J. Statist. Comput. Simul., 1997, Vol. 59, pp. 1–23
Reprints available directly from the publisher
Photocopying permitted by license only

MONTE CARLO FILTER USING THE GENETIC ALGORITHM OPERATORS

TOMOYUKI HIGUCHI

*The Institute of Statistical Mathematics, 4-6-7 Minami-Azabu,
Minato-ku, Tokyo 106, Japan*

(Received 29 February 1997; In final form 18 March 1997)

We consider the generalized state space model (GSSM) which is an extension of the state space model to the non-Gaussian and non-linear model. There are serious problems in the GSSM approach because of the need for numerical integration over a state space. A Monte Carlo method for filtering and smoothing, called the Monte Carlo Filter (MCF), has been proposed to overcome this numerical problem. It has been pointed out that there exists a close relationship between the MCF and the genetic algorithm (GA) and that an essential structure involved in the MCF is quite similar to that in the GA. In this study, we try to replace the step of the prediction by the mutation and crossover operators in the GA, and demonstrate their performance as the system noise. We furthermore propose a smoothing algorithm in which a massively simple parallel procedure plays an important role. The proposed method is first applied to a simple problem and then to a seasonal adjustment for quarterly data sets in order to illustrate its broad applicability.

Keywords: Bayesian approach: generalized state space model (GSSM): genetic algorithm (GA): self-organizing; seasonal adjustment

1. INTRODUCTION

The state space model (Anderson and Moore (1979)) has become a useful and powerful tool for modeling a non-stationary time series and then has been used for a wide variety of applications. because such models are typically very flexible thereby enabling a very good fit to data (e.g., see references in Gersch and Kitagawa (1988), West and Harrison (1989) and Harvey (1989)). In a case where the system is a linear and noise processes are Gaussian, optimal conditional mean estimates are provided by an efficient recursion known as Kalman filter and smoother (Kalman (1960)). Along the line of state space model, many authors attempted to consider extensions to nonlinear and non-Gaussian systems (e.g., see the references in Fahrmeir (1992)). In this article, we deal with this extended state space model which is sometimes called the generalized state space model (GSSM) (Kitagawa (1987, 1989)). The nonlinear model known as a dynamic generalized linear model (DGLM) (e.g., West *et al.* (1985), West and Harrison (1989), Fahrmeir (1992), Frühwirth-Schnatter (1994)) is expressed as a special case of the GSSM.

Consider the GSSM, in which we allow for nonlinear model structure and non-Gaussian distribution for specifying both system and observation noises,

$$\begin{array}{rl} \textit{system model} & x_n = f(x_{n-1}, v_n), \quad \text{and} \\ \textit{observation model} & y_n \sim r(\cdot|x_n), \quad n = 1, \ldots, N \end{array} \tag{1}$$

where x_n is the $k \times 1$ state vector, y_n is the univariate observation at discrete-time of n. Although we deal with only scaler observation, a description discussed below can be easily extend to the vector observation. $\{v_n\}$ is the l dimensional independently and identically distributed (i.i.d.) sample with $v_n \sim q(\cdot|\lambda_s)$, where λ_s is an unknown hyperparameter vector for describing a form of q (Lindley and Smith (1972)). $r(\cdot)$ is the conditional distribution of y_n, given x_n, with an unknown hyperparameter vector λ_o. Typically unknown forms for $q(\cdot)$ and $r(\cdot)$ are selected among competing candidates, for an example, the appropriate distribution families on a basis of maximizing the data-based model likelihood (e.g., Kitagawa (1987), Frühwirth-Schnatter (1994)).

In a framework of the GSSM, the distribution of x_n conditional on $Y_n = [y_1, \ldots, y_n]$, i.e., posterior density $p(x_n|Y_n)$ is easily determined from $p(x_{n-1}|Y_{n-1})$ by a sequential scheme (recursion) composed of two steps: prediction and filtering (Kitagawa (1987), Harvey (1989)). There are serious limitations to the approach by this recursive formula because of the need of numerical integration of order k (Fahrmeir (1992), Frühwirth-Schnatter (1994)). Actually, a simple but flexible approach based on a piecewise linear approximation to the conditional distribution is feasible only for lower dimensions ($k \leq 2$) (Kitagawa (1987)). To avoid numerical problems due to repeated multidimensional integrations, new algorithm based on a Monte Carlo method, called the *Monte Carlo Filter* (MCF) here, has been proposed (Kitagawa (1993, 1996), Gordon *et al.* (1993)). Gordon *et al.* (1993) called it the *Bootstrap filter*. In the MCF any conditional probability density function is approximated by many of their realizations. Alternative approximate filters to deal with higher-dimensional problems are attractive and see West and Harrison (1989). Carlin *et al.* (1992), Fahrmeir (1992), Frühwirth-Schnatter (1994) and references therein.

To review the MCF, suppose that $p(x_n|Y_{n-1})$ and $p(x_n|Y_n)$ are approximated by the m realizations

$$\begin{aligned} Z_P^{(n)} &\equiv \{z_{P,i}^{(n)}|\, i=1,\ldots,m\} \quad \text{and} \\ Z_F^{(n)} &\equiv \{z_{F,i}^{(n)}|\, i=1,\ldots,m\}, \end{aligned} \tag{2}$$

respectively. The subscripts, P and F, mean the prediction and filtering, respectively, and the superscript (n) is a discrete-time n. At each n the recursive calculation for estimating $p(x_n|Y_n)$ is realized in two steps (Kitagawa (1993, 1996), Gordon *et al.* (1993), Higuchi (1996)):

prediction
Get $Z_P^{(n)}$ of which each element is obtained by $z_{P,i}^{(n)} = f(z_{F,i}^{(n-1)}, v_i^{(n)})$.

filtering
Obtain $Z_F^{(n)}$ by resampling $z_{P,i}^{(n)}$ with the probability in proportion to

$$\begin{aligned} p(z^{(n)} = z_{P,i}^{(n)}|Y_n) &= \frac{r(y_n|z_{P,i}^{(n)}) \cdot (1/m)}{\sum_{i=1}^{m} r(y_n|z_{P,i}^{(n)}) \cdot (1/m)} \\ &= \frac{r(y_n|z_{P,i}^{(n)})}{\sum_{i=1}^{m} r(y_n|z_{P,i}^{(n)})} \,. \end{aligned} \tag{3}$$

Here $v_i^{(n)}$ is a realization drawn from $q(\cdot)$.

A similar structure to the algorithm of the MCF appears in the Genetic algorithm (GA) (Holland (1975)) that is a population-based search procedure developed in analogy to genetic laws and natural selection. In general the GA is characterized by keeping the m

candidates for optimal solution at each iteration composed of three steps: crossover, mutation, and reproduction (or selection) (Holland (1975)), Goldberg (1989), Davis (1991), Whitley (1994)). It has been pointed out in Higuchi (1996) that the filtering procedure is identical to the reproduction procedure by regarding $r(\cdot)/m$ as the evaluation function in the GA and that the prediction plays a similar role to mutation and crossover operators in giving a wide variety among population.

Using an analogy between the MCF and GA, an interpretation of the MCF from the viewpoint of the GA has been presented and several practical issues concerning its implementation has been investigated (Higuchi (1996)). For example, several schemes for the reproduction in the GA are available for the resampling scheme in the filtering procedure of the MCF. In this study, we make more ambitious use of the analogy to overcome the intrinsic problem of the MCF such that the sampling error inherent to the Monte Carlo approximation to any conditional distribution appears in an evaluation of the likelihood and makes it impractical to optimize the hyperparameter values according to the likelihood. One of possible mitigations is to an effort to estimate the likelihood more precisely by increasing the number of particles and performing an evaluation of the likelihood as many times as we can do. However, more sophisticated way to include a procedure capable of a self-tuning on the hyperparameter values in the MCF is desirable. Since the crossover, which is usually considered the distinguishing feature of the GA, possesses the *self-organizing* mechanism to reach an optimal solution, we will make use of this self-organizing effect in the prediction step of the MCF. Finally, the crossover operator will be expected to deal with more complicated situation where the system parameters depend on time.

Our goal throughout this article is to replace the prediction procedure by the genetic operators such as the crossover and mutation. Section 2 provides a methodology to accommodate the genetic operators to the GSSM. In Section 3, we present a method for the smoothing algorithm including genetic operators. In Section 4 we begin our illustrative analysis with a smoothing of a time series and consider an effect of mutation and crossover rates on a performance of the proposed method. In addition we apply it to more complicated model, seasonal adjustment.

2. GENERALIZED STATE SPACE MODEL WITH GENETIC OPERATOR

In this section we propose a new algorithm that accommodates the genetic operators to the MCF. For convenience we call it *Genetic Algorithm Filter* (GAF) henceforth.

2.1. Coding

Any type of the GA requires a design of effective codings that is problem dependent. In this study we consider an unsigned fixed-point integer coding with l_x bits which is usually adopted (e.g., Stoffa and Sen (1991)). A possible range of each component in the state vector $x_n(k')$ $(k'=1,\ldots,k)$, $[U_{\min,k'}, U_{\max,k'}]$, is encoded and mapped linearly onto the interval $[0, 2^{l_x}-1]$. Clearly, we have to control this specified range with care. Note that l_x is also chosen to give sufficient (or desired) precision for an estimate of each component of the state vector. In this study we set $l_x=8$ for all component.

2.2. Growth Operation

In the MCF, the state vector x_n evolves according to the system model at each time, and contributes to the displacement of the bulk of the distribution of x_n in the k-dimensional space. For example, when we treat the usual state space model with a system model given by $x_n=Fx_{n-1}+Gv_n$, the Kalman filter provides us with an estimate of the mean value of $p(x_n|Y_{n-1})$, $x_{n|n-1}$, defined by $x_{n|n-1}=Fx_{n-1|n-1}$, where F and G are $k\times k$ and $k\times l$ matrices, respectively, and $x_{n-1|n-1}$ is, of course, the mean of $p(x_{n-1}|Y_{n-1})$. This recursive form obviously indicates that the system model in the GSSM produces the bulk motion of the distribution of x_n. It turns out that, in the MCF, an apparent movement of the population at each step is induced by the systematic behavior of each particle which is driven by $f(\cdot)$ with $v_n=0$ in the prediction.

On the other hand, in the GA, there is no such movement in model space that is independent of any performance of optimization. We therefore have to include a special step for the GAF in which $z_{F,i}^{(n-1)}$ moves according to the system model just before undergoing the crossover. Specifically we denote the resulting particle by $z^{(n)}_{G,i}$; i.e., we obtain $z^{(n)}_{G,i}$ by

$$z^{(n)}_{G,i} = f(z^{(n-1)}_{F,i},\ v^{(n)}_i \equiv 0) \quad \text{for } i = 1, \ldots, m. \tag{4}$$

In this step $z^{(n-1)}_{F,i}$ suffers from no random change due to the system noise, and thus we call this step *growth operation.*

2.3. Crossover

We have to accommodate the genetic operators to the GAF with the multi-dimensional state vector. To explain it, let $z^{(n)}_{G,i}(k')$ denote the k'th component of $z^{(n)}_{G,i}$:

$$z^{(n)}_{G,i} = [z^{(n)}_{G,i}(1), \ldots, z^{(n)}_{G,i}(k)]', \tag{5}$$

where t is a transposition. Usually the binary codes for each component are concatenated resulting in a long string binary numbers, and then a crossover operator is performed on this long string. This means that a crossover site is selected somewhere within the long string. Such crossover operator is not suitable for the GAF, because it does not deal with a situation of the GSSM where some component is updated deterministically in the prediction process without any random change due to the system noise. Namely a crossover operator should be performed on each component with each crossover rate $P_C(k')$. We explain it by taking an example of the seasonal adjustment.

The primary object of the seasonal adjustment is to estimate a trend component with removing a seasonal effect in data, and usually achieved by decomposing a data into at least three components: the trend, seasonal, and observation noise components: $y_n = t_n + s_n + w_n$. We assume that the trend component satisfies a second order stochastically perturbed difference equation. In this study, we treat the quarterly data y_n, and then the seasonal component (of which the period is four) is represented by the four stochastic difference equation.

This decomposition is realized by using the following state space model (Kitagawa (1989)):

system model

$$\begin{bmatrix} t_n \\ t_{n-1} \\ s_n \\ s_{n-1} \\ s_{n-2} \end{bmatrix} = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} t_{n-1} \\ t_{n-2} \\ s_{n-1} \\ s_{n-2} \\ s_{n-3} \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_n(1) \\ v_n(2) \end{bmatrix}, \quad \text{and} \tag{6}$$

observation model

$$y_n = [1 \ 0 \ 1 \ 0 \ 0] \begin{bmatrix} t_n \\ t_{n-1} \\ s_n \\ s_{n-1} \\ s_{n-2} \end{bmatrix} + w_n, \quad n = 1, \ldots, N, \tag{7}$$

where $v_n(1)$ and $v_n(2)$ are assumed to have each own probability distribution function. Clearly $x_n = [t_n, t_{n-1}, s_n, s_{n-1}, s_{n-2}]^t$. While, in this model, two components, t_n and s_n suffer from a random effect due to v_n, other components are updated deterministically. As a result, these components free from a random change should be prohibited from receiving any genetic operator causing a random change. Taking account of this fact, we perform a crossover only on t_n and s_n in this case. In other words, a crossover is performed on other components with a probability of the crossover rate $P_C(k') = 0$. Generally a crossover is operated only on components which receive a random effect through a prediction process.

A detail explanation of the crossover we are proposing is as follows. First we select two parent particles (strings) $(z_{G,i}^{(n)}, z_{G,j}^{(n)})$ randomly and get $m/2$ pairs undergoing a mating. While there is a mating scheme in which some parent is permitted to mate twice (i.e., bigamy), three time, etc, but we allow a parent to mate only once with randomly selected spouse. This mating scheme is accomplished by permuting $z_{G,i}^{(n)}$ randomly and by paring two particles. An efficient algorithm for a random permutation is realized as follows. First we draw m uniform random numbers $\varepsilon_1, \ldots, \varepsilon_m$ between 0 and 1 and make a pair of $(z_{G,i}^{(n)}, \varepsilon_i)$. Next we sort m pairs in order of ε_i. Then we get the pair of $(z_{G,2i'-1}^{(n)}, z_{G,2i'}^{(n)})$ for $i' = 1, \ldots, m/2$.

There are various mating schemes to make an original GA more efficient (e.g., Goldberg (1989)). Most of them are, however, proposed on a basis of an optimization and then need not be considered in the GAF. In this study we propose a minor modification of the mating scheme mentioned above. The proposed mating scheme is that because a crossover has no effect on identical particles, we prohibit a parent from mating with an identical particle in order to keep a diversity within population (prohibition of intermarriage).

A crossover that is performed on the selected pair $(z^{(n)}_{G,2i'-1}, z^{(n)}_{G,2i'})$ proceeds as follows.

1. For each component, draw a uniform random number $U_C \sim U[0,1)$. If $U_C < P_C(k')$, do the following crossover.
   (a) Choose a crossing site l_C by drawing a uniform random integer between 1 and $l_x - 1$.
   (b) Swap all bits less than l_C inclusively.

The resulting crossover yields two new particles $(z^{(n)}_{C,2i'-1}, z^{(n)}_{C,2i'})$ which we call *offsprings* or *children*. It should be noticed that a subscript C is used to indicate the crossover.

2.4. Mutation

The following mutation procedure is performed on each particle $z^{(n)}_{C,i} (i = 1, \ldots, m)$. A probability of mutation is defined for each component according to the same reason as explained above. For a component that needs no stochastic fluctuation in the prediction process, we set a mutation rate $P_M(k') = 0$. A mutation is carried out as follows.

1. For each component, draw a uniform random number $U_M \sim U[0,1)$. If $U_M < P_M(k')$, do the following mutation.
   (a) Choose a mutation point l_M by drawing a uniform random integer between 1 and l_x.
   (b) Flip the l_M th bit.

The particle which goes through a process of the mutation is specified by $z^{(n)}_{M,i}$.

2.5. Analogy with Natural System

As mentioned above, the GA is a population-based search procedure developed in analogy to genetic laws and natural selection, and thus it is obvious that there exists a correspondence between the GA and natural system. In this study we interpret the operators involved in the GAF in terms of the terminology in natural system. Figure 1 shows a schematic explanation of the procedure of the GAF together with a correspondence of the operators to their terminology in natural system which is denoted in the parentheses. The procedures which are indicated by a shadow are introduced into the GAF to replace the prediction procedure of the MCF. Words of "Marriage" and "Birth" are pasted on this figure simply to help the readers understand a series of steps of the GAF.

While a qualitative interpretation of the parallel drown between the prediction in the MCF and the crossover in the GA is obvious, a quantitative discussion on the differences of these as an operator to act on the state vector is usually difficult. This is due to a fact that the

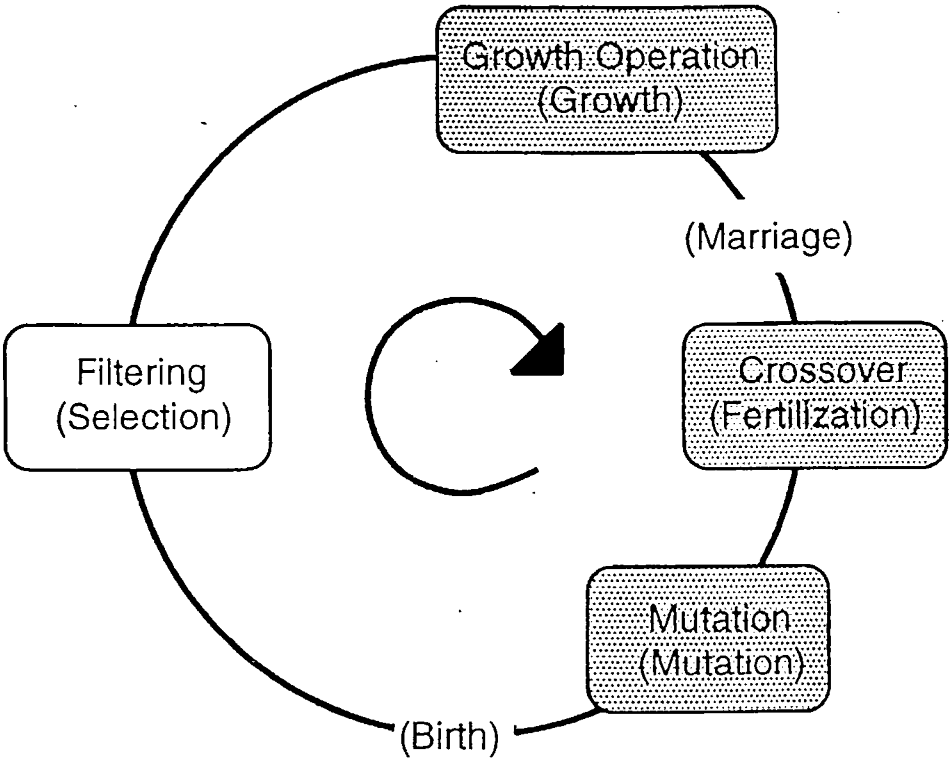

FIGURE 1 Analogy with natural system.

crossover, which is the distinguishing feature of the GA, is defined as a mutual action between two strings, and not as an action independent of an individual string like a system noise in the MCF. However, an understanding of the effect caused by the crossover in the GAF is possible in a context of the MCF by investigating the difference between the state vector before and after the crossover step. For example, when we consider the simplest GSSM that will be treated in Section 4.1, a quantity given by $\hat{v}_i^{(n)} = z_C^{(n)} - z_{G,i}^{(n)}$ has been examined and it was found that a distribution of $\hat{v}_i^{(n)}$ possesses the eminent non-Gaussian characteristics such as having a heavy tail (Higuchi (1996)). An analytic form of a distribution of $\hat{v}_i^{(n)}$, given special assumptions, can be also derived for this problem (Higuchi (1996)).

On the contrary to the crossover, the mutation is easily understood in the framework of the MCF, because the mutation is an operator which acts on each individual string without involving any other string. Actually, a coding scheme determines an analytic form of the distribution of the difference of the state vector before and after the mutation. The definition of the mutation clearly indicates that an apparent feature of the non-Gaussian distribution always appears as having much probability on giving larger noise, compared with a case of the Gaussian. Based on these results, the genetic operators such as the crossover and mutation are here employed in attempt to induce the stochastic fluctuation in the system model with the non-Guassian characteristics, without making any assumption on a form of $q(v_n|\lambda_s)$.

3. PARALLEL SMOOTHING PROCEDURE

In this section we describe a fixed-lag smoothing procedure, based on the original smoothing algorithm of the MCF (Kitagawa (1993, 1996)), capable of implementing it on a parallel computer easily. In fact, any calculation carried out at j^{th} parallel processor, described below, corresponds to the j^{th} trial on the single processor in this study. Gordon *et al.* (1993) has not dealt with the smoothing problem, but according to their line Doucet *et al.* (1995) has proposed the smoothing procedure that is identical to Kitagawa's one.

3.1. Estimate of the State Vector

At each parallel processor, we do the following calculation. The recursive calculation for the fixed L-lag smoothing begins by defining a quantity $T_i^{(-L+1:0)}$ for $i=1,\ldots,m$, called the

history

$$T_i^{(-L+1:0)} = [z_{\delta,i}^{(-L+1)}, z_{\delta,i}^{(-L+2)}, \ldots, z_{\delta,i}^{(-1)}, z_{\delta,i}^{(0)}] \tag{8}$$

where $z_{\delta,i}^{(-L+1)},\ldots,z_{\delta,i}^{(-1)}$ need not be specified, but only $z_{\delta,i}^{(0)}$, namely an initial distribution of the state vector, must be given prior to the following recursion.

The fixed-lag smoother is realized by repeating the following steps for $n=1,\ldots,N-L$. First we perform a growth, crossover and mutation operators to $Z_\delta^{(n-1)}$ and then get $Z_M^{(n)}$, where the population composed of $z_{\delta,i}^{(n-1)}(i=1,\ldots,m)$ is denoted by $Z_\delta^{(n-1)}$. Adding the particle $z_{M,i}^{(n)}$ to the end of histories $T_i^{(n-L:n-1)}$, we make a new history

$$U_i^{(n-L:n)} = [z_{\delta,i}^{(n-L)}, \ldots, z_{\delta,i}^{(n-1)}, z_{M,i}^{(n)}]. \tag{9}$$

Next we obtain $T_i^{\prime(n-L:n)}$ for $j=1,\ldots,m$ by resampling $U_i^{(n-L:n)}$ with the probability in proportion to

$$\begin{aligned}\gamma_i &= \frac{r(y_n|z_{M,i}^{(n)})\cdot(1/m)}{\sum_{i=1}^m r(y_n|z_{M,i}^{(n)})\cdot(1/m)}\\ &= \frac{r(y_n|z_{M,i}^{(n)})}{\sum_{i=1}^m r(y_n|z_{M,i}^{(n)})}\ .\end{aligned} \tag{10}$$

The denominator $(1/m)\cdot\sum_{i=1}^m r(y_n \mid z_{M,i}^{(n)})$ gives us a numerical evaluation of the likelihood of y_n, given Y_{n-1}, $p(y_n|Y_{n-1})$ (Kitagawa (1993, 1996), Higuchi (1996)). For simplicity, we henceforth denote its logarithmic value by $l^{\mathrm{MCF}(n)}$.

The each component of the resulting $T_i^{\prime(n-L:n)}$ is specified by

$$\begin{aligned} T_i^{\prime(n-L:n)} &= U_{i'}^{(n-L:n)}\\ \left[z_{\delta',i}^{(n-L)}, \ldots, z_{\delta',i}^{(n-1)}, z_{\delta',i}^{(n)}\right] &= \left[z_{\delta,i'}^{(n-L)}, \ldots, z_{\delta,i'}^{(n-1)}, z_{M,i'}^{(n)}\right], \end{aligned} \tag{11}$$

where i' is some i between 1 and m selected by a random number device. The first component $z_{\delta',i}^{(n-L)}$ is interpreted as a particle $z_{S,i}^{(n-L)}$ in a context of the fixed lag smoother. The estimate of the state vector is therefore determined by taking an average or median of each component of $z_{\delta',i}^{(n-L)}$, and denoted by $\hat{z}_S^{(n-L)}$. Once $\hat{z}_S^{(n-L)}$ is calculated, we set $T_i^{(n-L+1:n)} \leftarrow T_i'^{(n-L+1:n)}$ which means

$$T_i^{(n-L+1:n)} \leftarrow T_i'^{(n-L+1:n)}$$

$$\left[z_{\delta,i}^{(n-L+1)}, \ldots, z_{\delta,i}^{(n-1)}, z_{\delta,i}^{(n)}\right] \leftarrow \left[z_{\delta',i}^{(n-L+1)}, \ldots, z_{\delta',i}^{(n-1)}, z_{\delta',i}^{(n)}\right], \qquad (12)$$

and return to the growth operation mentioned above.

As a result of this fixed L-lag smoother, we get a trajectory of the state vector $\hat{z}_S^{(n)}$ for $n = 1, \ldots, N-L$ together with the log-likelihood l^{MCF} that is defined by

$$l^{\mathrm{MCF}} = \sum_{n=1}^{N} l^{\mathrm{MCF}(n)}. \qquad (13)$$

The estimation of the state vector for $n = (N-L+1), (N-L+2), \ldots, (N-1), (N)$ is determined by taking an average or median of each component of $z_{\delta,i}^{(n)}$ in $T_i^{(N-L+1:N)}$ for each n. An alternative way is to perform a backward fixed-lag smoother.

3.2. Final Estimate of the State Vector

The procedures for estimating $\hat{z}_S^{(n)}$ is common to all parallel processor (number of parallel processor is specified by N_p), and so that we specifically indicate $\hat{z}_S^{(n)}$ at the j^{th} parallel processor by $\hat{z}_{S,[j]}^{(n)}$. Accordingly the log-likelihood l^{MCF} obtained at the jth processor is also denoted by $l_{[j]}^{\mathrm{MCF}}$. In this study the final estimate of the state vector $z_{[*]}^{(n)}$ is defined by a trajectory with the maximum log-likelihood. We denote the parallel processor having the maximum l^{MCF} by $*$ and then $z_{[*]}^{(n)}$ is given by $\hat{z}_{S,[*]}^{(n)}$. In the actual application, there is no general criterion that can tell us the rough minimum N_p necessary for obtaining the satisfactory result. Several rules gathered from many experiences with various GSSM models are only available; usually $N_p \geq 100$ is recommended.

3.3. Efficiency of Implementations

The difference between the GAF and MCF exists only in the prediction step in the MCF. Consider we deal with only scaler system noise v_n in the system model. While the MCF requires the specification on the system noise distribution $q(v_n|\lambda_s)$, two values, P_C and P_M, should be specified for implementing the GAF. Since the number of hyperparameters, i.e., the dimension of λ_s, is two or three in a case of $q(v_n|\lambda_s)$ being the Pearson system that is frequently used in the GSSM (e.g., Kitagawa (1987)), an efficiency of implementing the GAF on the computer is almost same as that for the MCF. As for the required computational tasks under fixed $p(v_n|\lambda_s)$ and fixed value of (P_C, P_M), these two algorithms relies heavily on a pseudo random generator devices, and there is no significant difference between them as to the computational time.

As mentioned in Introduction, the value of l^{MCF}, that plays an important role in identifying the hyperparameters, is subject to an sampling error inherent to the Monte Carlo approximation to any conditional distribution. In particular, when we adopt the non-Gaussian distribution for $q(v_n|\lambda_s)$, the sampling error is too large to evaluate l^{MCF}. As a result, a large number of evaluations for l^{MCF}, usually achieved by using the parallel computer, is necessary for searching for the best hyperparameters values. However, we are usually interested in the estimate on the state vector and not in the fine search for the hyperparameter values. In other words, an assumption such that the state vector suffers the random fluctuation due to the system noise following $v_n \sim q(v_n|\lambda_s)$, is simply introduced for the sake of convenience in order to realize the stochastic fluctuation in the system model numerically. In such case practically, we are satisfied with a sequence of $p(x_n|Y_{n-1})$ $(n=1,\ldots,N)$ resulting in yielding the larger value of l^{MCF}.

The GAF takes advantage of this practical request. Since the random effect produced by the crossover in the GAF is determined by a mutual action among many particles, there appears the self-tuning on how wide the system gives a variety among population. In short, the characteristics of a stochastic fluctuation induced by the crossover can evolve and it is possible that the resulting random effect seen at time of $n=N$ shows quite different behavior from that at an initial time. Hence, the GAF is a robust approach in terms of obtaining the final estimate on the state vector, with much less effort made to search for the system parameters to give the larger l^{MCF}, rather than the MCF. As a result, the required computational time for the GAF is expected to be smaller than that for the MCF finally. Of course, if we deal faithfully with the GSSM, the GAF cannot be employed from this viewpoint.

When the hyperparameter is allowed for the time-varying structure, the usual type of the GSSM should be extended to include the time-dependence structure of hyperparameters, but it is difficult to determine how to realize this idea in the GSSM numerically. In contrast, based on a merit that the stochastic property of the random effect generated by the crossover is allowed to change itself, and then the GAF can deal with this problem. Although this self-tuning mechanism realized by the crossover is useful for an analysis of the time series with a gradual change in the system parameters, the crossover no longer produce enough variety among the population after the selction concentrates the m particles on a few particles. Once such concentration occurs, only mutation in the GAF can produce the variety among identical particles. Meanwhile, the system noise in the MCF always produces the variety even if the filtering step allows only a few particles to survive under a given circumstance.

4. APPLICATION

4.1. Simple Example

We illustrate the performance of the GAF by taking the following very simple model for smoothing a time series. Figure 2(a) demonstrates the artificially generated data which is obtained by adding the i.i.d. white Gaussian noise with a variance of 0.1 to the step function with a jump of 1 at $n = 51$. The GAF begins by giving a time series model expressed in terms of the GSSM. We consider the following model:

$$\begin{array}{rl} \textit{system model} & t_n = t_{n-1} + v_n \\ \textit{observation model} & y_n = t_n + w_n, \end{array} \tag{14}$$

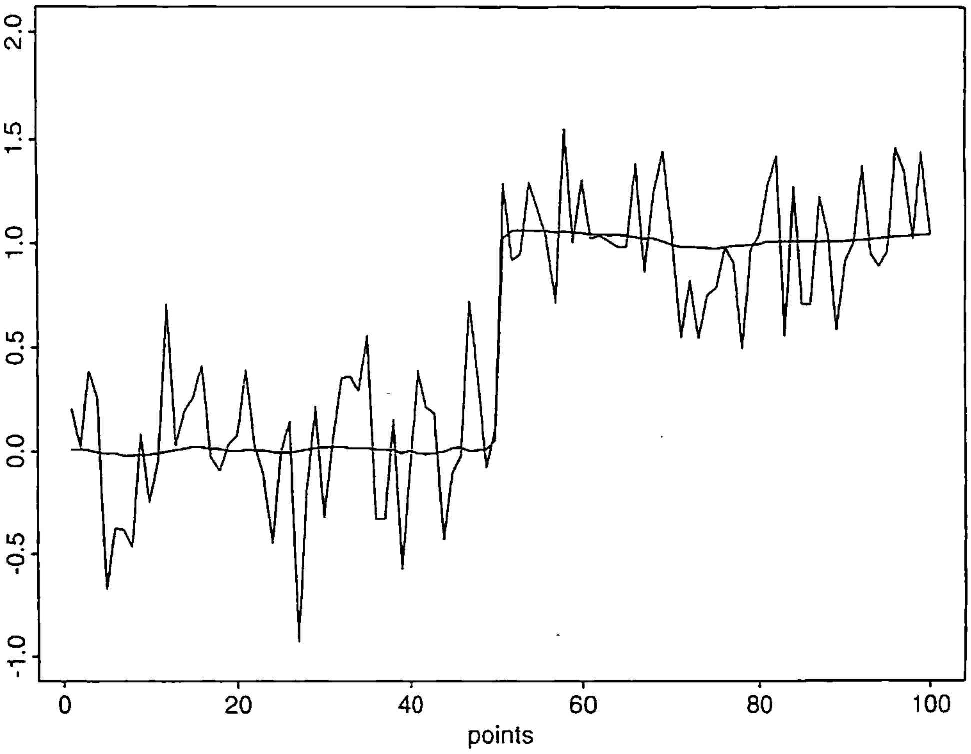

FIGURE 2a Artificially generated data y_n. The heavy curve indicates the result of the smoother based on the piecewise approximation to the conditional distribution (Kitagawa (1987)) for a case where $q(\cdot)$ and $r(\cdot)$ are a Cauchy and Gaussian, respectively. The optimal hyperparameters are determined by maximizing the log-likelihood.

where t_n is a trend component at time of n and then $x_n = [t_n]$. Accordingly v_n is the 1-dimensional white noise sequences, and thereby requiring only two control parameters, the crossover and mutation rates: P_C and P_M. It should be noticed in this case that no growth operation is needed because of the simplicity of the system model.

In this figure we show a curve which is obtained by applying the smoother based on the piecewise linear approximation to the conditional distribution (Kitagawa (1987)). Optimal forms for describing the distribution functions of both system and observation noises are objectively determined by maximizing the log-likelihood. For simplicity, we fix $r(y_n|t_n)\overset{\text{equiv}}{=} r(w_n = y_n - t_n)$ to be Gaussian. In this case, a Cauchy distribution is selected for the better system noise form, compared with the Gaussian distribution. Of course, an optimal value of the hyperparameter is determined so as to maximize the log-likelihood. The heavy curve shown in this figure is the result of the smoother for the best model.

The parallel smoothing procedure with $N_p = 100$ and $m = 10^3$ is applied against several values of (P_C, P_M) and yields $l_{[*]}^{\mathrm{MCF}}$ as a function of (P_C, P_M). A grid search in the (P_C, P_M) space is conducted and then the maximum $l_{[*]}^{\mathrm{MCF}}$ is obtained for $(P_C, P_M) = (0.05, 0.15)$. The heavy curve in Figure 2(b) shows $z_{[*]}^{(n)}$ with these optimal control parameter values. The solid curve is the worst process which is given by $\hat{z}_{S.[j]}^{(n)}$ with the minimum log-likelihood for the same value of (P_C, P_M). The broken curve is obtained by taking an average of all process with $(P_C, P_M) = (0.05, 0.15)$.

The discrepancy of the heavy curve from the result for the Cauchy distribution is quite small. The worst case suggests that, when a population cannot fit the drastic change in the environment such as the jump of the trend, a crossover operator is incapable of generating a new individual particle (string) with good fitness to the new environment after $n = 51$. Only mutation has a chance to drive the population toward $t_n \simeq 1$. Namely, unless a mutation operator forces a

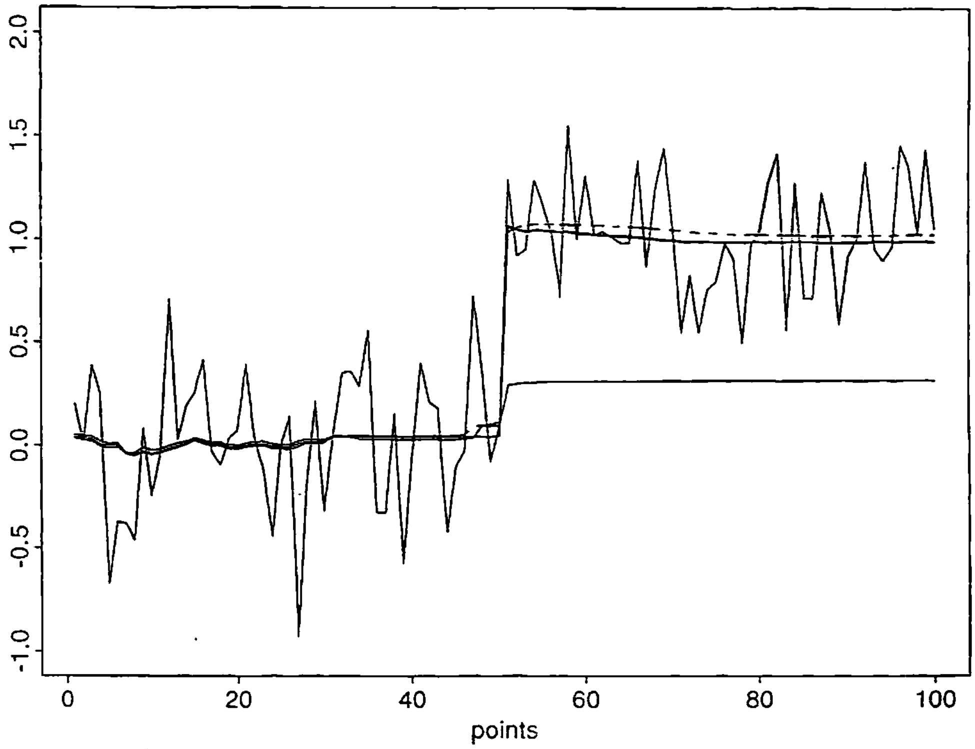

FIGURE 2b $z_{[*]}^{(n)}$ is defined by $\hat{z}_{S.[*]}^{(n)}$ that has a maximum log-likelihood among N_p processor. In addition, we superpose two curves $\hat{z}_{S.[\mathrm{mean}]}^{(n)}$ and $\hat{z}_{S.[\mathrm{worst}]}^{(n)}$ onto this figure. $\hat{z}_{S.[\mathrm{worst}]}^{(n)}$ is given by $\hat{z}_{S.[j]}^{(n)}$ with the minimum log-likelihood and $\hat{z}_{S.[\mathrm{mean}]}^{(n)}$ are determined by taking an average of all $\hat{z}_{S.[j]}^{(n)} (j = 1, \ldots, N_p)$.

particle to skip from $t_n \simeq 0$ to $t_n \simeq 1$, a reproduction is going on among individuals which cannot accommodate their-self to the data.

4.2. Effect of Mutation and Crossover Rate

Figure 3 shows a distribution of $l_{[j]}^{\mathrm{MCF}}$ as a function of P_C for the fixed value of $P_M = 0.15$. The dot and cross denote the maximum and mean values of $l_{[j]}^{\mathrm{MCF}}$, respectively. Each error bar corresponds to the 0.25, 0.5 (median), and 0.75 quantiles of $l_{[j]}^{\mathrm{MCF}}$, respectively. N_p under the fixed value of (P_C, P_M) is 100. While the mean (or median) is sensitive to P_C, the maximum value significantly changes in a response to the larger value of $0.7 < P_C$. This means that even inappropriate value of P_C has the small possibility to yield the relatively larger $l_{[j]}^{\mathrm{MCF}}$.

Figure 4(a) demonstrates the contour map of the maximum value of $l_{[j]}^{\mathrm{MCF}}$, $l_{[*]}^{\mathrm{MCF}}$, as a function of P_C and P_M. The horizontal and vertical

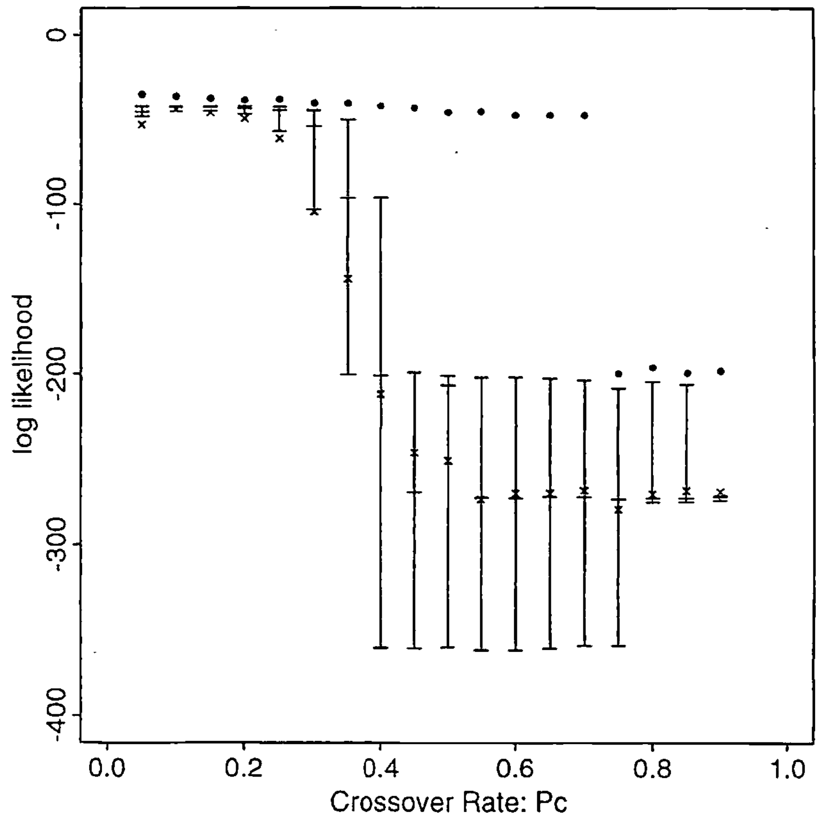

FIGURE 3 Distributions of $l_{[j]}^{\mathrm{MCF}}$ as a function of P_C for given $P_M = 0.15$. For fixed values P_C and P_M, $N_p = 100$. The dot and cross denote the maximum and mean of $l_{[j]}^{\mathrm{MCF}}$, respectively. Each error bar indicates the 0.25, 0.5 and 0.75 quantiles of $l_{[j]}^{\mathrm{MCF}}$, respectively.

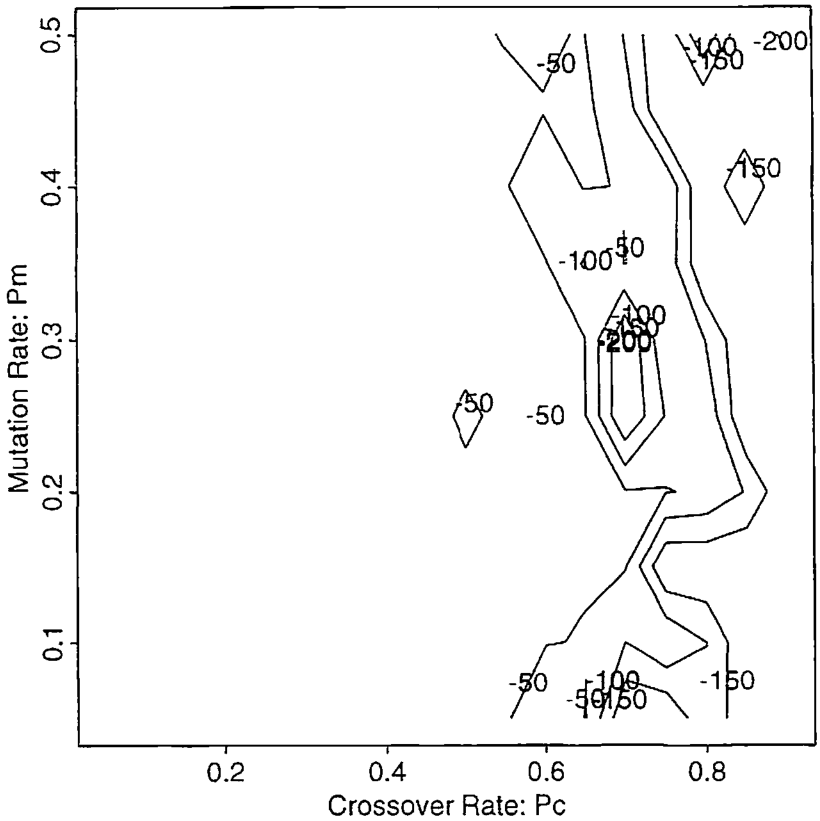

FIGURE 4a Contour of the maximum $l_{[j]}^{\mathrm{MCF}}$, $l_{[*]}^{\mathrm{MCF}}$, as a function of (P_C, P_M). For given (P_C, P_M), we calculate 100 values of $l_{[j]}^{\mathrm{MCF}}$ and determine $l_{[*]}^{\mathrm{MCF}}$. The section in Figure 3 is overlaid in this figure.

axes denote P_C and P_M, respectively. The section in the previous figure are overlaid in this figure. It is clearly seen that as demonstrated in previous figure, $l_{[*]}^{\mathrm{MCF}}$ is insensitive to the smaller P_C and shows a fairly flat in this region. It should be noticed that $l_{[*]}^{\mathrm{MCF}}$ depends mainly on P_C, not on P_M. We investigate in detail a dependence of $l_{[*]}^{\mathrm{MCF}}$ in the portion with P_C less than 0.6. It is again seen that a dependence of $l_{[*]}^{\mathrm{MCF}}$ on P_M is minor, compared with the variation of $l_{[*]}^{\mathrm{MCF}}$ against P_C. From this figure, we can also identify a tendency that $l_{[*]}^{\mathrm{MCF}}$ becomes smaller as increasing in P_C.

4.3. Seasonal Adjustment

We show two examples of an application of the GAF to the seasonal adjustment. First we consider the GSSM explained in the Section 2.3. Figure 5(a) is the increase of inventories of private company in Japan.

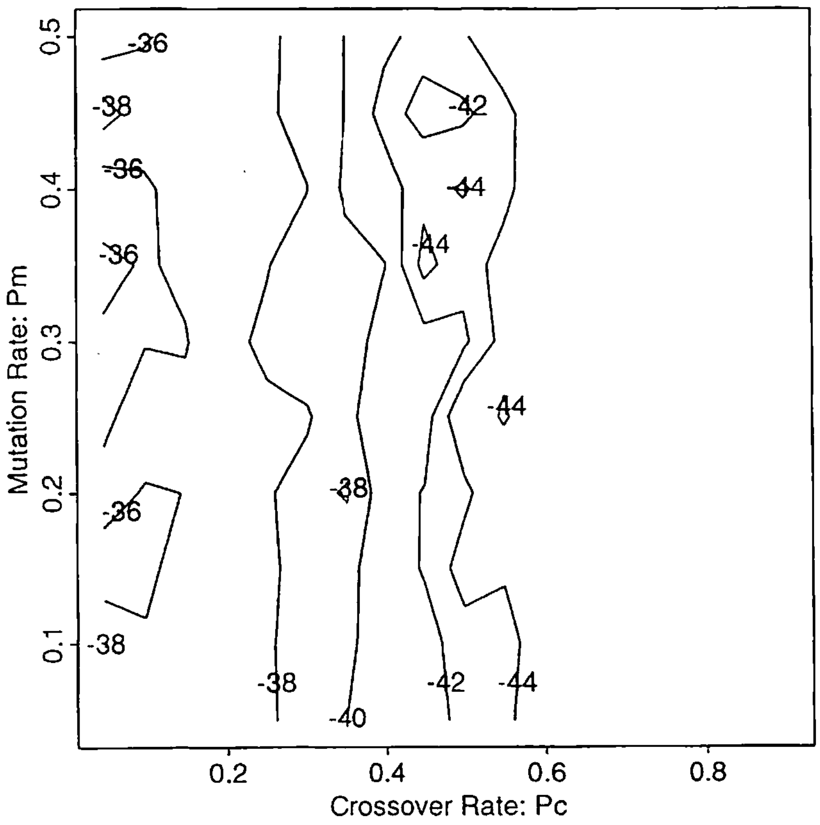

FIGURE 4b A detail dependence of $l^{\mathrm{MCF}}_{[\cdot]}$ in the portion with $P_C \leq 0.6$.

The trend jump up seen in this figure comes from the oil shock (crisis). The seasonal pattern also changed significantly during this period. Results of the GAF are demonstrated in Figures 5(a)–(c). Of course, the parallel smoothing procedure are made against several control parameters sets; in this case, there are four control parameters due to the presence of two system noises: $v_n(1)$ and $v_n(2)$. The estimated trend component is superposed in Panel (a). Panels (b) and (c) show the seasonal and residual components, respectively.

The next application is based on more complicated model in which another component c_n is added to the observation model: $y_n = t_n + s_n + c_n + w_n$. This component is used for representing a biennial oscillation with a period of T_c which can be expressed by the following special autoregressive (AR) model:

$$c_n = a \cdot c_{n-1} - c_{n-2} + v_n(3), \qquad (15)$$

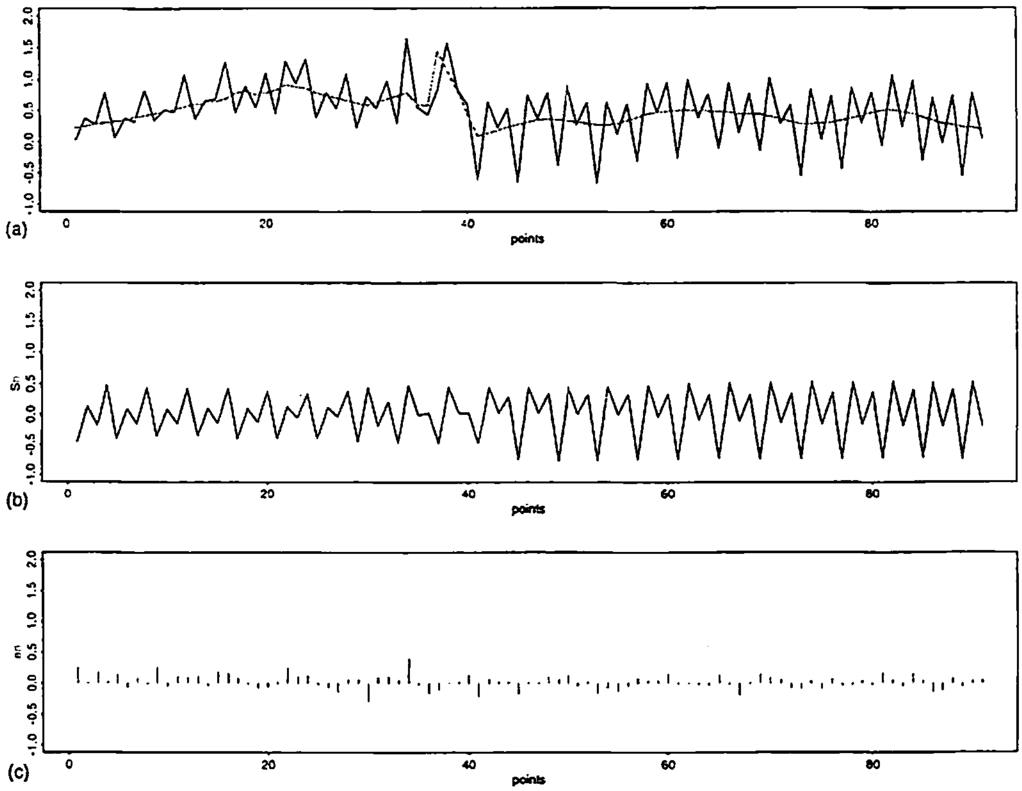

FIGURE 5 (a) The data and estimated trend component. (b) The seasonal component. (c) The residual component.

where $a=2\cos(2\pi/T_c)$. Accordingly, $x_n=[t_n, t_{n-1}, s_n, s_{n-1}, s_{n-2}, c_n, c_{n-1}]^t$. For the quarterly data, T_c is set $T_c=8$. This model is very suitable for the oscillation with the time-varying amplitude and with a drastic change in the phase (e.g., Higuchi *et al.* (1988), West and Harrison (1989) and West (1995)). Figure 6 shows an simulated data, original and estimated trend components. Figure 7 demonstrates an original given curve and estimated component for the trend, seasonal, and biennial components, respectively. These results are obtained with $m=3\times10^3$ and $N_p=10^3$. We can see a good agreement between the given and estimated curves.

5. CONCLUDING REMARKS

We would like to emphasize that because a final result $z_{[*]}^{(n)}$ depends weakly on control parameters, the GAF is simply conducted against

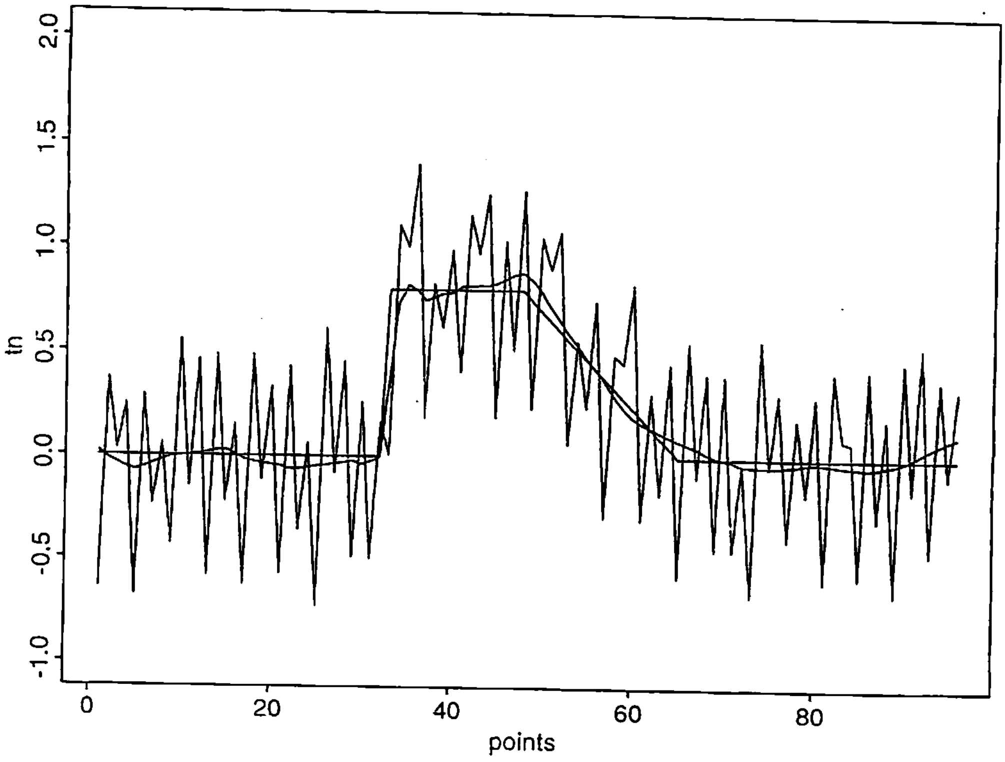

FIGURE 6 The data, given and estimated trend components.

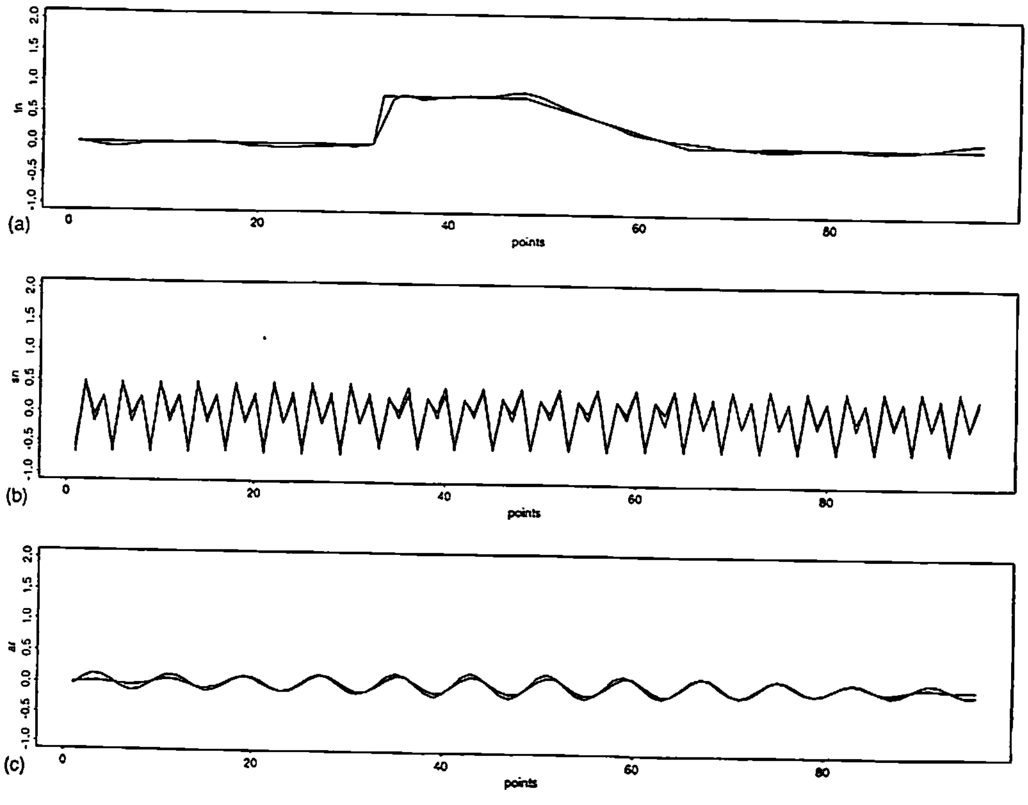

FIGURE 7 (a) The original and estimated curves for the trend component. (b) The original and estimated curves for the seasonal component. (c) The original and estimated curves for the biennial component.

several sets of (P_C, P_M) and does not require a fine optimization over control parameters. This is due to a fact that when N_p is sufficiently large, even inappropriate control parameters have a chance to yield a good result of $z_{[*]}^{(n)}$ that has a larger $l_{[*]}^{\mathrm{MCF}}$. This comes from a self-tuning on how much the system maintain the diversity among the population, so-called the *self-organizing effect* in the GAF. The applications of the GAF, proposed in this study, were limited only to three cases and then we need more numerical investigations as to whether the GAF can provides us with a satisfactory estimation for any kind of the GSSM.

Acknowledgments

I thank the referee for valuable comments on this article.

References

Anderson. B. D. O. and Moore. J. B. (1979) *Optimal Filtering*. Prentice-Hall, Englewood Cliffs. N. J.

Carlin. B. P.. Polson, N. G. and Stoffer. D. S. (1992) A Monte Carlo approach to nonnormal and nonlinear state-spacing modeling, *J. Am. Statist. Assoc.*, **87**(418), 493–500.

Davis, L. D. (1991) *Handbook of Genetic Algorithm*. Van Nostrand Reinhold. New York.

Doucet, A., Barat. E. and Duvaut, P. (1995) A Monte Carlo approach to recursive Bayesian state estimation. *Proc. IEEE Signal Processing/Athos Workshop on Higher Order Statistics*, 12–14, June, Girona, Spain.

Fahrmeir. L. (1992) Posterior mode estimation by extended Kalman filtering for multivariate dynamic generalized linear model. *J. Am. Statist. Assoc.*, **87**(418). 501–509.

Frühwirth-Schnatter. S. (1994) Applied state space modelling of non-Gaussian time series using integration-based Kalman filtering. *Statistics and Computing*, **4**, 259–269.

Gersch, W. and Kitagawa, G. (1988) Smoothness priors in time series, in *Bayesian Analysis of Time Series and Dynamic Models*. edited by J. C. Spall. pp. 431–476. Marcel Dekker, INC., New York and Basel.

Goldberg. D. E. (1989) Genetic Algorithm *in Search. Optimization and Machine Learning*. Addison-Wesley Publishing Company, Massachusetts.

Gordon, N. J., Salmond, D. J. and Smith, A. F. M. (1993) Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F*, **140**(2), 107–113.

Harvey, A. (1989) *Forcasting, Structural Time Series Models, and Kalman Filter*, Cambridge: Cambridge University Press.

Higuchi, T. (1996) Genetic Algorithm and Monte Carlo Filter (in Japanese and with English Abstract). *Proc. Institute of Statistical Mathematics*. **44**(1), 19–30.

Higuchi, T., Kita. K. and Ogawa, T. (1988) Bayesian statistical inference to remove periodic noise in the optical observations aboard a spacecraft, *Applied Optics*, **27**, 4514–4519.

Holland. J. H. (1975) *Adoption in natural and artificial systems*. Ann Arbor: The University of Michigan Press.

Kalman, R. E. (1960) A new approach to linear filtering and prediction problems, *Trans. ASME, J. Basic Eng.*, **82D**, 35–45.

Kitagawa, G. (1987) Non-Gaussian state space modelling of nonstationary time series (with discussion). *J. Am. Stat. Assoc.*, **79**, 1032–1063.

Kitagawa, G. (1989) Non-Gaussian seasonal adjustment, *Computers Math. Applic.*, **18**(6/7), 503–514.

Kitagawa, G. (1993) A Monte Carlo filtering and smoothing method for non-Gaussian Non-linear state space models, *Proc. 2nd. International U.S.–Japan joint seminar on statistical time series analysis*, 25–29 January, pp. 110–131, Honolulu, USA.

Kitagawa, G. (1996) Monte Carlo filter and smoother for non-gaussian nonlinear state space models, *J. Computational and Graphical Statistics*, **5**(1). 1–25.

Lindley, D. V. and Smith, A. F. M. (1972) Bayes estimates of the linear model (with discussion), *J. R. Statist. Soc. B*, **34**. 1–41.

Stoffa, P. L. and Sen, M. K. (1991) Nonlinear multiparameter optimization using genetic algorithms: inversion of plane wave seismonograms, *Geophysics*, **56**(11), 1794–1810.

West, M. (1995) Bayesian inference in cyclical component dynamic linear models, *J. Am. Statist. Assoc.*, **90**(432), 1301–1312.

West, M. and Harrison, P. J. (1989) *Bayesian forecasting and dynamic model*, Springer-Verlag, New York.

West, M., Harrison, P. J. and Migon, H. S. (1985) Dynamic generalized linear models and Bayesian forecasting (with discussion), *J. Am. Statist. Assoc.*, **80**(389), 73–97.

Whitley, D. (1994) A genetic algorithm tutorial, *Statistics and Computing*, **4**, 63–85.