Simulated Annealing

Chapter · March 2019

DOI: 10.1201/9780429504419-11

3 authors, including:

Prithwiraj Mal
National Institute of Fashion Technology, Hyderabad, India
48 PUBLICATIONS **146** CITATIONS

SEE PROFILE

Abhijit Majumdar
Indian Institute of Technology Delhi
210 PUBLICATIONS **5,169** CITATIONS

SEE PROFILE

11

Simulated Annealing

11.1 Introduction

Simulated annealing was first put forward by Metropolis et al. (1953) and successfully applied to the optimization problems by Kirkpatrick et al. (1983). It emulates the cooling process of molten metals through annealing. It is a random search technique that exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure. It begins with a starting temperature and by carefully controlling the rate of cooling it may lead to a global optimum. Unlike the genetic algorithm and particle swarm optimization which are population-based search methods, simulated annealing is a point-by-point search method.

Simulated annealing starts with a randomly selected current point in the search space, and the initial temperature T is kept at a high value. A new point is generated at random in the neighborhood of the current point. Then the difference in the function values ΔE at these two points is measured, and the Metropolis algorithm is used to accept or reject the point. If the new point is accepted, it is used as a starting point for the next step. However, if the new point is rejected, the original point is retained as a starting point of the next step. This is the end of the first iteration of simulated annealing. In the next iteration, again a new point is randomly generated in the vicinity of the current point, and the Metropolis algorithm is used to accept or reject the new point. In general, before reducing the temperature, a number of iterations are performed at a particular temperature. The temperature is lowered according to the cooling schedule, and the rate of cooling is done at a slower rate. The algorithm is stopped when the temperature becomes just lower than the termination temperature.

11.2 Simulated Annealing

Annealing means the tempering of a metal by heating above its melting point and then cooling it very slowly until it solidifies into a perfect crystalline

structure. The simulation of the annealing process is termed as *simulated annealing* (Kirkpatrick et al., 1983). At high temperature, the atoms in a molten metal have greater mobility, and they can move freely with respect to each other, but as the temperature is lowered, the movement of the atoms gets restricted. A fundamental question in statistical mechanics concerns the ultimate state of the material in the limit of low temperature: do the atoms remain fluid or solidify, and if they solidify, do they form a perfect crystalline solid, an imperfect crystalline solid, or a glass? The rate at which the temperature is reduced plays a key role in deciding the ultimate state of the material. The growth of a perfect crystal from a molten metal is achieved by careful annealing in which the temperature is lowered very slowly and a long time is spent at temperatures in the vicinity of the freezing point. Because of very slow cooling, the atoms get ordered gradually, and finally they form a crystal having the minimum energy level. Therefore, in order to achieve the absolute minimum energy level, the temperature needs to be reduced at a very slow rate. On the contrary, if the temperature of a molten metal is reduced at a faster rate, the resulting crystals will have many defects or may become a glass with no crystalline order. A defective crystal or a glass has a higher energy level. Figure 11.1 illustrates the effect of cooling rate on the ultimate state of a molten metal. A defective crystalline structure corresponds to local minimum configurations, whereas a perfect crystalline structure corresponds to a global minimum energy configuration.

Simulated annealing is analogous to an optimization procedure such that the energy of a state is the value of an objective function in a minimization

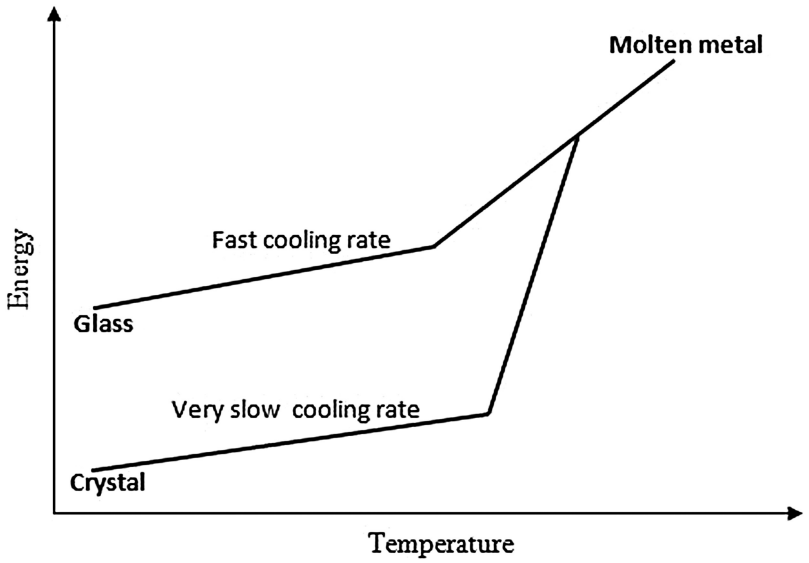

FIGURE 11.1
Effect of cooling rate.

problem, and the cooling phenomenon is simulated by controlling a temperature-like parameter introduced with the concept of a Boltzmann probability distribution (Deb, 2005). In statistical mechanics, the Boltzmann probability distribution gives the probability p_i of a physical system being in state i with energy E_i at temperature T by the following expression:

$$p_i = \frac{e^{-E_i/kT}}{\sum_{j=1}^{m} e^{-E_j/kT}} \tag{11.1}$$

where k is the Boltzmann constant, and m is the number of states accessible to the system.

This shows that states with lower energy will always have a higher probability of being occupied than the states with higher energy. According to the Boltzmann distribution, the probability of a state change p can be determined from the energy difference of the two states ΔE and temperature T as follows:

$$p = e^{-\Delta E/kT} \tag{11.2}$$

Figure 11.2 displays the plot of probability of state change with temperature, from which it is quite evident that at a high temperature, there is uniform

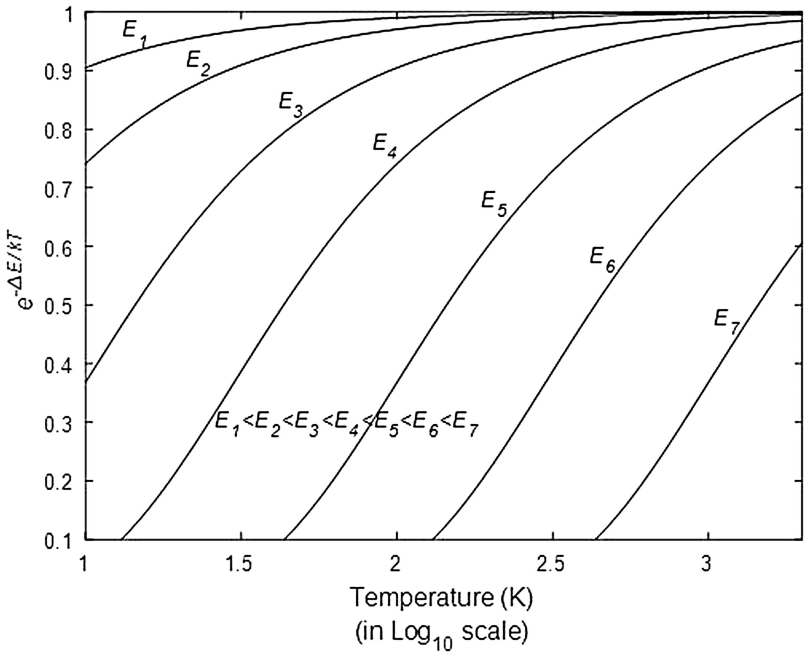

FIGURE 11.2
Probability of state change depending on the energy difference and temperature.

preference for all of the states, irrespective of energy; however, at a low temperature, there is a small probability of being at a high energy state. Thus, when the temperature approaches absolute zero, only the states with minimum energy have nonzero probability of occurrence. In simulated annealing, the search process follows the Boltzmann probability distribution where Boltzmann constant k is omitted and convergence of the algorithm is obtained by controlling the temperature T. The implementation of the Boltzmann probability distribution can be performed by using the Metropolis algorithm, which is a Monte Carlo simulation to generate sample states of a simulated thermodynamic system.

Metropolis et al. (1953) introduced a simple algorithm that can provide an efficient simulation of a collection of atoms in equilibrium at a given temperature. According to this algorithm, in each step, an atom is given a small displacement at random, and the resulting change in energy of the system ΔE is computed. If $\Delta E \leq 0$, the displacement is accepted, and the configuration with the displaced atom is used as the starting point of the next step. Otherwise, it is treated probabilistically using the Boltzmann distribution, that is, if $\Delta E > 0$, the probability that the configuration is accepted is $P = e^{-\Delta E/T}$. The random part of the algorithm is implemented by selecting a uniformly distributed random number P' in the range (0, 1). The selected number P' is compared with P. If $P' \leq e^{-\Delta E/T}$, the new configuration is retained, otherwise if $P' > e^{-\Delta E/T}$, the original configuration is used as the starting point of the next step. The Metropolis algorithm can also be used in the context of function minimization. Let us assume that at any moment the current point is $X^{(i)}$, which gives a function value $E^{(i)} = f(X^{(i)})$. Let us also assume that a new point $X^{(i+1)}$ is randomly created at the vicinity of the current point $X^{(i)}$. The function value of the point $X^{(i+1)}$ is $E^{(i+1)} = f(X^{(i+1)})$. The Metropolis algorithm says that the acceptance of the new point $X^{(i+1)}$ depends on the difference in the function values at these two points: $\Delta E = f(X^{(i+1)}) - f(X^{(i)})$. If $\Delta E \leq 0$, new point $X^{(i+1)}$ is accepted. In the context of function minimization, this is logical, because if the function value at $X^{(i+1)}$ is less than that at $X^{(i)}$, then the point $X^{(i+1)}$ is a better one and thus it is accepted. However, the situation becomes probabilistic when $\Delta E > 0$, which means that the function value at $X^{(i+1)}$ is higher than that at $X^{(i)}$. Had it been a traditional algorithm, the point $X^{(i+1)}$ would have not been accepted in this particular situation. But what is interesting about the Metropolis algorithm is that there exists some finite probability of selecting the point $X^{(i+1)}$, even if it is inferior to the point $X^{(i)}$. If $\Delta E > 0$, the probability that the new point $X^{(i+1)}$ is accepted is $P = e^{-\Delta E/T}$. This probability is compared with a randomly generated number P' in the range (0, 1). If $P' \leq P$, the new point $X^{(i+1)}$ is accepted; else if $P' > P$, the new point $X^{(i+1)}$ is rejected. At a high temperature, the probability of acceptance is more or less high for the points with largely disparate function values; therefore, any point has a fairly good chance of acceptance. On the contrary, at a low temperature, the probability of accepting a random point is becoming small; therefore,

the points with only small deviation in function value are accepted. With the intention of simulating the thermal equilibrium at every temperature, usually a number of iteration n is performed at a particular temperature. The temperature is then lowered step by step at a slower rate. The algorithm is terminated when temperature becomes just less than the minimum temperature.

11.2.1 Flowchart of Simulated Annealing

Step 1: Initialization

Choose an initial point X. Set initial temperature T to a sufficiently high value, cooling rate C_T, and minimum temperature $T_{\min}$. Also set $i = 0$. Estimate the function values for the current point $E(X)$.

Step 2: Generation of a New Point in the Neighborhood of Current Point

Create a new point $X^{(i+1)}$ randomly at the vicinity of the current point X. Estimate the function values for the new point $E(X^{(i+1)})$.

Step 3: Checking the Acceptance of the New Point

If $\Delta E = E(X^{(i+1)}) - E(X) < 0$, the new point is accepted, set $X = X^{(i+1)}$, else create a random number P' in between the range (0–1). If $P' \leq \exp(-\Delta E/T)$, the new point is accepted, set $X = X^{(i+1)}$, else the new point is rejected.

Set $i = i + 1$, and go to the next step.

Step 4: Lowering the Temperature as per the Cooling Schedule

Lower the temperature T according to the cooling schedule. If $T < T_{\min}$, terminate, or else go to step 2.

The flowchart of the simulated annealing algorithm is depicted in Figure 11.3.

11.3 Step-by-Step Working Principle of Simulated Annealing

Let us consider the following optimization problem to explain the working principle of simulated annealing:

$$\begin{aligned} &\text{Minimize} \quad f(x_1, x_2) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2 \\ &\text{Subject to} \quad -10 \leq x_1, x_2 \leq 10 \end{aligned} \tag{11.3}$$

where x_1 and x_2 are the real variables.

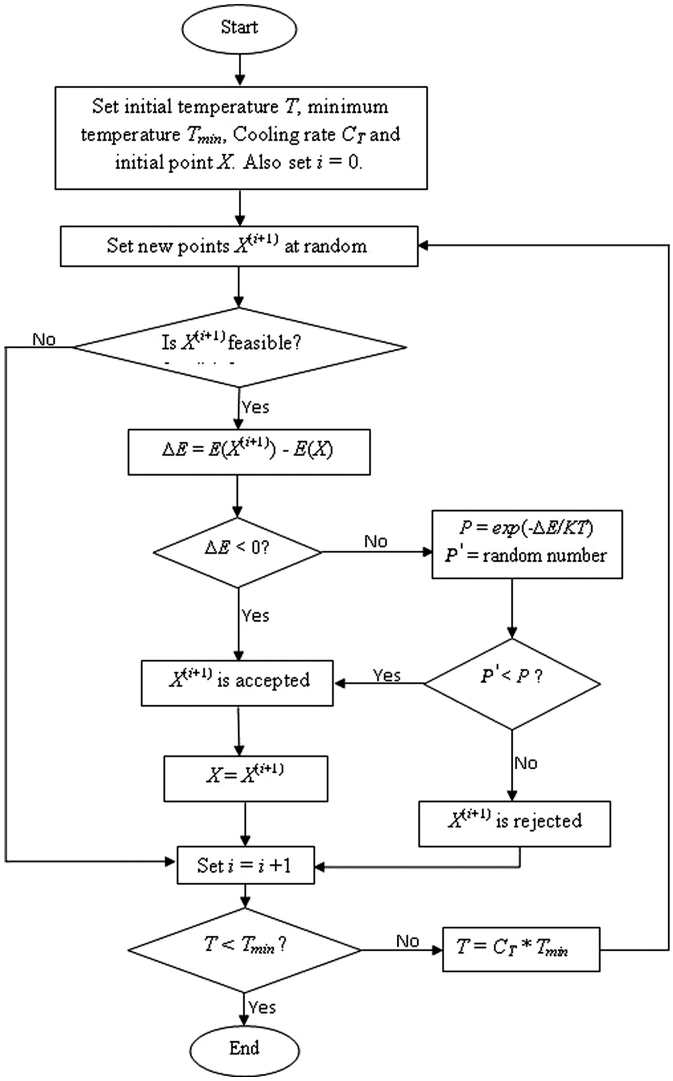

FIGURE 11.3
Flowchart of the simulated annealing algorithm.

Surface and contour plots of the objective function $f(x_1, x_2)$ are displayed in Figures 11.4 and 11.5, respectively. It can be shown that the minimum value of the function $f(x_1, x_2)$ is 0, which is obtained at $x_1 = 1$ and $x_2 = 3$. In both Figures 11.4 and 11.5, the optimum solution is marked by an asterisk (*).

Various steps of simulated annealing up to 10 iterations are discussed as follows for solving the optimization problem of Equation 11.3.

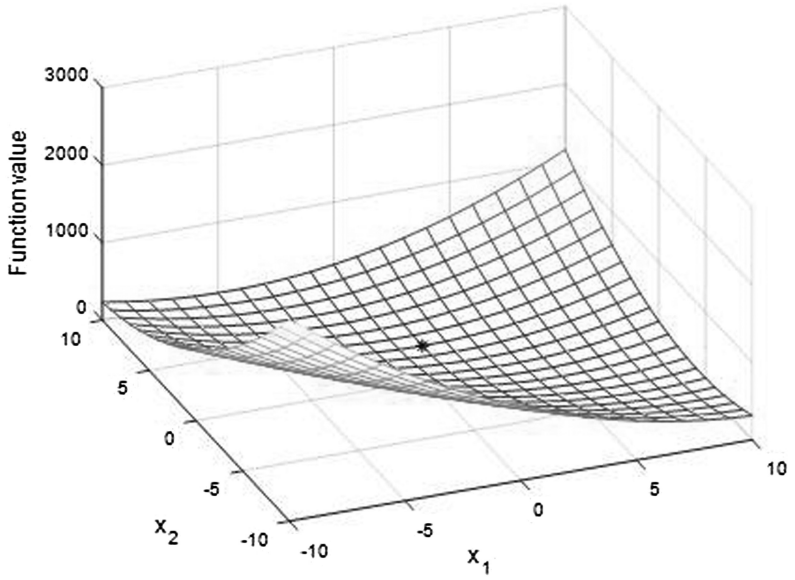

FIGURE 11.4
Surface plot of function $f(x_1, x_2) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$.

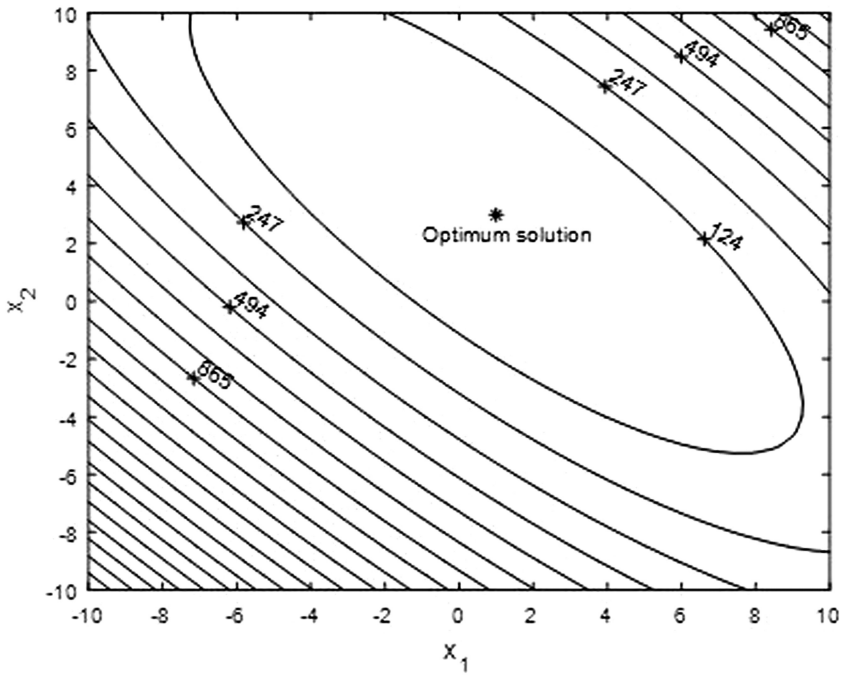

FIGURE 11.5
Contour plot of the function $f(x_1, x_2) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$.

Step 1: Initialization

We set the initial point $X^{(0)} = (0, 0)$, initial temperature $T = 1000$, and the initial value of iteration counter $i = 0$. Thus, the initial point has a function value $f(X^{(0)}) = 74$. Typical values of rate of cooling C_T, minimum temperature $T_{\min}$, and number of iterations to be performed at a particular temperature n are 0.9, 10^{-6}, and 1,000, respectively. However, for ease of explanation, in this example we have chosen $C_T = 0.5$, $T_{\min} = 1$, and $n = 1$.

Step 2: Generation of a New Point in the Neighborhood of Current Point

In this step, we generate a new point $X^{(1)}$ in the vicinity of the $X^{(0)}$. For this purpose, we select normally distributed random numbers for each variable. The mean (μ) of the normal distribution is kept at the current point, and the standard deviation (σ) is chosen in such a way that 3σ limits (i.e., 99.73% confidence limits) of each variable equal its maximum and minimum boundaries. Hence, 3σ limits of a normally distributed variable (x) will become (refer to Figure 11.6):

$$\mu + 3\sigma = x_{\max} \tag{11.4}$$

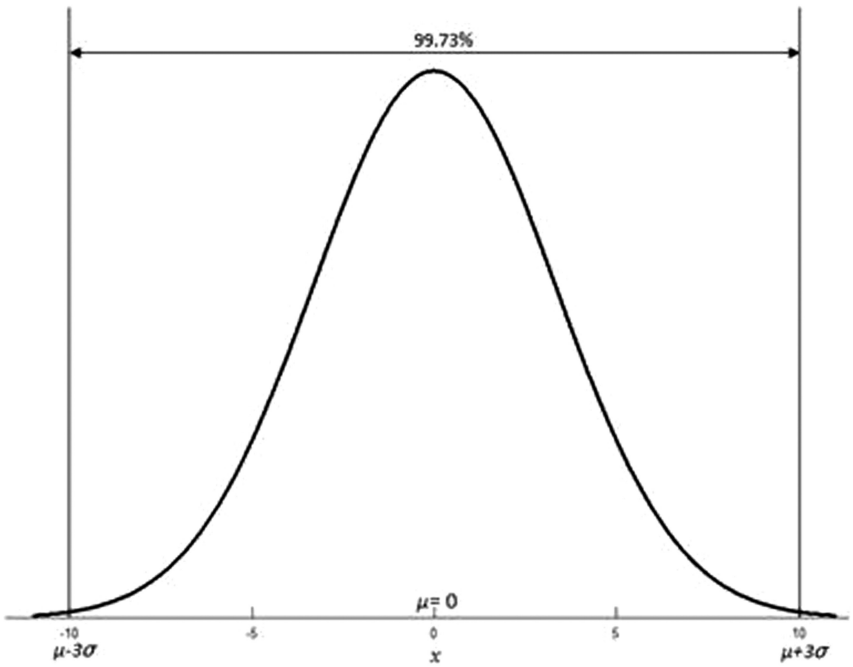

FIGURE 11.6
Normal distribution of variable x with mean = 0.

$$\mu - 3\sigma = x_{\min} \tag{11.5}$$

where $x_{\max}$ and $x_{\min}$ are the maximum and minimum boundaries of the variable. By solving Equations 11.4 and 11.5, we get

$$\sigma = \frac{(x_{\max} - x_{\min})}{6} \tag{11.6}$$

In this example, for both the variables (x_1, x_2), the current point $X^{(0)}$ maximum and minimum boundaries are (0, 0), (10, 10), and (−10, −10), respectively. Thus, each variable is assumed to have normal distribution with $\mu = 0$ and $\sigma = 3.3333$. With this distribution, let us suppose that we obtain two random numbers as $x_1 = -7.337$ and $x_2 = 0.53718$. Therefore, the new point is $X^{(1)} = (-7.337, 0.53718)$, and its function value is $f(X^{(1)}) = 542.11$.

Step 3: Checking the Acceptance of the New Point

At this step, we estimate the change in function value which is worked out as $\Delta E = f(X^{(1)}) - f(X^{(0)}) = 542.11 - 74 = 468.11 > 0$. Since $\Delta E > 0$, the acceptance of the new point is decided by the Metropolis algorithm. According to this algorithm, the probability of accepting new point (P) is $e^{-(468.1/1000)} = 0.6262$. We select a random number (P') between 0 and 1. Let us assume that $P' = 0.43091$. Since $P' < P$, the new point $X^{(1)} = (-7.337, 0.53718)$ is turned out to be accepted. We increment the iteration counter $i = 1$. This completes the first iteration of simulated annealing. In actual practice, a large number of iterations up to 1,000 is tried before lowering the temperature.

Step 4: Lowering the Temperature as per the Cooling Schedule

In this step, the temperature is lowered according to the cooling schedule. As we have selected the rate of cooling = 0.5, the new temperature (T) is set to $0.5 \times 1000 = 500$. Since $T > T_{\min}$, we go back to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

We randomly generate another point $X^{(2)}$ in the vicinity of the $X^{(1)}$. Using the same procedure as discussed in the preceding section, we generate two random numbers based on the normal distribution of two variables (x_1 and x_2) with $\mu = -7.337$ and 0.53718, respectively (refer to Figures 11.7 and 11.8). The value of σ for both variables is kept at 3.3333. Let us assume that we obtain two random numbers as $x_1 = -11.2205$ and $x_2 = 1.8084$. Since the value of variable x_1 is falling beyond the lower boundary, we reject this point. We set the iteration counter $i = 2$ and proceed to step 4. Therefore, after the second iteration, the new point $X^{(2)}$ remains at $X^{(1)}$.

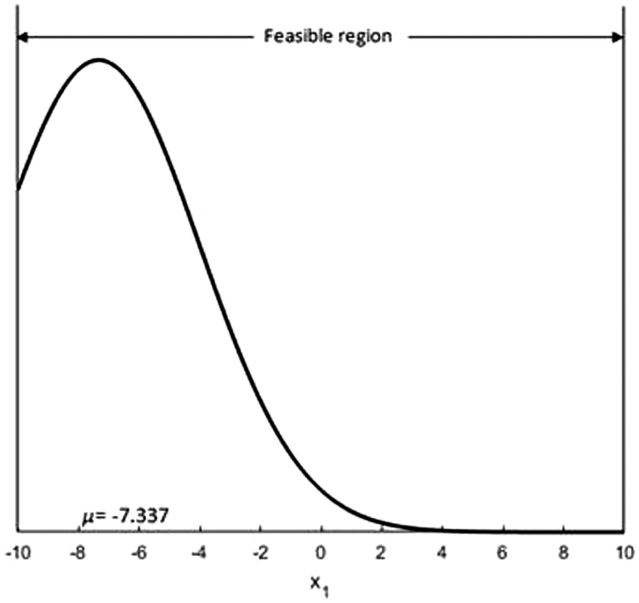

FIGURE 11.7
Normal distribution of variable x_1 with mean = −7.337.

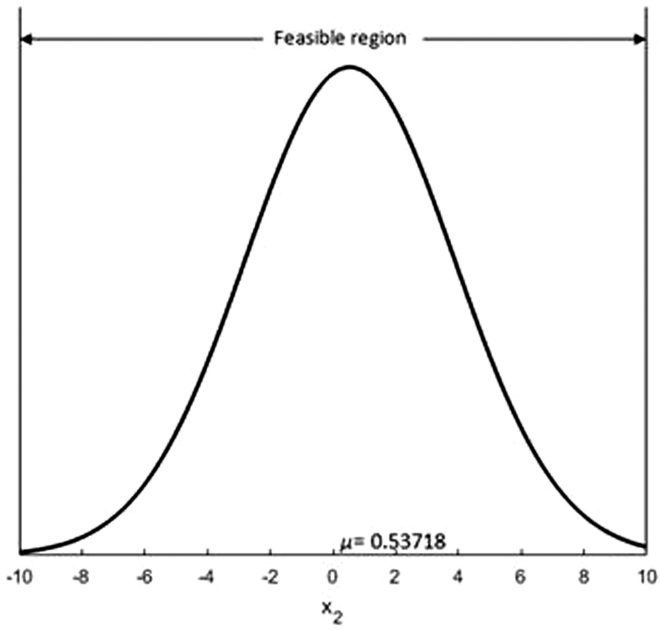

FIGURE 11.8
Normal distribution of variable x_2 with mean = 0.53718.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is reduced to $0.5 \times 500 = 250$. Since $T > T_{\min}$, we go back to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

The new point $X^{(3)}$ is again generated in the neighborhood of the current point $X^{(2)}$ following the same procedure as in the previous iteration. Suppose that the new point is $X^{(3)} = (-8.2649, 0.6315)$ and it has a function value $f(X^{(3)}) = 632.79$.

Step 3: Checking the Acceptance of the New Point

The change in function value is $\Delta E = f(X^{(3)}) - f(X^{(2)}) = 632.79 - 542.11 = 90.68 > 0$. Since $\Delta E > 0$, we generate a random number between the range (0, 1): $P' = 0.25622$. The probability of accepting the new point (P) is $e^{-(90.68/250)} = 0.69578$. Since $P' < P$, the new point $X^{(3)} = (-8.2649, 0.6315)$ will be accepted. We increment the iteration counter $i = 3$ and go to step 4.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is reduced to $0.5 \times 250 = 125$. Since $T > T_{\min}$, we go back to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

Suppose that the new point $X^{(4)} = (-10.5867, -1.9691)$ is found in the neighborhood of the current point $X^{(3)}$. This point is rejected; because the value of variable x_1 is falling beyond the lower boundary. We set the iteration counter $i = 4$ and proceed to step 4. Thus, after the fourth iteration, the new point $X^{(4)}$ remains at $X^{(3)}$.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is set to $0.5 \times 125 = 62.5$. Since $T > T_{\min}$, we go back to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

The new point $X^{(5)} = (-8.9895, 1.6117)$ is created in the neighborhood of the current point $X^{(4)}$, and it has a function value $f(X^{(5)}) = 619.54$.

Step 3: Checking the Acceptance of the New Point

The change in function value is $\Delta E = f(X^{(5)}) - f(X^{(4)}) = 619.54 - 632.79 < 0$. Since $\Delta E < 0$, point $X^{(5)}$ is accepted. We set iteration counter $i = 5$ and proceed to step 4.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is set to $0.5 \times 62.5 = 31.25$. Since $T > T_{\min}$, we proceed to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

The new point $X^{(6)} = (-5.4466, 2.3564)$ is generated in the vicinity of the current point $X^{(5)}$, and it gives a function value $f(X^{(6)}) = 243.05$.

Step 3: Checking the Acceptance of the New Point

The change in function value is $\Delta E = f(X^{(6)}) - f(X^{(5)}) = 243.05 - 619.54 < 0$. Since $\Delta E < 0$, point $X^{(6)}$ will be accepted. Thus, $i = 6$, and we move to step 4.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is set to $0.5 \times 31.25 = 15.625$. Since $T > T_{\min}$, we proceed to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

The new point $X^{(7)} = (-5.2255, -1.7712)$ is found in the vicinity of the current point $X^{(6)}$, and it gives a function value $f(X^{(7)}) = 545.22$.

Step 3: Checking the Acceptance of the New Point

The change in function value is $\Delta E = f(X^{(7)}) - f(X^{(6)}) = 545.22 - 243.05 = 302.17 > 0$. Since $\Delta E > 0$, we generate a random number between the range (0, 1): $P' = 0.086539$. The probability of accepting the new point (P) is $e^{-(302.17/15.625)} = 3.9924 \times 10^{-9}$. Since $P' > P$, the new point is rejected. We set the iteration counter $i = 7$ and go to step 4. Therefore, after the seventh iteration, the new point $X^{(7)}$ remains at $X^{(6)}$.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is reduced to $0.5 \times 15.625 = 7.8125$. Since $T > T_{\min}$, we proceed to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

The new point $X^{(8)} = (-3.2893, 5.386)$ is found in the vicinity of the current point $X^{(7)}$, which has a function value $f(X^{(8)}) = 38.581$.

Step 3: Checking the Acceptance of the New Point

The change in function value is $\Delta E = f(X^{(8)}) - f(X^{(7)}) = 38.581 - 545.22 < 0$. Since $\Delta E < 0$, point $X^{(8)}$ is accepted. We increment the iteration counter $i = 8$ and move to step 4.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is reduced to $0.5 \times 7.8125 = 3.9063$. Since $T > T_{\min}$, we proceed to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

The new point $X^{(9)} = (-1.4429, 2.6077)$ is found in the neighborhood of the current point $X^{(8)}$, which yields a function value $f(X^{(9)}) = 38.275$.

Step 3: Checking the Acceptance of the New Point

The change in function value is $\Delta E = f(X^{(9)}) - f(X^{(8)}) = 38.275 - 38.581 < 0$. Since $\Delta E < 0$, point $X^{(9)}$ is accepted. The iteration counter becomes $i = 9$, and we move to step 4.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is lowered to $0.5 \times 3.9063 = 1.9531$. Since $T > T_{\min}$, we go to step 2.

Step 2: Generation of a New Point in the Neighborhood of Current Point

The new point $X^{(10)} = (-5.0640, 0.2023)$ is found in the neighborhood of the current point $X^{(9)}$, and it yields a function value $f(X^{(10)}) = 358.72$.

Step 3: Checking the Acceptance of the New Point

The change in function value is $\Delta E = f(X^{(10)}) - f(X^{(9)}) = 358.720 - 38.275 = 320.44 > 0$. Since $\Delta E > 0$, we generate a random number between the range (0, 1): $P' = 0.98818$. The probability of accepting the new point (P) is $e^{-(320.44/1.9531)} \cong 0$. Since $P' > P$, the new point is rejected. We set iteration counter $i = 10$ and proceed to step 4. Hence, after the 10th generation, new point $X^{(10)}$ remains at $X^{(9)}$.

Step 4: Lowering the Temperature as per the Cooling Schedule

The temperature (T) is lowered to $0.5 \times 1.9531 = 0.97656$. Since $T < T_{\min}$, we terminate.

TABLE 11.1

A Summary of Search Points for First 10 Iterations

Iteration	Values of the Variables
$i = 0$	$X^{(0)} = (0, 0)$
$i = 1$	$X^{(1)} = (-7.337, 0.53718)$
$i = 2$	$X^{(2)} = (-7.337, 0.53718)$
$i = 3$	$X^{(3)} = (-8.2649, 0.6315)$
$i = 4$	$X^{(4)} = (-8.2649, 0.6315)$
$i = 5$	$X^{(5)} = (-8.9895, 1.6117)$
$i = 6$	$X^{(6)} = (-5.4466, 2.3564)$
$i = 7$	$X^{(7)} = (-5.4466, 2.3564)$
$i = 8$	$X^{(8)} = (-3.2893, 5.386)$
$i = 9$	$X^{(9)} = (-1.4429, 2.6077)$
$i = 10$	$X^{(10)} = (-1.4429, 2.6077)$

Table 11.1 gives a summary of the search points after each iteration. The progress of the search starting from the point (0, 0) up to the first 10 iterations is displayed in Figure 11.9.

The MATLAB coding for solving the aforesaid optimization problem given in Equation 11.3 using the simulated annealing method is presented in Section 11.5.

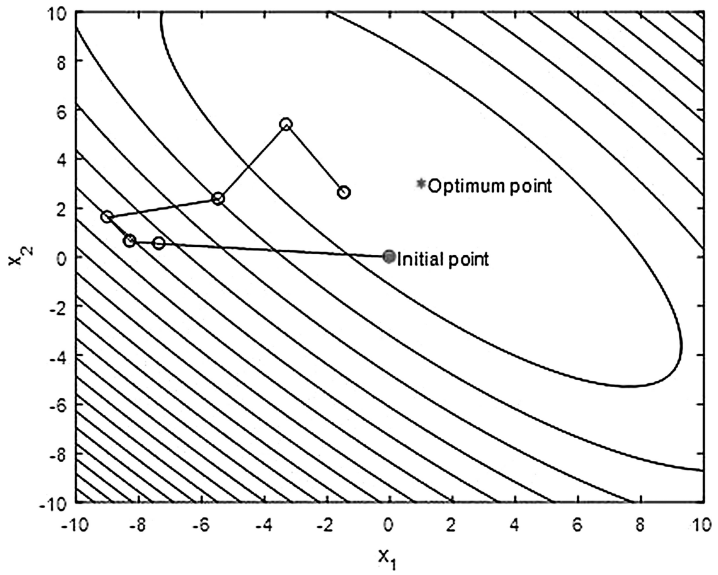

FIGURE 11.9
The progress of the search starting from the point (0, 0) up to the first 10 iterations.

11.4 Application of Simulated Annealing in Textiles

Das et al. (2012, 2013) used the simulated algorithm for engineering design of woven fabrics with minimum manufacturing cost and requisite quality. The optimization problem was formulated to minimize the fabric areal density (GSM) subjected to the inequality constraints composed of fabric tensile, bending, and shear properties, and an equality constraint of crimp balance. The optimization problem was solved using simulated annealing to find out the optimum values of governing parameters such as yarn counts, crimps, and threads spacing for the production of lightweight, medium weight, and heavyweight cotton fabrics. Das et al. (2014) also made an attempt for optimal product design in the textile spinning industry using simulated annealing.

Heckmann and Lengauer (1995) used a simulated annealing approach to the nesting problem in the textile manufacturing industry, which is basically a problem of placing a set of irregularly shaped pieces (called stencils) on a rectangular surface, such that no stencils overlap and the trim loss produced when cutting out the stencils is minimized. They pointed out that the simulated annealing algorithm has high performance and is easy to implement.

11.4.1 Application of Simulated Annealing in Yarn Engineering

Das and Ghosh (2015) used a simulated annealing algorithm for engineering design of cotton yarn with predefined strength by optimal selection of raw material and process parameters. The mechanistic model of yarn strength developed by Zurek (1975), Zurek et al. (1987), and Frydrych (1992) has been used for formulation of the optimization problem, which is discussed as follows.

The metric twist factor α_m is expressed as

$$\alpha_m = t\sqrt{\frac{Tt_y}{1000}} \tag{11.7}$$

where t is the yarn twist/meter, and Tt_y is the yarn linear density (tex).

The density of yarn ρ_y and α_m are related by the following expression (Barella, 1950):

$$\rho_y = 560 + 2.8\alpha_m \tag{11.8}$$

The nominal twist parameter is given as

$$g = 2\pi Rt = \sqrt{\frac{125.7}{\rho_y}}\left(\frac{\alpha_m}{100}\right) \tag{11.9}$$

where R is the yarn radius.

The equivalent fiber diameter is written as

$$d_f = \sqrt{\frac{4Tt_f}{\pi\rho_f}} \tag{11.10}$$

where Tt_f is linear density of a fiber (mg/m), and ρ_f is the fiber density (Kg/m^3); for cotton $\rho_f = 1520$ Kg/m^3.

The yarn diameter (in mm) can be expressed as

$$d_y = \sqrt{\frac{4Tt_y}{\pi\rho_y}} \tag{11.11}$$

The reduced twist parameter becomes

$$g_c = 2\pi\left(R - \frac{d_f}{4}\right)t = g\left(1 - \frac{1}{2}\frac{d_f}{d_y}\right) \tag{11.12}$$

The contraction factor is given by

$$s = \frac{\ln\sqrt{1+g_c^2}}{\sqrt{1+g_c^2}-1} \tag{11.13}$$

The coefficient of the yarn diameter contraction can be calculated according to the following formula:

$$u = 1.13 - \frac{0.0265}{g_c} - 0.12\sqrt[4]{100a_h} \tag{11.14}$$

Since a_h is unknown, it is first assumed that a_h is fiber breaking strain a_f, and the value of u is calculated. Knowing the approximated value of u, the strain of the breaking zone of the yarn is found from the following relationship:

$$a_h = \sqrt{\left(\frac{1+a_h}{s}\right)^2 - u^2 g_c^2} - 1 \tag{11.15}$$

Then a new value of u is calculated, and again a new value of a_h is found. This process is continued until the calculated values of u do not change.

The parameter of change of the fiber axis shape is written as

$$k = \frac{1+a_h}{u} \tag{11.16}$$

The critical value of the twist parameter is obtained as

$$g_r = \frac{g}{k}\left(1 - \frac{2}{u}\frac{d_f}{d_y}\right) \tag{11.17}$$

The radius of the fiber axis curvature in the external layer of yarn is given by

$$P = (Ru - d_f)\frac{1 + g_r^2}{g_r^2} \tag{11.18}$$

The z is a parameter that represents the change in yarn strength because the fibers are constrained from moving freely in the yarn. The numerical value z can be calculated from the formula as follows:

$$f(z) = \frac{4P}{\mu(c\eta l_f - \lambda)} = \frac{1 - z}{(2z+1)\ln\left(1 + \frac{1}{2z}\right) - 1} \tag{11.19}$$

where μ is the coefficient of friction between fibers; for cotton $\mu = 0.27$, c is a coefficient depending on the spinning system; $c = 1.1$ for combed yarn, η is the relative extent of fibers in yarn; $\eta = 0.9$ for combed yarn, l_f is the mean length of fibers, and λ is the length of fiber ends outside the yarn; it is assumed that $\lambda = 0.2$ mm.

The fracture zone length is expressed as

$$l_h = \frac{2P}{\mu}\ln\left(1 + \frac{1}{2z}\right) \tag{11.20}$$

The ratio of the specimen length y to the length of fracture zone l_h is

$$q = \frac{y}{l_h} \tag{11.21}$$

The parameter C depends on twist and the change in fiber axis shape while the yarn was being strained:

$$C = \frac{2k}{g^2}\ln\frac{\left[1.0253g^2 + 1\right]^{0.5} + \left[1.0253g^2 + k^2\right]^{0.5}}{\left[0.0253g^2 + 1\right]^{0.5} + \left[0.0253g^2 + k^2\right]^{0.5}} \tag{11.22}$$

The yarn stress at zero gauge length Q_h is calculated from the equation

$$Q_h = zQ(\varepsilon)C \tag{11.23}$$

where $Q(\varepsilon)$ is the breaking stress of the fiber.

The strength of a y mm long yarn sample Q_y is obtained by

$$Q_y = Q_h\left[1-3.64v_{Fh}\left(1-q^{-\frac{1}{7}}\right)\right] \tag{11.24}$$

where v_{Fh} is the variation coefficient of the breaking force in the length of the fracture zone, which can be approximately determined according to the relationship

$$v_{Fh} = \beta\sqrt{\frac{Tt_y}{Tt_f}} \tag{11.25}$$

where β is 1.35 for a combed spinning system. Here yarn specimen length y is taken as 500 mm.

The following optimization problem was formulated for 20 tex combed cotton yarn:

$$\begin{aligned} &\text{Minimize} && (Q_y - T_y)^2 \\ &\text{Subject to} && t^L \le t \le t^U \\ &&& Tt_f^L \le Tt_f \le Tt_f^U \\ &&& a_f^L \le a_f \le a_f^U \\ &&& Q(\varepsilon)^L \le Q(\varepsilon) \le Q(\varepsilon)^U \\ &&& l_f^L \le l_f \le l_f^U \end{aligned} \tag{11.26}$$

where Q_y is the predicted yarn strength, T_y is the target yarn strength, and the superscripts L and U refer to the values of lower and upper boundaries of the variables, respectively. The target value of yarn strength T_y was set to 20 gf/tex.

Simulated annealing was used to solve the optimization problem of Equation 11.26 in order to obtain the predefined yarn strength by searching the optimum values of cotton fiber parameters such as linear density Tt_f, breaking strain a_f, breaking stress $Q(\varepsilon)$, and mean length l_f, as well as yarn twist/meter t. The values of initial temperature T, minimum temperature $T_{\min}$, cooling rate C_T, and number of iterations at each temperature n for the simulated annealing algorithm were set to 1,000; 0.0001; 0.9; and 500, respectively.

Table 11.2 shows the lower and upper boundaries of the variables. Table 11.3 illustrates the optimum combination of fiber properties and twist/meter for the production of 20 tex yarns with target strength of 20 gf/tex. The value of yarn strength with the optimized parameters was obtained as 19.97 gf/tex.

TABLE 11.2

Boundary of Constraints

Controlling Parameters	Lower Boundary	Upper Boundary
Twist/meter (m^{-1})	650	700
Fiber linear density (tex)	0.120	0.150
Fiber breaking strain (%)	6.0	7.0
Average fiber length (mm)	25	33
Fiber breaking stress (gf/tex)	28	35

Source: Das, S. and Ghosh, A. 2015. *Fibres and Textiles in Eastern Europe*, 23(3), 51–53.

TABLE 11.3

Optimized Value of Constraints

Controlling Parameters	Optimized Values
Twist/meter	681.5
Fiber linear density (tex)	0.135
Fiber breaking strain (%)	6.75
Average fiber length	32.7
Fiber breaking strength (gf/tex)	34.3

Source: Das, S. and Ghosh, A. 2015. *Fibres and Textiles in Eastern Europe*, 23(3), 51–53.

11.5 MATLAB Coding

MATLAB Coding for Simulated Annealing

```
clc
clear all
close all
format short g
% Step-1: Initialization
x = [0  0];
vlb = [-10 -10];
vub = [10 10];
T = 1000;
Ct = 0.9;
T_min = 1e-6;
max_n = 1000;
i =0;
% Step-2: Generation of a new point in the neighborhood of
current point
sigma = [vub-vlb]/6;
while T > T_min
```

```
      for n = 1: max_n
          new_x = random('norm', x, sigma);
          if ((vlb <= new_x) & (new_x <= vub))
             fx = (x(1) + 2*x(2) -7).^2 + (2*x(1) + x(2)
-5).^2;
             fx_new = (new_x(1) + 2*new_x(2) -
7).^2+(2*new_x(1) + new_x(2) - 5).^2;
             % Step-3: Checking the acceptance of the new point
             del_E = (fx_new - fx);
               if del_E < 0
                  x = new_x;
                  else r = rand;
                  if r < = exp(-del_E / T)
                    x = new_x;
                  end
               end
            end
            i = i + 1;
         end
         % Step-4: Lowering the temperature as per the cooling
         schedule
         T = T*Ct;
end
i
x
f_val = (x(1) + 2*x(2) -7).^2 + (2*x(1) + x(2) -5).^2
```

The MATLAB code for simulated annealing was executed to solve the optimization problem given in Equation 11.3. We have set $X^{(0)} = (0, 0)$, $T = 1000$, $C_T = 0.9$, $T_{\min} = 10^{-6}$, and $n = 1000$. The temperature is reduced in steps, which has a geometric progression as follows:

$$1000, 1000 \times 0.9, 1000 \times 0.9^2, \ldots, 1000 \times 0.9^{(s-1)}$$

where s is the number of steps associated with temperature reduction. It can be seen that just after 197 steps, the temperature becomes less than 10^{-6}. As 1,000 iterations have been performed at a particular temperature, the program terminates after $197 \times 1000 = 197{,}000$ iterations. Figure 11.10 shows the progress in reducing the value of objective function over 197 steps, whereas Figure 11.11 depicts the same over 197,000 iterations. The algorithm eventually converges to the point (1.002, 2.994), which is very close to the actual optimum point (1, 3). The progress of the algorithm starting from the initial point to the end point is illustrated in Figure 11.12. It is clearly evident from Figure 11.12 that the search covers the whole space of variables; however, it ultimately concentrates in the neighborhood of the optimum point.

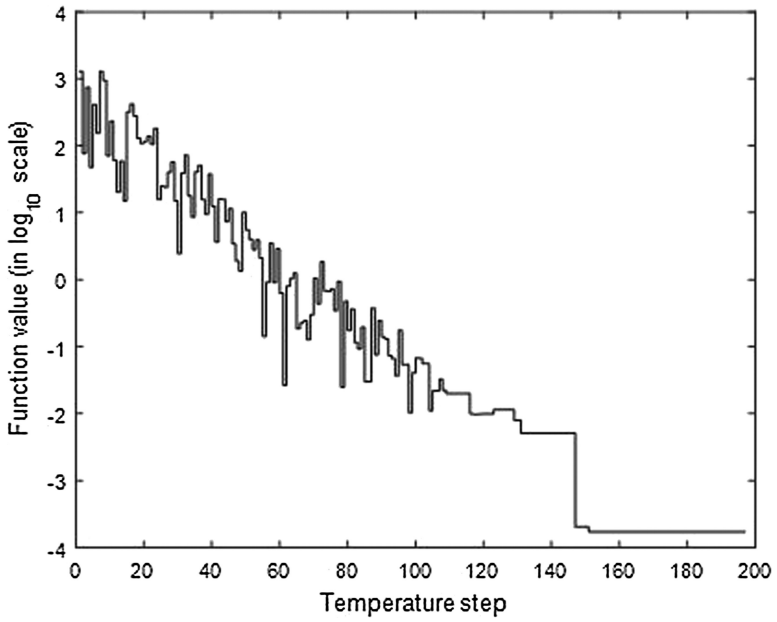

FIGURE 11.10
The progress of function value reduction with temperature step.

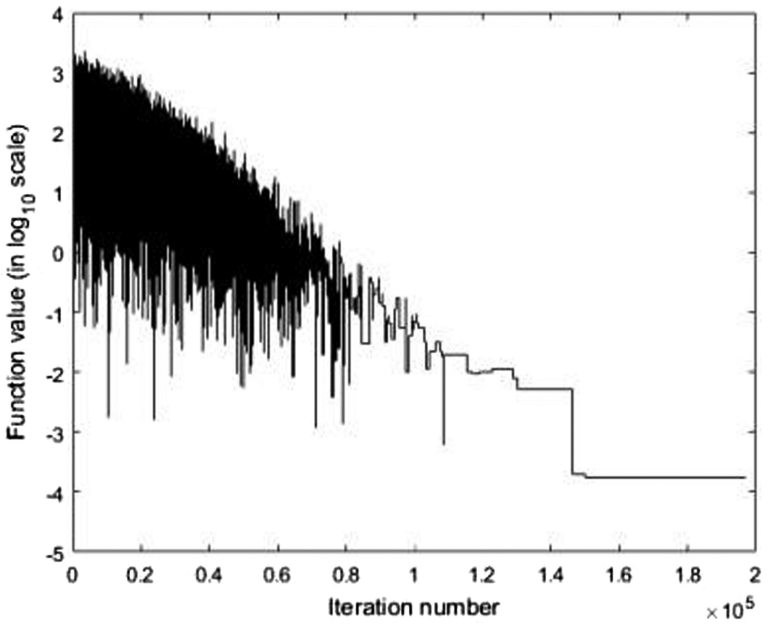

FIGURE 11.11
The progress of function value reduction with iteration number.

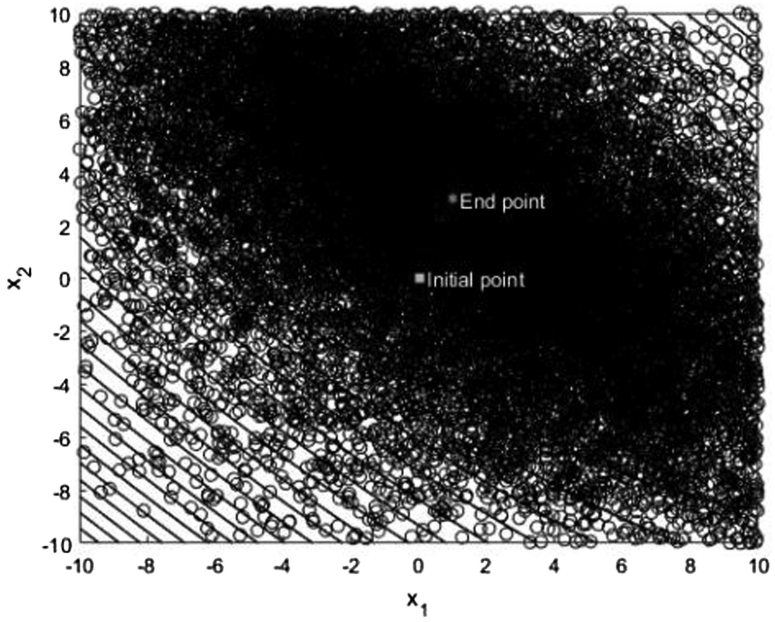

FIGURE 11.12
The progress of the search starting from the initial point to the end point.

11.6 Summary

An overview of the different steps of a simulated annealing algorithm is presented in this chapter. A worked-out problem brings out every step of the simulated annealing algorithm along with its coding in the MATLAB language. A survey of literature on the application of simulated annealing in textiles shows only little reported work is available. Finally, the results of a study carried out by the author on the application of simulated annealing in a yarn engineering problem are discussed.

References

Barella, A. 1950. Law of critical yarn diameter and twist influence on yarn characteristics. *Textile Research Journal*, 20, 249–253.

Das, S. and Ghosh, A. 2015. Cotton fibre-to-yarn engineering: A simulated annealing approach. *Fibres and Textiles in Eastern Europe*, 23(3), 51–53.

Das, S., Ghosh, A. and Banerjee, D. 2012. Selection of parameters for engineering design of woven fabrics using simulated annealing. *International Conference on Innovations in Engineering and Technology for Sustainable Development*, Sathyamangalam, India, September 3–5.

Das, S., Ghosh, A. and Banerjee, D. 2013. Engineering design of woven fabrics using non-traditional optimization methods: A comparative study. *Fibers and Polymers*, 14(9), 1562–1567.

Das, S., Ghosh, A. and Saha, B. 2014. Optimal product design of textile spinning industry using simulated annealing. *Fourth International Conference on Soft Computing for Problem Solving*, Silchar, India, December 27–29, 315–323.

Deb, K. 2005. *Optimization for Engineering Design: Algorithms and Examples*, Prentice Hall, New Delhi, India.

Frydrych, I. 1992. A new approach for predicting strength properties of yarn. *Textile Research Journal*, 62(6), 340–348.

Heckmann, R. and Lengauer, T. 1995. A simulated annealing approach to the nesting problem in the textile manufacturing industry. *Annals of Operations Research*, 57(1), 103–133.

Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P. 1983. Optimization by simulated annealing. *Science*, 220, 671–680.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. and Teller, E. 1953. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21, 1087–1092.

Zurek, W. 1975. *The Structure of Yarn*, Foreign Scientific Publications Department of the National Center for Scientific Technical and Economic Information, Warsaw, Poland.

Zurek, W., Frydrych, I. and Zakrzewski, S. 1987. A method of predicting the strength and breaking strain of cotton yarn. *Textile Research Journal*, 57, 439–444.