Meta-heuristic Methods for Outliers Detection in Multivariate Time Series

Domenico Cucina, Antonietta di Salvatore, Mattheos K. Protopapas

Department of Statistics, University of Rome La Sapienza, Piazzale Aldo Moro 5, I-00100 Roma, Italy

Abstract

Meta-heuristic methods to detect multiple additive outliers in multivariate time series are proposed. The implemented algorithms are: simulated annealing, threshold accepting and two different versions of genetic algorithms. In contrast with many of the existing methods, they do not require to specify a vector ARMA model for the data and are able to detect any number of potential outliers simultaneously reducing possible masking and swamping effects. A generalized AIC-like criterion has been used as objective function where the penalty constant has been suggested by both a simulation study and a theoretical approximation. The comparison and the performance of the proposed methods are illustrated by simulation studies and real data analysis. Simulation results show that the proposed approaches are able to handle patches of additive outliers.

Keywords: Genetic algorithms, Simulated annealing, Threshold accepting

1. Introduction

Outliers are commonly defined as observations which appear to be inconsistent with the remainder of the data set, and may be due to occasional unexpected events. The detection of outliers is an important problem in time series analysis because they can have adverse effects on model identification, parameter estimation (see Chang and Tiao [15]) and forecasting (see Chen and Liu [17]). The presence of just a few items of anomalous data can lead to model misspecification, biased parameter estimation, and poor forecasts. Therefore, it is essential to identify outliers data, estimate their magnitude and correct the time series, avoiding false identifications (i.e. observations

that are identified as outliers while they are not). Several approaches have been proposed in the literature for handling outliers in univariate time series. Among these methods we can distinguish those based on an explicit model (parametric approach) from the ones using non-explicit models (non-parametric approach). For the parametric approach, Fox [26] developed a likelihood ratio test for detecting outliers in a pure autoregressive model. Chang and Tiao [15], Chang et al. [16], Tsay [54, 55], Chen and Liu [17] extended this test to an autoregressive integrated moving-average (ARIMA) model and proposed an iterative procedure for detecting multiple outliers. For the non-parametric approach, Ljung [39], Ljung [40], Peña [47], Gómez et al. [32], Baragona and Battaglia [3] and Battaglia and Baragona [10] proposed specific procedures based on the relationship between additive outliers and linear interpolator, while Baragona et al. [5] used a genetic algorithm.

For multivariate time series, only three procedures have been proposed. Tsay et al. [53] proposed a sequential detection procedure, which we will call the TPP method, based on individual and joint likelihood ratio statistics; this method requires an initial specification of a vector ARMA model. Galeano et al. [27], Baragona and Battaglia [4] proposed a method based on univariate outlier detection applied to some useful linear combinations of the vector time series. The optimal combinations are found by projection pursuit in the first paper and independent component analysis (ICA) in the second one. Barbieri [7] used a Bayesian method and finally a graphical method was explored by Khattree and Naik [36].

Multiple outliers, especially those occurring close in time, often have severe masking effect (one outlier masks a second outlier) and smearing effect (misspecification of correct data as outliers) that can easily render the iterative outlier detection methods inefficient. A special case of multiple outliers is a patch of additive outliers. For univariate time series this problem has been addressed firstly by Bruce and Martin [14]. They define a procedure for detecting outlier patches by detecting blocks of consecutive observations. Other useful references for the patch detection are McCulloch and Tsay [44], Barnett et al. [8] and Justel et al. [35]. For multivariate time series, only Baragona and Battaglia [4] report simulation results for an outlier patch.

Unlike the univariate case where there are specific procedures on the identification of consecutive outliers, in multivariate time series framework, methods for identification of consecutive outliers do not exist.

We propose a class of meta-heuristic algorithms to overcome the difficulties of iterative procedures in detecting multiple additive outliers in multivari-

ate time series. This class includes: simulated annealing (SA)(Kirkpatrick et al. [37], Rayward-Smith et al. [48]), threshold accepting (TA) (Winker [58]) and genetic algorithm (GA) (Holland [34]; Goldberg [31]). These methods are illustrated in appendix. Our procedures are less vulnerable to the masking and smearing effects because they evaluate several outlier pattern where all observations that are possibly outlying ones are simultaneously considered. In this way, meta-heuristic methods deal efficiently the detection of patch of additive outliers.

Each outlier configuration is evaluated by a generalised AIC-criterion where the penalty constant is suggested by both a simulation study and a theoretical approximation. So, the meta-heuristic algorithms seem able to provide more flexibility and adaptation to the outlier detection problem.

2. Meta-heuristic methods

Many optimization problems do not satisfy the necessary conditions to guarantee the convergence of traditional numerical methods. For instance, in order to apply standard gradient methods to maximum likelihood estimation we need a globally convex likelihood function, however there are a number of relevant cases with non convex likelihood functions or functions with several local optima. Another class of "hard" problems is when the solution space is discrete and large. These problems are known as combinatorial problems. A simple approach for solving an instance of a combinatorial problem is to list all the feasible solutions, evaluate their objective function, and pick the best. However, for a combinatorial problem of a reasonable size, the complete enumeration of its elements is not feasible, and most available searching algorithms are likely to yield some local optimum as a result (Rayward-Smith et al. [48]).

Meta-heuristic algorithms are often used to solve such problem instances. These methods do not rely on a set of strong assumptions about the optimization problem, on the contrary, they are robust to changes in the characteristics of the problem. But, on the other side, they do not produce a deterministic solution but a high quality stochastic approximation to the global optimum.

In this work we are interested in the following methods: simulated annealing, threshold accepting and genetic algorithms. The first two are classified as *local search methods*. Classical local search algorithms move from an initial random solution ξ^c to another one, chosen in a neighborhood of ξ^c, that has

a better value of the objective function $f(\cdot)$. The procedure is iterated until a stopping criterion is satisfied. However, these algorithms may get stuck in local optima. To avoid this problem, the local search algorithms we employ here, sometimes accept worse solutions than the current one. Genetic algorithms have been initially developed by Holland [34] and they are classified as *population based* methods, or *evolutionary algorithms*. They work on a whole set of solutions that is adapted simultaneously by imitating the evolutionary process of species that reproduce sexually.
We give a brief sketch of the three methods.

2.1. Simulated annealing

Simulated annealing (SA) is a random search technique based on an analogy between the way a metal cools and freezes into a minimum energy crystalline structure (the annealing process) and the search for a minimum in a more general system. The ideas that form the basis of this method were first published by Metropolis et al. [45] in an algorithm to simulate the annealing process. Only thirty years later, Kirkpatrick et al. [37] suggested that this type of simulation could be used to search for feasible solutions of an optimization problem, with the objective of converging to an optimal solution which minimizes the objective function. Recent applications of the simulated annealing algorithm are discussed by Vera and Díaz-García [56], Depril et al. [20], Duczmal and Assunção [21] and Angelis et al. [2].
In analogy with the annealing process, simulated annealing is characterised by the presence of a control parameter T called *temperature*. A simulation starts by choosing the initial temperature T_0, a final temperature T_f close to zero and a *cooling schedule* whereby the parameter T is decreased. The simulation stops when the temperature T assumes the value T_f. Different cooling schedules are suggested in the literature; in our work the geometric schedule is used :

$$T_t = aT_{t-1}, \tag{1}$$

where a is a constant close to 1.
The algorithm proceeds by choosing an initial random solution ξ^c as the current solution. A new potential solution ξ^n is drawn randomly in the neighborhood of ξ^c through a random mechanism explained in section 3.2. The new potential solution ξ^n will always be accepted as the new current solution if its objective function is smaller. Moreover the algorithm also accepts an increase of the objective function, but only with a given probability (see

statements 7 and 8 in Algorithm 1). Note that the acceptance probability is a function of the temperature T. That way, when T goes to zero, the system will increasingly often select candidate solutions that decrease the objective function, and avoid those that increase it. Every value assigned to T (and affecting the acceptance probability) will be used for SA_{iter} different choices of potential solutions. The total number of iterationI_{tot}^{SA} is obtained as the number of different temperatures $N_{temperature}$ (function of T_0, T_f, a) times the number of steps SA_{iter}. In terms of the minimization problem, the algorithm for a simulated annealing heuristic consists of the steps reported in algorithm (1).

Algorithm 1 Pseudocode for simulated annealing.

1: Initialise T_0, T_f, a and SA_{iter}
2: Generate initial solution ξ^c
3: $T = T_0$
4: **while** $T > T_f$ **do**
5: **for** $r = 1$ to SA_{iter} **do**
6: Compute $\xi^n \in N(\xi^c)$ (neighbour to current solution)
7: Compute $\Delta = f(\xi^n) - f(\xi^c)$ and generate u from a uniform random variable between 0 and 1
8: **if** $\Delta < 0$ or $e^{-\Delta/T} > u$ **then**
9: $\xi^c = \xi^n$
10: **end if**
11: **end for**
12: $T \leftarrow aT$
13: **end while**

2.2. Threshold accepting

Threshold accepting (TA) was introduced by Dueck and Scheuer [22] as a deterministic analog to simulated annealing. They applied the algorithm to a Travelling Salesman Problem and argued that their algorithm is superior to classical simulated annealing. It is a refined local search procedure which escapes local optima by accepting solutions which are worse,but no more than a given threshold. The algorithm is deterministic as it uses a deterministic acceptance criterion instead of the probabilistic one used in simulated annealing for accepting worse solutions. The number of steps where we explore the neighborhood for improving the solution is fixed. The threshold is decreased

iteratively and reaches the value of zero after a given number of steps. The TA algorithm has an easy parameterization, it is robust to changes in problem characteristics and works well for many problem instances. . Threshold accepting has been successfully applied to different areas of statistics and econometrics (Winker and Fang [59], Fang et al. [25], Winker [57], Winker [58], Gilli and Winker [29], Maringer and Winker [43], Lin et al. [38], Lyra et al. [42], Winker et al. [60]). An extensive introduction to TA is given in Winker [58].

Algorithm (2) provides the pseudo-code for a prototype threshold accepting implementation for a minimization problem.

Algorithm 2 Pseudocode for Threshold Accepting.

1: Initialise N_t, TA_{iter},
2: Generate the sequence τ_h, $h = 1, \ldots, N_t$
3: Generate initial solution ξ^c
4: **for** $h = 1$ to N_t **do**
5: **for** $r = 1$ to TA_{iter} **do**
6: Compute $\xi^n \in N(\xi^c)$ (neighbour to current solution)
7: Compute $\Delta = f(\xi^n) - f(\xi^c)$ and generate u from a uniform random variable between 0 and 1
8: **if** $\Delta < 0$ or $\Delta < \tau_h$ **then**
9: $\xi^c = \xi^n$
10: **end if**
11: **end for**
12: **end for**

Comparing SA and TA algorithm we can see that, first, the sequence of temperatures T is replaced by a sequence of N_t thresholds τ_h with $h = 1, \ldots, N_t$ and, the most important, the statement 8 of algorithm (1) is replaced by:

$$\textbf{if} \quad \Delta < \tau_h \quad \textbf{then} \quad \xi^c = \xi^n.$$

In this case the total number of iteration I_{tot}^{TA} is obtained as the product of the number of different thresholds N_t and the number of times each thresholds is used, TA_{iter}.

A crucial element of TA is its threshold sequence since it determines TA's ability to overcome local optima. Basically, the idea is to accept ξ^n

if its objective function value is better or if it is not much worse than that of ξ^c where not much worse means the deterioration may not exceed some threshold τ defined by the threshold sequence. In extreme cases of threshold settings, the algorithm behaves like a classical local search algorithm (if all threshold values are set equal to zero) or like a random walk (if all values of the threshold sequence are set to a very large value). Althöfer and Koschnick [1] demonstrated the convergence of the TA algorithm under the hypothesis that an appropriate threshold sequence exists. But in their proof they do not provide a way to construct an appropriate sequence. Consequently, the threshold sequence is often chosen in a rather ad hoc approach. Two simple procedures can be used to generate the sequence of thresholds. In the first place, one could use a linear sequence decreasing to zero. The advantage of a linear threshold sequence consists in the fact, that for tuning purposes only the first value of the sequence has to be selected as it fixes the whole sequence. Alternatively, we can generate a sequence of selected thresholds using the a data driven method suggested in Winker and Fang [59]. This procedure is detailed in algorithm (3).

Algorithm 3 Pseudocode for generating threshold sequence.

1: Initialise N_t and M
2: **for** $r = 1$ to M **do**
3: Randomly choose solution ξ_r^c
4: Randomly choose neighbour solution $\xi_r^n \in N(\xi_r^c)$
5: Compute $\Delta_r =\mid f(\xi_r^c) - f(\xi_r^n) \mid$
6: **end for**
7: Compute the cumulative distribution function F of Δ_r, $r = 1, \ldots, M$
8: Compute the sequence of thresholds $\tau_i = F^{-1}(\frac{N_t - 1}{N_t}), i = 1, \ldots, N_t$

This method uses a two step process to construct the threshold sequence. For the first step a large number (M) of possible solutions ξ^c is generated at random. Then, we compute the distances between the values of the objective function at random point ξ_r^c and its neighbour ξ_r^n, $\Delta_r =\mid f(\xi_r^c) - f(\xi_r^n) \mid$, $r = 1, 2, \ldots, M$. In the second step the cumulative empirical distribution F of the distances Δ_r is computed. This distribution is an approximation of the distribution of local relative changes of the objective function. The thresholds τ_i are computed as the quantiles Q_i corresponding to percentiles $P_i = \frac{N_t - i}{N_t}, i = 1, \ldots, N_t$. The threshold sequence will be monotonically decreasing to zero.

2.3. Genetic Algorithm

Genetic algorithms (GA), inspired by Holland [34], imitate the evolution process of biological systems, to optimize a given function. A GA uses a set of candidate solutions, called *population*, instead of one single current solution. In GA terminology, any candidate solution is encoded via a numerical vector called *chromosome*.
The GA proceeds by updating the population of active chromosomes (the sets of current candidate solutions) in rounds, called generations. In each generation, some of the active chromosomes are selected (parents-chromosomes) to form the chromosomes of the next generation (children-chromosomes). The selection process is based on an evaluation measure called *fitness function*, linked to the objective function, that assigns to each chromosome a positive number. This fitness is the determining factor for the probability to select a chromosome as a parent. A higher fitness value leads to higher probability that the corresponding chromosome will be one of the parents used to form the children-chromosomes. Children are formed by recombining (*crossover*) the genetic material of their two parents-chromosomes and perhaps after a random alteration of some of the genes (single digits of the chromosome) which is called *mutation* (Holland [see 34], Goldberg [see 31, for a detailed description]). The general structure of genetic algorithms is shown in algorithm (4).

3. Algorithm Features

This section further describes the algorithms implementation we used for outlier detection. A successful implementation of meta-heuristic methods is certainly crucial to obtain satisfactory results. Before a meta-heuristic method can be applied to a problem some important decisions have to be made. The three meta-heuristic methods require a suitable encoding for the problem and an appropriate definition of objective function. In addition, the algorithms TA and SA require the structure of the neighborhood while for genetic algorithms, operators of selection, crossover and mutation have to be chosen. The following sections describe the choices made.

3.1. Solution Encoding

An appropriate encoding scheme is a key issue for meta-heuristic methods. For all algorithms we use a binary encoding for the solutions of the outliers problem as suggested in Baragona et al. [5]. Any solution ξ^c is a binary

Algorithm 4 Pseudocode for genetic algorithms.

Set population size (*pop*), probability of crossover (*pcross*), probability of mutation (*pmut*), number of generations (*gen*)
Generate initial population P of solutions
for $i = 1$ to *gen* **do**
  Evaluate each individual's fitness
  Initialise $P' = \emptyset$ (set of children)
  for j =1 to $\frac{pop}{2}$ **do**
    Select individuals x_a and x_b from P with probability proportional to their fitness
    Generate p_1 and p_2 from a uniform random variable $U(0, 1)$
    if $p_1 > pcross$ **then**
      Apply crossover to x_a and x_b to produce x_a^{child} and x_b^{child}
    else
      $x_a^{child} = x_a$ and $x_b^{child} = x_b$
    end if
    if $p_2 > pmut$ **then**
      Apply mutation to x_a^{child} and x_b^{child}
    end if
    $P' = P' \cup \{x_a^{child}, x_b^{child}\}$
  end for
  $P = P'$
end for

string of length N, where N is the number of observations of the time series: $\xi^c = (\xi_1^c, \xi_2^c, \ldots, \xi_N^c)$, where ξ_i^c takes the value 1 if at time i there is an outlier (we assume that all the s components are influenced) and 0 otherwise. Then, ξ^c represent a chromosome of GA and ξ_i^c a gene. Obviously, the number of outliers for a given time series is unknown. We allow for solutions with a maximum number of outliers equal to g. The value of g should be chosen according to the series length and every relevant a priori information on its accuracy and instability. The constant g should be chosen large enough to allow for the detection of any reasonable number of outliers in the series.

Binary encoding implies that the solution space Ω consists of $\sum_{k=0}^{g} \binom{N}{k}$ distinct elements, since the total number of outliers is limited to a constant g.We can see that Ω is really large even when g is considerably lower than the length of the time series. All our algorithms either severely penalise solutions with a maximum number of outliers larger then g , or do not consider such solutions at all. TA and SA algorithms are built so that they do not evaluate solutions with more than g outliers. With regard to the GA, chromosomes not belonging to Ω will be severely penalised subtracting a positive quantity (the penalty factor *pen*) to the fitness (function to be maximised), so that the algorithm tends to avoid these chromosomes. We set the value of *pen* to 1,000.

3.2. Neighbourhood search in simulated annealing and threshold accepting

Each solution $\xi^c \in \Omega$ has an associated set of *neighbours*, $N(\xi^c) \subset \Omega$, called the neighbourhood of ξ^c where every $\xi^n \in N(\xi^c)$ may be reached directly from ξ^c by an operation called *move*.Given the current solution ξ^c, its neighborhood is constructed using three different moves: add an outlier; remove an outlier; change the position of an outlier.Since a maximum of g outliers is allowed, moves are applied according to the current solution in the following way: if ξ^c doesn't contain outliers (i.e., it is a string where every bit is 0), algorithms can only introduce an outlier; if ξ^c contains less than g outliers, algorithms can add, remove or change the position of an outlier, with probability 1/3; if ξ^c already contains g outliers, algorithms cannot proceed adding an outlier but can only remove or change the position of one of them, with probability 1/2.

3.3. Objective function

Let $y_t = [y_{1,t}, \ldots, y_{s,t}]'$ be a vector time series generated from a Gaussian s-dimensional jointly second order stationary real-valued process Y_t, with

mean zero for each component, covariance matrix $\mathbf{\Gamma}_u$ and inverse covariance matrix $\mathbf{\Gamma i}_u$ for integer lag u. When outliers are present, y_t is perturbed and unobservable. We suppose that k perturbations $\omega_t = [\omega_{1,t}, \ldots, \omega_{s,t}]'$ impact the series y_t at time points t_j, $j = 1, \ldots, k$ such that at each t_j they affect all s components. The total number of outlying data is equal to $h = ks$. Denote the observed time series by $z_t = [z_{1,t}, \ldots, z_{s,t}]'$ generated by the observable multivariate stochastic process Z_t. Given a sample of N observations we may write the following model

$$z = y + \mathbf{X}\omega, \tag{2}$$

where $z = [z_1', \ldots, z_N']'$ is the vector obtained by stacking the s component observations at each time point, $y = [y_1', \ldots, y_N']'$ is the vector obtained by stacking the s component of the unobservable outlier free time series at each time point, $\omega = [\omega_{t_1}', \ldots, \omega_{t_k}']'$ is the vector obtained by stacking the s components of the k outliers and $\mathbf{X}$ is a $Ns \times h$ pattern design matrix defined as follows.
For each t_j with $j = 1, \ldots, k$, the $[(t_j - 1)s + r, (j - 1)s + r]$-th entry is one for $r = 1, \ldots, s$. All the remaining entries are zero.
Matrix $\mathbf{X}$ contains information about the perturbed time indices of a given outlier pattern. Thus, each feasible solution ξ corresponds to a matrix $\mathbf{X}$.

The natural logarithm of the likelihood for z may be written

$$L_{(z;\mathbf{X},\omega)} = -\frac{Ns}{2}\log(2\pi) - \frac{1}{2}\log(\det\mathbf{\Gamma}) - \frac{1}{2}(z - \mathbf{X}\omega)'\mathbf{\Gamma}^{-1}(z - \mathbf{X}\omega), \tag{3}$$

where $\mathbf{\Gamma}$ denotes the $Ns \times Ns$ block Toeplitz matrix with $\mathbf{\Gamma}_{i-j}$ as the (i, j)-th block. Assuming both $\mathbf{\Gamma}$ and $\mathbf{X}$ known, the maximisation of (3) with respect to ω yields:

$$\hat{\omega} = (\mathbf{X}'\mathbf{\Gamma}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{\Gamma}^{-1}z. \tag{4}$$

If we approximate $\mathbf{\Gamma}^{-1}$ with $\mathbf{\Gamma i}$ (Shaman [51]), where $\mathbf{\Gamma i}$ denotes the $Ns \times Ns$ block Toeplitz matrix with $\mathbf{\Gamma i}_{i-j}$ as the (i, j)-th block, the maximum likelihood estimate (4) of ω takes the form:

$$\hat{\omega} = (\mathbf{X}'\mathbf{\Gamma i}\mathbf{X})^{-1}\mathbf{X}'\mathbf{\Gamma i}z. \tag{5}$$

Since $\mathbf{\Gamma i}$ is unknown, we have to estimate it from the data. We may use two different ways to estimate the inverse covariance matrices. A first approach is based on the estimation of the spectral density matrix and then taking the Fourier transform of its inverse. The second one fits a high-order

vector autoregressive model to the data and derives estimates of the inverse covariance matrix from the estimated parameters of the model (Battaglia [9]). Bhansali [11] has shown that under reasonable regularity conditions both methods give consistent and asymptotically Gaussian estimates. We follow the second approach where the estimates of the inverse covariance are obtained as follows:

$$\mathbf{\Gamma i}_u = \begin{cases} \sum_{j=u}^{m} \hat{\Phi}'_{j-u} \hat{\Sigma}^{-1} \hat{\Phi}_j & 0 \leq u \leq m \\ 0 & u > m \\ \hat{\Gamma}_i(-u)' & u < 0 \end{cases} \tag{6}$$

where $\hat{\Phi}_1, \hat{\Phi}_2, ..., \hat{\Phi}_m$ are the LS estimates of the parameter matrices of the VAR(m) model, $\hat{\Sigma}$ is the estimated variance matrix of the noise and where we set $\hat{\Phi}_0 = -\mathbf{I}$.

If we look at the expression (6) can see that the estimate of the inverse covariance depends on estimates of autoregressive parameters and the estimated variance-covariance matrix $\hat{\mathbf{\Sigma}}$ of innovations. In the presence of outliers the residuals of VAR model are contaminated, hence $\hat{\mathbf{\Sigma}}$ may be biased. For obtaining a better estimate we use the $\alpha\%$ trimmed method. To compute the $\alpha\%$ trimmed variance-covariance matrix $\hat{\mathbf{\Sigma}}$, we first remove the 5% largest values (according to their absolute values) and then compute $\hat{\mathbf{\Sigma}}$ based on trimmed sample.

The natural logarithm of the maximised likelihood is obtained by replacing ω by $\hat{\omega}$ and $\mathbf{\Gamma}^{-1}$ by $\hat{\mathbf{\Gamma}}\mathbf{i}$ in (3) :

$$\hat{L}_{(z;\mathbf{X},\omega)} = -\frac{Ns}{2}\log(2\pi) - \frac{1}{2}\log(\det \hat{\mathbf{\Gamma}}\mathbf{i}) - \frac{1}{2} z' \hat{\mathbf{\Gamma}}\mathbf{i} z - \frac{1}{2}(\mathbf{X}'\hat{\mathbf{\Gamma}}\mathbf{i}z)'(\mathbf{X}'\hat{\mathbf{\Gamma}}\mathbf{i}\mathbf{X})^{-1}\mathbf{X}'\hat{\mathbf{\Gamma}}\mathbf{i}z. \tag{7}$$

The matrix $\hat{\mathbf{\Gamma}}\mathbf{i}$ is fixed for any outlier pattern $\mathbf{X}$, so that the maximised likelihood in (7) depends only on matrix $\mathbf{X}$. Since matrix $\mathbf{X}$ conveys all information about the outlier's location, it seems natural to detect the outlier pattern by determining the matrix $\mathbf{X}$ maximising the quadratic form in (7)

$$L_{\mathbf{X}} = \frac{1}{2}(\mathbf{X}'\hat{\mathbf{\Gamma}}\mathbf{i}z)'(\mathbf{X}'\hat{\mathbf{\Gamma}}\mathbf{i}\mathbf{X})^{-1}\mathbf{X}'\hat{\mathbf{\Gamma}}\mathbf{i}z. \tag{8}$$

Obviously the likelihood increases when the number of estimated parameters $\hat{\omega}$, i.e. the number of outliers, is increased. Thus, in a similar fashion as identification criteria for model selection (see Bhansali and Downham [12]), we contrast the likelihood with a linear function of the number of outliers.

So, the search of outliers in a multivariate series is equivalent to search the chromosome ξ or the design matrix X that minimizes the following objective function:

$$f(\xi) = -2L_{\mathbf{X}} + ch, \tag{9}$$

where c is an arbitrary constant and h is the actual number of outliers. The function $f(\xi)$ depends on both the matrix $\mathbf{X}$ and the penalty constant c. Different values are suggested in literature for the constant c (see Bhansali and Downham [12]). We propose two alternative approaches for selecting appropriate c values in Section (5.1.1).

In a genetic algorithm, the fitness function assigns a positive real number to any possible solution in order to evaluate its plausibility, therefore in the GA we adopt the following non-decreasing transform of (9):

$$fitness = exp(-f(\xi)/\beta) \tag{10}$$

where β is a parameter of scale. In the following experiments this parameter is set equal to 100.

3.4. Operators and other implementation issues in the genetic algorithms

We do not use the "standard" randomly generated initial populations (Goldberg [31]), while in the algorithms used here, the initial populations consist of chromosomes with just one outlier, different from each other (the size of the population is less than the number of observations). At the beginning, all possible single-outlier chromosomes are generated and sorted in terms of fitness value and the initial population consists of the chromosomes having the largest fitness. In this way we evaluate from the beginning the most promising one-outlier patterns (see Baragona et al. [5]).

The "roulette wheel" rule is used for parent selection. The probability of a chromosome being selected as a parent is proportional to the rank of its fitness. Each selected couple of parents will produce two "children" by methods of crossover and mutation.

The crossover operator used is "uniform crossover" Goldberg [31]. For each gene of the first child, one of the parents is selected at random (with equal probability of selection) and its corresponding gene is inherited at the same position. The other parent is used to determine the second child's corresponding gene.

Finally, a probability is chosen for randomly changing the value of each gene of the child-chromosome (mutation). In our encoding, where we have

only two admissible values for a gene ("0" and "1") the application of the mutation operator is pretty straightforward.

The entire population of chromosomes is replaced by the offsprings created by the crossover and mutation processes at each generation except for the best chromosome, which survives to the next generation. This *elitist* strategy ensures that the fitness will never decrease through generations (Rudolph [49]).

4. The TPP procedure

Let $y_t = [y_{1,t}, \ldots, y_{s,t}]'$ be a k-dimensional vector time series following the stationary and invertible vector autoregressive moving average (VARMA) model:

$$\Phi(B)y_t = \Theta(B)\epsilon_t, t = 1, \ldots, N, \tag{11}$$

where B is the backshift operator such that $By_t = y_{t-1}$, $\Phi(B) = (I - \Phi_1 B - \Phi_2 B^2 - \ldots \Phi_p B^p)$ and $\Theta(B) = (I - \Theta_1 B - \Theta_2 B^2 - \ldots \Theta_p B^p)$ are $k \times k$ matrix polynomials of finite degrees p and q and $\epsilon_t = (\epsilon_{1t}, \ldots, \epsilon_{kt})$ is a sequence of independent and identically distributed (iid) Gaussian random vectors with mean 0 and positive-definite covariance matrix Σ. For the VARMA model in equation (11), we have the AR representation $\Pi(B)y_t = \epsilon_t$ where $\Pi(B) = \Theta(B)^{-1}\Phi(B) = I - \sum_{i=1}^{\infty} \Pi_i B^i$.

Given an observed time series $\mathbf{z} = [z_1, \ldots, z_N]$ where $z_t = [z_{1,t}, \ldots, z_{s,t}]'$ Tsay et al. [53] generalized additive univariate outliers to the vector case in a direct manner using the representation

$$z_t = y_t + \omega I_t^{(h)} \tag{12}$$

where $I_t^{(h)}$ is a dummy variable such that $I_h^{(h)} = 1$ and $I_t^{(h)} = 0$ if $t \neq h$, $\omega = (\omega_1, \omega_2, \ldots, \omega_k)'$ is the size of the outlier, and y_t follows a VARMA model.

Tsay et al. [53] showed that when the model order is known, the estimate of the size of an additive multivariate outlier at time h is given by:

$$\hat{\omega}_{A,h} = -(\sum_{i=0}^{N-h} \hat{\Pi}_i' \Sigma^{-1} \hat{\Pi}_i)^{-1} \sum_{i=0}^{N-h} \hat{\Pi}_i' \Sigma^{-1} \tag{13}$$

The covariance matrix of this estimate is $\Sigma_{A,h}^{-1} = (\sum_{i=0}^{N-h} \hat{\Pi}_i' \Sigma^{-1} \hat{\Pi}_i)^{-1}$. Tsay et al. [53] proposed an iterative procedure similar to that of the univariate case to detect multivariate outliers. Assuming no outlier, the procedure starts building a multivariate ARMA model for the series under study and let $\hat{a}_t$ be the estimated residuals and $\hat{\Pi}_i$ the estimated coefficients of the autoregressive representation. The second step of the procedure requires the calculation of the test statistic:

$$J_{max} = \max_{1 \leq t \leq N} \{J_t\},$$

where $J_t = \hat{\omega}_{A,t}' \Sigma_{A,h}^{-1} \hat{\omega}_{A,h}$. As in the univariate case, if J_{max} is significant at time index t_0 we identify a additive multivariate outlier at t_0. Once an outlier is identified, its impact on underlying time series is removed, using the model in equation (12). The adjusted series is treated as a new time series and the detecting procedure is iterated. The TPP method terminates when no significant outlier is detected. Tsay et al. [53] used simulation to generate finite sample critical values of statistic J_{max}.

5. Performance of meta-heuristic methods

To test the performance of meta-heuristic algorithms for identifying outliers in multivariate time series we applied the proposed methods to simulated time series models of the class VARIMA. We consider eight vector VARMA models, four bivariate ($s = 2$) and four trivariate models ($s = 3$). The sample sizes used are $N = 200$ and $N = 400$. The models considered in this simulation study and reported in Galeano et al. [27], Lütkepohl [41], Tsay et al. [53] are listed below.

Model 1 - VAR(1) bivariate model: $\Phi_1 = \begin{bmatrix} 0.6 & 0.2 \\ 0.2 & 0.4 \end{bmatrix}$.

Model 2 - VAR(1) bivariate model: $\Phi_1 = \begin{bmatrix} 0.2 & 0.3 \\ -0.6 & 1.1 \end{bmatrix}$.

Model 3 - VAR(2) bivariate model: $\Phi_1 = \begin{bmatrix} 0.5 & 0.1 \\ 0.4 & 0.5 \end{bmatrix}$ $\Phi_2 = \begin{bmatrix} 0.0 & 0.0 \\ 0.25 & 0.0 \end{bmatrix}$.

Model 4 - VARMA(1,1) bivariate model: $\Phi_1 = \begin{bmatrix} 0.6 & 0.2 \\ 0.2 & 0.4 \end{bmatrix}$ $\Theta_1 = \begin{bmatrix} -0.7 & 0.2 \\ -0.1 & 0.4 \end{bmatrix}$.

Model 5 - VAR(1) trivariate model: $\Phi_1 = \begin{bmatrix} 0.6 & 0.2 & 0.0 \\ 0.2 & 0.4 & 0.0 \\ 0.6 & 0.2 & 0.5 \end{bmatrix}$.

Model 6 - VAR(1) trivariate model: $\Phi_1 = \begin{bmatrix} 0.2 & 0.3 & 0.0 \\ -0.6 & 1.1 & 0.0 \\ 0.2 & 0.3 & 0.6 \end{bmatrix}$.

Model 7 - VAR(2) trivariate model:

$$\Phi_1 = \begin{bmatrix} -0.3 & 0.15 & 0.95 \\ 0.0 & -0.15 & 0.3 \\ 0.0 & 0.2 & -0.25 \end{bmatrix} \qquad \Phi_2 = \begin{bmatrix} -0.15 & 0.1 & 0.9 \\ 0.0 & 0.0 & 0.0 \\ 0.0 & 0.35 & 0.0 \end{bmatrix}.$$

Model 8 - VARMA(1,1) trivariate model:

$$\Phi_1 = \begin{bmatrix} 0.6 & 0.2 & 0.0 \\ 0.2 & 0.4 & 0.0 \\ 0.6 & 0.2 & 0.5 \end{bmatrix} \qquad \Theta_1 = \begin{bmatrix} -0.7 & 0.0 & 0.0 \\ -0.1 & -0.3 & 0.0 \\ -0.7 & 0.0 & -0.5 \end{bmatrix}.$$

where the covariance matrix of the Gaussian noise is the identity matrix for seven models. For the Model 2, it has diagonal entries equal to 1.0 and all off-diagonal entries equal to -0.2.

We have considered three different outlier configurations. The first two instances have a small contamination: the first configuration has two isolated outliers at time indices $t = 100, 150$, and the second one has a patch of two outliers introduced at time indices $t = 100, 101$. The last one consists in a heavier contamination, that includes two isolated outliers and a patch of three outliers introduced at time indices $t = 40, 100, 101, 102, 150$. For the first two cases the size of each outlier is chosen equal to $\omega = (3.5, 3.5)'$ for bivariate models and is chosen equal to $\omega = (3.5, 3.5, 3.5)'$ for the trivariate models. When the contamination is heavier we set the size of each outlier equal to $\omega = (5.0, 5.0)'$ for bivariate models and we set $\omega = (5.0, 5.0, 5.0)'$ for the trivariate models. For each model, sample size and outliers configuration, we generate a set of 100 time series.

We may consider several criteria for evaluating the performance procedure. Since the proposed procedures are designed to detect the outliers avoiding false identifications, we used as criteria of evaluation the relative frequency of correct outlier detection, defined as a correct identification of outlier pattern. For the case of two outliers $(100, 150$ or $100, 101)$ this means the relative frequency of detecting both outliers and only them, while for

the case of five outliers the relative frequency of detecting all five outliers and only them. For each method, we include the relative frequency of partial correct configuration detection (the relative frequency of only one outlier correctly detected or the relative frequency of less than five outliers correctly detected) and the relative frequency of wrong identifications (i.e., solutions where at least one observation identified as outlier in fact is not).

To apply the algorithms we need to determine the values of two types of parameters, one concerning the outlier problem itself and the other one regarding the meta-heuristic algorithms. The parameters of the outlier detection problem are three: the constant c in (9), the order of the multivariate autoregressive process m in (6) and the maximum number of outliers g.

5.1. The problem of parameters tuning

5.1.1. The constant c

In order to obtain the critical values of the test statistics for outlier detection (in univariate and multivariate time series) one can rely on simulation, using a large number of series from different models (Tsay et al. [53], Galeano et al. [27]). Programs TRAMO and SCA, for example, have outlier detection routines that use critical values obtained by such a simulation study. In our work we follow the same idea to establish the value of the constant c through a Monte Carlo experiment.

We consider the eight vector VARMA models listed above and sample sizes $N = 200, 400$. For each model and sample size, we generate a set of 500 time series and apply the algorithms to each set, employing different values of c and recording the corresponding values of the type I error α (where α is the frequency of clean observations identified as outliers).

Table 1 provides the c values obtained via simulation according to different values of α, models, dimensions and sample sizes. We observed that the three meta-heuristic algorithms lead to similar simulation results, therefore in Table 1 we do not consider the effect of these algorithms on the constant c. Table 1 suggests the following observations. First, for each α, we see only minor differences in the c values among different models given dimension and sample size. Second, the estimated c values increase with the sample size N and decrease with the dimension s. In general, the sample size and the time series dimension are important factors affecting the behaviour of constant c, while the type of model does not seem to have a significant effect.

Table 1: Simulation study: c values corresponding to different type I error α

N	s	Model	α		
			0.10	0.05	0.01
200	2	1	7.17	7.68	9.53
		2	7.33	7.93	9.25
		3	7.29	7.89	9.20
		4	7.18	7.84	9.50
	3	5	5.71	6.13	7.03
		6	5.78	6.30	7.20
		7	5.72	6.20	7.50
		8	5.67	6.17	7.50
400	2	1	8.10	8.83	10.20
		2	8.05	8.59	10.50
		3	7.93	8.55	9.80
		4	7.57	8.19	9.68
	3	5	6.13	6.70	8.13
		6	6.23	6.78	8.13
		7	6.15	6.67	8.00
		8	5.80	6.33	7.80

In real application, it may be necessary to analyze time series with different sample sizes and different number of components. To address this need, we suggest a theoretical approximation to derive the constant c.

Let us consider a test where under the null hypothesis the time series is outlier free and under the alternative hypothesis a single outlier occurs at unknown time t. We may use as statistic test:

$$\Lambda_{max} = \max_{1 \leq t \leq N} \{\Lambda_t\},$$

where $\Lambda_t = (\mathbf{X_t'}\hat{\mathbf{\Gamma}}\mathbf{i}z)'(\mathbf{X_t'}\hat{\mathbf{\Gamma}}\mathbf{i}\mathbf{X_t})^{-1}(\mathbf{X_t'}\hat{\mathbf{\Gamma}}\mathbf{i}z)$ and $\mathbf{X_t}$ is the pattern design corresponding to just one outlier at time t. The statistic Λ_t is a quadratic form and is distributed approximately as a chi-squared random variable with s degrees of freedom under the null hypothesis of no outliers. The finite sample distribution of Λ_{max} is complicated because of the correlation between the Λ_t. We may obtain the approximate percentiles of Λ_{max} assuming the independence among the Λ_t (though a relatively strong hypothesis)

$$P(\Lambda_{max} < \lambda_\alpha) = [P(\chi_s^2 < \lambda_\alpha)]^N = 1 - \alpha$$

or

$$P(\chi_s^2 < \lambda_\alpha) = (1 - \alpha)^{1/N},$$

where λ_α is the $(1-\alpha)$th quantile of the chi-square distribution with s degrees of freedom. We reject the null hypothesis if Λ_{max} is greater than the quantile λ_α at the α significance level.

Now, a problem arises, when the value of N increases the quantity $(1 - \alpha)^{1/N} \to 1$ and $\lambda_\alpha \to \infty$. To solve this problem we approximate the distribution of Λ_{max} with the Gumbel distribution:

$$P\left(\frac{\Lambda_{max} - d_N}{c_N} < \nu_\alpha\right) = \exp(-e^{-\nu_\alpha}) = 1 - \alpha,$$

where $d_N = 2(\log N + (\frac{s}{2} - 1)\log(\log N) - \log\Gamma(\frac{s}{2}))$ and $c_N = 2$, and we obtain the quantiles for Λ_{max} as $\lambda_\alpha = c_n\nu_\alpha + d_N$.

Now we can choose the constant c so that, whenever the null hypothesis of no outlier is accepted, the fitness of the chromosome with no outlier is larger than the one of the best one-outlier chromosome, or similarly $\Lambda_{max} < cs$, therefore put $c = \lambda_\alpha/s$.

In Table 2 we observe that the resulting theoretical c values are always slightly larger than the simulated ones, so that by using them the test is

more conservative. The discrepancy between the theoretical and simulated c values may be caused by the dependence among the Λ_t variables.

The c values used in our simulation experiments are the simulated ones values reported in Table 2 corresponding to $\alpha = 0.05$

Table 2: Simulated and *theoretical* c values corresponding to different type I error α, dimensions s and sample sizes N

N	s	α		
		0.10	0.05	0.01
200	2	7.2	7.9	9.4
		7.5	*8.3*	*9.9*
	3	5.7	6.2	7.3
		5.9	*6.4*	*7.5*
400	2	7.9	8.5	10.0
		8.2	*8.9*	*10.6*
	3	6.0	6.6	8.0
		6.3	*6.7*	*8.0*

5.1.2. The parameters m and g

To determine the value of order m in (6) we used the FPE criterion (Lütkepohl [41]). Alternatively we could use Akaike's Information Criterion which differs from FPE essentially by a term of order $O(N^{-2})$ and thus the two criteria are almost equivalent for large N (Lütkepohl [41]).

The value of the parameter g should be chosen by taking into account the length of the time series and all other relevant information. The value g affects the choice of the iteration number. If we increase the value for g it seems reasonable to increase also the iteration number of the meta-heuristic algorithms because a larger solution space has to be explored. The selected value for g is 5 for all algorithms.

5.2. Meta-heuristic control parameters tuning

A correct choice of the value of the control parameters is important for the performance of the meta-heuristic algorithms. For the genetic algorithms,

choices have to be made for the crossover probability ($pcross$), mutation probability ($pmut$), population size (pop) and the number of generations or termination criterion (gen).

For the simulated annealing algorithm we have to determine the initial temperature (T_0), final temperature (T_f), number of internal loop iterations at any temperature (SA_{iter}), and the constant a in (1), characterising the cooling schedule. As reported in section (2.1), the number of evaluations of the objective function I_{tot}^{SA} depends on the choice of these parameters. Generally we establish a number of I_{tot}^{SA} and the parameters are chosen in order to meet this constraint.

Threshold accepting requires two parameters: the number of thresholds (N_t) and the number of internal loop iterations at any threshold (TA_{iter}). Also in this case, if we set I_{tot}^{TA}, N_t and TA_{iter} must be chosen in such a way that their product is equal to I_{tot}^{TA}.

Unfortunately, the correct choice of the suitable parameter values is a difficult task because a wide range of values needs to be considered for each parameter and some parameters may be correlated with each other. Few theoretical guidelines are available while experience with practical applications of meta-heuristic algorithms is offered by a vast literature.

Regarding the TA, two simple procedures that can be used to generate the threshold sequences are reported in section (2.2). First, one might use a linear threshold sequence decreasing to zero and, alternatively, one might use a data driven generation of the threshold sequence (see algorithm (3)) suggested by Winker and Fang [59]. In our simulation experiments we set the value of M in algorithm (3) to 2,000. There are several examples in literature suggesting that the two procedures are equivalent, while in some applications the method proposed by Winker and Fang [59] yields better results. As far as the number of thresholds N_t is concerned, Gilli and Winker [30] suggested the minimum value for N_t around 10. However, when the total number of iterations I_{tot}^{TA} becomes very large, N_t might be increased.

Some guidelines for the choice of GA parameters may be found in de Jong [19], Schaffer et al. [50], da Graça Lobo [18], Eiben et al. [24], South et al. [52]. de Jong [19] studies the effects of some control parameters of GA on its performance, concerning the population size, and the crossover and mutation probabilities. Using five different function optimisation scenarios, De Jong systematically varies these parameters, analyses the results and thus establishes guidelines for robust parameter choice. De Jong suggests population size $pop = 50$, probability of crossover $pcross = 0.6$, probability

of mutation $pmut = 0.001$ and the adoption of the elitist strategy. However, other empirical studies (Eiben et al. [24], South et al. [52], da Graça Lobo [18], Gao [28], Grefenstette [33]) indicate different values for these parameters.

Regarding the SA algorithm, the initial temperature must be set to a high value enough to allow a move to almost any neighbourhood state. However, if the temperature starts at too high a value then the search can move to any neighbour and thus transform the search (at least in the early stages) into a random search. Then, a very high initial temperature may influence the quality of the performance and the length of the computational time. If we know the maximum distance (objective function difference) between one neighbour and another then we can use this information to calculate a starting temperature. Another method, suggested in (Rayward-Smith, 1996), is to start with a very high temperature and cool it rapidly until about 60% of worst solutions are being accepted. This forms the real starting temperature and it can now be cooled more slowly. A similar idea, suggested in (Dowsland, 1995), is to rapidly heat the system until a certain proportion of worse solutions are accepted and then slow cooling can start. This can be seen to be similar to how physical annealing works in that the material is heated until it is liquid and then cooling begins (i.e. once the material is a liquid it is pointless carrying on heating it).

Theoretically, the cooling rate parameter a in (1) assumes values between 0 and 1, while Eglese [23] reports that values used in practice lie between 0.8 and 0.99. Park and Kim [46] suggest a systematic procedure, based on the simplex method for non linear programming, to determine parameter values.

In conclusion we can say that there is no uniformly best choice of parameters, but specific problems may require different values. Baragona et al. [6] suggest that a good choice may be obtained by considering a range of possible values for the same problems. In our applications these parameters values are chosen by a tuning experiment. For each algorithm, different combinations of parameters values are tried, keeping the number of the objective function evaluations constant. We select the parameter combination that yields the largest frequency of true outlier pattern detection.

5.2.1. A simulation experiment for tuning parameters

The remaining parameter values are chosen by means of a tuning experiment where a set of 200 time series with $N = 400$ have been generated by Model 2, and outliers at time indices 100 and 150 are analysed. All the algorithms run with a total of 2,000 evaluations of the objective function.

For the SA, the T_f is always kept equal to 0.05. Since T_f has the role of stopping criterion, a value close to zero seems reasonable, thus the probability of accepting a worse solution during the last iterations is very small. The examined values for a are [0.90, 0.94, 0.95, 0.96] and for T_0 are [2, 4, 6, 8, 10]. For each combination, the number of internal loop iterations SA_{iter} is equal to the ratio between the total number of evaluations of the objective function (2000) and the number of different temperatures (the number depending on T_0 and a). Table 3 shows the frequencies of correct identifications (based on 200 time series) for each pair of a and T_0. When decreasing the value of a, the best performance is obtained by increasing the value of T_0. The pair $a = 0.95$ and $T_0 = 8$ is used.

Table 3: SA tuning experiment: frequencies of correct identifications for different values of T_0 and a.

a	T_0				
	2	4	6	8	10
0.90	0.825	0.845	0.850	0.830	0.870
0.94	0.820	0.850	0.860	0.880	0.880
0.95	0.835	0.880	0.840	0.900	0.855
0.96	0.820	0.835	0.875	0.870	0.845

For the GA algorithms, we compare the frequency of the correct outlier pattern identification for 8 different combinations of population size *pop* and number of generations *gen*, keeping the mutation probability *pmut* and the crossover probability *pcross* constant for all experiments. The values considered for the population size are [10, 20, 30, 40, 50, 70, 100, 200], for the number of generations are [10, 20, 30, 40, 50, 70, 100, 200], while $pcross = 0.001$ and $pmut = 0.6$ (these values were suggested by de Jong [19]).

Table 4 suggests for the parameter *pop* an average value (between 70 and 100). In a second stage, different combinations of *pmut* and *pcross* are considered from $pmut = \{0.1, 0.01, 0.001, 0.0005\}$ and $pcross = \{0.4, 0.6, 0.8, 0.9\}$ whereas the population size and the number of generations are kept constant at 100 and 20, respectively. The results of some combinations of *pmut* and *pcross* are reported in table 4. The results indicate that better results are obtained for average values of crossover probability *pcross* and very low values, but not too much, of mutation probability *pmut*. Based on these results, we use as values: $pmut = 0.001$ and $pcross = 0.6$.

For TA algorithm, we compared a linear sequence of thresholds and a sequence generated by the method given in Winker and Fang [59]. The linear sequences were generated considering different initial thresholds and different rates of decrease. The initial thresholds {6, 8, 10, 14} are used while the values {0.90, 0.96} are considered as rates of decrease. For the method proposed by (Winker and Fang [59]) , we considered 8 combinations of the number of thresholds N_t and number of iterations SA_{iter} choices from N_t ={10, 20, 30, 40 , 50, 70, 100, 200} and SA_{iter} = {10, 20, 30, 40, 50, 70, 100, 200 }. With regard to the linear sequence, the results suggest to use a high threshold and a rate of decrease of the thresholds not very rapid. For the method proposed by (Winker and Fang [59]) the best result is obtained in correspondence to number of thresholds N_t equal to 100. However, there is not a constant improvement as the number of thresholds is incremented and also the differences are not very marked. Observing the thresholds provided by Winker and Fang [59] method, we observed that the initial threshold is large enough (slightly more than 14) and the thresholds decrease very slowly. This particular result depends on the type of problem considered. The value of the objective function for the solutions that belong to a neighborhood can be very different because the removal or insertion of a given anomaly can lead to great changes in the value of the AIC. This means that the distribution $F(\Delta)$ (see algorithm (3)) does not appear to be symmetrical around zero, but is asymmetric towards higher values. From these results it was decided to use a sequence of thresholds $N_t = 100$ obtained by the method of Winker.

Table 4: TA and GA tuning experiment: frequencies of correct identifications for different combinations of parameters.

TA		GA			
(N_t, TA_{iter})	f_{TA}	(pop, gen)	f_{GA}	$(pmut, pcross)$	f_{GA}
(10, 200)	0.860	(10,200)	0.815	(0.01,0.4)	0.850
(20,100)	0.865	(20,100)	0.830	(0.01,0.6)	0.875
(30,70)	0.860	(30,70)	0.850	(0.01,0.8)	0.835
(40,50)	0.880	(40,50)	0.850	(0.01,0.9)	0.825
(50,40)	0.875	(50,40)	0.840	(0.001,0.4)	0.880
(70,30)	0.885	(70,30)	0.885	(0.001,0.8)	0.880
(100,20)	0.885	(100,20)	0.885	(0.001,0.9)	0.850
(200,10)	0.855	(200,10)	0.880	(0.0005,0.6)	0.830

We summarize the parameter values used in the simulations. We imposed

that the objective (fitness) function were evaluated not more than 10,000 times: I_{tot}^{TA}= I_{tot}^{SA}=I_{tot}^{GA}=10,000. For the algorithm SA we chose $T_0 = 8.0$, $T_f = 0.05$, $SA_{iter} = 100$, $a = 0.95$. For the algorithm TA, we set $N_t = 100$ and $TA_{iter} = 100$. For the genetic algorithm we selected $pcross = 0.6$, $pmut = 0.001$, $pop = 100$, $gen = 100$. With $g = 5$, the solution space Ω is of order 2×10^9 when the sample size is $N = 200$, and it is of order 8×10^{10} when the sample size is $N = 400$ whereas the meta-heuristic algorithms reach a satisfying convergence to the optimum evaluating the objective function (fitness) no more than $10,000$ times.

6. Results

In Tables 5, 6 and 7 we report the results of the three meta-heuristic algorithms and the TPP detection procedure. In Tables 5 and 6, the rows labelled P_2 summarise the relative frequency of the correct outlier pattern (both outliers detected and only them), the rows labelled P_1 summarise the relative frequency of only one outlier correctly detected and the rows labelled E summarise the relative frequency of the solutions with wrong identifications (i.e., observations that are identified as outliers while they are not). The complement to one of the sum of these three frequencies is the frequency of the no outlier solution. In Table 7, the rows labelled P_5 summarise the relative frequency of the correct outlier pattern (all five outliers detected and only them), the rows labelled $P_{<5}$ summarise the relative frequency of less than five outliers correctly detected and the rows labelled E summarise the relative frequency of solutions with wrong identifications (i.e., observations that are identified as outliers while they are not). The complement to one of the sum of these three frequencies is again the frequency of the no outlier solution.

Table 5 shows that each of the four algorithms has a high percentage of success when the two outliers are far from each other ($t = 100, 150$). The frequencies of full identifications are nearly equivalent for the four methods. The results are mixed and no method seems uniformly superior to the others. For some models the frequency of correct identification of the TPP method is larger than the corresponding meta-heuristic frequency, while for other models the converse is true.

Table 6 reports simulation results concerning the outliers patch detection where outliers are introduced at time indices $t = 100, 101$. We can see from this table that for almost all models the meta-heuristic algorithms detect the

outlier patch with frequencies higher than those achieved by the TPP. Only for the model (7) the TPP method provides satisfactory results. Moreover, for almost all the models the TPP's frequency of wrong identification E is considerable larger than the corresponding frequencies achieved by meta-heuristic methods. In comparison to the preceding case (two outliers for each other) here the frequency of the no outlier solution is larger, and the largest for the TPP method. Finally, we can see that the frequencies P_2 for models with 200 observations are less than same models with 400 observations. This may be due to the fact that the solution space is larger and the meta-heuristic methods are were easily trapped in some local optimum.

In Table 7 are reported the results for the configuration with 5 outliers where three are consecutive. The configuration is very complex and very difficult to detect if the size of the outlier is not large enough. For this reason outlier sizes are set to 5.0 for the instants $40, 100, 101, 102, 150$. In the table 7 we can see that the relative frequencies of correct configuration P_5 obtained through the meta-heuristic methods are very different and depending on the model. For some models the relative frequency of correct outlier detection are very low.

To reduce the lack of convergence, we reported the simulations allowing for a total number of objective function (fitness) evaluations increased to 100,000 (instead of 10,000), both for the most complex configuration $(40, 100, 101, 102, 150)$ and for the simpler one $(100, 101)$.

Table 8-9 shows the results obtained for the configurations $100, 101$ and $40, 100, 101, 102, 150$ setting the number of evaluations equal to 100,000. We can see an improvement of the results in both cases but the increase of the frequencies of correct identification is very large for the case of 5 outliers. Now the relative frequencies of correct configuration detection obtained through the meta-heuristic methods are high and much larger than those obtained with the TPP method for seven of the eight models considered. For some models the correct pattern is always found (frequency P_5 assumes the value 1). The meta-heuristic algorithms show a better performance than the TPP also in the third configuration outliers (see Table 9).

Tables 8 and 9 evidently illustrate masking and smearing problems encountered by the TPP procedure when additive outliers exist in a patch. It has been noticed that this problems persist despite the size of outliers whereas the meta-heuristic methods improve their performance when the outliers are inserted with a bigger magnitude. Detecting a set of consecutive outliers seems much more difficult and affected by the underlying models.

The good performance of TPP in model 7 depends on the particular parameters of the model generating data. The three algorithms proposed here clearly outperform the TPP method to detect patch of additive outliers.

To understand the poor TPP's results, let us to consider the situation in which the time series follows a VAR(1) and there exists a patch of two additive outliers at time indices $t = T, T+1$, with magnitudes $\omega_t = \omega$ for $t = T, T+1$. Suppose that the model parameters are known, then the expected values of the perturbations at time indices $t = T, T+1$ are given by

$$E(\hat{\omega}_T) = \omega_T + \mathbf{\Gamma i}_0{}^{-1}\mathbf{\Gamma i}_1\omega_{T+1} = (I_s + \mathbf{\Gamma i}_0{}^{-1}\mathbf{\Gamma i}_1)\omega,$$
$$E(\hat{\omega}_{T+1}) = \omega_{T+1} + \mathbf{\Gamma i}_0{}^{-1}\mathbf{\Gamma i}_{-1}\omega_T = (I_s + \mathbf{\Gamma i}_0{}^{-1}\mathbf{\Gamma i}_{-1})\omega.$$

We observe that they are biased. The bias depends on the inverse covariance matrices and it may cause the masking effect. The good performance achieved by the TPP in model 7 may depend on the peculiar parameters of the models. On the contrary in our methods the estimates of the magnitude of outliers are unbiased.

6.1. Real time series data

In this subsection we illustrate the performance of the meta-heuristic procedures by analysing a real example. The data are the well-known gas-furnace series of Box et al. [13]. This bivariate time series consists of an input gas rate in cubic feet per minute and the CO_2 concentration in the outlet gas as a percentage, both measured at 9–second time intervals. There are 296 observations. The TPP method finds additive multivariate outliers at positions 42, 54, 113, 199, 235, 264. All the other algorithms, based on 1,000,000 objective function (fitness) evaluations (T_0= 8.0, T_f= 0.05, SA_{iter} = 10,000, $a = 0.95$, *gen*=30,000, *pop*=30, N_t=100 and TA_{iter} = 10,000, $g = 15$, $c = 8.2$ and $m = 6$) converge to the solution with 4 outliers at positions: 42, 54, 199 and 264. Additional information may be derived by looking also at the sub-optimal solutions. Table 10 displays the outliers patterns corresponding to the best ten solutions found after 1,000,000 objective function evaluations. It suggests that additional time indices may be considered as candidates for the true outlier positions, giving additional insight about the probably outlying observations. It turns out that for this series the TPP method has not

Table 5: Comparison of the algorithm performances: outliers at $t = 100, 150$ based on 10^4 iteration

	$N = 200$				$N = 400$			
	TA	SA	GA	TPP	TA	SA	GA	TPP
Model 1								
P_2	0.90	0.91	0.91	0.94	0.87	0.87	0.92	0.89
P_1	0.05	0.04	0.04	0.02	0.10	0.10	0.05	0.06
E	0.05	0.05	0.05	0.04	0.03	0.03	0.03	0.04
Model 2								
P_2	0.91	0.90	0.91	0.92	0.92	0.92	0.94	0.93
P_1	0.03	0.04	0.03	0.03	0.04	0.04	0.02	0.02
E	0.06	0.06	0.06	0.05	0.04	0.04	0.04	0.05
Model 3								
P_2	0.94	0.94	0.94	0.94	0.91	0.91	0.93	0.93
P_1	0.01	0.01	0.01	0.00	0.02	0.02	0.00	0.00
E	0.04	0.04	0.04	0.06	0.06	0.06	0.06	0.07
Model 4								
P_2	0.94	0.94	0.94	0.90	0.91	0.91	0.91	0.91
P_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E	0.06	0.06	0.06	0.10	0.09	0.09	0.09	0.09
Model 5								
P_2	0.90	0.90	0.90	0.93	0.94	0.94	0.94	0.94
P_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E	0.10	0.10	0.10	0.07	0.06	0.06	0.06	0.06
Model 6								
P_2	0.90	0.90	0.90	0.92	0.90	0.90	0.90	0.94
P_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E	0.10	0.10	0.10	0.08	0.10	0.10	0.10	0.06
Model 7								
P_2	0.95	0.94	0.95	0.94	0.90	0.90	0.90	0.93
P_1	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00
E	0.05	0.05	0.05	0.06	0.01	0.10	0.10	0.07
Model 8								
P_2	0.94	0.94	0.94	0.92	0.96	0.96	0.96	0.96
P_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E	0.06	0.06	0.06	0.08	0.04	0.04	0.04	0.04

P_2= frequency of event 'exactly two outliers found at times 100 and 150'
P_1= frequency of event 'exactly one outlier found at time 100 or at time 150'
E= frequency of solutions with wrong identifications

Table 6: Comparison of the algorithm performances: outliers at $t = 100, 101$ based on 10^4 iteration

	$N = 200$				$N = 400$			
	TA	SA	GA	TPP	TA	SA	GA	TPP
Model 1								
P_2	0.72	0.71	0.72	0.23	0.55	0.56	0.58	0.19
P_1	0.05	0.06	0.05	0.08	0.07	0.06	0.05	0.07
E	0.11	0.11	0.11	0.18	0.13	0.13	0.12	0.14
Model 2								
P_2	0.74	0.74	0.75	0.22	0.68	0.67	0.69	0.21
P_1	0.10	0.10	0.10	0.37	0.15	0.14	0.12	0.40
E	0.13	0.13	0.12	0.25	0.10	0.10	0.10	0.25
Model 3								
P_2	0.83	0.83	0.84	0.34	0.74	0.75	0.78	0.43
P_1	0.03	0.03	0.03	0.06	0.05	0.05	0.04	0.05
E	0.07	0.07	0.06	0.23	0.12	0.11	0.09	0.21
Model 4								
P_2	0.52	0.52	0.54	0.00	0.40	0.41	0.42	0.01
P_1	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01
E	0.21	0.21	0.19	0.30	0.29	0.28	0.27	0.41
Model 5								
P_2	0.89	0.89	0.89	0.55	0.83	0.82	0.83	0.55
P_1	0.00	0.00	0.00	0.08	0.01	0.02	0.01	0.11
E	0.11	0.11	0.11	0.23	0.15	0.15	0.15	0.23
Model 6								
P_2	0.84	0.84	0.84	0.55	0.81	0.81	0.82	0.52
P_1	0.01	0.01	0.01	0.00	0.01	0.01	0.00	0.01
E	0.13	0.13	0.13	0.32	0.17	0.17	0.17	0.35
Model 7								
P_2	0.92	0.92	0.92	0.90	0.88	0.88	0.88	0.87
P_1	0.01	0.01	0.02	0.02	0.00	0.00	0.00	0.04
E	0.07	0.07	0.07	0.08	0.12	0.12	0.12	0.09
Model 8								
P_2	0.91	0.91	0.91	0.10	0.89	0.89	0.91	0.03
P_1	0.00	0.00	0.00	0.15	0.00	0.00	0.00	0.08
E	0.09	0.09	0.09	0.70	0.11	0.11	0.09	0.88

P_2= frequency of event 'exactly two outliers found at times 100 and 150'
P_1= frequency of event 'exactly one outlier found at time 100 or at time 150'
E= frequency of solutions with wrong identifications

Table 7: Comparison of the algorithm performances: outliers at $t = 40, 100, 101, 102, 150$ based on 10^4 iteration

	$N = 200$				$N = 400$			
	TA	SA	GA	TPP	TA	SA	GA	TPP
Model 1								
P_2	0.60	0.58	0.63	0.32	0.32	0.32	0.37	0.24
P_1	0.28	0.30	0.25	0.39	0.48	0.48	0.44	0.46
E	0.12	0.12	0.12	0.29	0.20	0.20	0.19	0.30
Model 2								
P_2	0.75	0.00	0.00	0.29	0.68	0.00	0.00	0.27
P_1	0.13	0.00	0.00	0.45	0.20	0.00	0.00	0.50
E	0.12	0.00	0.00	0.26	0.12	0.00	0.00	0.23
Model 3								
P_2	0.72	0.75	0.76	0.28	0.47	0.47	0.49	0.35
P_1	0.15	0.13	0.12	0.29	0.24	0.24	0.23	0.25
E	0.13	0.12	0.12	0.43	0.29	0.29	0.28	0.40
Model 4								
P_2	0.23	0.22	0.26	0.01	0.20	0.21	0.23	0.00
P_1	0.31	0.32	0.31	0.22	0.21	0.20	0.20	0.19
E	0.46	0.46	0.43	0.77	0.59	0.59	0.57	0.81
Model 5								
P_2	0.84	0.84	0.85	0.55	0.72	0.71	0.72	0.54
P_1	0.03	0.03	0.02	0.13	0.08	0.09	0.08	0.15
E	0.13	0.13	0.13	0.32	0.20	0.20	0.20	0.31
Model 6								
P_2	0.95	0.95	0.95	0.41	0.90	0.90	0.90	0.40
P_1	0.02	0.02	0.02	0.04	0.02	0.02	0.02	0.03
E	0.03	0.03	0.03	0.55	0.08	0.08	0.08	0.57
Model 7								
P_2	1.00	1.00	1.00	0.91	1.00	1.00	1.00	0.90
P_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01
E	0.00	0.00	0.00	0.09	0.00	0.00	0.00	0.09
Model 8								
P_2	0.57	0.58	0.60	0.00	0.66	0.65	0.68	0.01
P_1	0.11	0.10	0.08	0.35	0.03	0.04	0.03	0.28
E	0.32	0.32	0.32	0.65	0.31	0.31	0.29	0.71

P_5= frequency of event 'exactly five outliers found at times 40, 100, 101, 102, 150'
$P_{<5}$= frequency of event 'some of correct outliers are detected'
E= frequency of solutions with wrong identifications

Table 8: Comparison of the algorithm performances: outliers at $t = 100, 101$ based on 10^5 iteration

	$N = 200$				$N = 400$			
	TA	SA	GA	TPP	TA	SA	GA	TPP
Model 1								
P_2	0.73	0.73	0.73	0.23	0.61	0.61	0.61	0.19
P_1	0.05	0.05	0.05	0.08	0.05	0.05	0.05	0.07
E	0.10	0.10	0.10	0.18	0.09	0.09	0.09	0.14
Model 2								
P_2	0.75	0.75	0.75	0.22	0.72	0.72	0.72	0.21
P_1	0.10	0.10	0.10	0.37	0.11	0.11	0.11	0.40
E	0.12	0.12	0.12	0.25	0.10	0.10	0.10	0.25
Model 3								
P_2	0.84	0.84	0.84	0.34	0.83	0.83	0.83	0.43
P_1	0.03	0.03	0.03	0.06	0.03	0.03	0.03	0.05
E	0.06	0.06	0.06	0.23	0.05	0.05	0.05	0.21
Model 4								
P_2	0.60	0.60	0.60	0.00	0.64	0.64	0.64	0.01
P_1	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.01
E	0.13	0.13	0.13	0.30	0.05	0.05	0.05	0.41
Model 5								
P_2	0.90	0.90	0.90	0.55	0.93	0.93	0.93	0.55
P_1	0.00	0.00	0.00	0.08	0.00	0.00	0.00	0.11
E	0.10	0.10	0.10	0.23	0.06	0.06	0.06	0.23
Model 6								
P_2	0.85	0.85	0.85	0.55	0.88	0.88	0.88	0.52
P_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01
E	0.13	0.13	0.13	0.32	0.10	0.10	0.10	0.35
Model 7								
P_2	0.92	0.92	0.92	0.90	0.88	0.88	0.88	0.87
P_1	0.01	0.01	0.02	0.02	0.00	0.00	0.00	0.04
E	0.07	0.07	0.07	0.08	0.12	0.12	0.12	0.09
Model 8								
P_2	0.93	0.93	0.93	0.10	0.96	0.96	0.96	0.03
P_1	0.00	0.00	0.00	0.15	0.00	0.00	0.00	0.08
E	0.07	0.07	0.07	0.70	0.04	0.04	0.04	0.88

P_2= frequency of event 'exactly two outliers found at times 100 and 150'
P_1= frequency of event 'exactly one outlier found at time 100 or at time 150'
E= frequency of solutions with wrong identifications

Table 9: Comparison of the algorithm performances: outliers at $t = 40, 100, 101, 102, 150$ based on 10^5 iteration

	$N = 200$				$N = 400$			
	TA	SA	GA	TPP	TA	SA	GA	TPP
Model 1								
P_2	0.89	0.90	0.95	0.32	0.80	0.80	0.92	0.24
P_1	0.06	0.05	0.00	0.39	0.09	0.09	0.00	0.46
E	0.05	0.05	0.05	0.29	0.11	0.11	0.08	0.30
Model 2								
P_2	0.86	0.86	0.87	0.29	0.84	0.85	0.87	0.27
P_1	0.10	0.10	0.09	0.45	0.12	0.11	0.09	0.50
E	0.09	0.04	0.04	0.26	0.04	0.04	0.04	0.23
Model 3								
P_2	0.95	0.97	0.99	0.28	0.86	0.90	0.94	0.35
P_1	0.02	0.00	0.00	0.29	0.04	0.02	0.00	0.25
E	0.03	0.03	0.01	0.43	0.10	0.08	0.06	0.40
Model 4								
P_2	0.74	0.73	0.75	0.01	0.82	0.82	0.84	0.00
P_1	0.14	0.15	0.13	0.22	0.05	0.05	0.05	0.19
E	0.12	0.12	0.12	0.77	0.13	0.13	0.11	0.81
Model 5								
P_2	0.97	0.97	1.00	0.55	0.96	0.96	1.00	0.54
P_1	0.00	0.00	0.00	0.13	0.00	0.00	0.00	0.15
E	0.03	0.03	0.00	0.32	0.04	0.04	0.00	0.31
Model 6								
P_2	1.00	1.00	1.00	0.41	0.98	0.98	1.00	0.40
P_1	0.00	0.00	0.00	0.04	0.02	0.02	0.00	0.03
E	0.00	0.00	0.00	0.55	0.00	0.00	0.00	0.57
Model 7								
P_2	1.00	1.00	1.00	0.91	1.00	1.00	1.00	0.90
P_1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01
E	0.00	0.00	0.00	0.09	0.00	0.00	0.00	0.09
Model 8								
P_2	0.93	0.92	0.95	0.00	0.93	0.93	0.94	0.01
P_1	0.02	0.03	0.00	0.35	0.02	0.02	0.01	0.28
E	0.05	0.05	0.05	0.65	0.05	0.05	0.05	0.71

P_5= frequency of event 'exactly five outliers found at times 40, 100, 101, 102, 150'
$P_{<5}$= frequency of event 'some of correct outliers are detected'
E= frequency of solutions with wrong identifications

Table 10: Meta-heuristic algorithm solutions for the gas–furnace series

Solution	$f(x)$	Locations
S_1	-53.82	42 54 199 264
S_2	-53.29	43 54 199 264
S_3	-51.42	42 54 199 235 264
S_4	-50.89	43 54 199 235 264
S_5	-50.10	42 54 113 199 264
S_6	-49.57	43 54 113 199 264
S_7	-48.55	42 55 199 264
S_8	-48.02	43 55 199 264
S_9	-47.78	42 54 198 264
S_{10}	-47.70	42 54 113 199 235 264

Table 11: Statistics of empirical distributions for different values of I (based on 100 runs)

I	TA				SA			
	$\hat{\mu}$	$\hat{\sigma}$	*best*	$q_{0.05}$	$\hat{\mu}$	$\hat{\sigma}$	*best*	$q_{0.05}$
100	-19.50	14.77	-44.59	-39.42	-14.81	13.98	-40.32	-36.24
500	-42.54	6.43	-53.82	-53.82	-33.21	7.65	-45.54	-44.58
1,000	-48.68	4.71	-53.82	-53.82	-39.10	6.60	-53.82	-48.69
5,000	-52.83	1.87	-53.82	-53.82	-52.79	1.92	-53.82	-53.82
10,000	-53.16	1.17	-53.82	-53.82	-53.16	1.15	-53.82	-53.82
I	GA_1				GA_2			
	$\hat{\mu}$	$\hat{\sigma}$	*best*	$q_{0.05}$	$\hat{\mu}$	$\hat{\sigma}$	*best*	$q_{0.05}$
100	-31.69	6.91	-44.92	-44.02				
500	-44.59	6.86	-53.82	-53.82				
1,000	-49.19	4.53	-53.82	-53.82				
5,000	-51.71	2.92	-53.82	-53.82				
10,000	-53.01	1.17	-53.82	-53.82				

given the best solution, but the ten-th one in order of decreasing objective function.

Let I denote the number of evaluations of the objective function. In order to compare the convergence of the algorithms we calculate, for different values of I (100, 500, 1,000, 5,000, 10,000), the empirical distribution, based on 100 restarts, of the best obtained objective function. Table 11 reports some relevant statistics (mean, standard deviation, best value and 5-th percentile) about the empirical distributions along the guidelines suggested by Gilli and Winker [30]. As I increases, the distributions shift to the left ($\hat{\mu}$ decreases) and become less dispersed ($\hat{\sigma}$ decreases). The GA show a better initial performance due to the favourable way the initial population is chosen, but the SA and the TA have a faster convergence speed.

At the last iteration ($I = 10,000$), the best value ($f(x) = -53.82$) is found in 59 out of 100 runs for the SA, in 58 out of 100 runs for the TA, in 46 out of 100 for the GA.

7. Conclusions

In this paper three meta-heuristic methods for detecting additive outliers in multivariate time series are proposed. Meta-heuristic algorithms, unlike other methods in literature, do not identify and remove outliers one at a time, but examine several proposed outlier patterns, where all observations are simultaneously considered. This feature seems to be effective in handling masking (meaning that one outlier hides others) and swamping (when outliers make other clean observations to appear outliers as well) effects caused by multiple outliers. Furthermore, our methods do not require the specification of an adequate multivariate model, which is usually a difficult task, especially when the data are contaminated by outliers. The procedures are illustrated by analysing artificial and real data sets. The results obtained from the simulation experiments seem to support the idea that the meta-heuristic algorithms constitute a valid approach to detect the time points where potential outliers in vector time series are located. In our experiment the meta-heuristic methods provide better results than the TPP method to identify outlier patch, while the results are similar for the case of well separated outliers. The examination of the "gas-furnace" data of Box and Jenkins yields satisfactory results. Comparing the results obtained by the detection procedure of Tsay et al. [53] with the best solution provided by meta-heuristic algorithms, we observe that they have in common four out of

six outliers locations. Such small discrepancy is caused by the difference between the two identification procedures. The efficiency of the meta-heuristic methods proposed in this study, depends crucially on the choice of appropriate values for some control parameters. The simulation and the theoretical study used for determining the value of parameter c, allows us to control for the type I error α. For any given value of α there is a corresponding value for c that does not depend on the underlying model. It only depends on the number of components (s) and the length of the time series. In the case of real data, given a value of α, the corresponding value of c, as reported in Table 2, can be used.

The presence of partial outliers, i.e., anomalies that affect only some components of the multivariate series, may be an issue to be considered for future developments. Moreover, an interesting further problem is the outlier identificability, that is, studying how large should the outliers size to ensure that the correct outlier configuration has the maximum fitness.

[1] Althöfer, I. and Koschnick, K. [1991], 'On the convergence of threshold accepting', *Applied Mathematics and Optimization* **24**, 183–195.

[2] Angelis, L., Bora-Senta, E. and Moyssiadis, C. [2001], 'Optimal exact experimental designs with correlated errors through a simulated annealing algorithm', *Computational Statistics and Data Analysis* **37**, 275–296.

[3] Baragona, R. and Battaglia, F. [1989], Identificazione e stima di dati anomali in serie temporali per mezzo di interpolatori lineari, Technical Report 19, Department of Statistical Sciences, University of Roma La Sapienza, Italy.

[4] Baragona, R. and Battaglia, F. [2007], 'Outliers detection in multivariate time series by independent component analysis', *Neural Computation* **19**, 1962–1984.

[5] Baragona, R., Battaglia, F. and Calzini, C. [2001], 'Genetic algorithms for the identification of additive and innovational outliers in time series', *Computational Statistics and Data Analysis* **37**, 1–12.

[6] Baragona, R., Battaglia, F. and Poli, I. [2011], *Evolutionary statistical procedures*, Springer-Verlag, Berlin.

[7] Barbieri, M. [1991], 'Outliers in serie temporali multivariate', *Quaderni di Statistica e Econometria* **13**, 1–11.

[8] Barnett, G., Kohn, R. and Sheather, S. [1997], 'Robust bayesian estimation of autoregressive-moving average models', *Journal of Time Series Analysis* **18**, 11–28.

[9] Battaglia, F. [1984], 'Inverse covariances of a multivariate time series', *Metron* **42**, 117–129.

[10] Battaglia, F. and Baragona, R. [1992], 'Linear interpolators and the outlier problem in time series', *Metron* **50**, 79–97.

[11] Bhansali, R. J. [1980], 'Autoregressive and window estimates of the inverse correlation function', *Biometrika* **67**, 551–566.

[12] Bhansali, R. J. and Downham, D. Y. [1977], 'Some properties of the order of an autoregressive model selected by a generalization of Akaike's FPE criterion', *Biometrika* **64**, 547–551.

[13] Box, G. E. P., Jenkins, G. M. and Reinsel, G. C. [1994], *Time Series Analysis. Forecasting and Control (3rd edition)*, Prentice Hall, Englewood Cliffs, New Jersey.

[14] Bruce, A. G. and Martin, D. [1989], 'Leave-k-out diagnostics for time series (with discussion)', *Journal of the Royal Statistical Society: Series B* **51**, 363–424.

[15] Chang, I. and Tiao, G. C. [1983], Estimation of time series parameters in the presence of outliers, Technical Report 8, University of Chicago, Statistics Research Center.

[16] Chang, I., Tiao, G. C. and Chen, C. [1988], 'Estimation of time series parameters in the presence of outliers', *Technometrics* **30**, 193–204.

[17] Chen, C. and Liu, L. [1993], 'Joint estimation of model parameters and outlier effects in time series', *Journal of the American Statistical Association* **88**, 284–297.

[18] da Graça Lobo, F. [2000], *The parameter-less genetic algorithm: rational and automated parameter selection for simplified genetic algorithm operation*, PhD thesis, Universidade Nova de Lisboa.

[19] de Jong, K. A. [1975], An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Phd thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI.

[20] Depril, D., Van Mechelen, I. and Mirkin, B. [2008], 'Algorithms for additive clustering of rectangular data tables', *Computational Statistics and Data Analysis* **52**, 4923–4938.

[21] Duczmal, L. and Assunção, R. [2004], 'A simulated annealing strategy for the detection of arbitrarily shaped spatial clusters', *Computational Statistics and Data Analysis* **45**, 269–286.

[22] Dueck, G. and Scheuer, T. [1990], 'Threshold accepting: A general purpose algorithm appearing superior to simulated annealing', *Journal of Computational Physics* **90**, 161–175.

[23] Eglese, R. [1990], 'Simulated annealing: a tool for operational research', *European Journal of Operational Research* **46**, 271–281.

[24] Eiben, A., Hinterding, R. and Michalewicz, Z. [1999], 'Parameter control in evolutionary algorithms', *IEEE Trans. on Evolutionary Computation* **3**, 124–141.

[25] Fang, K., Lin, D. K. J., Winker, P. and Zhang, Y. [2000], 'Uniform design:theory and application', *Technometrics* **42**, 237–248.

[26] Fox, A. J. [1972], 'Outliers in time series', *Journal of the Royal Statistical Society:Series B* **34**, 350–63.

[27] Galeano, P., Pena, D. and Tsay, R. S. [2006], 'Outlier detection in multivariate time series by projection pursuit', *Journal of the American Statistical Association* **101**, 654–669.

[28] Gao, Y. [2003], Population size and sampling complexity in genetic algorithms, *in* 'GECCO 2003 Workshop on Learning, Adaptation, and Approximation in Evolutionary Computation', Springer, Berlin, pp. 178–181.

[29] Gilli, M. and Winker, P. [2004], 'Applications of optimization heuristics to estimation and modelling problems', *Computational Statistics and Data Analysis* **47**, 211–223.

[30] Gilli, M. and Winker, P. [2009], Heuristic optimization methods in econometrics, *in* E. Kontoghiorghes and D. Belsley, eds, 'Handbook on computational econometrics', Wiley, Chichester, chapter 3, pp. 81–119.

[31] Goldberg, D. [1989], *Genetic algorithms in search optimization and machine learning*, Addison-Wesley, Reading, MA.

[32] Gómez, V., Maravall, A. and Peña, D. [1993], Computing missing values in time series, Working Paper 93-27 Statistics and Econometric Series 21, Departamento de Estadistica y Econometria Universidad Carlos III de Madrid.

[33] Grefenstette, J. [1986], 'Optimization of control parameters for genetic algorithms', *IEEE Systems, Man, and Cybernetics Society* **16**, 122–128.

[34] Holland, J. [1975], *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and AI.*, The University of Michigan, Ann Arbor, MI.

[35] Justel, A., Peña, T. and Tsay, R. [2001], 'Detection of outlier patches in autoregressive time series.', *Statistica Sinica* **11**, 651–673.

[36] Khattree, R. and Naik, D. [1987], 'Detection of outliers in bivariate time series data', *Communications in Statistics - Theory and Methods* **16(12)**, 3701–3714.

[37] Kirkpatrick, S., Gelat, C. D. and Vecchi, M. P. [1983], 'Optimization by simulated annealing', *Science* **220**, 671–680.

[38] Lin, D., Sharpe, C. and Winker, P. [2010], 'Optimized u-type designs on flexible regions', *Computational Statistics and Data Analysis* **54**, 1505–1515.

[39] Ljung, G. M. [1989], 'A note on the estimation of missing values in time series', *Communications in Statistics - Simulation and Computation* **18(2)**, 459–465.

[40] Ljung, G. M. [1993], 'On outlier detection in time series', *Journal of the Royal Statistical Society:Series B* **55**, 559–567.

[41] Lütkepohl, H. [1993], *Introduction to multiple time series analysis (2nd edition)*, Spinger-Verlag, Berlin.

[42] Lyra, M., Paha, J., Paterlini, S. and Winker, P. [2010], 'Optimization heuristics for determining internal rating grading scales', *Computational Statistics and Data Analysis* **54**, 2693–2706.

[43] Maringer, D. and Winker, P. [2009], 'The convergence of estimators based on heuristics: theory and application to a garch model', *Computational Statistics* **24**, 533–550.

[44] McCulloch, R. E. and Tsay, R. S. [1994], 'Bayesian analysis of autoregressive time series via the Gibbs sampler', *Journal of Time Series Analysis* **15**, 235–250.

[45] Metropolis, N., Rosenbluth, A. W., Teller, A. H. and Teller, E. [1953], 'Equation of state calculation by fast computing machines', *Journal of Chemical Physics* **21**, 1087–1091.

[46] Park, M. and Kim, Y. [1998], 'A systematic procedure for setting parameters in simulated annealing algorithm', *Computers and Operations Research* **25**, 207–217.

[47] Peña, D. [1990], 'Influential observations in time series', *Journal of Business and Economic Statistics* **8**, 235–241.

[48] Rayward-Smith, V., Osman, I., Reeves, C. and Smith, G. [1996], *Modern Heuristic Search Methods*, Wiley, Chichester, New York.

[49] Rudolph, G. [1994], 'Convergence analysis of canonical genetic algorithms', *IEEE Transactions on Neural Networks* **5**, 96–101.

[50] Schaffer, J., Caruana, R., Eshelman, L. and Das, R. [1989], A study of control parameters affecting online performance of genetic algorithms for function optimization, *in* 'Proceedings of the 3rd International Conference on Genetic Algorithm', Morgan Kaufmann, San Mateo, CA, pp. 51–60.

[51] Shaman, P. [1976], 'Approximations for stationary covariance matrices and their inverses with applications to arima models', *Annals of Statistics* **4**, 292–301.

[52] South, M., Wetherill, G. and Tham, M. [1993], 'Hitch-hiker's guide to genetic algorithms', *Journal of Applied Statistics* **20**, 153–175.

[53] Tsay, R., Peña, D. and Pankratz, A. [2000], 'Outliers in multivariate time series', *Biometrika* **87**, 789–804.

[54] Tsay, R. S. [1986], 'Time series model specification in the presence of outliers', *Journal of the American Statistical Association* **81**, 132–141.

[55] Tsay, R. S. [1988], 'Outliers, level shifts, and variance changes in time series', *Journal of Forecasting* **7**, 1–20.

[56] Vera, J. and Díaz-García, J. [2008], 'A global simulated annealing heuristic for the three-parameter lognormal maximum likelihood estimation', *Computational Statistics and Data Analysis* **52**, 5055–5065.

[57] Winker, P. [2000], 'Optimized multivariate lag structure selection', *Computational Economics* **16**, 87–103.

[58] Winker, P. [2001], *Optimization Heuristics in Econometrics and Statistics: A simple approach for complex problems with Threshold Accepting*, Wiley, New York.

[59] Winker, P. and Fang, K. [1997], 'Application of threshold accepting to the evaluation of the discrepancy of a set of points', *SIAM Journal on Numerical Analysis* **34**, 2028–2042.

[60] Winker, P., Lyra, M. and Sharpe, C. [2011], 'Least median of squares estimation by optimization heuristics with an application to the capm and a multi-factor model', *Computational Management Science* **8**, 103–123.