The implementation of simulated annealing combining gradient search in system identification

Yiqun Zou * **William Heath** **

* *School of Electrical and Electronic Engineering, University of Manchester, UK, M60 1QD(e-mail: Yiqun.Zou@postgrad.manchester.ac.uk).*
** *School of Electrical and Electronic Engineering, University of Manchester, UK, M60 1QD (e-mail: William.Heath@manchester.ac.uk).*

Abstract: A two-stage algorithm is proposed for system identification using a maximum likelihood criterion. The first stage is a modified simulated annealing algorithm that ensures the solution avoids local minima; the algorithm is tailored for the parameter identification problem. The second stage is a standard gradient descent algorithm that ensures fast and accurate convergence to the optimum. Simulation results are presented for both linear and nonlinear system identification. The performance is compared with a breeder genetic algorithm in both cases.

Keywords: System Identification, Simulated Annealing, Optimization

1. INTRODUCTION

N.B. Our main interests in this paper will be on time-invariant single input and single output dynamic system.

Many methods have been proposed in the field of system identification in last fifty years. Among them, maximum likelihood estimate might be one of the most popular techniques applied in practice. The basic theory of maximum likelihood estimate is summarized as: The information of the probability density functions $P(Y|\hat{\theta})$ of a series of random variables $Y = [\, y_1 \, \dots \, y_N \,]$ on the basis of the parameter estimate vector $\hat{\theta}$ is available. When Y is particularly known, the likelihood function is $L(\hat{\theta}) = P(Y|\hat{\theta})$. We regard $\hat{\theta}$ as the best estimate for true system dynamics θ_o if L is maximized at $\hat{\theta}$ under some mild assumptions[2].

We pick up a general family of model structure in [3] for demonstration

$$A(q)y(t) = \frac{B(q)}{F(q)}u(t) + \frac{C(q)}{D(q)}e(t) \tag{1}$$

where $y(t)$ $u(t)$ and $e(t)$ denote output, input signal and white noise. In estimation, the prediction error

$$\epsilon(t,\hat{\theta}) = y(t) - \hat{y}(t|\hat{\theta}) \tag{2}$$

also can be expressed as

$$\epsilon(t,\hat{\theta}) = \frac{\hat{D}(q)}{\hat{C}(q)}(\hat{A}(q)y(t) - \frac{\hat{B}(q)}{\hat{F}(q)}u(t)) \tag{3}$$

For convenience, we introduce the auxiliary variables

$$w(t) = \frac{\hat{B}(q)}{\hat{F}(q)}u(t) \tag{4a}$$

$$v(t) = \frac{\hat{C}(q)}{\hat{D}(q)}\epsilon(t) \tag{4b}$$

where

$$\begin{aligned}
\hat{A}(q) &= \quad 1 + \hat{a}_1 q^{-1} + \ldots + \hat{a}_{n_a} q^{-n_a} && (5a)\\
\hat{B}(q) &= \quad \hat{b}_1 q^{-1} + \ldots + \hat{b}_{n_b} q^{-n_b} && (5b)\\
\hat{C}(q) &= \quad 1 + \hat{c}_1 q^{-1} + \ldots + \hat{c}_{n_c} q^{-n_c} && (5c)\\
\hat{D}(q) &= \quad 1 + \hat{d}_1 q^{-1} + \ldots + \hat{d}_{n_d} q^{-n_d} && (5d)\\
\hat{F}(q) &= \quad 1 + \hat{f}_1 q^{-1} + \ldots + \hat{f}_{n_f} q^{-n_f} && (5e)
\end{aligned}$$

Here q is the forward shift operator with the superscript $\hat{}$ denoting the estimate efficient. Figure 1 illustrates the general structure of identification. Transforming (2) to its vectorial expression in terms of the coefficients for the system filters in (5), we get

$$\epsilon(t,\hat{\theta}) = y(t) - \varphi^T(t,\hat{\theta})\hat{\theta} \tag{6}$$

and

$$\hat{y}(t|\hat{\theta}) = \varphi^T(t,\hat{\theta})\hat{\theta} \tag{7}$$

where $\varphi^T(t,\hat{\theta}) = [\, -y(t-1) \, \dots \, -y(t-n_a) \; u(t-1) \, \dots \, u(t-n_b) \; \epsilon(t-1,\hat{\theta}) \, \dots \, \epsilon(t-n_c,\hat{\theta}) \; -v(t-1,\hat{\theta}) \, \dots \, -v(t-n_d,\hat{\theta}) \; -\omega(t-1,\hat{\theta}) \, \dots \, -\omega(t-n_f,\hat{\theta}) \,]$ while $\hat{\theta} = [\, \hat{a}_1 \, \dots \, \hat{a}_{n_a} \; \hat{b}_1 \, \dots \, \hat{b}_{n_b} \; \hat{c}_1 \, \dots \, \hat{c}_{n_c} \; \hat{d}_1 \, \dots \, \hat{d}_{n_d} \; \hat{f}_1 \, \dots \, \hat{f}_{n_f} \,]^T$. We treat $Y = [\, y(1) \, \dots \, y(t-1) \; u(1) \, \dots \, u(t) \,]$ as a given set of data. Then the likelihood function turns to be

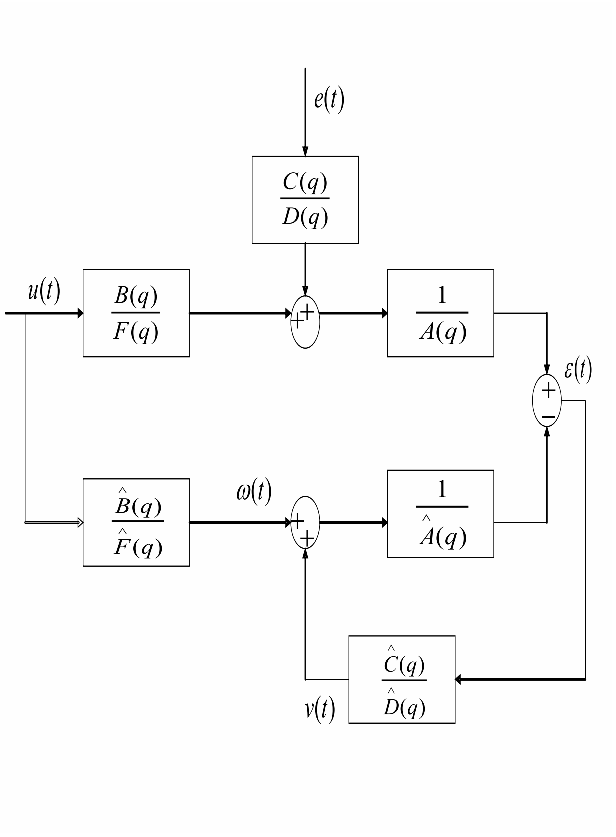

Fig. 1. General Identification Model

$$L(\hat{\theta}) = \prod_{t=1}^{N} P(y(t)|\ y(1);\ \ldots;\ y(t-1);\ \hat{\theta})$$
$$= \prod_{t=1}^{N} f(\epsilon(t,\hat{\theta}),\hat{\theta}) \quad (8)$$

where $f(\epsilon(t,\hat{\theta},\hat{\theta}))$ is the probability density function of $\epsilon(t)$ assumed to be independent and normally distributed. Furthermore the maximization of (8) could alternatively be achieved by minimizing its simplified negative log-form

$$l(\hat{\theta}) = constant + \frac{1}{2N}\sum_{t=1}^{N}\epsilon^2(t) \quad (9)$$

Based on the knowledge above, estimation problem has been converted into optimization of the loss function

$$V_N(\hat{\theta}) = \frac{1}{2N}\sum_{t=1}^{N}\epsilon^2(t) \quad (10)$$

Common gradient-based methods[4][5] utilizing $\frac{\partial V_N(\hat{\theta})}{\partial \hat{\theta}}$ work effectively well on optimizing some model cases. However, for other models, local minimum points of the loss function may exist. A following survey for the non-existence of local convergence suggests
(1). The landscape of **ARMA** models($B(q) = D(q) = F(q) = 1$ for 3)is convex[6].
(2). There will be no local minimum point for **ARX** models ($C(q) = D(q) = F(q) = 1$)[2].
(3). There will also be no local minimum point for **ARARX** models($C(q) = F(q) = 1$) model provided a large enough signal-noise-ratio(**SNR**)[7].
(4). For **OE** models($A(q) = C(q) = D(q) = 1$), the landscape is convex when the input signal $u(t)$ is white noise[7].
(5). For **BJ** models($A(q) = 1$), if $F(q)$ can be written as $1 + f_1q^{-1}$, there will be no local minimum point[7].
When the conditions above are not satisfied, gradient search could be possibly stuck at the basin of those stationary points. In this case we will be misled by wrong description of the system.

To tackle this dilemma, a few so-called "direct search" (gradient free) techniques are broadly examined in recent years, such as genetic algorithm[8][9] and Tabu algorithm[10]. However compared to the above algorithms, simulated annealing always plays an auxiliary role[11][12] rather than the "backbone" as it is in other areas[13]. After scrupulous scrutiny, we discover as a powerful optimization technique, simulated annealing can be brought in system identification. In this paper, we will present a novel independent implementation of simulated annealing followed by iterative gradient search to secure the final estimate accuracy in system identification.

2. THE SCHEME OF SIMULATED ANNEALING WITH GRADIENT SEARCH IN SYSTEM IDENTIFICATION

The term "SIMULATED ANNEALING" appeared in [14]. However its basic idea was already included in Metropolis-Hastings algorithm[15] dating back to 1953. The mechanism of simulated annealing is to decide whether to replace the current solution with the trial solution selected randomly in the nearby neighborhood after each iteration decided by the difference between the cost function, i.e., the loss function in the scope of system identification, values of two solutions and annealing temperature T. This decreases gradually in the annealing period. At the start when T_o is moderately large, there is a very high probability that some "uphill" movements will be accepted. Once T asymptotically approaches zero, only "downhill" replacement will occur. In short, simulated annealing algorithm is like a rolling ball over a mountainous landscape as its energy continuously reduces on the way.

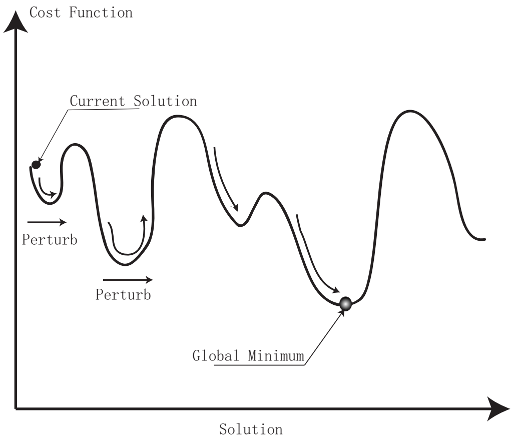

Fig. 2. General Simulated Annealing in $2-D$ landscape[16]

Before directly bringing in normal simulated annealing to system identification, we have to ensure the stability of the criterion $V_N(\hat{\theta})$, i.e., to maintain the roots of estimation filters polynomials $\hat{C}(q)$ $\hat{D}(q)$ and $\hat{F}(q)$ in the unit circle(see (1) and (3)). A so-called "Threshold Acceptance Rule"[17] typically designed which is characterized by

$$P_{acc} = \begin{cases} 1 & if\ \Delta V \leq T \\ 0 & if\ \Delta V > T \end{cases} \tag{11}$$

is eligible for this goal. To show this, first let us define $\hat{\theta}_{now}$ and $\hat{\theta}_{trial}$ as a stable current and its neighboring trial filter vector individually. In (6), the forms of filters are transformed from polynomials(see (5)) into the column vector $\hat{\theta}$. Thus a tight stability control over the update of $\hat{\theta}$ especially on the vectorial bit $[\ \hat{c}_1\ \ldots\ \hat{c}_{n_c}\ \hat{d}_1\ \ldots\ \hat{d}_{n_d}\ \hat{f}_1\ \ldots\ \hat{f}_{n_f}\ ]$ should be enforced. Since the stability of filters is judged by whether $V_N(\hat{\theta}_{trial})$ is bounded or not, the job could be done by comparing the difference improvement ΔV between $V_N(\hat{\theta}_{trial})$ and $V_N(\hat{\theta}_{now})$ with the annealing temperature T. As it is seen in (11), if ΔV is less than T in which case it means $\hat{\theta}_{trial}$ is stable, the update will be granted. All the potential updates leading to unstable results will be discarded in the probability of 1. Meanwhile the current $\hat{\theta}_{now}$ is retained until the next stable neighboring candidate is found or simulated annealing ceases. Generally, provided certain large temperature "T" and infinite iterative times, the algorithm will guarantee the final convergence to the global minimum point(true system values). In summary, we suggest the following steps for the algorithm:

Stage 1: Select initial values Algorithms normally should be assigned initial values to start off operation. In this particular case, we choose the parameter vector θ and the primitive cooling temperature T_o, etc.

Stage 2: Choose cooling scheme To speed up the computation, we choose fast annealing technique[18]:

$$T = \frac{To}{1+k} \tag{12}$$

where k indicates the iteration time.

Stage 3: Calculate trial solution in neighborhood In this step, a trial solution is picked up randomly in the nearby neighborhood of θ. A scalar constant is set to adjust the step size of the change between θ_{tri} and θ. Using MATLAB language, the numerical relationship is described as below:

$$\hat{\theta}_{trial} = \hat{\theta}_{now} + \lambda \times randn(dim(\hat{\theta}_{now}), 1) \tag{13}$$

where $dim(\ )$ denotes the dimension of vector $\hat{\theta}_{now}$. Empirically, we let λ equal to 0.1 here.

Stage 4: Replacement Judgement Based on $\hat{\theta}_{trial}$ and $\hat{\theta}_{now}$, we compare ΔV between $V_N(\hat{\theta}_{trial})$ and $V_N(\hat{\theta}_{now})$ with T. The replacement $\hat{\theta}_{now} = \hat{\theta}_{trial}$ will be approved if and only if $\Delta V \leq T$ holds. Otherwise the same $\hat{\theta}$ will be taken into next iteration.

Stage 5: Ending Rule of Simulated Annealing Terminate the simulated annealing if the stopping criterion below is meet or a maximum iteration time is reached. Otherwise go back to Stage 2 to continue the programme. The stop criterion is

$$\frac{|V_N(\hat{\theta}_i) - V_N(\hat{\theta}_{i-n})|}{V_N(\hat{\theta}_i)} < \varepsilon \tag{14}$$

where $V_N(\hat{\theta}_i)$ and $V_N(\hat{\theta}_{i-n})$ both are the loss function values on different iteration times between which $V_N(\hat{\theta}_i)$ is behind $V_N(\hat{\theta}_{i-n})$ n times. And ε is a very small positive constant. Both n and ε are previously defined.

Stage 6: Iterative Gradient Search Our algorithm is further extended by normal iterative gradient search[3]. The iterations are stopped when the expected improvement of loss function value measured in percent at next iteration is less than the tolerance which is set to be $\bar{\varepsilon}$ beforehand.

3. LINEAR SYSTEM IDENTIFICATION

To show the advantage of simulated annealing, it will be compared with a breeder genetic algorithm[1] as a benchmark in the scope of both linear and nonlinear system identification illustrated by individual example. In addition, if the landscape is smooth, it is appropriate to terminate both algorithms with gradient search.

In linear identification only, we provide system order and time delay already available, and employ simulated annealing with gradient search method as compared with genetic algorithm in identification. The first linear case in the open loop is governed by an $\boldsymbol{OE}(A(q) = C(q) = D(q) = 1)$ model where

$$B_o(q) = \quad q^{-1} \tag{15a}$$

$$F_o(q) = \quad 1 - 1.4q^{-1} \quad +0.4q^{-2} \tag{15b}$$

$$u(t) = \quad (1 - 0.98q^{-2} \quad +0.2401q^{-4} \quad)v(t) \tag{15c}$$

are given. Here $v(t)$ and $e(t)$ are both white noise with q the forward shift operator. It is argued by [7] that the loss function $V_N(\theta)$ for (14) has one local minimum point whose polynomials are

$$\hat{B}(q) = \quad -0.23q^{-1} \tag{16a}$$

$$\hat{F}(q) = \quad 1 + 1.367q^{-1} \quad +0.513q^{-2} \tag{16b}$$

Hence we could assign the initial vector $\hat{\theta}_{ini} = [\ b_1\ \ldots\ b_{n_b}\ f_1\ \ldots\ f_{n_f}\ ]^T = [\ -0.23\ 1.367\ 0.513\ ]^T$.

The trace of loss function on logarithmic scale $V_N(\hat{\theta})$ for simulated annealing over 4676 times is given in Fig 3. Meanwhile, in terms of genetic algorithm, fig 4 shows the smallest loss function value in current generation trend against the product of iteration times and the size of population, i.e. a "revised" scale unit since genetic algorithm applies a set of vectors in evolution. For the purpose of comparison, the vector amongst the initial generation corresponding to the smallest loss function will also be let equal to $\hat{\theta}_{ini}$. Regarding to mean-square-error(MSE), the parameter generated by our approach gives 0.941 which is much less than both 0.9755 contributed by genetic algorithm and 0.9773 by first-step simulated annealing.

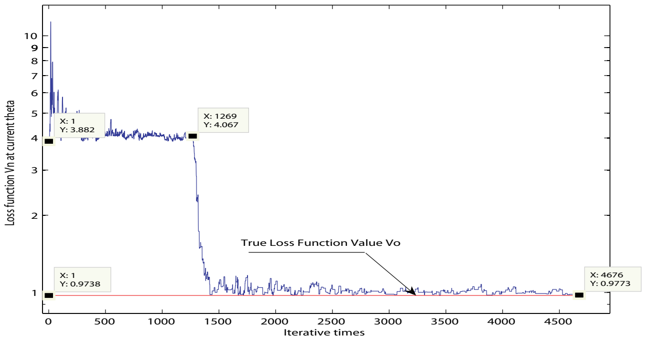

Fig. 3. Trace of $V_N(\hat{\theta})$ in simulated annealing optimization for **OE** model(At the 1269^{th} iteration, the search hops from local minimum basin into the global minimum one)

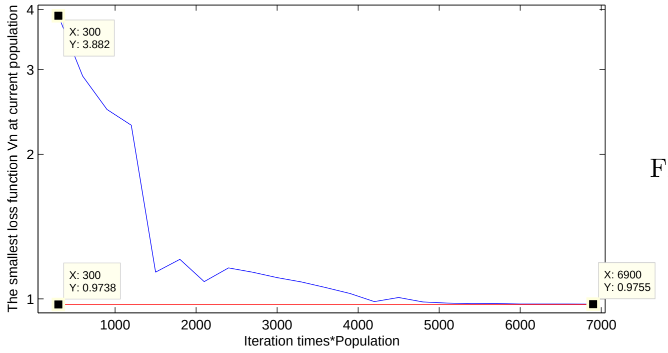

Fig. 4. Trace of $V_N(\hat{\theta})$ in genetic algorithm optimization for **OE** model

Table 1. Numerical Results between two methods for **OE** model

Contents	θ_{ini}	θ_{fin}
SA with GS	$[0.23\ 1.367\ 0.513]^T$	$[\,1.011\ \ -1.395\ 0.487\,]^T$
SA	$[0.23\ 1.367\ 0.513]^T$	$[\,0.962\ \ -1.408\ 0.499\,]^T$
GA(best)	$[0.23\ 1.367\ 0.513]^T$	$[\,0.973\ \ -1.407\ 0.498\,]^T$

Next consider a high order **BJ**$(A(q)=1)$ model in which

$$B_o(q) = \quad q^{-1} \qquad (17a)$$

$$C_o(q) = \quad 1+0.2q^{-1} \qquad (17b)$$

$$D_o(q) = \quad 1+q^{-1} \quad +0.25q^{-2} \qquad (17c)$$

$$F_o(q) = \quad 1-1.4q^{-1} \quad +0.69q^{-3} \quad -0.24q^{-4} \qquad (17d)$$

$$u(t) = \quad (1-1.47q^{-2} \quad +0.72q^{-4} \quad -0.12q^{-6} \quad)v(t) \qquad (17e)$$

are given. Here $v(t)$ and $e(t)$ are both assumed once more to be white noise. From simulation, we know the system has its stationary-point filters as

$$\hat{B}(q) = \quad -0.3q^{-1} \qquad (18a)$$

$$\hat{C}(q) = \quad 1+0.25q^{-1} \qquad (18b)$$

$$\hat{D}(q) = \quad 1+0.17q^{-1} \quad -0.41q^{-2} \qquad (18c)$$

$$\hat{F}(q) = \quad 1+1.87q^{-1} \quad +1.25q^{-2} \quad +0.59q^{-3}+0.24q^{-4} \qquad (18d)$$

Analogous to **OE** system case, the initial vector for simulated annealing is allocated with the local minimum point.

Again the trace of loss function $V_N(\hat{\theta})$ for both algorithms are shown individually in fig 5 and 6. For simulated annealing, dramatic droppings are observed at the $1269th$ and $679th$ iteration after the quasi-global-convergence from figure 3 and 5 respectively which means at this moment the search has hopped out the basin of local minimum. Jointly table 2 gives the numeric indices details. From the two diagrams, we get the **MSE** for the combined method is 1.2394 compared to 1.3213 by bare simulated annealing 1.4093 by genetic algorithm over a population of 2300.

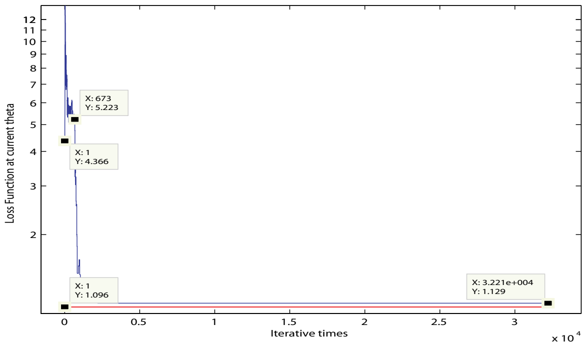

Fig. 5. Trace of $V_N(\hat{\theta})$ in simulated annealing optimization for **BJ** model

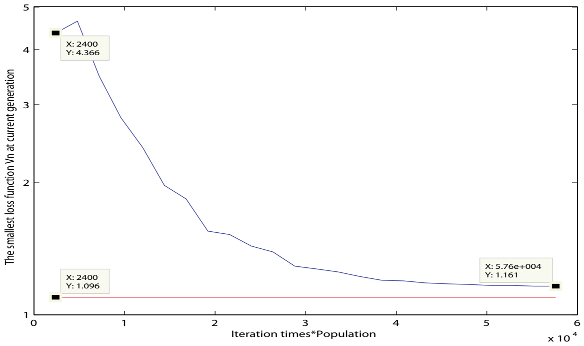

Fig. 6. Trace of $V_N(\hat{\theta})$ in simulated annealing optimization for **BJ** model

Table 2. Numerical Results between two methods for **BJ** model

Contents	θ_{ini}	θ_{fin}
SA with GS	$[-0.3\ 0.2453\ 0.1656$ $-0.4099\ 1.8663\ 1.2470$ $0.5933\ 0.2440]^T$	$[1.0110\ 0.1088\ 0.8904$ $0.1581\ \ -1.3940\ \ -0.0038$ $0.6856\ \ -0.2413]^T$
SA	$[-0.3\ 0.2453\ 0.1656$ $-0.4099\ 1.8663\ 1.2470$ $0.5933\ 0.2440]^T$	$[1.0251\ \ -0.4453\ 0.3426$ $-0.2562\ \ -1.3081\ \ -0.2279$ $0.9163\ \ -0.3319]^T$
GA	$[-0.3\ 0.2453\ 0.1656$ $-0.4099\ 1.8663\ 1.2470$ $0.5933\ 0.2440]^T$	$[0.9646\ \ -0.5888\ 0.1637$ $-0.3869\ \ -1.5138\ 0.3646$ $0.2717\ \ -0.0741]^T$

Result Analysis: As we define above, both curves start from the same $V_N(\hat{\theta})$. From the identification results, we could draw the conclusion that if the initial generation of vectors are all poorly located for genetic algorithm, simulated annealing is better than genetic algorithm regarding to judge criterion **MSE** and computation effort in linear system identification.

4. NONLINEAR SYSTEM IDENTIFICATION

Nonlinear system identification has been of interest in scientific research and engineering area. It has been proved that a **Volterra** function provides a general expression for causal, nonlinear, time invariant systems[19]. In discrete time domain, a noise corrupted mth order **Volterra** series is conveyed as following[20]

$$\begin{aligned} y(t) &= e(t) + \sum_{k_1=0}^{\tau-1} g_1(k_1)u(t-k_1) \\ &+ \sum_{k_1=0}^{\tau-1}\sum_{k_2=0}^{\tau-1} g_2(k_1,k_2)u(t-k_1)u(t-k_2) + \dots \\ &+ \sum_{k_1=0}^{\tau-1}\dots\sum_{k_m=0}^{\tau-1} g_m(k_1,\dots,k_m)\prod_{j=1}^{m} u(t-k_j) \qquad (19) \end{aligned}$$

where $y(t)$, $u(t)$ and $e(t)$ are the output, input data and white noise respectively, τ is the time delay. The kernels $g_d(\dots,k_i,\dots,k_j,\dots)(1 \le k \le m)$ are assumed to have the reciprocity property as

$$g_d(\dots,k_i,\dots,k_j,\dots) = g_d(\dots,k_j,\dots,k_i,\dots) \qquad (20)$$

which means k_i and k_j are interchangeable. Furthermore, (2) could also represent (19) after the denotation of $\hat{\theta} = [\dots\ \hat{g}_d(k_1,\dots,\ k_i,\ \dots,k_d)\ \dots]^T$ and $\varphi(t) = [\dots \prod_{i=1}^{d} u(t-k_i)\ \dots]^T$ are established. Hence the relationship could be derived

$$\frac{\partial V_N(\hat{\theta})}{\partial\hat{\theta}} = \frac{1}{N}\sum_{t=1}^{N}\epsilon(t)\frac{\partial\epsilon(t)}{\partial\hat{\theta}} = -\frac{1}{N}\sum_{t=1}^{N}\epsilon(t)\varphi(t) \qquad (21)$$

Example: consider the following **Volterra** expansion example

$$\begin{aligned} y(t) &= -0.64u(t) + 0.89u(t-1) - 0.95u(t-2) \\ &- 1.4u^2(t-1) + 1.06u(t-1)u(t-2) \\ &- 0.5u^3(t) + u^3(t-1) - 1.2u^3(t-2) + e(t) \qquad (22) \end{aligned}$$

In simulation, we assume the maximum time delay $\tau = 5$ and system order $m = 3$. The total number of kernels $N = \sum_{j=1}^{m}\binom{N+j-1}{j} = 55$ to be identified will be so large as m and τ increase that a beforehand model selection will offer better reference for reducing insignificant terms and save computation time. In [20] Abbas and Bayoumi designed an adaptive genetic algorithm in which the size of population is set to be equal with N. In the following evolution, insignificant kernel would be assigned to zero which will not go into next generation since its correlation coefficient does not pass the threshold test. However their theory is not suitable for our case because for simulated annealing there is only one initial. Hence we adopted the classic orthogonal search method[21]. Gram-Schmidt orthogonalization is continuously applied to the truncate **Volterra** to generate new orthogonal terms. Then the corresponding coefficients of orthogonal terms are technically calculated to reduce the mean-square-error. Those elements causing the most remarkable reductions in the mean-square-error will be recognized as the candidates in the "unknown" nonlinear system. Their location in ascending order and other estimate details are recorded in Table 3. The **MSE** given by simulated annealing followed by gradient search is 0.9860 still less than 1.0074 by first-step simulated annealing and 0.9888 by genetic algorithm over a population of 1000.

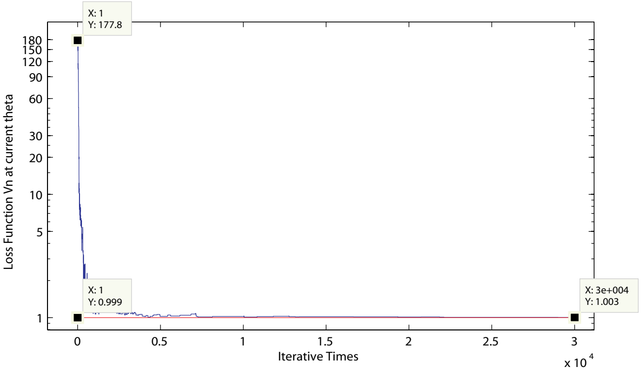

Fig. 7. The optimization of $V_N(\hat{\theta})$ by simulated annealing for nonlinear identification

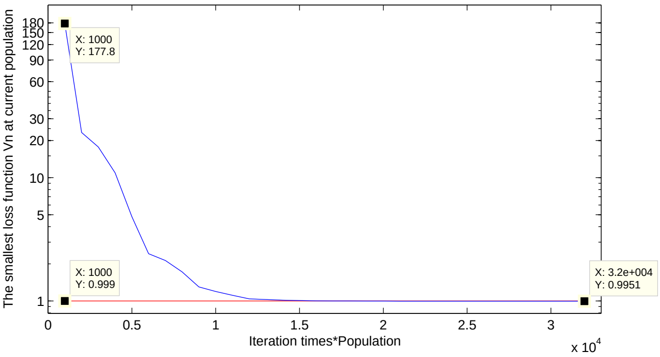

Fig. 8. The optimization of $V_N(\hat{\theta})$ by genetic algorithm for nonlinear identification

Table 3. Numerical Results between two methods for nonlinear systems

Location	True Value	Initial	SA with GS	SA	GA
1	-0.64	randn	-0.5921	-0.5824	-0.5482
2	0.89	randn	0.9332	0.9084	0.9185
3	-0.95	randn	-0.8934	-0.8522	-0.8804
11	-1.4	randn	-1.400	-1.4422	-1.3958
12	1.06	randn	1.0882	1.069	1.089
21	-0.5	randn	-0.5020	-0.5002	-0.5136
36	1	randn	0.9884	0.9812	0.9926
46	-1.2	randn	-1.2115	-1.223	-1.2138

Result Analysis: In this particular nonlinear system identification, genetic algorithm is preferable than simulated annealing with respect to **MSE** while they almost reach the optima at the same iteration time(3e+004 and 3e2+004 accordingly). And the influence of gradient search at the second stage seems not as remarkable as in the linear examples.

5. CONCLUSION AND FUTURE WORK

In this paper, simulated annealing followed by a normal gradient search is firstly employed to optimize the loss function $V_N(\hat{\theta})$. From the simulation in section 3 and 4, we draw the conclusions as follows
(1). Simulated annealing is comparable with genetic algorithm in optimizing the loss function under relatively fair initial conditions.
(2). Simulated annealing adopts only one starting point rather than a relatively large population of initial vectors as in genetic algorithm. Hence the computation burden decreases.
(3). Hill-climbing movements which takes the search out from the basin of local minimum points into the vicinity of the global minimum point(true value) are allowed in simulated annealing.
(4). Simulated annealing is conceptually easy to manipulate compared to genetic algorithm.
(5). The second-step gradient search is necessary because it secures the final estimation accuracy by gradient search especially in linear system identification when our important assumption holds.
Hence simulated annealing combined with gradient search could be used as a powerful algorithm in linear and nonlinear system identification. In future, we are looking forward to implement it on more complicated linear and nonlinear model cases.

REFERENCES

[1] O.Castillo *et al.* Application of a breeder genetic algorithm for system identification in an adaptive finite impulse response filter. In *Proceedings of the The 3rd NASA/DoD Workshop on Evolvable Hardware*, page 146, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7695-1180-5.

[2] G.C.Goodwin *et al.* Conditions for local convergence of maximum likelihood estimation for armax models. 13th IFAC Symposium on System Identification, 2001.

[3] L.Ljung. *System Idenfication:Theory for the User, 2nd Edition.* Prentice Hall, 1999.

[4] N.Gupta and R.Mehra. Computational aspects of maximum likelihood estimation and reduction in sensitivity function calculations. *IEEE Transactions on Automatic Control*, 19(6):774–783, December 1974. ISSN 0018-9286.

[5] R.Kashyap. and R.Nasburg. Parameter estimation in multivariate stochastic difference equations. *IEEE Transactions on Automatic Control*, 19(6):784–797, December 1974. ISSN 0018-9286.

[6] K.Astrom and T.Soderstrom. Uniqueness of the maximum likelihood estimates of the parameters of an ARMA model. *IEEE Transactions on Automatic Control*, 19(6):769–773, December 1974.

[7] T.Söderström. On the uniqueness of maximum likelihood identification. *Automatica*, 11:193–197, 1975.

[8] V.Duong and A.R.Stubberud. System identification by genetic algorithm. In *Aerospace Conference Proceedings, 2002. IEEE*, volume 5, pages 5–2331, 2002. doi: 10.1109/AERO.2002.1035405.

[9] C.Z.Han, Y.Li and Y.N.Dang. Nonlinear system identification with genetic algorithms. In *Proceedings of 3^{rd} World Congress on Intelligent Control and Automation*, pages 597–601, 2000.

[10] A.Srikaew, S.Sujitjorn, D.Puangdownreong, K.N.Areerak and P.Totarong. System identification via adaptive tabu search. *Industrial Technology*, 2: 915–920, 2002.

[11] I.K.Jeong and J.J.Lee. Adaptive simulated annealing genetic algorithm for system identification. *Engineering Applications of Artificial Intelligence*, 9(5):523–532, 1996.

[12] K.C. Tan, Y.Li, D.J. Murray-Smith and K.C. Sharman. System identification and linearisation using genetic algorithms with simulated annealing. In *Genetic Algorithms in Engineering Systems: Innovations and Applications, 1995. GALESIA. First International Conference on (Conf. Publ. No. 414)*, pages 164–169, Sheffield, September 1995.

[13] T.W.Manikas and J.T.Cain. Genetic algorithms vs. simulated annealing: A comparison of approaches for solving the circuit partitioning problem. Technical report, Department of Electrical Engineering, The University of Pittsburgh, 1996.

[14] C.D.Gelatt, S.Kirkpatrick and M.P.Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[15] M.N.Rosenbluth, A.H.Teller, N.Metropolis, A.W.Rosenbluth and E.Teller. Equations of state calculations by fast computing machines. 1953.

[16] D.J.Wales and J.P.K.Doye. Global Optimization by Basin-Hopping and the Lowest Energy Structures of Lennard-Jones Clusters Containing up to 110 Atoms. *Journal of Physical Chemistry A*,101:5111–5116, 1997.

[17] P.Sibani, P.Salamon and R.Frost. Facts, Conjectures,and Improvements for Simulated Annealing. S.I.A.M, 2002.

[18] H.Szu and R.Hartley. Fast simulated annealing. *Physics Letters A*, 122(3,4):157–162, 1987.

[19] V.Z.Marmarlis. Identification of nonlinear systems using laguuerre expansion of kernels. *Annals of Biomedical Engineering*, 21:573–589, 1993.

[20] H.M.Abbas and M.M.Bayoumi. Volterra system identification using adaptive genetic algorithm. *Applied Soft Computing*, 5:75–86, 2004.

[21] M.J.Korenberg. A robust orthogonal alogrithm for system identification and time-series analysis. *Biological Cybernetics*, 60:267–276, 1989.