Available online at www.sciencedirect.com
ScienceDirect
Computational Statistics & Data Analysis 52 (2007) 109–120
COMPUTATIONAL STATISTICS & DATA ANALYSIS
www.elsevier.com/locate/csda

GSA-based maximum likelihood estimation for threshold vector error correction model

Zheng Yang[a,*], Zheng Tian[a,b], Zixia Yuan[a]

[a] *Department of Applied Mathematics, Northwestern Polytechnical University, Xi'an 710072, China*
[b] *National Key Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100080, China*

Received 4 July 2005; received in revised form 28 April 2007; accepted 6 June 2007
Available online 9 June 2007

Abstract

The log-likelihood function of threshold vector error correction models is neither differentiable, nor smooth with respect to some parameters. Therefore, it is very difficult to implement maximum likelihood estimation (MLE) of the model. A new estimation method, which is based on a hybrid algorithm and MLE, is proposed to resolve this problem. The hybrid algorithm, referred to as genetic-simulated annealing, not only inherits aspects of genetic-algorithms (GAs), but also avoids premature convergence by incorporating elements of simulated annealing (SA). Simulation experiments demonstrate that the proposed method allows to estimate the parameters of larger cointegrating systems. Additionally, numerical results show that the hybrid algorithm does a better job than either SA or GA alone.

Keywords: Threshold; Vector error correction model; Maximum likelihood estimation; Genetic-simulated annealing

1. Introduction

Balke and Fomby (1997) introduced the bivariate threshold vector error correction model (TVECM) as an extension of the linear cointegration model (Johansen, 1988). The TVECM incorporating threshold-nonlinearity in cointegration allows for nonlinear adjustment to long-run equilibrium. Before introducing the TVECM representation, we briefly review the concept of cointegration or "long-run equilibrium". Let x_t be a $p \times 1$ dimensional time series integrated of order 1, or simply $I(1)$. The vector time series x_t is said to be cointegrated if there exists a $p \times 1$ vector of parameters β, called cointegrating vector, such that $\beta' x_t$ is stationary or $I(0)$.

Let us consider the following general two-regime threshold cointegrated model with one cointegrating relation,

$$\Delta x_t = \mu + (\alpha + \varepsilon \cdot I\{z_{t-1} > \gamma\}) z_{t-1} + \sum_{i=1}^{l-1} \Gamma_i' \Delta x_{t-i} + u_t, \quad t = 1, 2, \ldots, n, \tag{1}$$

where $\Delta x_t = x_t - x_{t-1}$, μ is an intercept term, α and ε are both $p \times 1$ adjustment vectors, $I\{\cdot\}$ is the indicator function, $z_{t-1} = \beta' x_{t-1}$ is the error correction term, γ is the threshold parameter, $\Gamma_i (i = 1, \ldots, l-1)$ are $p \times p$ coefficient

* Corresponding author. Tel.: +86 29 88494347; fax: +86 29 88491214.
E-mail address: yzeagle@163.com (Z. Yang).

0167-9473/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.csda.2007.06.003

matrices, and $\{u_t\}$ is a sequence of independent and identically distributed random vectors with mean zero and positive definite covariance matrix Σ.

One of the key challenges consists in estimating the parameters of the TVECM, since the likelihood function is not smooth and differentiable with respect to the threshold parameter γ and the cointegrating vector β. In order to estimate the threshold parameter of the TVECM, Lo and Zivot (2001) proposed sequential multivariate least squares estimation. The estimation was implemented under a required assumption that the TVECM had a known cointegrating vector. Hansen and Seo (2002) developed a quasi-maximum likelihood estimation (quasi-MLE) approach for the threshold cointegrating model. Their method involved a combination of grid search and MLE. In the case $p = 2$, grid search might be effective.[1] But in higher dimensional cases, grid search becomes ineffective. Furthermore, the precision of the numerical estimates would be reduced when the parameters maximizing the log-likelihood function do not happen to lie exactly on a grid point. Therefore, the grid search could be replaced by a genetic algorithm (GA) in higher dimensional cases.

GA has been successfully used in a wide range of differentiable, nondifferentiable, and discontinuous optimization problems encountered in engineering and economic applications (see, e.g. Baragona et al., 2004, Winker and Gilli, 2004). However, GA has two major limitations (Wong and Wong, 1994, Jeong and Lee, 1996). First, the performance might deteriorate as the problem size grows. In fact, with a growing size of the problem, GA requires a larger population to obtain a satisfactory solution. Second, premature convergence might occur when the GA cannot find the optimal solution due to loss of some important characters (genes) in candidate solutions.

A hybrid algorithm, which combines aspects of GA and simulated annealing (SA) (Kirkpatrick et al., 1983) is proposed to overcome the limitations of GA. The performance of GA can be improved by introducing more diversity among the chromosomes in the early stage of the solution process so that premature convergence can be eliminated. To implement this idea, the Metropolis acceptance test technique from SA is adopted into GA. The new hybrid algorithm, referenced to as genetic-simulated annealing or GSA, has been shown to overcome the poor convergence properties of GA and outperform GA or SA along by Chen et al. (1998).

This paper proposes a new method combining GSA and MLE for estimating the parameters of the TVECM, which is motivated by Lo and Zivot (2001) and Hansen and Seo (2002). Initially, the threshold parameter γ and the cointegrating vector β are estimated by GSA. Subsequently, the remaining parameters are computed by maximum likelihood estimation. In comparison with sequential multivariate least squares, this method implements the estimation when both the cointegrating vector and the threshold parameter are unknown. More importantly, the method is unrestricted with respect to the dimension of the model.

The rest of this paper is organized as follows. In Section 2, the quasi-maximum likelihood estimation is introduced. Section 3 presents GSA and the detailed steps for the parameter estimation of TVECM. The simulation examples in Section 4 show that the method works well, and GSA performs better than GA and SA alone in terms of the mean and the standard errors of the estimated parameters. Section 5 concludes.

2. Quasi-MLE for TVECM

Let $\theta = (\mu, \varepsilon, \alpha, \Gamma, \Sigma) \in \Theta$ denote the parameter vector, where $\Gamma = (\Gamma_1', \Gamma_2', \ldots, \Gamma_{l-1}')'$. The log-likelihood function, with the auxiliary condition that u_t is normally distributed, is given by

$$L_n(\gamma, \beta, \theta) = -\frac{n}{2}\log|\Sigma| - \tfrac{1}{2}\sum_{t=1}^{n} u_t(\gamma, \beta, \theta)'\Sigma^{-1}u_t(\gamma, \beta, \theta), \tag{2}$$

where

$$u_t = \Delta x_t - \mu - (\alpha + \varepsilon \cdot I\{z_{t-1} > \gamma\})z_{t-1} - \sum_{i=1}^{l-1}\Gamma_i'\Delta x_{t-i}.$$

Let $\Upsilon = [\gamma_L, \gamma_U]$ denote the empirical support of z_{t-1}, and let B be a $p-1$ dimensional space which can be interpreted as a (large) confidence interval for β constructed from the linear estimate $\tilde{\beta}$ (for example, based on the

[1] A convenient normalization condition is to set the first element of β equal to unity. In case $p = 2$, the parameters (γ, β_2) were estimated using grid search.

asymptotic normal approximation). We denote $\hat{\theta}(\gamma, \beta)$ as the MLE of θ for known $(\gamma \times \beta) \in (\Upsilon \times B)$, where $\Upsilon \in \mathscr{R}^1$ and $B \in \mathscr{R}^{p-1}$. The first-order conditions of the likelihood function are given by

$$\frac{\partial L_n(\gamma, \beta, \hat{\theta})}{\partial \mu'} = \sum_{t=1}^{n} \hat{u}_t' \hat{\Sigma}^{-1} = 0, \tag{3}$$

$$\frac{\partial L_n(\gamma, \beta, \hat{\theta})}{\partial \varepsilon'} = \sum_{t=1}^{n} I\{z_{t-1} > \gamma\} z_{t-1} \hat{u}_t' \hat{\Sigma}^{-1} \hat{\alpha} = 0, \tag{4}$$

$$\frac{\partial L_n(\gamma, \beta, \hat{\theta})}{\partial \alpha'} = \sum_{t=1}^{n} z_{t-1} \hat{u}_t' \hat{\Sigma}^{-1} = 0, \tag{5}$$

$$\frac{\partial L_n(\gamma, \beta, \hat{\theta})}{\partial \Gamma_i'} = \sum_{t=1}^{n} \Delta x_{t-i} \hat{u}_t' \hat{\Sigma}^{-1} = 0, \quad i = 1, 2, \ldots, l-1, \tag{6}$$

and

$$\hat{\Sigma} = n^{-1} \sum_{t=1}^{n} \hat{u}_t' \hat{u}_t, \tag{7}$$

where $\hat{u}_t = u_t(\gamma, \beta, \hat{\theta})$ in Eq. (1).

The concentrated likelihood function is

$$\begin{aligned} L_n(\gamma, \beta) &= L_n(\gamma, \beta, \hat{\theta}(\gamma, \beta)) \\ &= -\frac{n}{2} \log |\hat{\Sigma}(\gamma, \beta)| - \frac{np}{2}. \end{aligned} \tag{8}$$

The MLE estimators $(\hat{\gamma}, \hat{\beta})$ of the parameters (γ, β) are defined as any values of (γ, β) minimizing $\log |\hat{\Sigma}(\gamma, \beta)|$. Then, the MLE for θ is $\theta = \hat{\theta}(\hat{\gamma}, \hat{\beta})$.

The criterion function (8) is not smooth, so conventional gradient hill-climbing algorithms are not suitable for its maximization. In the case $p = 2$, Hansen and Seo (2002) use a grid search over the two-dimensional space $(\Upsilon \times B)$ to find $(\hat{\gamma}, \hat{\beta}_2)$ as the values of (γ, β_2) on this grid which yield the largest value of the likelihood function (8).

3. GSA Algorithm

List of principle symbols

L_n	log-likelihood function
F	fitness function
ΔF	the change in fitness
f_i	normalized fitness of the ith chromosome
Δ	amount of deteriorated fitness measure
T	temperature
T_0	initial temperature
λ	temperature reduction factor
p_c	crossover probability
p_m	mutation probability
N_p	population size
g	number of iteration (generation)

GSA is a hybrid method that inherits the parallel search of GA by incorporating the Metropolis rule of SA. GSA enables a powerful implementation that avoids the inherent weaknesses of optimization processes of both GA and SA.

By exploiting the population mechanism and crossover/mutation operator of GA, the algorithm is naturally parallel. By exploiting the local selection strategy of SA, the risk of premature convergence of GA is reduced.

For later illustrating the hybrid algorithm, a few terms need to be clarified.

Gene and *chromosome* are two basic concepts in evolution theory. Every gene and chromosome, respectively, represents a parameter of the estimation problem and a "candidate solution". To avoid the need to convert binary numbers into their decimal equivalents for the SA operator, this paper proposes the use of floating-point (real-coded) numbers to present each chromosome instead of binary representation (Zhou and Wang, 2005).

In the present estimation, let $\gamma, \beta_2, \ldots, \beta_{p-1}$ and β_p denote genes. The symbol "y" denotes a chromosome, which is made up of p genes, i.e. $y = (\gamma, \beta_2, \ldots, \beta_p)$. During each iteration g, GSA maintains a population of potential chromosomes (solutions) $P(g) = \{y_1(g), y_2(g), \ldots, y_{N_p}(g)\}$, where N_p denotes the population size.

Population denotes a set of the chromosomes from the species of "candidate solution". There is no clear indication as to how large a population should be. If the population is too large, there may be difficulties in storing the data, but if the population is too small, there may not be enough chromosomes for good crossovers. In the experiments, the populations range from 10 to 100.

Fitness function evaluates the "fitness" of a chromosome as a solution to the optimization problem. The objective function of the estimation problem is the likelihood function L_n in (8). For estimating parameters of the model using GSA, we propose the following fitness function:

$$F(y) = L_n(\gamma, \beta) + C, \tag{9}$$

where C is a positive integer that guarantees a positive fitness, i.e. $C > |L_n(\hat{\gamma}, \hat{\beta})|$.

The population of chromosomes evolves from generation to the next through four main operators including *selection*, *crossover*, *mutation and the Metropolis acceptance rule*.

Selection. Chromosomes of the current population are selected as suitable subjects for development of the next generation based on their fitness. For each chromosome $y_j (j = 1, \ldots, N_p)$ in the current population $P(g)$, evaluate the fitness measure by the fitness function $F(y_j)$. Let F_j denote the fitness of the jth chromosome, its normalized fitness f_j is formed:

$$f_j = F_j / \Sigma F_i. \tag{10}$$

Assign the normalized fitness measure of the chromosomes as the probabilities for the chromosomes to be chosen as parents for the production of the "next" generation. The Roulette wheel method (Goldberg, 1989) is used to realize the selection. To ensure that at least one of the best chromosomes is reproduced to the next generation, the elitist strategy (Gudla and Ganguli, 2005) is also used at the same time.

Crossover. Crossover is a mechanism for probabilistic inheritance of useful information from two fit individuals to their offspring. The main idea is that the genetic information of a good solution is spread over the entire population. Thus, the best solution can be obtained by thoroughly combining the chromosomes in the population. Crossover operation achieves recombination of the genetic material. The recombination process includes domain specific knowledge to enforce the inheritance of desirable features from individuals of the current population. The following crossover operators have been used in this work.

In order to take advantage of the real-coded representation to increase the convergence rate of the GSA, we use flat crossover (Goldberg, 1991). For every y in the population, draw a random number r from [0, 1]; if $r \leqslant p_c$, where p_c is the crossover probability, y is selected for crossover. The chromosomes $y_i = (y_{1,i}, y_{2,i}, \ldots, y_{p,i})'$ and $y_j = (y_{1,j}, y_{2,j}, \ldots, y_{p,j})'$ selected for crossover are randomly paired off to produce two offspring y_i' and y_j' per mating as follows:

$$\begin{cases} y'_{k,i} = r_c * y_{k,i} + (1 - r_c) * y_{k,j}, \\ y'_{k,j} = r_c * y_{k,j} + (1 - r_c) * y_{k,i}, \end{cases} \tag{11}$$

where $k = 1, 2, \ldots, p$, and r_c is a random number from [0, 1].

Mutation. Mutation is a means of introducing new information into the population. For each chromosome y, we draw a random number r from [0, 1]. If $r \leqslant p_m$, where p_m is the probability of mutation, y is selected for mutation.

If r_{m} is a random number from [0, 1], the result of the mutation is

$$y'_{k,i} = y_{k,i} + \delta(k), \tag{12}$$

where $k = 1, 2, \ldots, p$, and $\delta(k) = 0.1 * r_m$ denotes a perturbation.

Metropolis acceptance rule is one of the important core concepts of the GSA algorithm, which simulates the annealing process at a given temperature (Metropolis et al., 1953). In the neighborhood of two selected parent chromosomes, two child chromosomes were produced using the crossover operator and mutation operator. One child and one parent are taken out from the four chromosomes arbitrarily. If the fitness value of the child is better than the fitness value of the parent, then the child chromosome is accepted and used to generate a new chromosome for the next generation. If the child has a lower fitness in comparison to the parent, the Metropolis rule accepts the child on a probabilistic basis. A random number is generated in the range 0 to 1. If this random number is smaller than $\mathrm{e}^{\Delta F/T}$, where $\Delta F = F_{\mathrm{child}} - F_{\mathrm{parent}}$ is the amount of deterioration between the child and parent and T is the current temperature, the inferior child is accepted. The same procedure is applied to the other pair of child and parent. This criterion for accepting the new chromosome is known as the Metropolis criterion.

Initial temperature. The GSA algorithm starts from a lower temperature than the SA algorithm. If this initial value of temperature is too high, it causes a waste of processing time. The temperature parameter is initialized using the procedure described in Youssef et al. (2001). Let M denote a constant number of moves in the neighborhood of the current solution. Then, the fitness difference for each move i, ΔF_i is given by $\Delta F_i = F_i - F_{i-1}$. Let M_{u} and M_{d} be the number of uphill and downhill moves, respectively ($M = M_{\mathrm{u}} + M_{\mathrm{d}}$). The average ΔF_{d} is then given by

$$\Delta F_{\mathrm{d}} = \frac{1}{M_{\mathrm{d}}} \sum_{i=1}^{M_{\mathrm{d}}} \Delta F_i .$$

In order to keep the probability P_0 of accepting uphill moves high in the initial stage of GSA, we estimate the values of the temperature parameter by substituting the value of P_0 in the following expression derived from the Metropolis function:

$$T_0 = \frac{\Delta F_{\mathrm{d}}}{\ln(P_0)}, \tag{13}$$

where $P_0 \approx 1 (P_0 = 0.999)$. The GSA started from a lower temperature to make the Metropolis function $\mathrm{e}^{\Delta F/T}$ lower than the SA. Since the GSA took the population evolution, the candidate solution could be easily found even though $\mathrm{e}^{\Delta F/T}$ was small.

S*topping criterion.*The global searching is stopped when generations $g = Num$, where *Num* is the predetermined number of generations, or the final $T_g < 0.001$, where $T_g = \lambda T_{g-1}$ and λ denotes the cooling rate.

The GSA algorithm is implemented with Matlab 7.0 for Windows on a Pentium IV, the structure of the GSA for the parameter estimation is given below. For the pseudo codes for SA and GA see Winker and Gilli (2004).

Procedure of the GSA.
Initialize population $P(0)$ of size N_p, initialize T_0, and set $g = 0$;
while stopping criteria not met **do**
  Evaluate fitness of the population $P(g)$.
  for $j = 1$ to N_p **do**
    Select two chromosomes $P'(g)$ from $P(g)$ randomly
    Apply the crossover operator on $P'(g)$ to produce $P''(g)$
    Perform mutation operator on $P''(g)$ to produce $P'''(g)$
    Evaluate fitness of chromosomes $P'''(g)$
    Perform Metropolis rule for $P'''(g)$ and $P'(g)$ to generate two chromosomes of the next generation $P(g + 1)$
  end for
  $g = g + 1$, $T_g = \lambda T_{g-1}$
end while

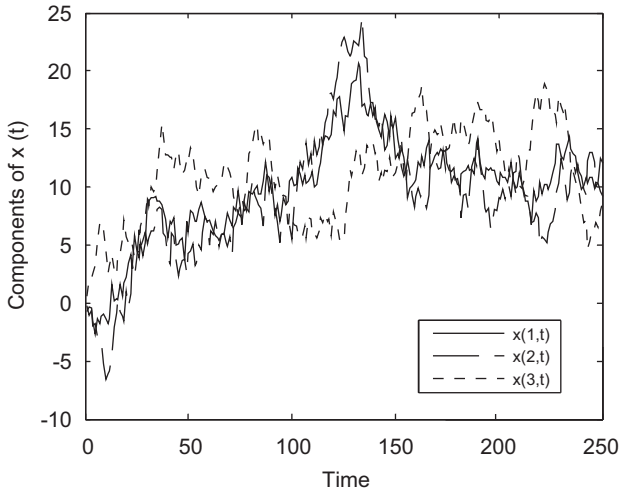

Fig. 1. Time plots of the simulated series. The curves of x_{1t}, x_{2t} and x_{3t} are displayed by solid, dashed and dotted lines, respectively.

4. Simulation examples

4.1. A simulated data

A Monte Carlo experiment is performed to illustrate our algorithm proposed in this paper. The experiment is based on a three-dimensional threshold error correction model while the data are generated from the first-order VAR. Let $x_t = (x_{1t}, x_{2t}, x_{3t})'$ and $z_t = \beta' x_t$, the two-regimes threshold model is

$$\Delta x_t = \mu + (\alpha + \varepsilon \cdot I\{z_{t-1} > \gamma\}) z_{t-1} + \Gamma_1' \Delta x_{t-1} + u_t, \tag{14}$$

where

$$\mu = \begin{pmatrix} 0.0 \\ 0.0 \\ 0.0 \end{pmatrix}, \quad \beta = \begin{pmatrix} 1.0 \\ -0.7 \\ -0.3 \end{pmatrix}, \quad \alpha = \begin{pmatrix} -0.3 \\ 0.5 \\ -0.4 \end{pmatrix}, \quad \varepsilon = \begin{pmatrix} 0.1 \\ -0.1 \\ 0.2 \end{pmatrix}, \quad \Gamma_1 = \begin{pmatrix} 0.4 & 0.0 & -0.1 \\ 0.0 & 0.2 & 0.0 \\ 0.0 & 0.0 & 0.3 \end{pmatrix},$$

and set $\gamma = -0.4$. We generated a single realization of the error variables $u_t \sim N(0, \Sigma)$, where Σ is an identity covariance matrix.

We consider two sample sizes, $n = 100$ and 250. Following data generating process (14), the simulated data are displayed in Fig. 1 with sample size $n = 250$. In Fig. 2, we illustrate the nondifferentiability of the criterion function. Fig. 2 (a) plots criterion (9) as a function of γ with β concentrated out, and Fig. 2 (b) plots the criterion as a function of β_2 and β_3 with γ concentrated out, notice that $\beta_1 = 1.0$.

4.2. Parameter estimation by GSA

In order to compare the results, we estimate the parameters γ, β_2 and β_3 using SA, GA and GSA, respectively. The chosen values of the parameters in the algorithms are provided in Table 1. For instance, the population size of 30 for GA in Table 1 means a population of 30 feasible and valid candidate solutions, and in each generation the number of offspring produced is the same as the number of parents. The probability of crossover P_c is set to 80%, and the probability of mutation P_m to 10% for GA. In the experiment, the stopping criteria in GSA and GA are set to a fixed number of generations of 450. And SA stops if the final temperature T_g is less than 0.001.

The average, worst and best log-likelihood provided by the algorithms in 1000 replications for the given single realization are summarized in Table 2. The results show that GSA is more robust than SA and GA for the parameter estimation. And the average value found by GSA is higher than those found by SA and GA, which confirms that GSA is well capable of determining the global or near-global optimum solution. Although GA can find the best log-likelihood in

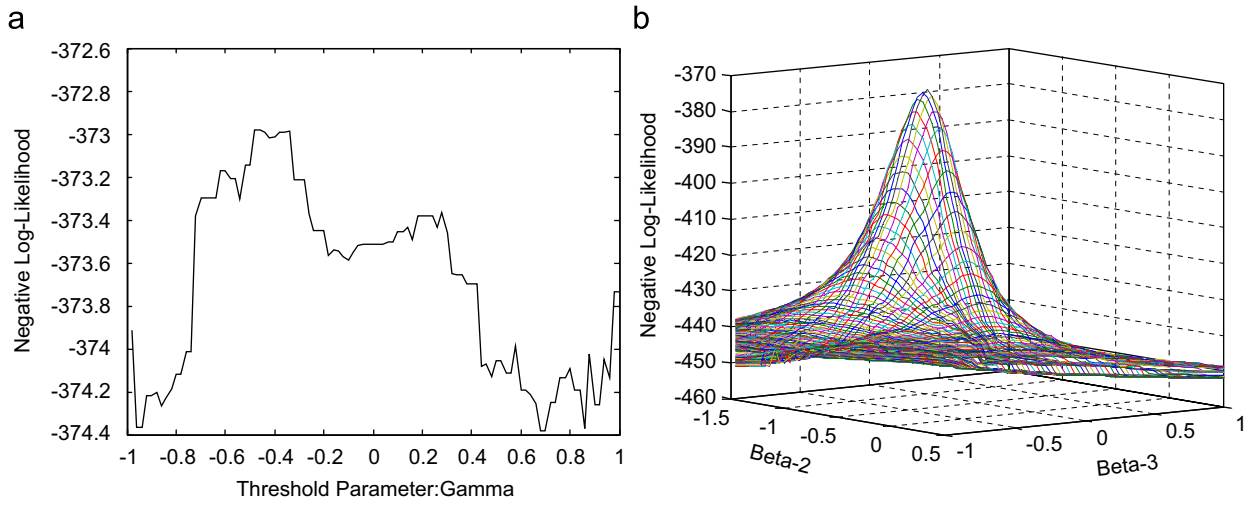

Fig. 2. Concentrated negative log-likelihood, (a) γ and (b) β_2 and β_3.

Table 1
Parameter values for the execution of three algorithms

Adapted parameter	SA	GA	GSA
Population size (N_p)	1	30	10
Number of iteration (g)		450	450
Probability of crossover (P_c)		0.8	0.6
Probability of mutation (P_m)		0.1	0.1
Initial temperature (T_0)	1000		100
Cooling rate (λ)	0.98		0.60

Table 2
Comparison of the best, worst and average log-likelihood values

Algorithms	$n = 100$			$n = 250$		
	Best	Average	Worst	Best	Average	Worst
SA	−156.4331	−157.9354	−181.4879	−372.8944	−374.7210	−448.7880
GA	−156.2712	−156.9070	−166.2627	−372.8347	−374.8128	−419.7776
GSA	−156.2516	−156.3082	−156.6139	−372.8236	−372.9273	−373.3054

Table 3
Execution times(s) of the three algorithms

Algorithms	$n = 100$			$n = 250$		
	Shortest	Average	Longest	Shortest	Average	Longest
SA	31.6785	36.5594	46.4806	30.4563	38.7634	44.0562
GA	25.3052	26.0045	27.6281	26.0890	27.3622	27.8248
GSA	23.2527	23.8972	24.5360	23.8656	24.6941	25.7463

one execution, it can also produce a lower quality in another execution due to the premature convergence phenomenon. Therefore, the performance of GA appears to be not as reliable as GSA.

Table 3 gives an overview on the relative execution times required by the three different algorithms. GSA is the fastest algorithm as can be observed based on the parameter settings from Table 1. The execution time is variable depending

Table 4
Distribution of estimation (threshold and cointegrating vector)

	Algorithms	Mean	RMSE	MAE	Percentiles (%)				
					5	25	50	75	95
$n = 100$									
$\hat{\gamma} - \gamma$	SA	0.0836	0.6910	0.5814	−0.9714	−0.4647	0.1124	0.7986	1.0550
	GA	−0.0023	0.1124	0.0854	−0.1867	−0.0808	0.0051	0.0703	0.1284
	GSA	−0.0064	0.0116	0.0074	−0.0280	−0.0127	−0.0007	0.0002	0.0019
$\hat{\beta}_2 - \beta_2$	SA	−0.0001	0.0295	0.0136	−0.0242	−0.0168	−0.0008	0.0045	0.0215
	GA	0.0141	0.0595	0.0320	−0.0290	−0.0111	−0.0019	0.0197	0.0807
	GSA	−0.0021	0.0116	0.0086	−0.0155	−0.0087	−0.0042	−0.0010	0.0151
$\hat{\beta}_3 - \beta_3$	SA	−0.0033	0.0426	0.0153	−0.0324	−0.0150	−0.0043	0.0014	0.0077
	GA	−0.0144	0.0483	0.0260	−0.0774	−0.0215	−0.0074	0.0006	0.0055
	GSA	−0.0064	0.0116	0.0074	−0.0280	−0.0127	−0.0007	0.0002	0.0019
$n = 250$									
$\hat{\gamma} - \gamma$	SA	0.0878	0.3349	0.2418	−0.3141	−0.0950	0.0193	0.1251	0.6091
	GA	0.1006	0.1431	0.1192	−0.0762	0.0373	0.1047	0.1561	0.2323
	GSA	0.0789	0.1265	0.1086	−0.0757	0.0316	0.0973	0.1405	0.2019
$\hat{\beta}_2 - \beta_2$	SA	0.0094	0.1610	0.0230	−0.0135	−0.0100	−0.0062	−0.0035	−0.0002
	GA	−0.0088	0.0191	0.0131	−0.0480	−0.0109	−0.0089	−0.0049	0.0019
	GSA	−0.0070	0.0079	0.0073	−0.0106	−0.0093	−0.0086	−0.0048	−0.0007
$\hat{\beta}_3 - \beta_3$	SA	0.0061	0.0117	0.0066	−0.0020	0.0014	0.0068	0.0086	0.0100
	GA	0.0139	0.0341	0.0156	−0.0026	0.0043	0.0074	0.0100	0.0172
	GSA	0.0057	0.0069	0.0059	0.0006	0.0030	0.0061	0.0088	0.0103

on the different parameter settings, such as the populations size N_p. We just present them to provide a rough estimate of the relative time consumption of the three algorithms. Note that the execution times are not much influenced by the actual sample size.

We consider the estimation of three parameters, the threshold parameter γ and the cointegrating vector β_2 and β_3. In Table 4, we report the mean, root mean squared error (RMSE), mean absolute error (MAE) following Hansen and Seo (2002), all with regard to the parameters of the data generating process and selected percentiles of each estimator in 1000 simulation replications.

The results in Table 4 indicate that GSA outperforms SA and GA by contrasting the means and standard errors of the parameter estimation. First, for $n = 100$, there are no statistically significant differences in bias among the three estimation procedures. For $n = 250$, the estimators using GSA are more accurate than those using SA and/or GA. Second, GSA has a slightly lower RMSE and MAE. Finally, the estimator based on GSA has considerably less dispersion than those using SA and/or GA. It is interesting to remark that the estimates based on GSA and GA are influenced much less by the choice of initial values than the estimates based on SA.

For illustrating the performance of the three algorithms, the estimated values of one replication are displayed in Fig. 3. Fig. 3(a) is a plot of the estimated values of the log-likelihood L_n versus the number of iterations or generations. Similarly, Fig. 3(b)–(d) plot the estimated values of parameters γ, β_2 and β_3, respectively.

From Fig. 3(a), it can be shown that GSA has quickly converged to the largest log-likelihood value after about 100 iterations while GA converges to the largest value after about 200 iterations. In contrast, SA finds the maximum log-likelihood value about 300 iterations. The slow speed of convergence for the SA is partly due to the fact that the new solutions are selected at three directions in the neighborhood of the current solution, which results in a small fitness difference between the original solution and the new solution. Hence, SA exhibits high fluctuation since solutions with inferior fitness are accepted even at relatively low temperature according to the Metropolis rule. Fig. 3(c) and (d) shows that all three algorithms converge to the exact values of β_2 and β_3, but GA and SA in Fig. 3(b) do not converge to the true value of γ. This happens because there is a plateau of log-likelihood values with the regard to the threshold γ in the range of $(-0.5, -0.3)$, see Fig. 2(a). As a matter of fact GSA exhibits much better performance than SA and GA in Fig. 3.

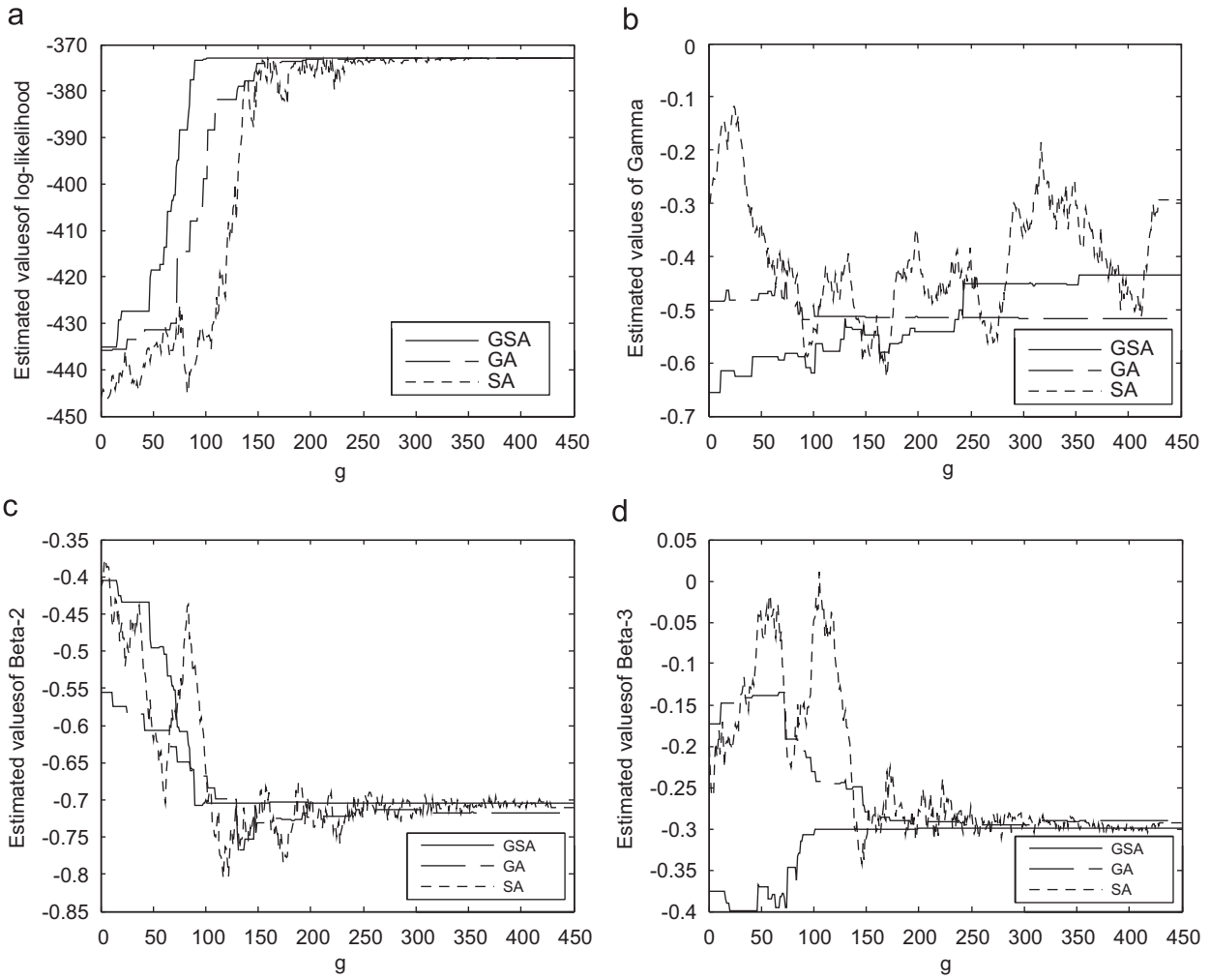

Fig. 3. Comparison of estimation of GSA, SA and GA. (a) Estimated values of log-likelihood L_n, (b) estimated values of γ, (c) estimated values of β_2, (d) estimated values of β_3. GSA is displayed by solid line. GA is plotted by dashed line and SA is dotted line.

4.3. Parameter estimation by MLE

We consider in this subsection the remaining parameters in the TVECM using MLE. According to Eqs. (3)–(7), the adjustment vectors α and ε, the matrices of coefficients Γ_1 and the covariance matrix Σ are estimated following the estimated values of parameters γ, β_2 and β_3. To save space, we report the results for the adjustment vectors α and ε only.

In Table 5 we report the mean, RMSE, MAE and selected percentiles of each estimator in 1000 optimization runs for α and ε. The results of estimation clearly show that GSA-based MLE outperforms GA-based MLE and SA-based MLE. The method seems to be very accurate, since these mean values are close to zero and the RMSE, MAE are smaller than for the other methods from 1000 executions.

4.4. Robustness of GSA-based MLE

So far, we analyze the performance of GSA-based MLE in a single realization. Next, we consider the impact of some changes in the simulated data generated by the model (14). In order to check the robustness of the process, two cases are considered.

Table 5
Estimation of adjustment vectors

	Algorithms	Mean	RMSE	MAE	Percentiles (%)				
					5	25	50	75	95
$n = 100$									
$\hat{\alpha}_1 - \alpha_1$	SA	−0.0739	0.2386	0.1171	−0.1367	−0.1163	−0.1037	−0.0933	−0.0323
	GA	−0.1372	0.1787	0.1372	−0.3880	−0.1252	−0.1060	−0.0962	−0.0738
	GSA	−0.1058	0.1071	0.1058	−0.1504	−0.1114	−0.1043	−0.0975	−0.0895
$\hat{\alpha}_2 - \alpha_2$	SA	−0.0478	0.5896	0.0734	−0.0154	0.0052	0.0111	0.0197	0.0284
	GA	0.0251	0.0685	0.0357	−0.0290	−0.0009	0.0112	0.0313	0.0753
	GSA	0.0115	0.0186	0.0145	−0.0145	0.0034	0.0101	0.0198	0.0288
$\hat{\alpha}_3 - \alpha_3$	SA	0.0561	0.3774	0.0803	−0.0618	−0.0234	0.0371	0.0481	0.0555
	GA	0.0432	0.0567	0.0492	0.0154	0.0333	0.0464	0.0589	0.0673
	GSA	0.0440	0.0452	0.0440	0.0302	0.0379	0.0431	0.0516	0.0580
$\hat{\varepsilon}_1 - \varepsilon_1$	SA	−0.0937	0.0995	0.0971	−0.1410	−0.0961	−0.0915	−0.0844	−0.0744
	GA	−0.0819	0.0884	0.0861	−0.1112	−0.0970	−0.0901	−0.0820	−0.0621
	GSA	−0.0919	0.0992	0.0919	−0.1073	−0.0950	−0.0910	−0.0877	−0.0839
$\hat{\varepsilon}_2 - \varepsilon_2$	SA	0.0421	0.0603	0.0494	0.0201	0.0413	0.0470	0.0516	0.0572
	GA	0.0331	0.0613	0.0519	−0.0932	0.0334	0.0474	0.0579	0.0698
	GSA	0.0481	0.0492	0.0481	0.0304	0.0421	0.0472	0.0535	0.0622
$\hat{\varepsilon}_3 - \varepsilon_3$	SA	0.1142	0.1182	0.1142	0.0877	0.0934	0.0963	0.1398	0.1716
	GA	0.0972	0.0996	0.0972	0.0809	0.0881	0.0906	0.0983	0.1253
	GSA	0.0912	0.0913	0.0912	0.0863	0.0883	0.0889	0.0941	0.0976
$n = 250$									
$\hat{\alpha}_1 - \alpha_1$	SA	0.1265	0.7709	0.1265	0.0408	0.0452	0.0493	0.0548	0.0597
	GA	0.0501	0.0533	0.0514	0.0167	0.0477	0.0517	0.0550	0.0625
	GSA	0.0503	0.0504	0.0503	0.0420	0.0477	0.0511	0.0534	0.0546
$\hat{\alpha}_2 - \alpha_2$	SA	−0.0494	0.0545	0.0528	−0.0616	−0.0574	−0.0524	−0.0445	−0.0414
	GA	−0.0493	0.0561	0.0536	−0.0724	−0.0582	−0.0531	−0.0471	−0.0395
	GSA	−0.0520	0.0524	0.0520	−0.0603	−0.0561	−0.0528	−0.0492	−0.0427
$\hat{\alpha}_3 - \alpha_3$	SA	−0.0497	0.4263	0.0500	−0.0180	−0.0105	−0.0068	−0.0034	0.0004
	GA	−0.0163	0.0433	0.0194	−0.0854	−0.0141	−0.0085	−0.0035	0.0020
	GSA	−0.0079	0.0096	0.0082	−0.0170	−0.0117	−0.0077	−0.0041	−0.0003
$\hat{\varepsilon}_1 - \varepsilon_1$	SA	0.0235	0.0298	0.0235	0.0139	0.0182	0.0215	0.0252	0.0281
	GA	0.0294	0.0388	0.0294	0.0157	0.0204	0.0228	0.0259	0.0448
	GSA	0.0221	0.0222	0.0221	0.0176	0.0202	0.0217	0.0241	0.0256
$\hat{\varepsilon}_2 - \varepsilon_2$	SA	0.0373	0.0579	0.0452	0.0273	0.0352	0.0431	0.0474	0.0508
	GA	0.0255	0.0624	0.0494	−0.0713	0.0331	0.0407	0.0459	0.0514
	GSA	0.0417	0.0422	0.0417	0.0316	0.0367	0.0429	0.0459	0.0496
$\hat{\varepsilon}_3 - \varepsilon_3$	SA	0.0298	0.0346	0.0298	0.0247	0.0268	0.0277	0.0296	0.0318
	GA	0.0328	0.0373	0.0328	0.0237	0.0270	0.0281	0.0297	0.0426
	GSA	0.0288	0.0289	0.0288	0.0264	0.0275	0.0282	0.0294	0.0319

Case 1: We generate 1000 different realizations with the sample size $T = 250$ by changing the disturbance terms u_t in the data generating process (14).[2]

Case 2: The data generating processes (14) are tried for various sample sizes $T = 100$, 250 and 500. The parameters are varied with (i) ($\gamma = 0.2$, $\beta_3 = -0.0$); (ii) ($\gamma = 0.0$, $\alpha_2 = -0.8$) and (iii) ($\beta_2 = 0.2$, $\beta_3 = -0.6$).

[2] A referee kindly suggested analyzing the robustness of Case 1.

Table 6
Mean and RMSE of the estimator

	SA-based MLE		GA-based MLE		GSA-based MLE	
	Mean	RMSE	Mean	RMSE	Mean	RMSE
$\hat{\gamma}-\gamma$	0.1032	0.3765	0.0963	0.1846	0.0852	0.1318
$\hat{\beta}_2-\beta_2$	0.0121	0.1089	−0.0095	0.0154	−0.0077	0.0082
$\hat{\beta}_3-\beta_3$	0.0072	0.0128	0.0147	0.0295	0.0063	0.0076
$\hat{\alpha}_1-\alpha_1$	0.0936	0.2813	0.0428	0.0524	0.0376	0.0403
$\hat{\alpha}_2-\alpha_2$	−0.0554	0.0607	−0.0531	0.0572	−0.0515	0.0520
$\hat{\alpha}_3-\alpha_3$	−0.0486	0.0834	−0.0206	0.0358	−0.0092	0.0116
$\hat{\varepsilon}_1-\varepsilon_1$	0.0242	0.0283	0.0283	0.0331	0.0251	0.0267
$\hat{\varepsilon}_2-\varepsilon_2$	0.0484	0.0626	0.0381	0.0716	0.0456	0.0462
$\hat{\varepsilon}_3-\varepsilon_3$	0.0326	0.0410	0.0405	0.0488	0.0322	0.0346

Under Case 1, we apply all three methods to each single realization. The mean and RMSE are still the measures of approximation quality for the estimated parameters when a robust measure is needed. The experimental results for the estimated parameters γ, β, α and ε reported in Table 6 indicate that the three different heuristics provide estimators qualitatively not different from those in Tables 4 and 5. As expected, the estimators related to the GSA method are smaller than those for SA and GA, i.e. the GSA method is more robust than the other methods. Under Case 2, we apply the optimization heuristics with 1000 replications to the same realization generated by the three different parameters, respectively. We omit the estimated results since they are very similar to the results in Tables 4 and 5.

5. Conclusion

The parameter estimation for the TVECM has been developed in the literature and practitioners are usually faced with the problem of restricting the dimension of the models. Consequently, there is a need for more powerful numerical algorithms that do not restrict the form of the objective function. The quasi-maximum likelihood estimation here combining GSA and MLE has not been previously used in the context of econometric estimation. In this paper we illustrated the applicability of the method with examples from some simulated realizations.

For a nondifferential problem such as MLE of the TVECM, the potential contribution of the proposed method is twofold. First, it can remove the restrictions on the dimension of the TVECM, and it may be applicable in some other nonlinear econometric models such as the smooth transition error correction models. Second, the core of the proposed method is the GSA, which takes advantage of the good performance of GA and SA to eliminate the shortcomings of either GA or SA alone. Through the operations of selection, crossover, mutation and the Metropolis acceptance rule, the GSA provides a more comprehensive search that efficiently converged to the true coefficients. The effectiveness and the flexibility are becoming more important as estimation methodology continues to advance for the development of nonlinear econometric models.

Acknowledgments

The authors would like to thank the editor, an associate editor, three anonymous referees, Manferd Gilli and Peter Winker for useful suggestions and criticisms. This work is supported in part by the National Natural Science Foundation of China (No. 60375003) and the Aeronautics and Astronautics Basal Science Foundation of China (No. 03I53059).

References

Balke, N.S., Fomby, T.B., 1997. Threshold cointegration. Internat. Econom. Rev. 38, 627–645.

Baragona, R., Battaglia, F., Cucina, D., 2004. Fitting piecewise linear threshold autoregressive models by means of genetic algorithms. Comput. Statist. Data Anal. 47, 277–295.

Chen, H., Flann, N.S., Watson, D.W., 1998. Parallel genetic simulated annealing: a massively parallel SIMD algorithm. IEEE Trans. Parallel Distrib. Syst. 9, 126–136.

Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Reading, MA.

Goldberg, D.E., 1991. Real-coded genetic algorithms, virtual alphabets, and blocking. Complex Syst. 5, 139–167.
Gudla, P.K., Ganguli, R., 2005. An automated hybrid genetic-conjugate gradient algorithm for multimodal optimization problems. Appl. Math. Comput. 167, 1457–1474.
Hansen, B., Seo, B., 2002. Testing for two-regime threshold cointegration in vector error correction models. J. Econometrics 110, 293–318.
Jeong, I.K., Lee, J.J., 1996. Adaptive simulated annealing genetic algorithm for system identification. Engng Appl. Artificial lntelligence 9 (5), 523–532.
Johansen, S., 1988. Statistical analysis of cointegration vectors. J. Econom. Dynamics Control 12, 231–254.
Kirkpatrick, S., Gelatt, C., Vecchi, M., 1983. Optimization by simulated annealing. Science 220 (4598), 498–516.
Lo, M., Zivot, E., 2001. Threshold cointegration and nonlinear adjustment to the law of one price. Macroeconomic Dynamics 5 (4), 533–576.
Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E., 1953. Equation of state calculations by fast computing machines. J. Chem. Phys. 21, 1087–1092.
Youssef, H., Sadiq, M.S., Adiche, H., 2001. Evolutionary algorithms, simulated annealing and tabu search: a comparative study. Engng. Appl. Artificial Intellingence 14, 167–181.
Zhou, X., Wang, J., 2005. A genetic method of LAD estimation for models with censored data. Comput. Statist. Data Anal. 48, 451–466.
Winker, P., Gilli, M., 2004. Applications of optimization heuristics to estimation and modelling problems. Comput. Statist. Data Anal. 47 (2), 211–223.
Wong, K.P., Wong, Y.W., 1994. Genetic and genetic/simulated-annealing approaches to economic dispatch. IEE Proc. Gener. Transm. Distrib. 141 (5), 507–513.