Journal of Hyperstructures 7 *(Spec. 2nd CSC2017)* (*2018*), *60-66.*
ISSN: 2322-1666 print/2251-8436 online

SOLVING ILL-CONDITIONED LINEAR EQUATIONS USING SIMULATED ANNEALING METHOD

MOJTABA MORADI

Abstract. The purpose of this paper is to using the Simulated Annealing method to solving a linear equations system which have an ill-conditioned coefficients matrix. A linear equation system is called ill-conditioned if its condition number be large. By using a matrix scaling, the linear equation system transforms into a linear equation system with less condition number. Matrix balancing is performed by Simulated Annealing algorithm. The efficiency of this method is investigated by numerical examples. Numerical results show that Simulated Annealing can reduce the condition number of equations.

Key Words: Condition Number, Matrix Scaling, Simulated Annealing, Linear Equations System.

2010 Mathematics Subject Classification: 15A06 , 65F35, 65F22, 68W20.

1. Introduction

Simulated Annealing algorithm (SA) is a local search algorithm that can escape from local optima. Simplicity of implementation and convergence properties, accepting worse solutions to jump out of any local optimums has made SA one of the most widely used methods in the last two decades. The first idea, which later became the basis of the SA algorithm, was first introduced by Metropolis in 1953 based on the process of cooling and annealing of materials (mainly crystalline metals). But this algorithm was first formally introduced by Kirkpatrick et al., inspired by

Received: 15 January 2018, Accepted: 21 January 2018. Communicated by A. Yousefian Darani;

∗Address correspondence to M. Moradi; E-mail: mmoradi@guilan.ac.ir.

the annealing phenomenon. Of course, after two years from KirkPatrick, Cerny also stated in an independent article that he developed this algorithm for the first time [1]. In statistical mechanics, the relationship between atomic structure, disorder (or entropy) and temperature during the process of cooling the material are studied. Annealing is a heat treatment process to above the upper critical temperature and slow cooling down. In physics, compressing, heating, and cooling of material are a physical process in which a solid is heated in a container to become fluid, then its heat gradually decreases. In this way, all particles can arrange themselves at the lowest level of energy. This situation occurs when the heating be sufficient and the cooling be also slow.

The result of Simulated Annealing algorithm is not dependent on the initial state, and it can be obtained by finding a value close to the optimal solution. The upper limit of the execution time of the algorithm can also be determined. In each step of the SA algorithm, a solution is created and is compared with the best obtained solution. Better solutions are always accepted, while part of non-improving solutions are also accepted in order to avoid getting stuck in local optima. The probability of accepting non-improving solutions depends on the temperature parameter. The key feature of SA is that using of hill climbing moves to escape the trap of local optima. By decreasing the temperature parameter to zero (increasing repetitions), the melting movements will be less and less and the distribution of the corresponding solutions will be heterogeneous with a consistent Markov Chain, whose stable states will be the same as the optimal global solutions, which requires that the extreme probabilities of these states greater than zero. Therefore, in optimization problems, temperature is used as a control parameter. [2, 3]

The steps taken by the SA algorithm to solving a minimizing problem can be summarized as follows:

Step 0) Before starting the algorithm, specify the following: Cooling schedule (including number of algorithm repeats per temperature, M) frequency of temperature changes, temperature reduction mechanism, initial temperature value $t_0 \geq 0$, final temperature value, method of producing neighboring solutions.
Step 1) Create random initial solution, $w \in \Omega$
Step 2) Set $k = 0$ as the initial value of the frequency change of the temperature.

Step 3) Set $m = 0$ as the number of repetitions at each temperature.
Step 4) Create a new solution in the neighborhood of the current solution, $(w' \in N(w))$.
Step 5) Calculate the value of $\Delta_{w,w'} = f(w') - f(w)$.
Step 6) If $\Delta_{w,w'} \leq 0$ then replace w by w', otherwise replace w by w' with probability $\exp(-\Delta_{w,w'}/t_k)$. In any case, save the best solution.
Step 7) $(m \leftarrow m + 1)$
Step 8) If $m = M$, go to Step 9, otherwise go to Step 4.
Step 9) Set $(k \leftarrow k + 1)$ and reduce the temperature according to the cooling schedule.
Step 10) If the condition for termination of the algorithm is established, stop the algorithm and find the best solution as the optimal algorithm's solution, otherwise go back to Step 3.

One of SA's characteristics is no working with multiple solutions at each iteration over demographic algorithms such as Genetic Algorithm. It should be noted that it cannot be claimed that the SA algorithm is the fastest and most efficient method for solving the optimization problems, but it is simpler to apply over other algorithms such as an ant colony algorithm or neural networks.

Worse solutions that may lead to a global optimum are selected with special probability. With repeating the algorithm, this probability become less and less, until we feel that there will be no further progress, for example, in two consecutive repetitions of this algorithm, the difference between the two solutions will be less. In this case, we reach the stability temperature.

In determining the parameters of this algorithm, we should note the following points: If T be small at first generations, the algorithm stops in local solutions. Because at the beginning of the algorithm, the chance of accepting a solution with a higher energy level is reduce. If T be large at first generations, many "bad" trades are accepted, and a large part of solution space is accessed and it will leading to a waste of time at the beginning of the algorithm. Because the probability of accepting solutions with higher energy levels will be constantly randomly walk from one solution to another. [1]

2. Using Simulated Annealing algorithm for solving linear equations

Consider linear equation system $Ax = b$, which A is a nonsingular and an ill-conditioned matrix. A small change in b or A may change

the vector of the solution completely. The measure of ill-condition of a matrix, defined as:

$$cond(A) = ||A||\ \ ||A^{-1}||.$$

If the condition number is large, then the matrix is said to be **ill-conditioned**. Instead of solving $Ax = b$, that A is an ill-conditioned matrix, one can solve the linear equation system $(D_1 A\, D_2)\, y = D_1\, b$ and then obtain the value of $x = D_2 y$. D_1 and D_2 are diagonal matrices that must be chosen so that the matrix $D_1 A D_2$ has a condition number less than the matrix A. In this case, we say that the matrix scaling is taken. matrix scaling was first proposed by Sinkhorn in 1964 by publishing an article on balancing certain positive matrices [4].

Matrix scaling is often done by multiplying each row and column in a scalar. That is, it is often assumed to D_1 and D_2 be diagonal matrices.

The purpose of this paper is to using Simulated Annealing algorithm to determine D_1 and D_2 such that the condition number of the matrix $D_1 A D_2$ is minimized. The problem of optimizing matrix scaling has been studied by Baur, Braatz and Murari, Buzinger, and other authors [8, and 9].

The goal of this paper is solving min : $\|D_1 A D_2\|$, where it is a single-objective problem. If the square matrix A be $n \times n$, then the problem variables are equal to $2n$. That is, the unknowns $d_{11}, d_{12}, ..., d_{1n}$ (elements of the main diagonal of D_1) and $d_{21}, d_{22}, ..., d_{2n}$ (elements of the main diagonal of D_2) must be determined. The algorithm for solving this problem is as follows:

1) Get square matrix A and set $t_0 = 100$; $M = 3$.
2) Generate D_1 and D_2 as the initial solution randomly from a uniform distribution on$(-100, 100)$.
3) Set $m = 0$.
4) Generate two vectors d_1 and d_2 randomly according to uniform distribution on $(-0.1\,,\ 0.1)$, and then calculate the new solutions $ND_1 = D_1 + d_1$ and $ND_2 = D_2 + d_2$.
5) Calculate the value of $\Delta = \|ND_1\ A\ ND_2\| - \|D_1 A D_2\|$.
6) If $\Delta < 0$, new solution is accepted, otherwise accept the new solution with probability $\exp(-\Delta/t_k)$. Save new solution in D_1 and D_2.
7) Set $m \leftarrow m + 1$.
8) If $m < M$ then go to 4.
9) Set $k \leftarrow k + 1$and $t_k = 0.95 \times t_{k-1}$.

10) If $k < 100$ then go to 4.
11) Report the last accepted solution as optimal solution.

3. Numerical Results

It is well know that the condition number of each matrix is always greater than or equal to 1. We want to show that, if possible, Simulated Annealing method has the ability to reduce the condition number of a matrix close to 1 using the concept of scaling.

Example 1: The condition number of

$$A = \begin{pmatrix} 10^5 & 0 \\ 0 & 10^{-5} \end{pmatrix}$$

is large and equal to 10^{10}.

The diagonal matrices are obtained using Simulated Annealing as follows.

$$D_1 = \begin{pmatrix} 10.310808169156953 & 0 \\ 0 & 145.5323354581699 \end{pmatrix}$$

And

$$D_2 = \begin{pmatrix} -1.253708204201532\text{e}-07 & 0 \\ 0 & -88.823866894639240 \end{pmatrix}$$

that $cond(D_1 A D_2) = 1.000000000001375$ is close to 1.

Example 2: The condition number of following matrix is 61.983866769659270.

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 6 \end{pmatrix}$$

From [5], we know that the lowest possible value (optimal value) of the condition number obtained from the matrix scaling is equal to $(3 + \sqrt{10})^2 = 37.973665961010280$. We now want to examine the efficiency of the Simulated Annealing algorithm for this matrix. The results of this method are as follows:

$$D_1 = \begin{pmatrix} 407.6211814788877 & 0 & 0 \\ 0 & 373.1143253579575 & 0 \\ 0 & 0 & -192.4281752057432 \end{pmatrix}$$

and

$$D_2 = \begin{pmatrix} -234.7621137716025 & 0 & 0 \\ 0 & -214.2286722377142 & 0 \\ 0 & 0 & 111.0194306184976 \end{pmatrix}$$

and

$$cond(D_1AD_2) = 37.973967559242126$$

The absolute error is equal to$3.0160\mathrm{e}-04$ and the relative error is equal to $7.9423\mathrm{e}-06$.

Example 3: The condition number of following matrix is equal to 460.2704442412975.

$$A = \begin{pmatrix} 0.0926612 & 17.0784926 & 0.3127063 & 12.7526810 \\ 1.7811361 & 54.0213314 & 1.4953060 & 14.7655003 \\ 0.3460217 & 0.0680433 & 0.2626770 & 0.0227214 \\ 1.3745248 & 45.1500312 & 0.0505958 & 1.4314422 \end{pmatrix}$$

Chin-Chieh and Ch., John in 2008, using a duplicate algorithm called SCALGM, managed to reduce the condition number to14.4856257. Using Simulated Annealing method, we were able to find a better solution (though close to their solution). The scaled matrix has a condition number equal to 14.48547757. The diagonal matrices D_1 and D_2 respectively are as follows:

$$\begin{pmatrix} -0.5834716587683 & 0 & 0 & 0 \\ 0 & 0.2831906690374 & 0 & 0 \\ 0 & 0 & 1.5658855853424 & 0 \\ 0 & 0 & 0 & 0.3113680681914 \end{pmatrix}$$

and

$$\begin{pmatrix} 0.5906753296597 & 0 & 0 & 0 \\ 0 & -0.0189122833584 & 0 & 0 \\ 0 & 0 & 0.7391127768269 & 0 \\ 0 & 0 & 0 & 0.0529508083825 \end{pmatrix}$$

4. Conclusion

The ill-conditioned linear equation system is sensitive and small changes may produce a new solution that is very different from the previous one. On the other hand, the round-off error may leads to an error in the solution as a very minor change. For this reason, providing an approach that impedes the growth of errors is especially important.

The condition number of coefficient matrix in a linear equation system was introduced as a measure of ill-conditionality. By multiplying the left and right of the coefficient matrix by $D_1$and D_2, we arrived the D_1AD_2 that the condition number of this matrix is the lowest possible (optimal value) The elements of $D_1$and D_2 was determined by Simulated

Annealing algorithm. This algorithm is able to reduce the condition number based on the concept of matrix scaling.

References

[1] M. C. Trosset , *What is Simulated Annealing*, Optimization and Engineering, **2** (2001), 201–203.

[2] C. Blum and A. Roli, *Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison*, ACM Computing Surveys, **35** (2003), 268–308.

[3] F. Glover and G. Kochenberger, *Handbook of Metaheuristics*, Kluwer Academic Publishers, Norwell (2002).

[4] R. Sinkhorn, *A relationship between arbitrary positive matrices and stochastic matrices*, Annals of Mathematical Statistics, **35** (1964), 876–879.

[5] F. L. Bauer, *Optimally scaled matrices*. Numerische Mathematik, **5** (1963), 73–87.

[6] D. R Braatz and M. Morari, *Minimizing the Euclidean Condition Number*. SIAM Journal on Control and Optimization, **32** (1994), 1763–1768.

[7] P. A. Businger, *Matrices Which Can Be Optimally Scaled*. Numerische Mathematik , **12** (1968), 346–348.

[8] Ch. Chin-Chieh and P. Ch. John *An Approximate Equation for the Condition Numbers of Well-scaled Matrices*. Proceedings of The 2008 IAJC-IJME International Conference.

[9] G. A. Watson, *An Algorithm for Optimal*. Scaling of Matrices. IMA J. Numer. Anal , **11** (1991), 481–492.

Mojtaba Moradi
Department of industrial engineering, University of Guilan, Rudsar, Iran
Email: mmoradi@guilan.ac.ir