepsilon\sum_{i=1}^{S}\left(\frac{dn_i}{dt}\right)\left(\frac{da_i}{dt}\right). \quad (21.15)$$

Since $n_i = f^{-1}(a_i)$, we can expand the derivative of n_i as follows:

$$\frac{dn_i}{dt} = \frac{d}{dt}[f^{-1}(a_i)] = \frac{d}{da_i}[f^{-1}(a_i)]\frac{da_i}{dt}. \quad (21.16)$$

Now Eq. (21.15) can be rewritten

$$\frac{d}{dt}V(\mathbf{a}) = -\varepsilon \sum_{i=1}^{S} \left(\frac{dn_i}{dt}\right)\left(\frac{da_i}{dt}\right) = -\varepsilon \sum_{i=1}^{S} \left(\frac{d}{da_i}[f^{-1}(a_i)]\right)\left(\frac{da_i}{dt}\right)^2 . \tag{21.17}$$

If we assume that $f^{-1}(a_i)$ is an increasing function, as it would be for an operational amplifier, then

$$\frac{d}{da_i}[f^{-1}(a_i)] > 0 . \tag{21.18}$$

From Eq. (21.17), this implies that

$$\frac{d}{dt}V(\mathbf{a}) \le 0 . \tag{21.19}$$

Thus, if $f^{-1}(a_i)$ is an increasing function, $dV(\mathbf{a})/dt$ is a negative semidefinite function. Therefore, $V(\mathbf{a})$ is a valid Lyapunov function.

Invariant Sets

Now we want to apply LaSalle's Invariance Theorem to determine equilibrium points for the Hopfield network. The first step is to find the set Z (Eq. (20.19)).

$$Z = \{\mathbf{a}\text{: } dV(\mathbf{a})/dt = 0,\ \mathbf{a} \text{ in the closure of } G\} \tag{21.20}$$

This set includes all points at which the derivative of the Lyapunov function is zero. For now, let's assume that G is all of $\Re^S$.

We can see from Eq. (21.17) that such derivatives will be zero if the derivatives of all of the neuron outputs are zero.

$$\frac{d\mathbf{a}}{dt} = \mathbf{0} \tag{21.21}$$

However, when the derivatives of the outputs are zero, the circuit is at equilibrium. Thus, those points where the system "energy" is not changing are also points where the circuit is at equilibrium.

This means that the set L, the largest invariant set in Z, is exactly equal to Z.

$$L = Z \tag{21.22}$$

Thus, all points in Z are potential attractors.

Some of these features will be illustrated in the following example.

Example

Consider the following example from Hopfield's original paper [Hopf84]. We will examine a system having an amplifier characteristic

$$a = f(n) = \frac{2}{\pi}\tan^{-1}\left(\frac{\gamma\pi n}{2}\right) . \tag{21.23}$$

We can also write this expression as

$$n = \frac{2}{\gamma\pi}\tan\left(\frac{\pi}{2}a\right) . \tag{21.24}$$

Assume two amplifiers, with the output of each connected to the input of the other through a unit resistor, so that

$$R_{1,2} = R_{2,1} = 1 \text{ and } T_{1,2} = T_{2,1} = 1 . \tag{21.25}$$

Thus we have a weight matrix

$$\mathbf{W} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} . \tag{21.26}$$

If the amplifier input capacitance is also set to 1, we have

$$\varepsilon = R_i C = 1 . \tag{21.27}$$

Let us also take $\gamma = 1.4$ and $I_1 = I_2 = 0$. Therefore

$$\mathbf{b} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} . \tag{21.28}$$

Recall from Eq. (21.8) that the Lyapunov function is

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} + \sum_{i=1}^{S}\left\{\int_0^{a_i} f^{-1}(u)du\right\} - \mathbf{b}^T\mathbf{a} . \tag{21.29}$$

The first term of the Lyapunov function, for this example, is

$$-\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} = -\frac{1}{2}\begin{bmatrix} a_1 & a_2 \end{bmatrix}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} a_1 \\ a_2 \end{bmatrix} = -a_1 a_2 . \tag{21.30}$$

The third term is zero, because the biases are zero. The ith part of the second term is

$$\int_0^{a_i} f^{-1}(u)du = \frac{2}{\gamma\pi}\int_0^{a_i} \tan\left(\frac{\pi}{2}u\right)du = \frac{2}{\gamma\pi}\left[-\log\left[\cos\left(\frac{\pi}{2}u\right)\right]\frac{2}{\pi}\right]_0^{a_i} . \tag{21.31}$$

This expression can be simplified to

$$\int_0^{a_i} f^{-1}(u)du = -\frac{4}{\gamma\pi^2}\log\left[\cos\left(\frac{\pi}{2}a_i\right)\right]. \tag{21.32}$$

Finally, substituting all three terms into Eq. (21.29), we have our Lyapunov function:

$$V(\mathbf{a}) = -a_1 a_2 - \frac{4}{1.4\pi^2}\left[\log\left\{\cos\left(\frac{\pi}{2}a_1\right)\right\} + \log\left\{\cos\left(\frac{\pi}{2}a_2\right)\right\}\right]. \tag{21.33}$$

Now let's write out the network equation (Eq. (21.6)). With $\varepsilon = 1$ and $\mathbf{b} = \mathbf{0}$, it is

$$\frac{d\mathbf{n}}{dt} = -\mathbf{n} + \mathbf{W}\mathbf{f}(\mathbf{n}) = -\mathbf{n} + \mathbf{W}\mathbf{a}. \tag{21.34}$$

If we substitute the weight matrix of Eq. (21.26), this expression can be written as the following pair of equations:

$$dn_1/dt = a_2 - n_1, \tag{21.35}$$

$$dn_2/dt = a_1 - n_2. \tag{21.36}$$

The neuron outputs are

$$a_1 = \frac{2}{\pi}\tan^{-1}\left(\frac{1.4\pi}{2}n_1\right), \tag{21.37}$$

$$a_2 = \frac{2}{\pi}\tan^{-1}\left(\frac{1.4\pi}{2}n_2\right). \tag{21.38}$$

Now that we have found expressions for the system Lyapunov function and the network equation of operation, let's investigate the network behavior. The Lyapunov function contour and a sample trajectory are shown in Figure 21.3.

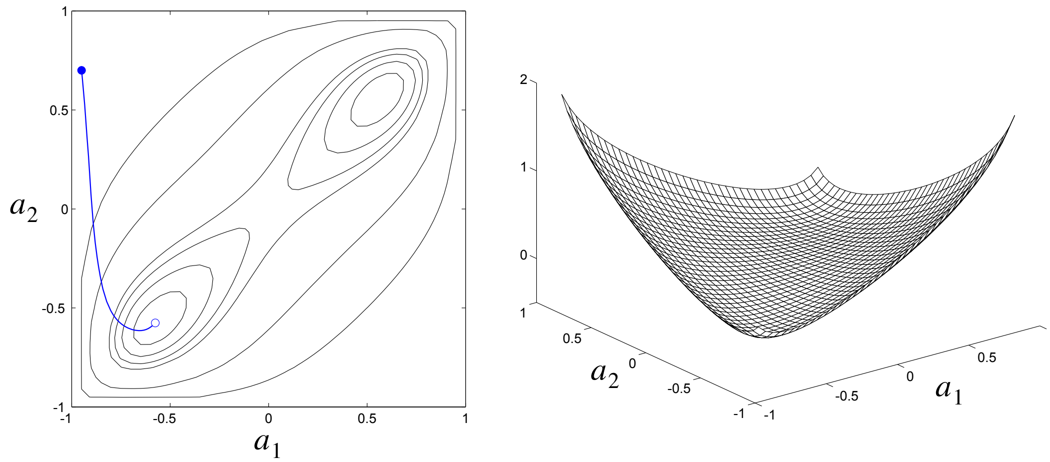


Figure 21.3 Hopfield Example Lyapunov Function and Trajectory

The contour lines in this figure represent constant values of the Lyapunov function. The system has two attractors, one in the lower left and one in the upper right of Figure 21.3. Starting from the upper left, the system converges, as shown by the blue line, to the stable point at the lower left.

Figure 21.4 displays the time response of the two neuron outputs.

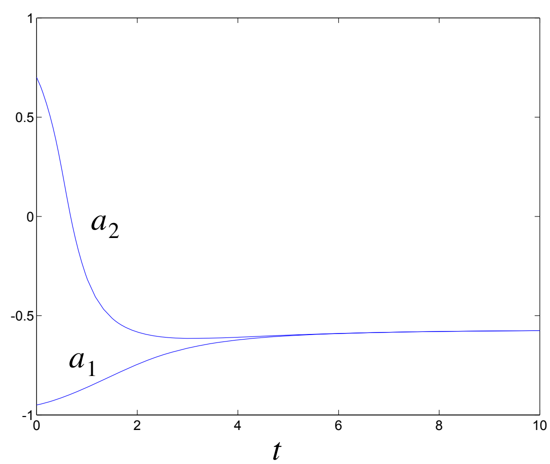


Figure 21.4 Hopfield Example Time Response

Figure 21.5 displays the time response of the Lyapunov function. As expected, it decreases continuously as the equilibrium point is approached.

The system also has an equilibrium point at the origin. If the network is initialized anywhere on a diagonal line drawn from the upper-left corner to the lower-right corner, the solution converges to the origin. Any initial conditions that do not fall on this line, however, will converge to one of the solutions in the lower-left or upper-right corner. The solution at the origin is a saddle point of the Lyapunov function, not a local minimum. We will discuss this problem in a later section. Figure 21.6 displays a trajectory that converges to the saddle point.

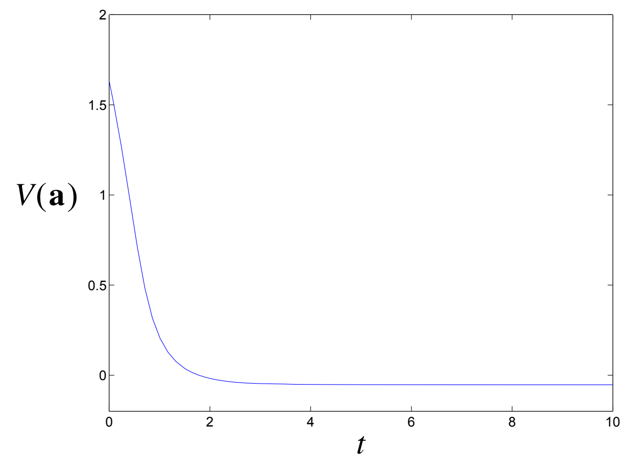


Figure 21.5 Lyapunov Function Response

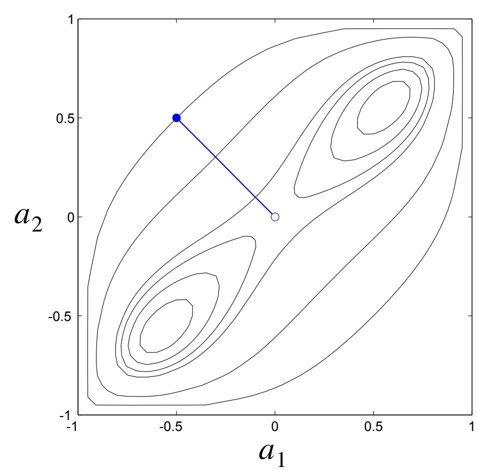


Figure 21.6 Hopfield Convergence to a Saddle Point

To experiment with the Hopfield network, use the Neural Network Design Demonstration Hopfield Network (`nnd18hn`).

This example has provided some insight into the Hopfield attractors. In the next section we will analyze them more carefully.

Hopfield Attractors

In the example network in the previous section we found that the Hopfield network attractors were stationary points of the Lyapunov function. Now we want to show that this is true in the general case. Recall from Eq. (21.21) that the potential attractors of the Hopfield network satisfy

$$\frac{d\mathbf{a}}{dt} = \mathbf{0} . \tag{21.39}$$

How are these points related to the minima of the Lyapunov function $V(\mathbf{a})$? In Chapter 8 (Eq. (8.27)) we showed that the minima of a function must be stationary points (i.e., gradient equal to zero). The stationary points of $V(\mathbf{a})$ will satisfy

$$\nabla V = \left[\frac{\partial V}{\partial a_1} \; \frac{\partial V}{\partial a_2} \; \ldots \; \frac{\partial V}{\partial a_S}\right]^T = \mathbf{0} , \tag{21.40}$$

where

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} + \sum_{i=1}^{S}\left\{\int_0^{a_i} f^{-1}(u)du\right\} - \mathbf{b}^T\mathbf{a} . \tag{21.41}$$

If we follow steps similar to those we used to derive Eq. (21.13), we can find the following expression for the gradient:

$$\nabla V(\mathbf{a}) = [-\mathbf{W}\mathbf{a} + \mathbf{n} - \mathbf{b}] = -\varepsilon\left[\frac{d\mathbf{n}(t)}{dt}\right] . \tag{21.42}$$

The ith element of the gradient is therefore

$$\frac{\partial}{\partial a_i}V(\mathbf{a}) = -\varepsilon\frac{dn_i}{dt} = -\varepsilon\frac{d}{dt}([f^{-1}(a_i)]) = -\varepsilon\frac{d}{da_i}[f^{-1}(a_i)]\frac{da_i}{dt} . \tag{21.43}$$

Notice, incidentally, that if $f^{-1}(a)$ is linear, Eq. (21.43) implies that

$$\frac{d\mathbf{a}}{dt} = -\alpha\nabla V(\mathbf{a}) . \tag{21.44}$$

Therefore, the response of the Hopfield network is steepest descent. Thus, if you are in a region where $f^{-1}(\mathbf{a})$ is approximately linear, the network solution approximates steepest descent.

We have assumed that the transfer function and its inverse are monotonic increasing functions. Therefore,

$$\frac{d}{da_i}[f^{-1}(a_i)] > 0 . \tag{21.45}$$

From Eq. (21.43), this implies that those points for which

$$\frac{d\mathbf{a}(t)}{dt} = \mathbf{0}, \tag{21.46}$$

will also be points where

$$\nabla V(\mathbf{a}) = \mathbf{0}. \tag{21.47}$$

Therefore, the attractors, which are members of the set L and satisfy Eq. (21.39), will also be stationary points of the Lyapunov function $V(\mathbf{a})$.

Effect of Gain

The Hopfield Lyapunov function can be simplified if we consider those cases where the amplifier gain γ is large. Recall that the nonlinear amplifier characteristic for our previous example was

$$a = f(n) = \frac{2}{\pi}\tan^{-1}\left(\frac{\gamma\pi n}{2}\right). \tag{21.48}$$

This function is displayed in Figure 21.7 for four different gain values.

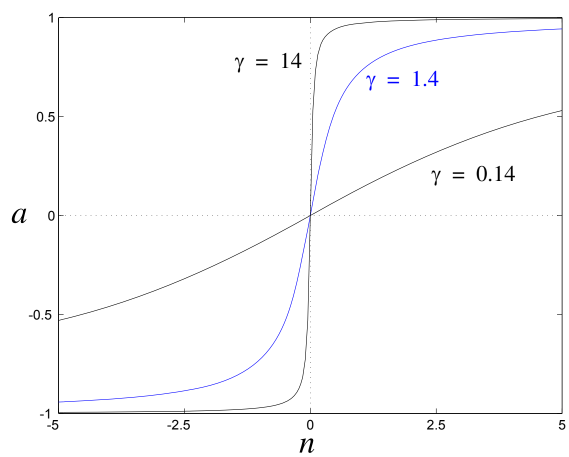


Figure 21.7 Inverse Tangent Amplifier Characteristic

The gain γ determines the steepness of the curve at $n = 0$. As γ increases, the slope of the curve at the origin increases. As γ goes to infinity, $f(n)$ approaches a signum (step) function.

Now recall from Eq. (21.8) that the general Lyapunov function is

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} + \sum_{i=1}^{S}\left\{\int_0^{a_i} f^{-1}(u)du\right\} - \mathbf{b}^T\mathbf{a}. \tag{21.49}$$

For our previous example,

$$f^{-1}(u) = \frac{2}{\gamma\pi}\tan\left(\frac{\pi u}{2}\right). \tag{21.50}$$

Therefore, the second term in the Lyapunov function takes the form

$$\int_0^{a_i} f^{-1}(u)du = \frac{2}{\gamma\pi}\left[\frac{2}{\pi}\log\left(\cos\left(\frac{\pi a_i}{2}\right)\right)\right] = -\frac{4}{\gamma\pi^2}\log\left[\cos\left(\frac{\pi a_i}{2}\right)\right]. \tag{21.51}$$

A graph of this function is shown in Figure 21.8 for three different values of the gain. Note that as γ increases the function flattens and is close to 0 most of the time. Thus, as the gain γ goes to infinity, the integral in the second term of the Lyapunov function will be close to zero in the range $-1 < a_i < 1$. This allows us to eliminate that term, and the *high-gain Lyapunov function* then reduces to

High-Gain Lyapunov Function

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} - \mathbf{b}^T\mathbf{a}. \tag{21.52}$$

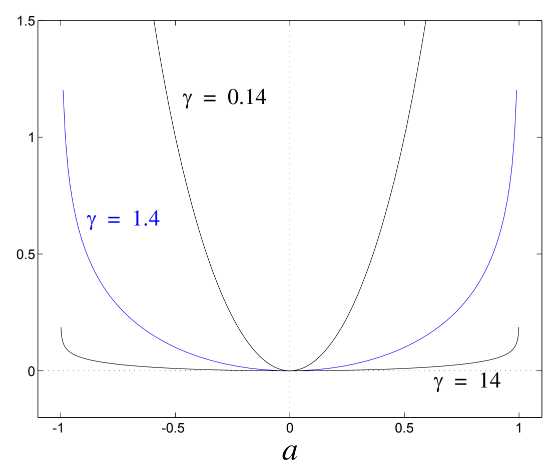


Figure 21.8 Second Term in the Lyapunov Function

By comparing Eq. (21.52) with Eq. (8.35), we can see that the high-gain Lyapunov function is, in fact, a quadratic function:

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} - \mathbf{b}^T\mathbf{a} = \frac{1}{2}\mathbf{a}^T\mathbf{A}\mathbf{a} + \mathbf{d}^T\mathbf{a} + c, \tag{21.53}$$

where

$$\nabla^2 V(\mathbf{a}) = \mathbf{A} = -\mathbf{W},\ \mathbf{d} = -\mathbf{b} \text{ and } c = 0. \tag{21.54}$$

This is an important development, for now we can apply our results from Chapter 8 on quadratic functions to the understanding of the operation of Hopfield networks.

Recall that the shape of the surface of a quadratic function is determined by the eigenvalues and eigenvectors of its Hessian matrix. The Hessian matrix for our example Lyapunov function is

$$\nabla^2 V(\mathbf{a}) = -\mathbf{W} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}. \tag{21.55}$$

The eigenvalues of this Hessian matrix are computed as follows:

$$\left|\nabla^2 V(\mathbf{a}) - \lambda \mathbf{I}\right| = \begin{vmatrix} -\lambda & -1 \\ -1 & -\lambda \end{vmatrix} = \lambda^2 - 1 = (\lambda + 1)(\lambda - 1). \tag{21.56}$$

Thus, the eigenvalues are $\lambda_1 = -1$ and $\lambda_2 = 1$. It follows that the eigenvectors are

$$\mathbf{z}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ and } \mathbf{z}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \tag{21.57}$$

What does the surface of the high-gain Lyapunov function look like? We know, since the Hessian matrix has one positive and one negative eigenvalue, that we have a saddle point condition. The surface will have a negative curvature along the first eigenvector and a positive curvature along the second eigenvector. The surface is shown in Figure 21.9.

The function does not have a minimum. However, the network is constrained to the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$ by the amplifier transfer function. Therefore, there will be constrained minima at the two corners of the hypercube

$$\mathbf{a} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ and } \mathbf{a} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}. \tag{21.58}$$

When the gain is very small, there is a single minimum at the origin (see Exercise E21.1). As the gain is increased, two minima move out from the origin toward the two corners given by Eq. (21.58). Figure 21.3 displays an intermediate case, where the gain is $\gamma = 1.4$. The minima in that figure occur at

$$\mathbf{a} = \begin{bmatrix} 0.57 \\ 0.57 \end{bmatrix} \text{ and } \mathbf{a} = \begin{bmatrix} -0.57 \\ -0.57 \end{bmatrix}. \tag{21.59}$$

In the general case, where there are more than two neurons in the network, the high-gain minima will fall in certain corners of the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$. We will discuss the general case in more detail in later sections, after we describe the Hopfield design process.

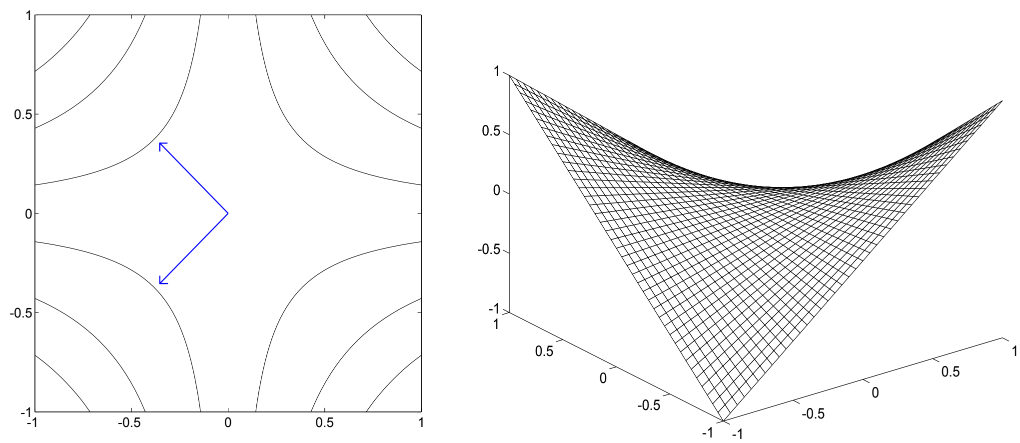


Figure 21.9 Example High Gain Lyapunov Function

Hopfield Design

The Hopfield network does not have a learning law associated with it. It is not trained, nor does it learn on its own. Instead, a design procedure based on the Lyapunov function is used to determine the weight matrix.

Consider again the high-gain Lyapunov function

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} - \mathbf{b}^T\mathbf{a}. \qquad (21.60)$$

The Hopfield design technique is to choose the weight matrix $\mathbf{W}$ and the bias vector $\mathbf{b}$ so that V takes on the form of a function that you want to minimize. Convert whatever problem you want to solve into a quadratic minimization problem. Since the Hopfield network will minimize V, it will also solve the original problem. The trick, of course, is in the conversion, which is generally not straightforward.

Content-Addressable Memory

Content-Addressable Memory

In this section we will describe how a Hopfield network can be designed to work as an associative memory. The type of associative memory we will design is called a *content-addressable memory*, because it retrieves stored memories on the basis of part of the contents. This is in contrast to standard computer memories, where items are retrieved based on their addresses. For example, if we have a content-addressable data base that contains names, addresses and phone numbers of employees, we can retrieve a complete data entry simply by providing the employee name (or perhaps a partial name). The content-addressable memory is effectively the same as the autoassociative memory described in Chapter 7 (see page 7-10), except that in this chapter we will use the recurrent Hopfield network instead of the linear associator.

Suppose that we want to store a set of prototype patterns in a Hopfield network. When an input pattern is presented to the network, the network should produce the stored pattern that most closely resembles the input pattern. The initial network output is assigned to the input pattern. The network output should then converge to the prototype pattern closest to the input pattern. For this to happen, the prototype patterns must be minima of the Lyapunov function.

Let's assume that the prototype patterns are

$$\{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_Q\}. \tag{21.61}$$

Each of these vectors consists of S elements, having the values 1 or -1. Assume further that $Q \ll S$, so that the state space is large, and that the prototype patterns are well distributed in this space, and so will not be close to each other.

In order for a Hopfield network to be able to recall the prototype patterns, the patterns must be minima of the Lyapunov function. Since the high-gain Lyapunov function is quadratic, we need the prototype patterns to be (constrained) minima of an appropriate quadratic function. We propose the following quadratic performance index:

$$J(\mathbf{a}) = -\frac{1}{2}\sum_{q=1}^{Q}([\mathbf{p}_q]^T\mathbf{a})^2. \tag{21.62}$$

If the elements of the vectors $\mathbf{a}$ are restricted to be ± 1, this function is minimized at the prototype patterns, as we will now show.

Assume that the prototype patterns are orthogonal. If we evaluate the performance index at one of the prototype patterns, we find

$$J(\mathbf{p}_j) = -\frac{1}{2}\sum_{q=1}^{Q}([\mathbf{p}_q]^T\mathbf{p}_j)^2 = -\frac{1}{2}([\mathbf{p}_j]^T\mathbf{p}_j)^2 = -\frac{S}{2}. \tag{21.63}$$

The second equality follows from the orthogonality of the prototype patterns. The last equality follows because all elements of $\mathbf{p}_j$ are ± 1.

Next, evaluate the performance index at a random input pattern $\mathbf{a}$, which is presumably not close to any prototype pattern. Each element in the sum in Eq. (21.62) is an inner product between a prototype pattern and the input. The inner product will increase as the input moves closer to a prototype pattern. However, if the input is not close to any prototype pattern, then all terms of the sum in Eq. (21.62) will be small. Therefore, $J(\mathbf{a})$ will be largest (least negative) when $\mathbf{a}$ is not close to any prototype pattern, and will be smallest (most negative) when $\mathbf{a}$ is equal to any one of the prototype patterns.

We have now found a quadratic function that accurately indicates the performance of the content-addressable memory. The next step is to choose the weight matrix $\mathbf{W}$ and bias $\mathbf{b}$ so that the Hopfield Lyapunov function V will be equivalent to the quadratic performance index J.

If we use the supervised Hebb rule to compute the weight matrix (with target patterns being the same as input patterns) as

$$\mathbf{W} = \sum_{q=1}^{Q} \mathbf{p}_q(\mathbf{p}_q)^T, \tag{21.64}$$

and set the bias to zero

$$\mathbf{b} = \mathbf{0}, \tag{21.65}$$

then the Lyapunov function is

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} = -\frac{1}{2}\mathbf{a}^T\left[\sum_{q=1}^{Q} \mathbf{p}_q(\mathbf{p}_q)^T\right]\mathbf{a} = -\frac{1}{2}\sum_{q=1}^{Q} \mathbf{a}^T\mathbf{p}_q(\mathbf{p}_q)^T\mathbf{a}. \tag{21.66}$$

This can be rewritten

$$V(\mathbf{a}) = -\frac{1}{2}\sum_{q=1}^{Q} [(\mathbf{p}_q)^T\mathbf{a}]^2 = J(\mathbf{a}). \tag{21.67}$$

Therefore, the Lyapunov function is indeed equal to the quadratic performance index for the content-addressable memory problem. The Hopfield network output will tend to converge to the stored prototype patterns (among other possible equilibrium points, as we will discuss later).

As noted in Chapter 7, the supervised Hebb rule does not work well if there is significant correlation between the prototype patterns. In that case the pseudoinverse technique has been suggested. Another design technique, which is beyond the scope of this text, is given in [LiMi89].

In the best situation, where the prototype patterns are orthogonal, every prototype pattern will be an equilibrium point of the network. However, there will be many other equilibrium points as well. The network may well converge to a pattern that is not one of the prototype patterns. A general rule is that, when using the Hebb rule, the number of stored patterns can be no more than 15% of the number of neurons.The reference [LiMi89] discusses more complex design procedures, which minimize the number of spurious equilibrium points.

In the next section we will analyze the location of the equilibrium points more closely.

Hebb Rule

Let's take a closer look at the operation of the Hopfield network when the Hebb rule is used to compute the weight matrix and the prototype patterns are orthogonal. (The following analysis follows the discussion in the Chapter 7, Problem P7.5.) The supervised Hebb rule is given by

$$\mathbf{W} = \sum_{q=1}^{Q} \mathbf{p}_q (\mathbf{p}_q)^T . \tag{21.68}$$

If we apply the prototype vector $\mathbf{p}_j$ to the network, then

$$\mathbf{W}\mathbf{p}_j = \sum_{q=1}^{Q} \mathbf{p}_q (\mathbf{p}_q)^T \mathbf{p}_j = \mathbf{p}_j (\mathbf{p}_j)^T \mathbf{p}_j = S\mathbf{p}_j , \tag{21.69}$$

where the second equality holds because the prototype patterns are orthogonal, and the third equality holds because each element of $\mathbf{p}_j$ is either 1 or -1. Eq. (21.69) is of the form

$$\mathbf{W}\mathbf{p}_j = \lambda \mathbf{p}_j . \tag{21.70}$$

Therefore, each prototype vector is an eigenvector of the weight matrix and they have a common eigenvalue of $\lambda = S$. The eigenspace X for the eigenvalue $\lambda = S$ is therefore

$$X = \operatorname{span}\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_Q\} . \tag{21.71}$$

This space contains all vectors that can be written as linear combinations of the prototype vectors. That is, any vector $\mathbf{a}$ that is a linear combination of the prototype vectors is an eigenvector.

$$\begin{aligned}
\mathbf{W}\mathbf{a} &= \mathbf{W}\{\alpha_1\mathbf{p}_1 + \alpha_2\mathbf{p}_2 + \cdots + \alpha_Q\mathbf{p}_Q\} \\
&= \{\alpha_1\mathbf{W}\mathbf{p}_1 + \alpha_2\mathbf{W}\mathbf{p}_2 + \cdots + \alpha_Q\mathbf{W}\mathbf{p}_Q\} \\
&= \{\alpha_1 S\mathbf{p}_1 + \alpha_2 S\mathbf{p}_2 + \cdots + \alpha_Q S\mathbf{p}_Q\} \\
&= S\{\alpha_1\mathbf{p}_1 + \alpha_2\mathbf{p}_2 + \cdots + \alpha_Q\mathbf{p}_Q\} = S\mathbf{a}
\end{aligned} \tag{21.72}$$

The eigenspace for the eigenvalue $\lambda = S$ is Q-dimensional (assuming that the prototype vectors are independent).

The entire space R^S can be divided into two disjoint sets [Brog85],

$$R^S = X \cup X^{\perp}, \tag{21.73}$$

where $X^{\perp}$ is the orthogonal complement of X. (This is true for any set X, not just the one we are considering here.) Every vector in $X^{\perp}$ is orthogonal to every vector in X. This means that for any vector $\mathbf{a} \in X^{\perp}$,

$$(\mathbf{p}_q)^T \mathbf{a} = 0, \;\; q = 1, 2, \ldots, Q. \tag{21.74}$$

Therefore, if $\mathbf{a} \in X^{\perp}$,

$$\mathbf{W}\mathbf{a} = \sum_{q=1}^{Q} \mathbf{p}_q (\mathbf{p}_q)^T \mathbf{a} = \sum_{q=1}^{Q} (\mathbf{p}_q \cdot 0) = \mathbf{0} = 0 \cdot \mathbf{a}. \tag{21.75}$$

So $X^{\perp}$ defines an eigenspace for the repeated eigenvalue $\lambda = 0$.

To summarize, the weight matrix has two eigenvalues, S and 0. The eigenspace for the eigenvalue S is the space spanned by the prototype vectors. The eigenspace for the eigenvalue 0 is the orthogonal complement of the space spanned by the prototype vectors.

Since (from Eq. (21.54)) the Hessian matrix for the high-gain Lyapunov function V is

$$\nabla^2 V = -\mathbf{W}, \tag{21.76}$$

the eigenvalues for $\nabla^2 V$ will be $-S$ and 0.

The high-gain Lyapunov function is a quadratic function. Therefore, the eigenvalues of the Hessian matrix determine its shape. Because the first eigenvalue is negative, V will have negative curvature in X. Because the second eigenvalue is zero, V will have zero curvature in $X^{\perp}$.

What do these results say about the response of the Hopfield network? Because V has negative curvature in X, the trajectories of the Hopfield network will tend to fall into the corners of the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$ that are contained in X.

Note that if we compute the weight matrix using the Hebb rule, there will be at least two minima of the Lyapunov function for each prototype vector. If $\mathbf{p}_q$ is a prototype vector, then $-\mathbf{p}_q$ will also be in the space spanned by the prototype vectors, X. Therefore, the negative of each prototype vector will be one of the corners of the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$ that are contained in X. There will also be a number of other minima of the Lyapunov function that do not correspond to prototype patterns.

The minima of V are in the corners of the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$ that are contained in X. These corners will include the prototype patterns, but they will also include some linear combinations of the prototype patterns. Those minima that are not prototype patterns are often referred to as *spurious patterns*. The objective of Hopfield network design is to minimize the number of spurious patterns and to make the basins of attraction for each of the prototype patterns as large as possible. A design method that is guaranteed to minimize the number of spurious patterns is described in [LiMi89].

Spurious Patterns

To illustrate these principles, consider again the second-order example we have been discussing, where the connection matrix is

$$\mathbf{W} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{21.77}$$

Suppose that this had been designed using the Hebb rule with one prototype pattern (obviously not an interesting practical case):

$$\mathbf{p}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}. \tag{21.78}$$

Then

$$\mathbf{W} = \mathbf{p}_1(\mathbf{p}_1)^T = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}. \tag{21.79}$$

Notice that

$$\mathbf{W}' = \mathbf{W} - \mathbf{I} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{21.80}$$

corresponds to our original connection matrix. More about this in the next section.

The high-gain Lyapunov function is

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} = -\frac{1}{2}\mathbf{a}^T\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}\mathbf{a}. \tag{21.81}$$

The Hessian matrix for $V(\mathbf{a})$ is

$$\nabla^2 V(\mathbf{a}) = -\mathbf{W} = \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix}. \qquad (21.82)$$

Its eigenvalues are

$$\lambda_1 = -S = -2 \text{, and } \lambda_2 = 0 \text{,} \qquad (21.83)$$

and the corresponding eigenvectors are

$$\mathbf{z}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ and } \mathbf{z}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \qquad (21.84)$$

The first eigenvector, corresponding to the eigenvalue $-S$, represents the space spanned by the prototype vector:

$$X = \{\mathbf{a}: a_1 = a_2\}. \qquad (21.85)$$

The second eigenvector, corresponding to the eigenvalue 0, represents the orthogonal complement of the first eigenvector:

$$X^{\perp} = \{\mathbf{a}: a_1 = -a_2\}. \qquad (21.86)$$

The Lyapunov function is displayed in Figure 21.10.

This surface has a straight ridge from the upper-left to the lower-right corner. This represents the zero curvature region of $X^{\perp}$. Initial conditions to the left or to the right of the ridge will converge to the points

$$\mathbf{a} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \text{ or } \mathbf{a} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \qquad (21.87)$$

respectively. Initial conditions exactly on this ridge will stabilize where they start. This situation is the same as that for our original example (see Figure 21.9), except that in that case, initial points on the sloping ridge converged to the origin, instead of remaining where they started (see Figure 21.6). Initial points to the right or to the left of the ridge, in both systems, converge to the prototype design points. Thus, the convergence of our original system, and the convergence of the system with zero diagonal elements, are identical in every important aspect. We will investigate this further in the next section.

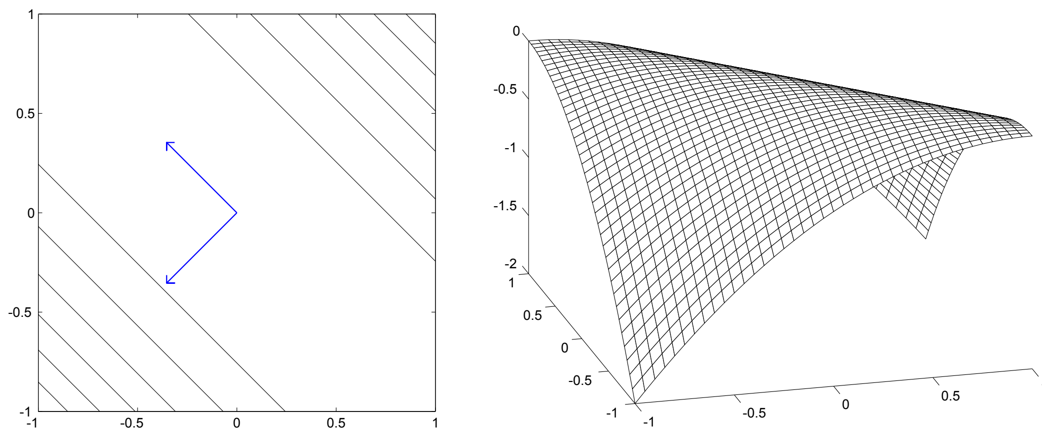


Figure 21.10 Example Lyapunov Function

Lyapunov Surface

In many discussions of the Hopfield network the diagonal elements of the weight matrix are set to zero. In this section we will analyze the effect of this operation on the Lyapunov surface. (See also Chapter 7, Exercise E7.5.)

For the content-addressable memory network, all of the diagonal elements of the weight matrix will be equal to Q (the number of prototype patterns), since the elements of each $\mathbf{p}_q$ are ± 1. Therefore, we can zero the diagonal by subtracting Q times the identity matrix:

$$\mathbf{W}' = \mathbf{W} - Q\mathbf{I} . \tag{21.88}$$

Let's investigate how this change affects the form of the Lyapunov function. If we multiply this new weight matrix times one of the prototype vectors we find

$$\mathbf{W}'\mathbf{p}_q = [\mathbf{W} - Q\mathbf{I}]\mathbf{p}_q = S\mathbf{p}_q - Q\mathbf{p}_q = (S - Q)\mathbf{p}_q . \tag{21.89}$$

Therefore, $(S - Q)$ is an eigenvalue of $\mathbf{W}'$, and the corresponding eigenspace is X, the space spanned by the prototype vectors.

If we multiply the new weight matrix times a vector from the orthogonal complement space, $\mathbf{a} \in X^{\perp}$, we find

$$\mathbf{W}'\mathbf{a} = [\mathbf{W} - Q\mathbf{I}]\mathbf{a} = \mathbf{0} - Q\mathbf{a} = -Q\mathbf{a} . \tag{21.90}$$

Therefore, $-Q$ is an eigenvalue of $\mathbf{W}'$, and the corresponding eigenspace is $X^{\perp}$.

To summarize, the eigenvectors of $\mathbf{W}'$ are the same as the eigenvectors of $\mathbf{W}$, but the eigenvalues are now $(S - Q)$ and $-Q$, instead of S and 0. Therefore, the eigenvalues of the Hessian matrix of the modified Lyapunov function, $\nabla^2 V'(\mathbf{a}) = -\mathbf{W}'$, are $-(S-Q)$ and Q.

This implies that the energy surface will have negative curvature in X and positive curvature in $X^{\perp}$, in contrast with the original Lyapunov function, which had negative curvature in X and zero curvature in $X^{\perp}$.

A comparison of Figure 21.9 and Figure 21.10 demonstrates the effect on the Lyapunov function of setting the diagonal elements of the weight matrix to zero. In terms of system performance, the change has little effect. If the initial condition of the Hopfield network falls anywhere off of the line $a_1 = -a_2$, then, in either case, the output of the network will converge to one of the corners of the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$, which consists of the two points $\mathbf{a} = \begin{bmatrix} 1 & 1 \end{bmatrix}^T$ and $\mathbf{a} = \begin{bmatrix} -1 & -1 \end{bmatrix}^T$.

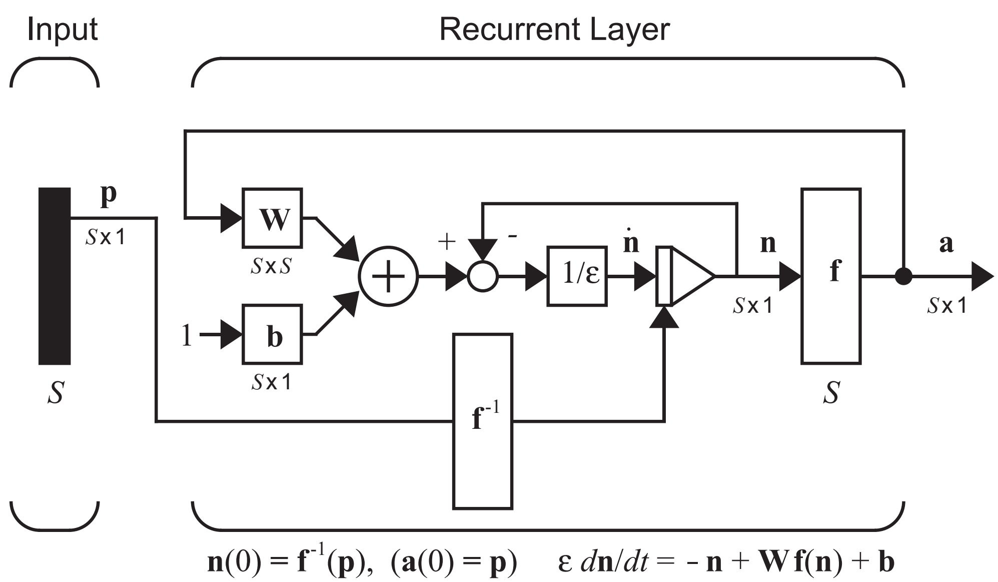
If the initial condition falls exactly on the line $a_1 = -a_2$, and the weight matrix $\mathbf{W}$ is used, then the network output will remain constant. If the initial condition falls exactly on the line $a_1 = -a_2$, and the weight matrix $\mathbf{W}'$ is used, then the network output will converge to the saddle point at the origin (as in Figure 21.6). Neither of these results is desirable, since the network output does not converge to a minimum of the Lyapunov function. Of course, the only case in which the network converges to a saddle point is when the initial condition falls exactly on the line $a_1 = -a_2$, which would be highly unlikely in practice.

Summary of Results

Hopfield Model

$$\varepsilon\frac{d\mathbf{n}(t)}{dt} = -\mathbf{n}(t) + \mathbf{W}\mathbf{a}(t) + \mathbf{b}$$

$$\mathbf{a}(t) = \mathbf{f}(\mathbf{n}(t))$$



Lyapunov Function

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} + \sum_{i=1}^{S}\left\{\int_0^{a_i} f^{-1}(u)du\right\} - \mathbf{b}^T\mathbf{a}$$

$$\frac{d}{dt}V(\mathbf{a}) = -\varepsilon\sum_{i=1}^{S}\left(\frac{d}{da_i}[f^{-1}(a_i)]\right)\left(\frac{da_i}{dt}\right)^2$$

If $\frac{d}{da_i}[f^{-1}(a_i)] > 0$, then $\frac{d}{dt}V(\mathbf{a}) \le 0$.

Invariant Sets

The Invariant Set Consists of the Equilibrium Points.

$$L = Z = \{\mathbf{a} : d\mathbf{a}/dt = \mathbf{0},\ \mathbf{a} \text{ in the closure of } G\}$$

Hopfield Attractors

The Equilibrium Points Are Stationary Points.

$$\text{If } \frac{d\mathbf{a}(t)}{dt} = \mathbf{0}\text{, then } \nabla V(\mathbf{a}) = \mathbf{0}.$$

$$\nabla V(\mathbf{a}) = [-\mathbf{W}\mathbf{a} + \mathbf{n} - \mathbf{b}] = -\varepsilon\left[\frac{d\mathbf{n}(t)}{dt}\right]$$

High-Gain Lyapunov Function

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} - \mathbf{b}^T\mathbf{a}$$

$$\nabla^2 V(\mathbf{a}) = -\mathbf{W}$$

Content-Addressable Memory

$$\mathbf{W} = \sum_{q=1}^{Q} \mathbf{p}_q(\mathbf{p}_q)^T \text{ and } \mathbf{b} = \mathbf{0}$$

Energy Surface (Orthogonal Prototype Patterns)

Eigenvalues/Eigenvectors of $\nabla^2 V(\mathbf{a}) = -\mathbf{W}$ Are

$$\lambda_1 = -S\text{, with eigenspace } X = \text{span}\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_Q\}.$$

$$\lambda_2 = 0\text{, with eigenspace } X^{\perp}.$$

$X^{\perp}$ is defined such that for any vector $\mathbf{a} \in X^{\perp}$, $(\mathbf{p}_q)^T\mathbf{a} = 0,\ q = 1, 2, \ldots, Q$

Trajectories (Orthogonal Prototype Patterns)

Because the first eigenvalue is negative, $V(\mathbf{a})$ will have negative curvature in X. Because the second eigenvalue is zero, $V(\mathbf{a})$ will have zero curvature in $X^{\perp}$. Because $V(\mathbf{a})$ has negative curvature in X, the trajectories of the Hopfield network will tend to fall into the corners of the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$ that are contained in X.

Solved Problems

P21.1 Assume the binary prototype vectors

$$\mathbf{p}_1 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}, \qquad \mathbf{p}_2 = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix}.$$

i. Design a continuous-time Hopfield network (specify connection weights) to recognize these patterns, using the Hebb rule.

ii. Find the Hessian matrix of the high-gain Lyapunov function for this network. What are the eigenvalues and eigenvectors of the Hessian matrix?

iii. Assuming large gain, what are the stable equilibrium points for this Hopfield network?

i. First calculate the weight matrix from the reference vectors, using the supervised Hebb rule.

$$\mathbf{W} = \mathbf{p}_1(\mathbf{p}_1)^T + \mathbf{p}_2(\mathbf{p}_2)^T = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 \\ -1 & -1 & 1 & 1 \\ -1 & -1 & 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{bmatrix},$$

which simplifies to

$$\mathbf{W} = \begin{bmatrix} 2 & 0 & 0 & -2 \\ 0 & 2 & -2 & 0 \\ 0 & -2 & 2 & 0 \\ -2 & 0 & 0 & 2 \end{bmatrix}.$$

ii. The Hessian of the high-gain Lyapunov function, from Eq. (21.54), is the negative of the weight matrix:

$$\nabla^2 V(\mathbf{a}) = \begin{bmatrix} -2 & 0 & 0 & 2 \\ 0 & -2 & 2 & 0 \\ 0 & 2 & -2 & 0 \\ 2 & 0 & 0 & -2 \end{bmatrix}.$$

The prototype patterns are orthogonal ($[\mathbf{p}_1]^T\mathbf{p}_2 = 0$). Thus, the eigenvalues are $\lambda_1 = -S = -4$ and $\lambda_2 = 0$. The eigenspace for $\lambda_1 = -4$ is

$$X = \text{span}\{\mathbf{p}_1, \mathbf{p}_2\}.$$

The eigenspace for $\lambda_2 = 0$ is the orthogonal complement of X:

$$X^{\perp} = \text{span}\left\{\begin{bmatrix}1\\1\\1\\1\end{bmatrix}, \begin{bmatrix}1\\-1\\-1\\1\end{bmatrix}\right\},$$

where we have chosen two vectors that are orthogonal to both $\mathbf{p}_1$ and $\mathbf{p}_2$.

iii. The stable points will be $\mathbf{p}_1, \mathbf{p}_2, -\mathbf{p}_1, -\mathbf{p}_2$ since the negative of the prototype patterns will also be equilibrium points. There may be other equilibrium points, if other corners of the hypercube lie in the span $\{\mathbf{p}_1, \mathbf{p}_2\}$. There are a total of $2^4 = 16$ corners of the hypercube. Four will fall in X and four will fall in $X^{\perp}$. The other corners are partly in X and partly in X^T.

P21.2 Consider a high-gain Hopfield network with a weight matrix and bias given by

$$\mathbf{W} = \begin{bmatrix}-1 & -1\\-1 & -1\end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix}1\\-1\end{bmatrix}.$$

i. Sketch a contour plot of the high-gain Lyapunov function for this network.

ii. If the network is given the initial condition $\begin{bmatrix}1 & 1\end{bmatrix}^T$, where will the network converge?

i. First consider the high-gain Lyapunov function

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T\mathbf{W}\mathbf{a} - \mathbf{b}^T\mathbf{a}.$$

The Hessian matrix is

$$\nabla^2 V(\mathbf{a}) = -\mathbf{W} = \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix}.$$

Next, we need to compute the eigenvalues and eigenvectors:

$$\left|\nabla^2 V(\mathbf{a}) - \lambda \mathbf{I}\right| = \left|\begin{bmatrix} 1-\lambda & 1 \\ 1 & 1-\lambda \end{bmatrix}\right| = \lambda^2 - 2\lambda + 1 - 1 = \lambda(\lambda - 2).$$

The eigenvalues are $\lambda_1 = 0$ and $\lambda_2 = 2$.

Now we can find the eigenvectors. For $\lambda_1 = 0$,

$$[\nabla^2 V(\mathbf{a}) - \lambda_1 \mathbf{I}]\mathbf{z}_1 = \mathbf{0},$$

and therefore

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}\mathbf{z}_1 = \mathbf{0} \text{ or } \mathbf{z}_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Similarly, for $\lambda_2 = 2$,

$$[\nabla^2 V(\mathbf{a}) - \lambda_2 \mathbf{I}]\mathbf{z}_2 = \mathbf{0}$$

and therefore

$$\begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}\mathbf{z}_2 = \mathbf{0} \text{ or } \mathbf{z}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

So the term

$$-\frac{1}{2}\mathbf{a}^T \mathbf{W} \mathbf{a}$$

has zero curvature in the direction $\mathbf{z}_1$ and positive curvature in the direction $\mathbf{z}_2$.

Now we have to account for the linear term. First plot the contour without the linear term, as in Figure P21.1.

The linear term will cause a negative slope in the direction of

$$\mathbf{b} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Therefore everything will curve down toward $\begin{bmatrix} 1 & -1 \end{bmatrix}^T$, as is shown in Figure P21.2.

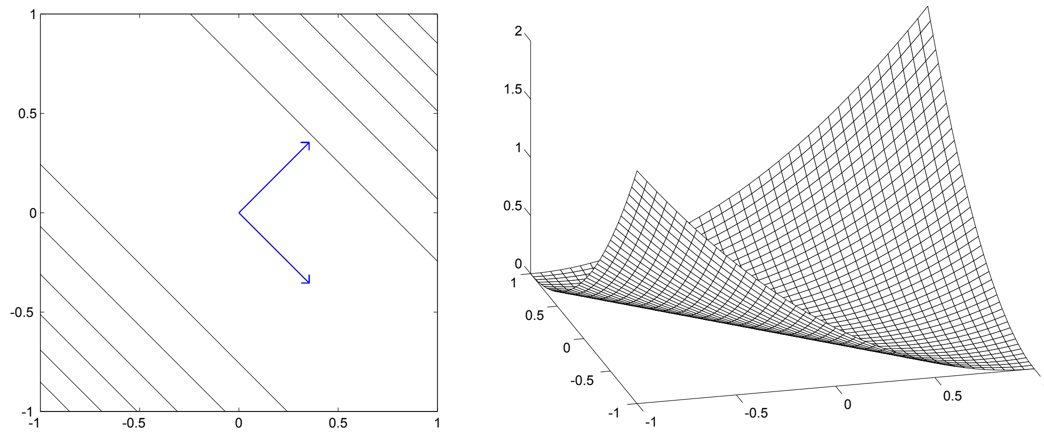


Figure P21.1 Contour Without Linear Term

ii. All trajectories will converge to $\begin{bmatrix}1 & -1\end{bmatrix}^T$, regardless of the initial conditions. As we can see in Figure P21.2, the energy function has only one minimum, which is located at $\begin{bmatrix}1 & -1\end{bmatrix}^T$. (Keep in mind that the output of the network is constrained to fall within the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$.)

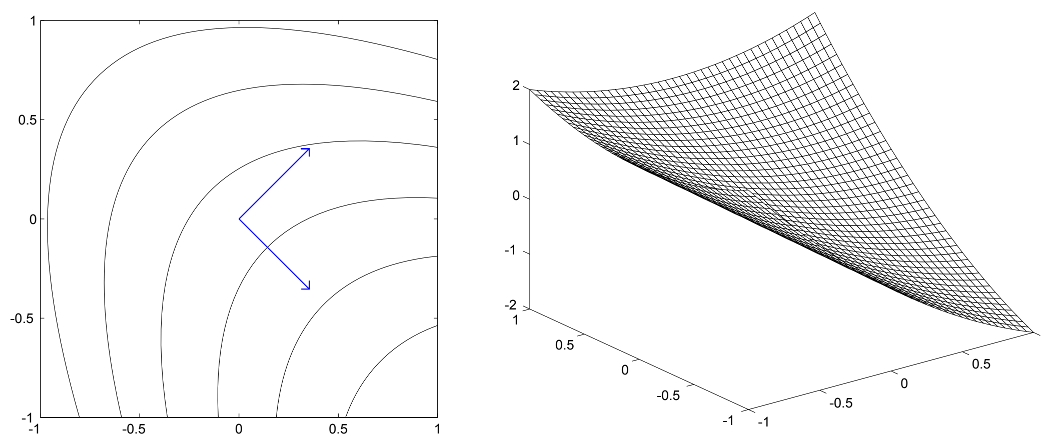


Figure P21.2 Contour Including Linear Term

P21.3 Consider the following prototype vectors.

$$\mathbf{p}_1 = \begin{bmatrix}1\\1\end{bmatrix} \quad \mathbf{p}_2 = \begin{bmatrix}-1\\1\end{bmatrix}$$

i. Design a Hopfield network to recognize these patterns.

ii. Find the Hessian matrix of the high-gain Lyapunov function for this network. What are the eigenvalues and eigenvectors of the Hessian matrix?

iii. What are the stable points for this Hopfield network (assume large gain)? What are the basins of attraction?

iv. How well does this network perform the pattern recognition problem?

i. We will use the Hebb rule to find the weight matrix.

$$\mathbf{W} = \mathbf{p}_1(\mathbf{p}_1)^T + \mathbf{p}_2(\mathbf{p}_2)^T = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} + \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

The bias is set to zero.

$$\mathbf{b} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

ii. The Hessian matrix of the high-gain Lyapunov function is the negative of the weight matrix.

$$\nabla^2 V(\mathbf{a}) = -\mathbf{W} = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix}$$

By inspection, we can see that there is a repeated eigenvalue.

$$\lambda_1 = \lambda_2 = -S = -2$$

The eigenvectors will then be

$$\mathbf{z}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \text{ and } \mathbf{z}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

or any linear combination. (The entire $\Re^2$ is the eigenspace for the eigenvalue $\lambda = -2$.)

iii. From Chapter 8 we know that when the eigenvalues of the Hessian are equal, the contours will be circular. Because the eigenvalues are negative, the function will have a single maximum at the origin. There will be four minima at the four corners of the hypercube $\{\mathbf{a}: -1 < a_i < 1\}$. There are also four saddle points. The high-gain Lyapunov function is displayed in Figure P21.3.

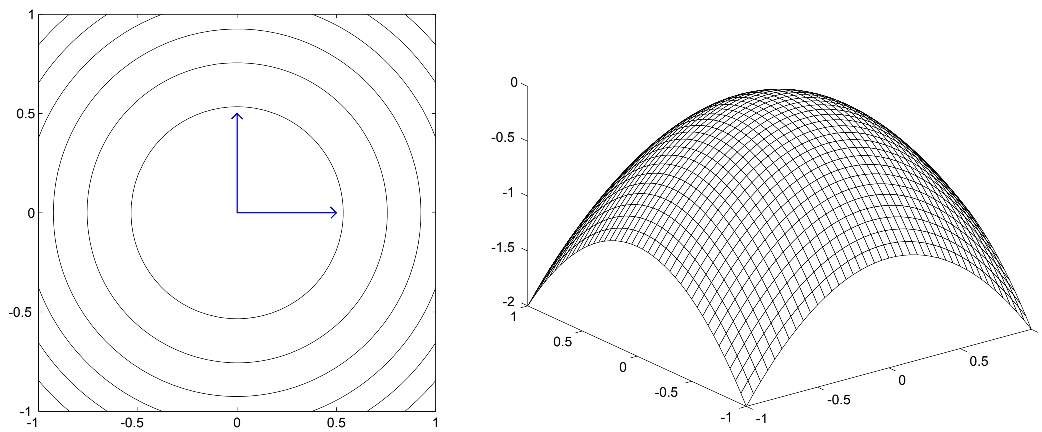


Figure P21.3 High-Gain Lyapunov Function for Problem P21.3

There are a total of nine stationary points. We could use the corollary to LaSalle's Invariance Theorem to show that the maximum point at the origin has a basin of attraction that only includes the origin itself. Therefore it is not a stable equilibrium point. The saddle points have regions of attraction that are lines. (For example, the saddle point at $\begin{bmatrix}-1 & 0\end{bmatrix}^T$ has a region of attraction along the negative a_1 axis.) The four corners of the hypercube are the only attractors that have two-dimensional regions of attraction. The region of attraction for each corner is the corresponding quadrant of the hypercube. Figure P21.4 shows the low-gain Lyapunov function (with gain $\gamma = 1.4$) and illustrates convergence to a saddle point and to a minimum.

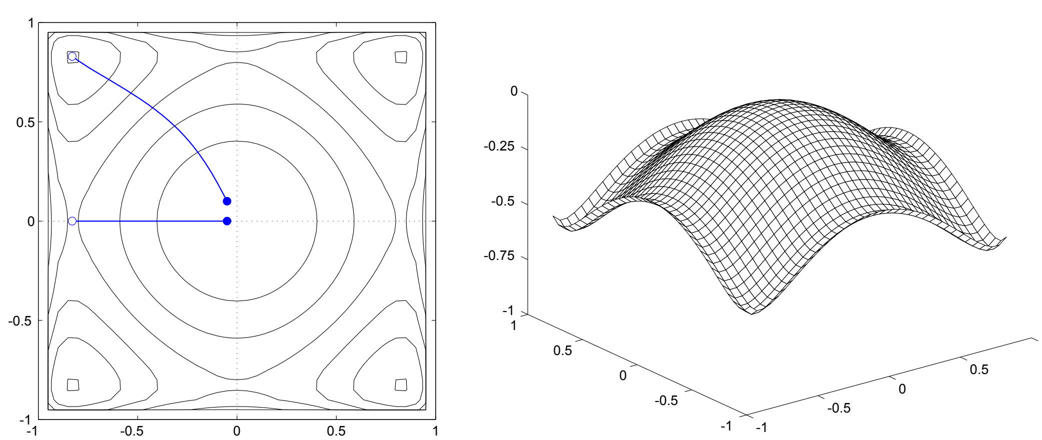


Figure P21.4 Lyapunov Function for Problem P21.3

iv. The network does not do a good job on the pattern recognition problem. Not only does it recognize the two prototype patterns, but it also "recognizes" the other two corners of the hypercube as well. The network will converge to whichever corner is closest to the input pattern, even though we only wanted it to store the two prototype patterns. Since every possible two-bit pattern has been stored, the network is not very useful. This is not unexpected, since the number of patterns that the Hebb rule is expected to store is only 15% of the number of neurons. Since we only have two neurons, we can't expect to successfully store many patterns. Try Exercise E21.2 for a better network.

P21.4 A Hopfield network has the following high-gain Lyapunov function:

$$V(\mathbf{a}) = -\frac{1}{2}(7(a_1)^2 + 12a_1a_2 - 2(a_2)^2).$$

i. **Find the weight matrix.**

ii. **Find the gradient vector of the Lyapunov function.**

iii. **Find the Hessian matrix of the Lyapunov function.**

iv. **Sketch a contour plot of the Lyapunov function.**

v. **Sketch the path that a steepest descent algorithm would follow for $V(\mathbf{a})$ with an initial condition of $\begin{bmatrix} 0.25 & 0.25 \end{bmatrix}^T$.**

i. $V(\mathbf{a})$ is a quadratic function, which can be rewritten as

$$V(\mathbf{a}) = -\frac{1}{2}(7(a_1)^2 + 12a_1a_2 - 2(a_2)^2) = -\frac{1}{2}\mathbf{a}^T\begin{bmatrix} 7 & 6 \\ 6 & -2 \end{bmatrix}\mathbf{a}.$$

Therefore the weight matrix is

$$\mathbf{W} = \begin{bmatrix} 7 & 6 \\ 6 & -2 \end{bmatrix}.$$

ii. Since $V(\mathbf{a})$ is a quadratic function, we can use Eq. (8.38) to find the gradient.

$$\nabla V(\mathbf{a}) = -\begin{bmatrix} 7 & 6 \\ 6 & -2 \end{bmatrix}\mathbf{a}$$

iii. From Eq. (8.39), the Hessian is

$$\nabla^2 V(\mathbf{a}) = -\begin{bmatrix} 7 & 6 \\ 6 & -2 \end{bmatrix} = \begin{bmatrix} -7 & -6 \\ -6 & 2 \end{bmatrix}.$$

iv. To compute the eigenvalues,

$$|\nabla^2 V(\mathbf{a}) - \lambda\mathbf{I}| = \left|\begin{bmatrix} -7-\lambda & -6 \\ -6 & 2-\lambda \end{bmatrix}\right| = \lambda^2 + 5\lambda - 50 = (\lambda + 10)(\lambda - 5).$$

The eigenvalues are $\lambda_1 = -10$ and $\lambda_2 = 5$.

Now we can find the eigenvectors. For $\lambda_1 = -10$,

$$[\nabla^2 V(\mathbf{a}) - \lambda_1\mathbf{I}]\mathbf{z}_1 = \mathbf{0},$$

and therefore

$$\begin{bmatrix} 3 & -6 \\ -6 & 12 \end{bmatrix}\mathbf{z}_1 = \mathbf{0} \text{ or } \mathbf{z}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}.$$

Similarly, for $\lambda_2 = 5$,

$$[\nabla^2 V(\mathbf{a}) - \lambda_2\mathbf{I}]\mathbf{z}_2 = \mathbf{0}$$

and therefore

$$\begin{bmatrix} -12 & -6 \\ -6 & -3 \end{bmatrix}\mathbf{z}_2 = \mathbf{0} \text{ or } \mathbf{z}_2 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}.$$

Note that this is a saddle point case, since $\lambda_1 < 0 < \lambda_2$. There will be negative curvature along $\mathbf{z}_1$ and positive curvature along $\mathbf{z}_2$. The contour plot of the high-gain Lyapunov function is shown in Figure P21.5.

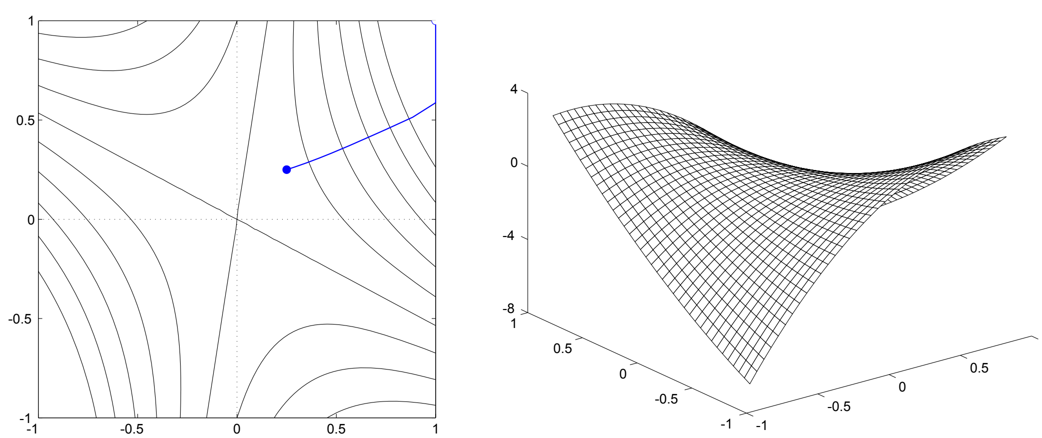


Figure P21.5 High-Gain Lyapunov Func. & Steepest Descent Trajectory

v. The steepest descent path will follow the negative of the gradient and will be perpendicular to the contour lines, as we saw in Chapter 9. When the trajectory hits the edge of the hypercube, it follows the edge down to the minimum point. The resulting trajectory is shown in Figure P21.5.

The high-gain Lyapunov function is only an approximation, since it assumes infinite gain. As a comparison, Figure P21.6 illustrates the Lyapunov function, and the Hopfield trajectory, for a gain of 0.5.

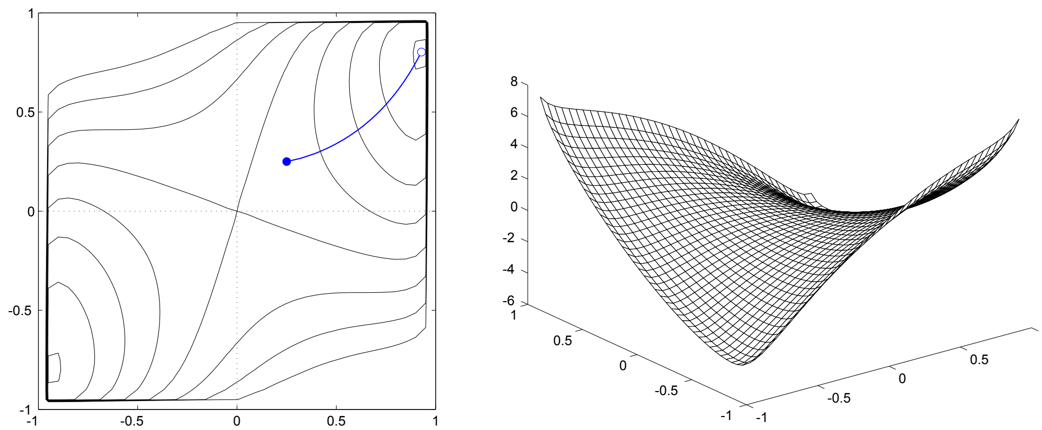


Figure P21.6 Lyapunov Function & Hopfield Trajectory

P21.5 The Hopfield network has been used for applications other than content-addressable memory. One of these other applications is analog-to-digital (A/D) conversion [HoTa86]. The function of the A/D converter is to take an analog signal y, and convert it into a series of bits (zeros and ones). For example, a two-bit A/D converter would try to approximate the signal y as follows:

$$y \cong \sum_{i=1}^{2} a_i 2^{(i-1)} = a_1 + a_2 2,$$

where a_1 and a_2 are allowed values of 0 or 1. (This A/D converter would approximate analog values in the range from 0 to 3, with a resolution of 1.) Tank and Hopfield suggest the following performance index for the A/D conversion process:

$$J(\mathbf{a}) = \frac{1}{2}\left(y - \sum_{i=1}^{2} a_i 2^{(i-1)}\right)^2 - \frac{1}{2}\left(\sum_{i=1}^{2} 2^{2(i-1)} a_i (a_i - 1)\right),$$

where the first term represents the A/D conversion error, and the second term forces a_1 and a_2 to take on values of 0 or 1.

Show that this performance index can be written as the Lyapunov function of a Hopfield network and define the appropriate weight matrix and bias vector.

The first step is to expand the terms of the performance index.

$$\left(y - \sum_{i=1}^{2} a_i 2^{(i-1)}\right)^2 = y^2 - 2y \sum_{i=1}^{2} a_i 2^{(i-1)} + \sum_{j=1}^{2} \sum_{i=1}^{2} a_i a_j 2^{(i-1)+(j-1)},$$

$$\left(\sum_{i=1}^{2} 2^{2(i-1)} a_i (a_i - 1)\right) = \sum_{i=1}^{2} (a_i)^2 2^{2(i-1)} - \sum_{i=1}^{2} a_i 2^{2(i-1)}$$

If we substitute these terms back into the performance index we find

$$J(\mathbf{a}) = \frac{1}{2}\left(y^2 + \sum_{j=1}^{2} \sum_{\substack{i=1 \\ i \neq j}}^{2} a_i a_j 2^{(i-1)+(j-1)} + \sum_{i=1}^{2} a_i (2^{2(i-1)} - 2^i y)\right).$$

The first term is not a function of $\mathbf{a}$. Therefore, it does not affect where the minima will occur, and we can ignore it.

We now want to show that this performance index takes the form of a high-gain Lyapunov function:

$$V(\mathbf{a}) = -\frac{1}{2}\mathbf{a}^T \mathbf{W} \mathbf{a} - \mathbf{b}^T \mathbf{a}.$$

This will be the case if

$$\mathbf{W} = \begin{bmatrix} 0 & -2 \\ -2 & 0 \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} y - \frac{1}{2} \\ 2y - 2 \end{bmatrix}.$$

In this Hopfield network, unlike the content-addressable memory, the input to the network is the scalar y, which is then used to compute the bias vector. In the content-addressable memory, the inputs to the network were vector patterns, which became the initial conditions on the network outputs.

Note that in this network the transfer function must limit the output to the range $0 < a < 1$. One transfer function that could be used is

$$f(n) = \frac{1}{(1 - e^{-\gamma n})}.$$

Epilogue

In this chapter we have introduced the Hopfield model, one of the most influential neural network architectures. One of the reasons that Hopfield was so influential was that he emphasized the practical considerations of the network. He described how the network could be implemented as an electrical circuit, and VLSI implementations of Hopfield-type networks were built at an early stage.

Hopfield also explained how the network could be used to solve practical problems in pattern recognition and optimization. Some of the applications that Hopfield proposed for his networks were: content-addressable memory [Hopf82], A/D conversion [TaHo86] and linear programming and optimization tasks, such as the traveling salesman problem [HoTa85].

One of Hopfield's key contributions was the application of Lyapunov stability theory to the analysis of his recurrent networks. He also showed that, for high-gain amplifiers, the Lyapunov function for his network was a quadratic function, which was minimized by the network. This led to several design procedures. The idea behind the development of the design techniques was to convert a given task into a quadratic minimization problem, which the network could then solve.

The Hopfield network is the last topic we will cover in any detail in this text. However, we have certainly not exhausted all of the important neural network architectures. In the next chapter we will give you some ideas about where to go next to explore the subject further.

Further Reading

[Ande72] J. Anderson, "A simple neural network generating an interactive memory," *Mathematical Biosciences*, vol. 14, pp. 197–220, 1972.

Anderson proposed a "linear associator" model for associative memory. The model was trained, using a generalization of the Hebb postulate, to learn an association between input and output vectors. The physiological plausibility of the network was emphasized. Kohonen published a closely related paper at the same time [Koho72], although the two researchers were working independently.

[AnSi77] J. A. Anderson, J. W. Silverstein, S. A. Ritz and R. S. Jones, "Distinctive features, categorical perception, and probability learning: Some applications of a neural model," *Psychological Review*, vol. 84, pp. 413–451, 1977.

This article describes the brain-state-in-a-box neural network model. It combines the linear associator network with recurrent connections to form a more powerful autoassociative system. It uses a nonlinear transfer function to contain the network output within a hypercube.

[CoGr83] M. A. Cohen and S. Grossberg, "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 13, no. 5, pp. 815–826, 1983.

Cohen and Grossberg apply LaSalle's Invariance Theorem to the analysis of the stability of competitive neural networks. The network description is very general, and the authors show how their analysis can be applied to many different types of recurrent neural networks.

[Gros67] S. Grossberg, "Nonlinear difference-differential equations in prediction and learning theory," *Proceedings of the National Academy of Sciences*, vol. 58, pp. 1329–1334, 1967.

This early work of Grossberg's discusses the storage of information in dynamically stable configurations.

[Hopf82] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational properties," *Proceedings of the National Academy of Sciences*, vol. 79, pp. 2554–2558, 1982.

This is the original Hopfield neural network paper, which signaled the resurgence of the field of neural networks. It describes a discrete-time network that behaves as a content-addressable memory. Hopfield demonstrates that the network evolves so as to minimize a specific Lyapunov function.

[Hopf84] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proceedings of the National Academy of Sciences*, vol. 81, pp. 3088–3092, 1984.

Hopfield demonstrates how an analog electrical circuit can function as a model for a large network of neurons with a graded response. The Lyapunov function for this network is derived and is used to design a network for use as a content-addressable memory.

[HoTa85] J. J. Hopfield and D. W. Tank, " 'Neural' computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141–154, 1985.

This article describes the application of Hopfield networks to the solution of optimization problems. The traveling salesman problem, in which the length of a trip through a number of cities with only one visit to each city is minimized, is mapped onto a Hopfield network.

[Koho72] T. Kohonen, "Correlation matrix memories," *IEEE Transactions on Computers*, vol. 21, pp. 353–359, 1972.

Kohonen proposed a correlation matrix model for associative memory. The model was trained, using the outer product rule (also known as the Hebb rule), to learn an association between input and output vectors. The mathematical structure of the network was emphasized. Anderson published a closely related paper at the same time [Ande72], although the two researchers were working independently.

[LiMi89] J. Li, A. N. Michel and W. Porod, "Analysis and synthesis of a class of neural networks: Linear systems operating on a closed hypercube," *IEEE Transactions on Circuits and Systems*, vol. 36, no. 11, pp. 1405–1422, November 1989.

This article investigates a class of neural networks described by first-order linear differential equations defined on a closed hypercube (Hopfield-like networks). Wanted and unwanted equilibrium points fall at the corners of the cube. The authors discuss design procedures that minimize the number of spurious equilibrium points.

[McPi43] W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics.*, vol. 5, pp. 115–133, 1943.

This article introduces the first mathematical model of a neuron in which a weighted sum of input signals is compared to a threshold to determine whether or not the neuron fires.

[TaHo86] D. W. Tank and J. J. Hopfield, "Simple 'neural' optimization networks: An A/D converter, signal decision circuit and a linear programming circuit," *IEEE Transactions on Circuits and Systems*, vol. 33, no. 5, pp. 533–541, 1986.

The authors describe how Hopfield neural networks can be designed to solve certain optimization problems. In one example the Hopfield network implements an analog-to-digital conversion.

Exercises

E21.1 In the Hopfield network example starting on page 18-8 we used a gain of $\gamma = 1.4$. Figure 21.3 displays the Lyapunov function for that example. The high-gain Lyapunov function for the example is shown in Figure 21.9.

i. Show that the minima of the Lyapunov function for this example will be located at points where $n_1 = n_2 = f(n_1) = f(n_2)$. (Use Eq. (21.42) and set the gradient of $V(\mathbf{a})$ to zero.)

ii. Investigate the change in location of the minima as the gain is varied from $\gamma = 0.1$ to $\gamma = 10$.

iii. Sketch the contour plot for several different values of gain in this interval. You will probably need to use MATLAB for this.

E21.2 In Problem P21.3 we used the supervised Hebb rule to design a Hopfield network to recognize the following patterns:

$$\mathbf{p}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \mathbf{p}_2 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}.$$

If we use another design rule [LiMi89], we find the following weight matrix and bias

$$\mathbf{W} = \begin{bmatrix} 1 & 0 \\ 0 & -10 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} 0 \\ 11 \end{bmatrix}.$$

i. Graph the contour plot for the high-gain Lyapunov function, if this weight matrix and bias are used.

ii. Discuss the difference between the performance of this Hopfield network and the one designed in Problem P21.3.

iii. Write a MATLAB M-file to simulate the Hopfield network. Use the `ode45` routine. Plot the responses of this network for several initial conditions.

E21.3 A Hopfield network has the following high-gain Lyapunov function:

$$V(\mathbf{a}) = -\frac{1}{2}((a_1)^2 + 2a_1a_2 + 4(a_2)^2 + 6a_1 + 10a_2).$$

i. Find the weight matrix and bias vector for this network.

ii. Find the gradient and Hessian for $V(\mathbf{a})$.

iii. Sketch a contour plot of $V(\mathbf{a})$.

iv. Find the stationary point(s) for $V(\mathbf{a})$. Use the corollary to LaSalle's Invariance Theorem to find as much information as you can about basins of attraction for any stable equilibrium points.

E21.4 In Problem P21.5 we demonstrated how a Hopfield network could be designed to operate as an A/D converter.

i. Sketch the contour plot of the high-gain Lyapunov function for the two-bit A/D converter network using an input value of $y = 0.5$. Locate the minimum points.

ii. Repeat part (i) for an input value of $y = 2.5$.

iii. Use the answers to parts (i) and (ii) to explain how the network will operate. Will the network perform the A/D conversion correctly?

E21.5 Assume the binary prototype vectors

$$\mathbf{p}_1 = \begin{bmatrix} -1 & 1 & 1 & -1 \end{bmatrix}^T, \mathbf{p}_2 = \begin{bmatrix} 1 & -1 & 1 & -1 \end{bmatrix}^T.$$

i. Design a continuous-time Hopfield network (specify connection weights and biases only) to recognize these patterns, using the Hebb rule.

ii. Find the Hessian matrix of the high-gain Lyapunov function for this network. What are the eigenvalues and eigenvectors of the Hessian matrix? (This requires very little computation.)

iii. Assuming large gain, what are the stable equilibrium points for this Hopfield network?

E21.6 Repeat Exercise E21.5 for the following prototype vectors.

i. $\mathbf{p}_1 = \begin{bmatrix} 1 & -1 & 1 & 1 \end{bmatrix}^T, \mathbf{p}_2 = \begin{bmatrix} 1 & 1 & 1 & -1 \end{bmatrix}^T.$

ii. $\mathbf{p}_1 = \begin{bmatrix} -1 & 1 & 1 & -1 \end{bmatrix}^T, \mathbf{p}_2 = \begin{bmatrix} 1 & 1 & -1 & -1 \end{bmatrix}^T.$

iii. $\mathbf{p}_1 = \begin{bmatrix} -1 & -1 & 1 & 1 & -1 & -1 \end{bmatrix}^T, \mathbf{p}_2 = \begin{bmatrix} -1 & -1 & -1 & 1 & 1 & 1 \end{bmatrix}^T.$

E21.7 Consider a high-gain Hopfield network with weight matrix and bias given by:

$$\mathbf{W} = \begin{bmatrix} 1 & 3 \\ 3 & 9 \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} 3 \\ -1 \end{bmatrix}.$$

i. Sketch a contour plot of the high-gain Lyapunov function for this network.

ii. If the network is given the following initial condition, where will the network converge?

$$\mathbf{a}(0) = \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}$$

E21.8 Design a high-gain Hopfield network (give the weights and the biases) with only one stable equilibrium point:

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

Explain your procedure, and show all steps. (Do not use the Hebb rule.)

E21.9 Consider a high-gain Hopfield network with weight matrix and bias given by:

$$\mathbf{W} = \begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

i. Sketch a contour plot of the high-gain Lyapunov function for this network.

ii. Assuming a large gain, what are the stable equilibrium points for this Hopfield network? What can you say about the basins of attraction for these stable equilibrium points? Explain your answers.

E21.10 Repeat E21.9 for the following weight and bias:

$$\mathbf{W} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

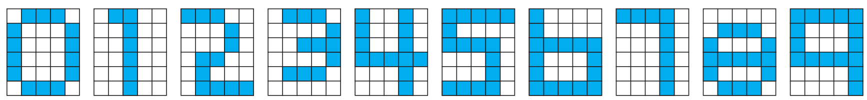
» 2 + 2
ans =
4

E21.11 In Exercise E7.11 we asked the question: How many prototype patterns can be stored in one weight matrix? Repeat that problem using the Hopfield network. Begin with the digits "0" and "1". (The digits are shown at the end of this problem.) Add one digit at a time up to "6", and test how often the correct digit is reconstructed after randomly changing 2, 4 and 6 pixels.

i. First use the Hebb rule to create the weight matrix for the digits "0" and “1”. Then randomly change 2 pixels of each digit and apply the noisy digits to the network. Repeat this process 10 times, and record the percentage of times in which the correct pattern (without noise) is produced at the output of the network. Repeat as 4 and 6 pixels of each digit are modified. The entire process is then repeated when the digits “0”, “1” and “2” are used. This continues, one digit at a time, until you test the network when all of the digits “0” through “6” are used. When you have completed all of the tests, you will be able to plot three curves showing percentage error versus number of digits stored, one curve each for 2, 4 and 6 pixel errors.

ii. Repeat part (i) using the pseudoinverse rule (see Chapter 7), and compare the results of the two rules.

iii. For extra credit, repeat part (i) using the method described in [LiMi89]. In that paper it is called Synthesis Procedure 5.1.



$\mathbf{p}_1$ $\mathbf{p}_2$ $\mathbf{p}_3$ $\mathbf{p}_4$ $\mathbf{p}_5$ $\mathbf{p}_6$ $\mathbf{p}_7$ $\mathbf{p}_8$ $\mathbf{p}_9$ $\mathbf{p}_{10}$