10.14 Simulated Annealing

When we address the design of high dimensional and complex pattern classifiers we find that analytical methods or even techniques that use local derivatives for gradient descent turn out to be inadequate. We know that error surfaces of non-linear systems in high dimensional spaces have multiple minima, and finding the true global minimum is indeed a difficult task. Exhaustive search in solution space to find a good set of parameters is almost impossible to use due to the sheer number of possibilities that arise in real world problems. In addition, an issue that accompanies the increasing complexity of the problem is that we usually have less training data and less prior knowledge [136]. Sophisticated search techniques need to be adopted in order to find acceptable solutions. In this section we discuss a stochastic search method called *simulated annealing* that affords one possible approach to solve this problem. Our discussion will eventually lead us to the formulation of the Boltzmann machine, a neural network model that employs stochastic methods in its operation. Other stochastic techniques such as genetic algorithms shall be discussed in Chapter 15.

Annealing allows systems of magnets or atoms in alloys to search low energy configurations. It involves heating the solid in question to high temperatures whereby random configurations are explored. This is followed by gradually reducing the temperature of the system towards zero. This allows the system to relax into a low energy configuration because even at moderately high temperatures the system favours regions in configuration space that have a lower energy. These regions are thus more likely to contain the global minimum. Consequently, as the temperatures are lowered the system has a high probability of searching the optimal configuration.

Simulated annealing provides a general framework for the optimization of the behaviour of complex systems. It operates by introducing noise in a controllable fashion into the operational dynamics of the system, for robust iterative search. In the treatment that follows, we refer to a specific combination of neuron states as a *configuration* of the network. Different *configurations* of the network—each being nothing but a particular instance of a signal vector—are indexed by γ. Clearly, in an n-node network there are at most 2^n configurations. The basic idea underlying simulated annealing is to generate different configurations of the system at various values of a control parameter called the *temperature*, and to gradually reduce the value of this parameter to search for an optimal or *ground state* solution to the problem. The simplest way to generate multiple configurations of a system at an energy E and temperature T is to use the Metropolis algorithm [386].

10.14.1 An Important Result from Statistical Mechanics

In general, the low energy configurations will be very few. We know that these will be precisely those corresponding to the vectors that are encoded into the network and other spurious memories. However, there are many more possible configurations that correspond to higher energies. In fact, the number of possible configurations increases exponentially with an increase in energy. From statistical mechanics we know that a system is in thermal equilibrium when a configuration γ with energy E_γ occurs with a probability:

$$P(\gamma) = \frac{e^{-E_\gamma/T}}{\sum_{\gamma'} e^{-E_{\gamma'}/T}} = \frac{e^{-E_\gamma/T}}{\mathcal{Z}(T)} \tag{10.1}$$

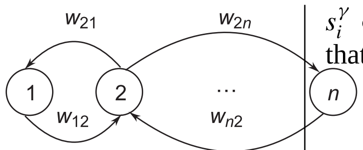
$P(\gamma)$ is an increasing function of $e^{-E_\gamma/T}$, and therefore decreases as E_γ increases.

where the various possible configurations are indexed by γ'. $P(\gamma)$ is referred to as the *Boltzmann Probability Distribution*. In Eq. 10.1, $\mathcal{Z}(T)$ is called the *partition function* which plays the role of a normalizing constant. The numerator of Eq. 10.1 is called the *Boltzmann factor*, and for physical systems there is a Boltzmann constant k_B $(= 1.38 \times 10^{-23}$ J/K), that converts temperature to an energy in the expression. In the present treatment we can safely ignore this constant by assuming that we will be working with scaled temperatures in our simulations. T then represents a scaled temperature which includes the Boltzmann constant.

10.14.2 Understanding the Procedure

Our discussion on simulated annealing will be in the context of neural networks, although it is worth pointing out that it is an optimization technique that can be applied to a wide variety of real world problems.

At present, we return to the classic quadratic energy optimization problem. To remain on familiar ground, we will consider a Hopfield network architecture with a set of given connections w_{ij}, and bipolar signal states $s_i^\gamma \in \{-1, +1\}$, $(1 \leq i \leq n)$. Then, we are required to find values of s_i^γ, s_j^γ that minimize the Lyapunov energy:



Hopfield network showing a few connections for neuron 2.

$$E_\gamma = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij} s_i^\gamma s_j^\gamma \tag{10.2}$$

where $w_{ij} = w_{ji}$, and $w_{ii} = 0$. Note that the energy has been specifically indexed by the configuration γ. Each configuration will have its own energy that depends upon the signal states.

The simulated annealing method for finding an optimal configuration of neuron states given a set of weights is based on the physical annealing metaphor. It involves the following basic steps:

The occasional acceptance of a higher energy state allows the algorithm to escape from local minima and *explore* other potential regions in the solution space. To implement the state transition from a lower energy state to a higher energy state, we simply have to compare the value of $e^{-\Delta E^{(I)}/T}$ with a random number generated between 0 and 1. If the number generated is less than $e^{-\Delta E^{(I)}/T}$, then accept the transition; else reject the transition.

- Randomize neuron states once in the beginning, and initialize the temperature to a high value.
- Choose a neuron I randomly from the network.
- Compute the energy $E_\mathcal{A}$ of the present configuration $\mathcal{A}$.
- Flip the state of neuron I to generate a new configuration $\mathcal{B}$.
- Compute the energy $E_\mathcal{B}$ of configuration $\mathcal{B}$.
- If $E_\mathcal{B} < E_\mathcal{A}$ then accept the state change for neuron I. Otherwise accept the state change for neuron I with a probability $e^{-\Delta E^{(I)}/T}$ where $\Delta E^{(I)} = E_\mathcal{B} - E_\mathcal{A}$.
- Continue selecting and testing neurons randomly, and set their states several times in this way until a *thermal equilibrium* is reached.
- Finally, lower the temperature and repeat the procedure.

This procedure continues until the temperature reaches a very small value.

Note that the algorithm works because at very high temperatures since $e^{-E_\gamma/T} \approx 1$, $P(\gamma) \approx \frac{1}{2^n}$. What this really means is that at high temperatures all configurations are somewhat equally likely. Also, at high temperatures transitions to energetically unfavourable states are frequent. However, at lower temperatures transitions to energetically unfavourable states become less frequent and the search becomes more like the usual descent procedures. Finally, if the cooling is sufficiently slow, the network has a very high probability of finding itself in an optimal configuration that represents a minimum energy configuration.

From an implementation point of view, note that the transition probability is dependent on $\Delta E^{(I)}$. Therefore only those neurons need to be considered which are directly connected to the neuron I under consideration. The change in energy $\Delta E^{(I)}$ which occurs when neuron I flips its state from s_I to $-s_I$ is:

$$\Delta E^{(I)} = -(-s_I)\sum_{j=1}^{n} w_{Ij}s_j + \left(s_I \sum_{j=1}^{n} w_{Ij}s_j\right) \tag{10.3}$$

Here we have used the expression for energy comprising terms related to neuron I only (see Eq. 10.42).

$$= 2s_I \sum_{j=1}^{n} w_{Ij}s_j \tag{10.4}$$

$$= 2s_I \sum_{j=1}^{n} w_{jI}s_j \tag{10.5}$$

$$= 2s_I x_I \tag{10.6}$$

where we have used the weight symmetry condition $w_{jI} = w_{Ij}$. Then the simulated annealing procedure outlined above can be recast into algorithmic form as shown in Table 10.1.

A critical aspect of the algorithm is the choice of initial temperature and the *annealing schedule*. A typical choice of the annealing schedule is

$$T_{k+1} = cT_k \tag{10.7}$$

where $0 < c < 1$. A typical working range of c is $0.8 < c < 0.9$ which is found to work well for real world problems. The initial temperature should be chosen high and the maximum iteration index $k_{\max}$ as large as possible.

Note that although we have discussed the simulated annealing optimization algorithm in the context of the Hopfield network, it is a general technique that can be used to optimize any non-linear cost functions.

It should be straightforward to write the MATLAB code segment for the stochastic simulated annealing algorithm of Table 10.1 (see Review Question 10.17).

10.14.3 Energy Optimization of Hopfield CAM

Much intuition into the working of the algorithm is to be gained from the interesting simulation example that follows.

In this example we encode two vectors into a six dimensional Hopfield network using bipolar outer product encoding. The vectors are:

Table 10.1 Stochastic simulated annealing algorithm applied to a Hopfield network

Given	A set of binary vectors $\{A_i\}_{i=1}^{Q}$ to be encoded into a Hopfield CAM using bipolar encoding.
Encode	↬ $\mathbf{W} = \sum_{k=1}^{Q} X_k X_k^T - Q\mathbf{I}$
Initialize	↬ T_0, $k = 0$, $k_{\max}$ (temperature, iteration, iteration limit) S_0, c (signal vector, temperature contraction)

Iterate

```
↻Repeat
{
  ↻Repeat
  {
    ⇝ Select neuron $I$ randomly.
    ⇝ Compute: $\Delta E^{(I)} = 2s_I \sum_{j=1}^{n} w_{jI} s_j$
    ⇝ if $\Delta E^{(I)} < 0$, $s_I = -s_I$
       else if $e^{-\Delta E^{(I)}/T_k} >$ rand [0,1), $s_I = -s_I$
  } until(all nodes are polled several times)
  ⇝ Reduce temperature: $T_{k+1} = cT_k$
} until ($k = k_{\max}$ or stopping criterion met)
```

$A_1 = (110001)$ and $A_2 = (101010)$. The resultant weight matrix is:

$$\mathbf{W} = \begin{pmatrix} 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & -2 & 0 & -2 & 2 \\ 0 & -2 & 0 & 0 & 2 & -2 \\ -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2 & 2 & 0 & 0 & -2 \\ 0 & 2 & -2 & 0 & -2 & 0 \end{pmatrix} \tag{10.8}$$

The first thing we do is to calculate the probabilities of the system being in different configurations at a particular temperature. This is done in accordance with Eq. 10.1. Figure 10.1 portrays these configuration-wise probability plots at various temperatures.

In Fig. 10.21, the horizontal axis plots the configuration number from 0 to 63—for six bits we have 64 combinations. So we start with zero and go to 63. Energies for each of these configurations are plotted in Fig. 10.1(e). Note that the lowest energy corresponds to the encoded associations A_1, A_2, and their complements A_1', A_2': $E_{A_1} = E_{A_2} = E_{A_1'} = E_{A_2'} = -20$. In Figs 10.1(a)–(d) the probabilities are plotted for temperatures 10, 5, 1, and 0.1. As is to be expected, at higher temperatures, the probability of the

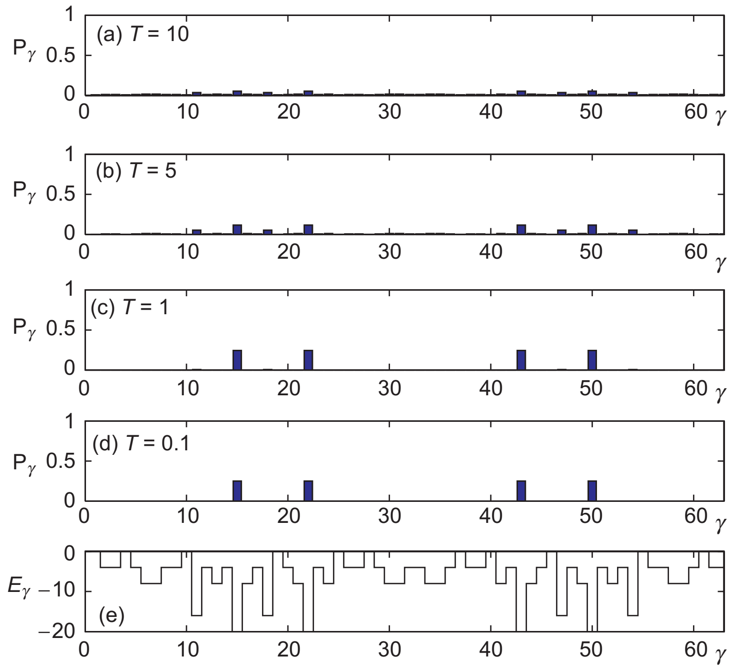


Fig. 10.21 Probability estimates for different configurations at four temperatures.
Note: (a) $T = 10$ (b) $T = 5$ (c) $T = 1$ (d) $T = 0.1$. (e) Staircase plot of the energy levels of different attractors or configurations of the network. The maximum energy is 0 and the minimum energy is –20

system being in a particular state is almost the same for all states. However, as the temperature is reduced, the probability of the system being in lower energy states increases. At $T = 0.1$ the probability gets roughly divided between the four lowest energy states, while that of other 60 states is almost negligible. In our case, these four states are the two memories and their two spurious complements that were encoded into the weight matrix.

Okay, now let us do the annealing. We start with a temperature $T = 5$, and use the contraction $T_{k+1} = 0.99T_k$. Figure 10.2(a) plots the annealing schedule, while Fig. 10.2(b) shows the energy transitions as the system moves from state to state during the annealing cycles. Notice from Fig. 10.2(b) that at higher temperatures, there are very many transitions that take place from lower to higher energy. This is what lends simulated annealing its search power. As the temperature is gradually lowered, the system begins to settle into lower energy states until it finally settles down into one of the four attractors which have the lowest energy of -20. Note that the system might settle into any one of the four -20 energy configurations in this case. In fact, as predicted by Fig. 10.1(e), the system has an equal probability of finding itself in one of the lowest energy states. However, which one it settles to cannot be predicted in advance since the search is stochastic.

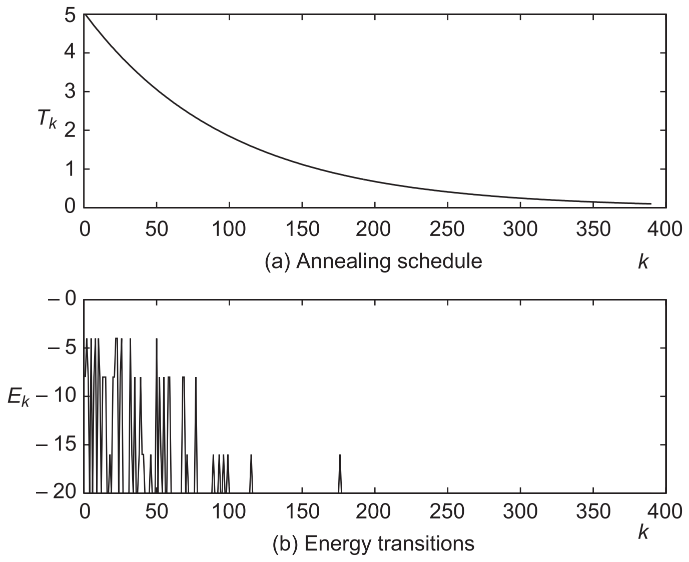


Fig. 10.22 A plot of the energy transitions made by the system during the annealing process.
Note: As the temperature is lowered, the number of transitions to higher energy states becomes more infrequent