Int. J. Appl. Math. Comput. Sci., 2014, Vol. 24, No. 1, 133–149
DOI: 10.2478/amcs-2014-0011

amcs

AN ALGORITHM FOR REDUCING THE DIMENSION AND SIZE OF A SAMPLE FOR DATA EXPLORATION PROCEDURES

PIOTR KULCZYCKI [*,**], SZYMON ŁUKASIK [*,**]

[*] Systems Research Institute
Polish Academy of Sciences, ul. Newelska 6, 01-447 Warsaw, Poland
e-mail: {kulczycki,slukasik}@ibspan.waw.pl

[**] Department of Automatic Control and Information Technology
Cracow University of Technology, ul. Warszawska 24, 31-155 Cracow, Poland

The paper deals with the issue of reducing the dimension and size of a data set (random sample) for exploratory data analysis procedures. The concept of the algorithm investigated here is based on linear transformation to a space of a smaller dimension, while retaining as much as possible the same distances between particular elements. Elements of the transformation matrix are computed using the metaheuristics of parallel fast simulated annealing. Moreover, elimination of or a decrease in importance is performed on those data set elements which have undergone a significant change in location in relation to the others. The presented method can have universal application in a wide range of data exploration problems, offering flexible customization, possibility of use in a dynamic data environment, and comparable or better performance with regards to the principal component analysis. Its positive features were verified in detail for the domain's fundamental tasks of clustering, classification and detection of atypical elements (outliers).

Keywords: dimension reduction, sample size reduction, linear transformation, simulated annealing, data mining.

1. Introduction

Contemporary data analysis avails of a broad and varied methodology, based on both traditional and modern, often specialized statistical procedures, currently ever more supported by the significant possibilities of computational intelligence. Apart from the classical methods—fuzzy logic and neural networks—metaheuristics such as genetic algorithms, simulated annealing, particle swarm optimization, and ants algorithms (Gendreau and Potvin, 2010) are also applied here more widely. Proper combination and exploitation of the advantages of these techniques allows in practice an effective solution of fundamental problems in knowledge engineering, particularly those connected with exploratory data analysis. More and more frequently the process of knowledge acquisition is realized using multidimensional data sets of a large size. This stems from the dynamic growth in the amount of information collected in database systems requiring permanent processing.

The extraction of knowledge from extensive data sets is a highly complex task. Here difficulties are mainly related to limits in the efficiency of computer systems for large size samples and problems exclusively connected with the analysis of multidimensional data. The latter arise mostly from a number of phenomena occurring in data sets of this type, known in the literature as "the curse of multidimensionality". Above all, this includes the exponential growth in the sample size necessary to achieve appropriate effectiveness of data analysis methods with an increasing dimension (the empty space phenomenon), as well as the vanishing difference between near and far points (norm concentration) using standard Minkowski distances (François *et al*., 2007). As previously mentioned, the data set size can be reduced mainly to speed up calculations or make them possible at all (Czarnowski and Jędrzejowicz, 2011). In the classical approach, this is realized mostly with sampling methods or advanced data condensation techniques.

Useful algorithms have also been worked out allowing the problem to be simplified by decreasing its dimensionality. Therefore, let X denote a data matrix of dimension $m \times n$:

$$X = [x_1|x_2|\ldots|x_m]^T, \tag{1}$$

with particular m rows representing realizations of an n-dimensional random variable[1]. The aim of reducing a dimension is to transform the data matrix in order to obtain its new representation of the dimension $m \times N$, where N is considerably (from the point of view of the conditioning of a problem in question) smaller than n. This reduction can be achieved in two ways, either by choosing N most significant coordinates/features (feature selection) or through the construction of a reduced data set, based on initial features (feature extraction) (Inza *et al*., 2000; Xu and Wunsch, 2009). The latter can be treated as more general—the selection is a particularly simple case of extraction. Noteworthy among extraction procedures are linear methods, where the resulting data set Y is obtained through the linear transformation of the initial set (1), therefore using the formula

$$Y = XA, \tag{2}$$

where A is a matrix of dimension $n \times N$, as well as nonlinear methods for which the transformation can be described by a nonlinear function $g : \mathbb{R}^n \to \mathbb{R}^N$. This group also contains the methods for which such a functional dependence, expressed explicitly, does not exist.

Studies on the efficiency of extraction procedures carried out in the literature on the subject show that nonlinear methods, despite having a more general mathematical apparatus and higher efficiency in the case of artificially generated specific sets of data, frequently achieve significantly worse results for real samples (Maaten, 2009). The goal of this paper is to develop a universal method of reducing the dimension and size of a sample designed for use in data exploration procedures. The reduction of the dimension will be implemented using a linear transformation on the condition that it affects as little as possible the mutual positions of the original and resulting samples' elements.

The focus of this contribution is to offer an alternative for the state-of-the-art linear method of principal component analysis (Sumi *et al*., 2012), with unsupervised character and the possibility to customize the algorithm's features. To this aim a novel version of the heuristic method of parallel fast simulated annealing will be researched. Moreover, those elements of a random sample which significantly change their position following transformation will be eliminated or assigned less weight for the purposes of further analysis. This concept achieves an improvement in the quality of knowledge discovery and, possibly, a reduction in the sample size. The effectiveness of the presented method will be verified for fundamental procedures in exploratory data analysis: clustering, classification and detection of atypical elements (outliers). Many aspects considered in this paper were initially proposed by Łukasik and Kulczycki (2011) as well as Kulczycki and Łukasik (2014) in their basic form.

[1]Particular coordinates of a random variable clearly constitute one-dimensional random variables, and if the probabilistic aspects are not the subject of research then in data analysis these variables are given the terms "feature" or "attribute".

2. Preliminaries

2.1. Reduction in the dimension and sample size. The dimension can be reduced in many ways. Correctly sorting the procedures applied here requires, therefore, a wide range of criteria to be taken into account. Firstly, the aforementioned systematic for linear and nonlinear methods is associated with the character of the dependence between the initial and reduced data sets. Most important of these, being even a reference linear procedure for dimension reduction, is Principal Component Analysis (PCA). Among linear methods most often mentioned is MultiDimensional Scaling (MDS). Reduction procedures are often considered with respect to the ease of description of the mapping between the initial and reduced data sets. This can be defined *explicitly* (which allows generalizing the reduction procedure on points not belonging to the initial data set), as well as given only *implicitly*, i.e., through a reduced representation of elements of the initial data set.

The type of method chosen has particular significance in the cases of data analysis tasks, where a continuous influx of new information is present. In this form of the problem, reduction methods belonging to the first of the above groups are preferred. The third division of transformation procedures is related to their level of relationship with the data analysis algorithms used in the next step.

It is worth noting here universal techniques which, through an analogy to machine learning methods, can be termed as unsupervised. These work autonomously, without using results of exploration procedures (Bartenhagen *et al*., 2010; Aswani Kumar and Srinivas, 2006). The second category concerns algorithms dedicated to particular techniques in data analysis, in particular considering class labels. Here statistical methods as well as heuristic procedures of optimization, e.g., evolutionary algorithms or simulated annealing, are often used (Tian *et al*., 2010). For an overview of the existing dimension reduction techniques one could refer to Cunningham (2007) or Aswani Kumar (2009).

A reduction in the data set size can be realized with a wide range of sampling or grouping methods. The former most often use random procedures or stratified sampling (Han and Kamber, 2006). The latter apply either classical clustering techniques or special procedures for data condensation problems. There also exists a significant number of methods for size reduction which take into account additional knowledge, for example, concerning

whether elements belong to particular classes (Kulczycki and Kowalski, 2011; Wilson and Martinez, 2000). Moreover, methods dedicated to particular analytical techniques, for example, kernel estimators (Kulczycki, 2005; Wand and Jones, 1995), have been developed (see, e.g., Deng *et al*., 2008).

The method presented in this paper is based on a concept of dimension reduction which is linear *explicitly* defined and of universal purpose. Its closest equivalents can be the principal component analysis method (due to its linear and unsupervised nature), feature selection using evolutionary algorithms (Saxena *et al*., 2010) and the projection method with preserved distances (Sammon, 1969; Strickert *et al*., 2005; Vanstrum and Starks, 1981), with respect to the similar quality performance criterion. A natural priority for the dimension reduction procedure is maintaining distances between particular data set elements—a wide range of methods treat this as a quality index.

Typical for this group of algorithms is the classic multidimensional scaling, also known as principal coordinates analysis. It is a linear method which creates the analytical form of the transformation matrix A, minimizing the performance index

$$S(A) = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \left(d_{ij}^2 - \delta_{ij}(A)^2 \right), \tag{3}$$

where d_{ij} denotes the distance between the elements x_i and x_j of the initial data set, while $\delta_{ij}(A)$ are respective distances in the reduced data set. A different strategy is required when searching for a solution with different structural characteristics or the performance index, or a nonlinear relation between the initial and reduced data sets. This type of procedure is termed multidimensional scaling, as mentioned before. A model example of this is nonlinear Sammon mapping, which (due to the application of a simple gradient algorithm) allows us to find a reduced representation of the investigated data set, ensuring the minimization of the so-called Sammon stress:

$$S_S(A) = \frac{1}{\sum_{i=1}^{m-1} \sum_{j=i+1}^{m} d_{ij}} \times \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \frac{\left(d_{ij} - \delta_{ij}(A)\right)^2}{d_{ij}}. \tag{4}$$

Such a criterion enables a more homogeneous treatment of small and large distances (Cox and Cox, 2000), while the value $S_S(A)$ is further normalized to the interval $[0, 1]$. An alternative index, also considered in the context of MDS, is the so-called raw stress, defined by

$$S_R(A) = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \left(d_{ij} - \delta_{ij}(A)\right)^2. \tag{5}$$

Multidimensional scaling methods are mostly nonlinear procedures. However, the task was undertaken to formulate the problem of minimization of the indexes (4) and (5) with the assumed linear form of transformation.

The first example of this technique is the algorithm for finding a linear projection described by (Vanstrum and Starks, 1981). They apply an iterative method of steepest descent, which gives in consequence better results than PCA in the sense of the index (4). A similar procedure was investigated for the function (5), with the additional possibility to successively supplement a data set (Strickert *et al*., 2005). In both the cases the approach applied did not account for the multimodality of the stress function. To avoid becoming trapped in a local minimum, one can use the appropriate heuristic optimization strategy. In particular, for minimization of the index (4), Saxena *et al*. (2010) use the evolutionary algorithm. The solution for this investigation is, however, only to choose the reduced features set.

A more effective approach seems to be the concept of their extraction—being more general, it will be the subject of investigation in this paper. In the construction of the algorithm presented here, an auxiliary role is played by an unsupervised technique of feature selection using to this aim an appropriate measure of the similarity index of maximal compression of information (Pal and Mitra, 2004). It is based on the concept of dividing features into clusters, with the similarity criterion of features defined by the aforementioned index. This division is based on the algorithm of k-nearest neighbors, where it is recommended that $k \cong n - N$. The number of clusters achieved then approaches N, although it is not strictly fixed, but in a more natural manner adapted to a real data structure.

Another aspect of the procedure presented here is a reduction in the size of the sample (1). Conceptually, the closest technique is the condensation method (Mitra *et al*., 2002). It is unsupervised, and to establish the importance of elements it takes into account their respective distances. In this case the algorithm of k-nearest neighbors is also applied, where the similarity measure between sample elements is the Euclidean distance. Within this algorithm, in the data set there are iteratively found prototype points, or points for which the distance r to the k-th nearest neighbor is the smallest. With every iteration, elements closer than $2r$ from the nearest prototype point are eliminated.

2.2. Simulated annealing algorithm. Simulated Annealing (SA) is a heuristic optimization algorithm, based on the iterative technique of local search with an appropriate criterion for accepting solutions. This allows establishing a valid solution for every iteration, mostly using the performance index value for the previous and current iteration, and a variable parameter called the

annealing temperature, which decreases in time. Within the scheme of the algorithm it is possible to accept a valid solution a worse than the previous one, which reduces the danger of the algorithm getting stuck at local minimums. It is assumed, however, that the probability of accepting such worse solution should decrease over time. All of the above traits contain the so-called Metropolis rule, which is most often applied as an acceptance criterion in simulated annealing algorithms.

Let therefore $Z \subset \mathbb{R}^t$ denote the set of admissible solutions to a certain optimization problem, while the function $h : Z \to \mathbb{R}$ is its performance index, hereinafter referred to as cost. Furthermore, let $k = 0, 1, \ldots$ stand for the number of iteration, whereas $T(k) \in \mathbb{R}$, $z(k) \in Z$, $c(k) = h(z(k))$, $z_0(k) \in Z$, $c_0(k) = h(z_0(k))$ denote respectively the temperature and the solution valid for the iteration k and its cost, and also the best solution found to date and its cost. Under the above assumptions the basic variant of the SA algorithm can be described as Algorithm 1. The procedure for the Metropolis rule is realized by Algorithm 2.

Algorithm 1. Simulated annealing.

Generate(T(1),z(0))
$c(0) \leftarrow$ Evaluate_quality($z(0)$)
$z_0(0) \leftarrow z(0)$
$c_0(0) \leftarrow c(0)$
$k \leftarrow 1$
repeat
  $z(k) \leftarrow$ Generate_neighbor($z(k-1)$)
  $c(k) \leftarrow$ Evaluate_quality($z(k)$)
  $\Delta c \leftarrow c(k) - c(k-1)$
  $z(k) =$Metropolis_rule($\Delta c, z(k), z(k-1), T(k)$)
  if $c(k) < c_0(k-1)$ **then**
    $z_0(k) \leftarrow z(k)$
    $c_0(k) \leftarrow c(k)$
  else
    $z_0(k) \leftarrow z_0(k-1)$
    $c_0(k) \leftarrow c_0(k-1)$
  end if
  Calculate($T(k+1)$)
  stop_condition = Check_stop_condition()
  $k \leftarrow k+1$
until stop_condition = **false**
return $k_{\text{stop}} = k-1$, $z_0(k_{\text{stop}})$, $c_0(k_{\text{stop}})$

The SA algorithm requires in the general case the assumption of the appropriate initial temperature value, a formula of its changes associated with an accepted method of generating a neighboring solution, and a condition for ending the procedure. However, in particular applications one should also define other functional elements, such as the method of generating the initial solution and the form of the quality index. The first group of tasks will now be discussed, while the second—as specific for the application of the SA algorithm investigated here—will be the subject of a detailed analysis in Section 3. Numerous fundamental and applicational works have resulted in the creation of many variants of the algorithm described here. Their main difference is the scheme for temperature changes and the method for obtaining a neighboring solution.

Algorithm 2. Metropolis rule.

if $\Delta c < 0$ **then**
  return $z(k)$
else
  if Random_number_in_(0,1) $< \exp(-\Delta c/T(k))$ **then**
    return $z(k)$
  else
    return $z(k-1)$
  end if
end if

A standard approach is the classical simulated annealing algorithm, also known as the Boltzmann Annealing (BA) algorithm. This assumes an iterative change in temperature according to a logarithmic schedule and generation of a subsequent solution by adding to the current one the value of step $\Delta z \in \mathbb{R}^t$, which is a realization of a t-dimensional pseudorandom vector with the normal distribution. The BA algorithm, although effective in the general case, has a large probability of acceptance of worse solutions, even in the final phase of the search process. This allows effective escape from local minima of a cost function and guarantees asymptotic convergence to a global one (Geman and Geman, 1984), while also causing that the procedure represents, in some sense, random search of the space of admissible solutions. For the SA algorithm to be more deterministic in character, and at the same time to keep convergence to the optimal solution, the following scheme for the temperature change can be applied:

$$T(k+1) = \frac{T(1)}{k+1}, \tag{6}$$

together with the generation of neighboring solution using the Cauchy distribution,

$$g(\Delta z) = \frac{T(k)}{(\Delta z^2 + T(k)^2)^{(t+1)/2}}. \tag{7}$$

The procedure defined by the above elements is called Fast Simulated Annealing (FSA) (Szu and Hartley, 1987). It will be a base—in the framework of this paper—for the dimension reduction algorithm. The problem of practical implementation of FSA is effective generation of random numbers with a

multidimensional Cauchy distribution. The simplest solution is the application, for each dimension of the vector, of a one-dimensional number generator with the same distribution. This strategy was used in the Very Fast Simulated Annealing (VFSA) algorithm, expanded later within the framework of a complex procedure of adaptive simulated annealing (Ingber, 1996). Such a concept has, however, a fundamental flaw: the step vectors generated here concentrate near the axes of the coordinate system.

An alternative could be to use a multidimensional generator based on the transformation of the Cartesian coordinate system to a spherical one. It is suggested here that the step vector $\Delta z = [\Delta z_1, \Delta z_2, \ldots, \Delta z_t]$ should be obtained by generating first the radius r of the hypersphere, using the method of inverting the Cauchy distribution function described with the spherical coordinates, and then selecting the appropriate point on the t-dimensional hypersphere. The second phase is realized by randomly generating the vector $u = [u_1, u_2, u_t]^T$ with coordinates originating from the one-dimensional normal distribution $u_i \sim N(0, 1)$, and then the step vector Δz is obtained by

$$\Delta z_i = r\frac{u_i}{|u|}, \quad i = 1, 2, \ldots, t. \tag{8}$$

The presented procedure ensures a symmetric and multidirectional generation scheme, with heavy tails of distribution, which in consequence causes effective exploration of a solution space (Nam *et al.*, 2004). Taking the above into account, it has been applied in the algorithm investigated in this paper. Establishing an initial temperature is vital for correct functioning of the simulated annealing algorithm. It implies the probability of acceptance of a worse solution at subsequent stages of the search in the solution space. The subject literature tends to suggest choosing the initial temperature so that the probability of acceptance of a worse solution at the first iteration, denoted hereinafter as $P(1)$, is relatively large. These recommendations are not absolute, however, and different proposals can be found in the literature, for example, close to 1.0 (Aarts *et al.*, 1997), around 0.8 (Ben-Ameur, 2004) or even only 0.5 (Kuo, 2010).

Often in practical applications of SA algorithms the temperature value is fixed during numerical experiments. An alternative is to choose a temperature according to a calculational criterion which has the goal of obtaining the $T(1)$ value on the basis of a set of pilot iterations, consisting of generating the neighbor solution $z(1)$ so that the assumed $P(1)$ value is ensured. For this purpose one can, analyzing the mean difference in cost between the solutions $z(1)$ and $z(0)$, denoted by $\overline{\Delta c}$, calculate the temperature $T(1)$ value by substituting $\overline{\Delta c}$ to the right-side of the inequality in the Metropolis rule defining the probability of the worse solutions acceptance:

$$P(1) = \exp\Big(-\frac{\overline{\Delta c}}{T(1)}\Big), \tag{9}$$

thus in consequence

$$T(1) = -\frac{\overline{\Delta c}}{\ln P(1)}. \tag{10}$$

The mean difference in cost can be replaced with, e.g., the standard deviation of the cost function value, marked as $\overline{\sigma}_c$, also estimated on the basis of the set of pilot iterations (Sait and Youssef, 2000). A problem which appears in the case of SA algorithms dedicated to minimizing functions with real arguments (including the aforementioned BA, FSA, VFSA and ASA) is the dependence of the strategy for generating a neighbor solution on temperature. Therefore, both the standard deviation $\overline{\sigma}_c$ and the mean $\overline{\Delta c}$ are directly dependent on it.

The application of the formula (10) is not possible here, and in the case of these algorithms the initial temperature value is usually arbitrary. This paper proposes a different strategy based on the generation of a set of pilot iterations. It allows the value $T(1)$ to be obtained with the assumption of any initial probability of worse solution acceptance. As important as the choice of the initial temperature is the determination of the iteration at which the algorithm should be terminated. The simplest (although not flexible and often requiring too detailed knowledge of the investigated task) stop criterion is reaching a previously assumed number of iterations or a satisfactory cost function value. An alternative could be to finish the algorithm when following a certain number of iterations the best obtained solution is not improved, or the use of an appropriate statistical method based on the analysis of cost function values as they are obtained. The last concept is universal and (desirable among heuristic algorithms stop criteria) related to the expected result of their works. This usually consists of calculating the estimator of the expected value of the global minimum $\hat{c}_{\min}$ and finishing the algorithm in the iteration k, when the difference between it and the discovered smallest value $c_0(k)$ is not greater than the assumed positive ε, so if

$$|c_0(k) - \hat{c}_{\min}| \leq \varepsilon. \tag{11}$$

One recent technique using this type of strategy is the algorithm proposed by Bartkuté and Sakalauskas (2009). In order to calculate the value $\hat{c}_{\min}$ an estimator is applied here based on order statistics (David and Nagaraja, 2003). This algorithm constitutes a universal and effective tool for a wide range of stochastic optimization techniques. Such a method, used as part of the FSA procedure, will now be described.

Let therefore $\{c_0(k), c_1(k), c_2(k), \ldots, c_r(k)\}$ denote the ordered non-decreasing set of r lowest cost function values, obtained during k iterations of the algorithm. In the case of an algorithm convergent to a global minimum, the condition $\lim_{k\to\infty} c_j(k) = c_{\min}$ is fulfilled for every $j \in \mathbb{N}$, while the sequences $c_j(k)$ can be applied to construct the aforementioned estimator value $\hat{c}_{\min}$. This estimator makes use of the assumption of asymptotic convergence of the order statistic distribution to the Weibull distribution, and in the iteration k takes the general form:

$$\hat{c}_{\min}(k) = c_0(k) - \frac{\frac{2t}{\beta-1}}{r}(c_r(k) - c_0(k)). \tag{12}$$

The parameter β, occurring above, is termed a homogenous coefficient of the cost function h around its minimum. On additional assumptions, in calculational practice one can take $\beta = 2$ (Zhigljavsky and Žilinskas, 2008). The confidence interval for the cost function minimum, at the assumed significance level $\delta \in (0, 1)$, is of the form

$$\Big[c_0(k) - \frac{(1-(1-\delta)^{1/r})^{\beta/t}}{1-(1-(1-\delta)^{1/r})^{\beta/t}}(c_r(k) - c_0(k)), c_0(k)\Big]. \tag{13}$$

Bartkuté and Sakalauskas (2009) suggest that the point estimator (12) can be replaced by the confidence interval (13) with the algorithm being stopped when the confidence interval width is less than the aforementioned, assumed value ε. Such an idea, modified for the specific problem under investigation, will be applied here. The simulated annealing procedure can be easily parallelized, whether for required calculations or in the scheme of establishing subsequent solutions. While parallelizing the SA algorithm is not a new idea and was already investigated a few years after its creation (Azencot, 1992), it needs to be adapted for particular applicational tasks (Alba, 2005). At present the suitability of the Parallel Simulated Annealing (PSA) algorithm continuously increases with the common availability of multicore systems. In the algorithm worked out in this paper, a variant will be taken with parallel generation of neighbor solutions, assuming that the number of SA threads equals that of available processing units.

3. Procedure for reducing the dimension and sample size

The algorithm investigated in this paper consists of two functional components: a procedure for reducing the dimension and a way of allowing the sample size to be decreased. They are implemented sequentially, with the latter dependent on the results of the former. The reduction of the sample size is optional here.

3.1. Procedure for dimension reduction. The aim of the algorithm under investigation is a decrease in the dimensionality of the data set elements, represented by the matrix X with the form specified by the formula (1), so of the dimension $m \times n$, where m means the data set size and n the dimension of its elements. In consequence, the reduced form of this set is represented by the data matrix Y of the dimension $m \times N$, while N denotes the assumed reduced dimension of elements, appropriately less than n. The procedure for reducing the dimension is based on the linear transformation (2). For the purposes of notation used in the simulated annealing algorithm, matrix A is denoted as the row vector

$$\Big[a_{11}, a_{12}, \ldots, a_{1N}, a_{21}, a_{22}, \ldots, a_{2N}, \ldots, a_{n1}, a_{n2}, \ldots, a_{nN}\Big], \tag{14}$$

which represents the current solution $z(k) \in \mathbb{R}^{(nN)}$ in any iteration k.

In order to generate neighbor solutions, a strategy was used based on the multidimensional generator of the Cauchy distribution (formulas (7) and (8)). The quality of the obtained solution can be evaluated with the application of the cost function h, which is the function of the raw stress S_R given by the dependence (5), where the elements of the matrix Y are defined on the basis of Eqn. (2). The alternative possibility of using the Sammon stress (4) for this purpose was also examined. The developed procedure requires firstly that the basic parameters should be specified: the dimension of the reduced space N, a coefficient defining directly the maximum allowed width of the confidence interval ε_w for the stop criterion based on the order statistics, the number of processing threads of the FSA procedure p_{thread}, the initial scaling coefficient (step length) for the multidimensional Cauchy generator T_{scale}, as well as the probability of acceptance of a worse solution $P(1)$ in the first iteration of the FSA algorithm.

Starting the algorithm requires moreover the generation of the initial solution $z(0)$. To this aim, the feature selection procedure of Pal and Mitra (2004), described in the previous section, was realized. Here $k = n - N$ should be assumed, which in consequence usually results in obtaining approximately N clusters. The aforementioned procedure leads to obtaining the auxiliary vector $b \in \mathbb{R}^n$, whose particular coordinates characterize the number of the cluster to which the coordinate from the original space was mapped, as well as the vector $b_r \in \mathbb{R}^n$ of binary values $b_r(i) \in \{0, 1\}$ for $i = 1, 2, \ldots, n$, defining whether a given feature was chosen as a representative of the cluster to which it belongs, in which case $b_r(i) = 1$, or not, then $b_r(i) = 0$. The auxiliary vectors b and b_r can be used in the algorithm considered for generating the initial solution in two ways:

1. Each of N features of the initial solution is a

linear combination of features mapped to one of N clusters—to define the form of the matrix A one can use

$$a_{ij} = \begin{cases} 1 \text{ if } b(i) = j, \\ 0 \text{ if } b(j) \neq j \end{cases} \tag{15}$$

for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, N$.

2. Each of N features of the initial solution is given as a representative for one of N clusters—the form of the matrix A is then defined as

$$a_{ij} = \begin{cases} 1 \text{ if } b_r(i) = 1 \text{ and } b(i) = j, \\ 0 \text{ if } b_r(i) = 0 \end{cases} \tag{16}$$

for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, N$.

The possibility of applying both of the above ways of generating an initial solution—the first called a linear combination of features and the second referred to as feature selection—is a subject of detailed experimental analysis concerning dimensional reduction, described in Section 4. After generating the initial solution, in order to carry out the simulated annealing algorithm, the temperature $T(1)$ should be established for the first iteration. To this aim, the technique presented in the previous section can be followed, allowing us at the start to obtain the assumed initial value of the probability of worse solution acceptance $P(1)$. In the case of the algorithm for generating neighbor solutions, it is not recommended to use the relation resulting from the equality (9). As mentioned in the previous section, this is implied by the dependence of a formula for generating a neighbor solution on the annealing temperature. In order to avoid this inconvenience, an additional coefficient T_{scale} was introduced, being the parameter of the Cauchy distribution in the first iteration of the FSA algorithm (also known as an initial step length), and furthermore the temperature occurring in the generating distribution was scaled.

The coefficient T_{scale} is thus used as a parameter of the random numbers generator, with the aim of calculating a set of pilot iterations (the size of this set is assumed to be 100). These iterations consist of the generation of an appropriate number of transitions from $z(0)$—worse in the sense of the cost function used—to the neighbor solution $z(1)$, and the establishment of the mean value of the cost difference $\overline{\delta c}$ between $z(1)$ and $z(0)$. This value is inserted to the formula (10), through which the initial temperature can be obtained. Moreover, in order to find the assumed shape of the generated distribution, in the first iteration of the FSA algorithm an additional scaling coefficient is calculated:

$$c_{\text{temp}} = \frac{-\Delta c}{\ln P(1) T_{\text{scale}}}. \tag{17}$$

In consequence, in the first iteration of the actual algorithm, in order to generate a neighbor solution, the scaled temperature $T(1)/c_{\text{temp}}$ (therefore T_{scale}) is used, and for the Metropolis rule—it is just the value $T(1)$. Similar scaling takes place during the generation of neighbor solutions in each consecutive iteration of the FSA algorithm. Thanks to this kind of operation it becomes possible to fix the initial probability of acceptance of a worse solution, determined by the coefficient $P(1)$, while retaining the additional possibility of establishing—by assuming the value T_{scale}—the parameter of the initial spread of values obtained from a pseudorandom numbers generator.

All iterations of the FSA algorithm have been parallelized using a strategy with parallel generation of neighbor solutions. So each of p_{thread} threads creates a solution neighboring the one established in the previous iteration $z(k-1)$. This occurs with the application of a random generator with a multidimensional Cauchy distribution. For all threads, the annealing temperature is identical and equals $T(k)/c_{\text{temp}}$. Furthermore, every thread realizes the procedure for the Metropolis rule, accepting or rejecting its own obtained neighbor solutions.

The next two steps of the algorithm are performed in sequence. So, first the current solution is established for the SA algorithm. The procedure for this is to choose as a current solution either the best from those better than that the ones found in the previous iteration obtained by different threads, or if such a solution does not exist–a random selection of one of the worse solutions. Thus, the current solution, together with the temperature updated according to the formula (6), is used in the next iteration of the FSA algorithm as the current solution. This kind of strategy can be classified as a method of parallel processing based on speculative decomposition.

The last step performed as part of a single iteration is verifying the criterion for stopping the SA procedure. To this aim, the confidence interval for the minimum value of the cost function, given by the formula (13), is calculated. The order statistics used for interval estimation have the order r assumed to be 20, in accordance with the proposals of Bartkuté and Sakalauskas (2009). As a significance level δ for the confidence interval defined by the formula (13), a typical value 0.99 is assumed. The width of the confidence interval is compared with the threshold value ε calculated at every iteration with

$$\varepsilon = 10^{-\varepsilon_w} c_0(k). \tag{18}$$

Finally, the simulated annealing procedure is terminated if

$$\frac{(1-(1-\delta)^{1/r})^{\beta/t}}{1-(1-(1-\delta)^{1/r})^{\beta/t}} (c_r(k) - c_0(k)) > \varepsilon, \tag{19}$$

with the notation introduced at the end of Section 2. Finding the threshold value ε based on the formula (19)

allows the adaptation of such a criterion to the structure of a specific data set under investigation. The sensitivity of the above procedure can be set by assuming the value of the exponent $\varepsilon_w \in \mathbb{N}$, one of the arbitrarily fixed parameters of the procedure worked out in this paper. It is worth noting that the nature of the method presented here for dimension reduction enables the establishment of the "contribution" which particular elements of the data set Y make to the final value $c_0(k_{\text{stop}})$. This fact will be used in the procedure for reducing the sample size, which will be presented in the next section. A graphical description of the algorithm is summarized in Fig. 1.

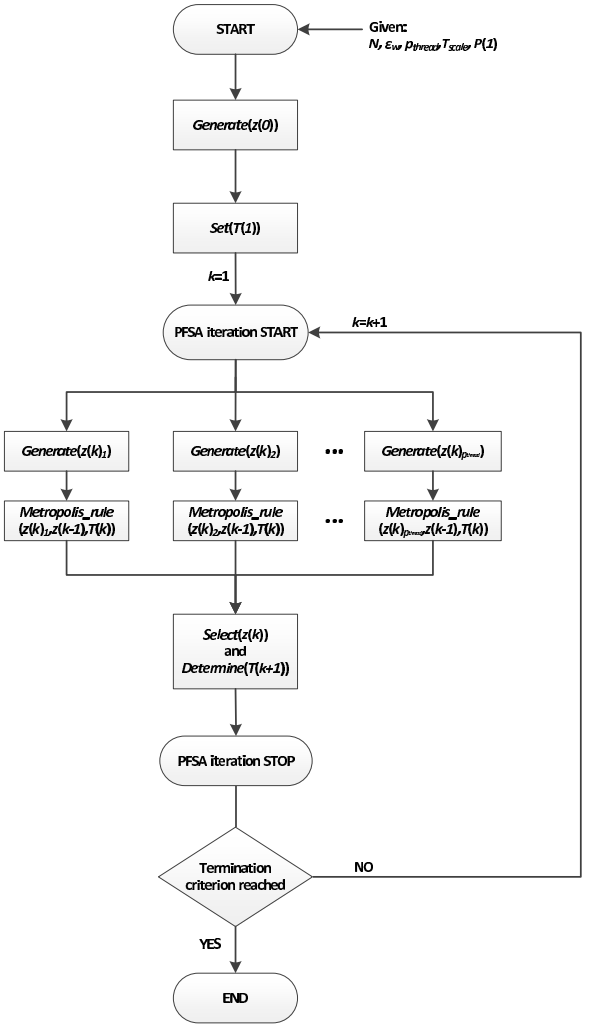


Fig. 1. Dimension reduction algorithm.

3.2. Procedure for sample size reduction. In the case of the dimension reduction method presented above, some sample elements may be subject to an undesired shift with respect to the others and, as a result, may noticeably worsen the results of an exploratory data analysis procedure in the reduced space $\mathbb{R}^N$. A measure of the location deformation of the single sample element x_i compared to the others, resulting from the transformation (2), is the corresponding stress value $c_0(k_{\text{stop}})$ calculated for this point (called the stress per point) (Borg and Groenen, 2005). In the case of the raw stress it is given by

$$c_0(k_{\text{stop}})_i = S_R(A)_i = \sum_{\substack{j=1 \\ j\neq i}}^{m} \left(d_{ij} - \delta_{ij}(A)\right)^2, \qquad (20)$$

whereas for the Sammon stress it takes the form

$$\begin{aligned} &c_0(k_{\text{stop}})_i \\ &\quad = S_S(A)_i \\ &\quad = \frac{1}{\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} d_{ij}} \sum_{\substack{j=1 \\ j\neq i}}^{m} \frac{\left(d_{ij} - \delta_{ij}(A)\right)^2}{d_{ij}}. \end{aligned} \qquad (21)$$

It is worth noting that in both cases these values are nonzero, except for the case (unattainable in practice) of "perfect" matching of respective distances of elements in the original and reduced spaces. The values $c_0(k_{\text{stop}})_i$ for particular elements can be used to construct a set of weights, defining the adequacy of their location in a reduced space. Let therefore w_i represent nonnegative weight mapped with the element x_i. Taking the above into account, it is calculated according to the following formula:

$$w_i = \frac{m\frac{1}{c_0(k_{\text{stop}})_i}}{\sum_{i=1}^{m}\frac{1}{c_0(k_{\text{stop}})_i}}. \qquad (22)$$

The normalization which occurs in the above dependence guarantees the condition

$$m = \sum_{i=1}^{m} w_i. \qquad (23)$$

The weights in this form contain information as to the degree to which a given sample element changed its relative location compared with the rest, where the larger the weight, the more relatively adequate its location, and its significance should be greater for procedures of exploratory data analysis carried out in a space of a reduced dimension. The weights' values which are calculated on the basis of the above formulas can be used for further procedures of data analysis. They also allow the following method of reducing the sample size. From the reduced data set Y one can remove those m_{el} elements for which their respective weights fulfill the condition $w_i < W$ with assumed $W > 0$. Intuitively, $W = 1$ is justified taking into account the formula (23), and this results in the elimination of elements corresponding to the values w_i less than the mean. In conclusion, conjoining the methods from Sections 3.1 and 3.2 enables a data set with a reduced dimension as well as size to be obtained, with the degree of compression implied by the values of parameters N and W.

3.3. Comments and suggestions. In the case of the procedure for reducing the dimension and sample size presented here, efforts were made to limit the number of parameters, whose arbitrary selection is always a significant practical problem for heuristic algorithms. At the same time, conditioning data analysis tasks, in which the procedure investigated here will be applied, means that, from a practical point of view, it is useful to propose specific values of those parameters with an analysis of the influence of their potential changes.

One of the most important arbitrarily assumed parameters is the reduced space dimension N. It can be fixed initially using one of the methods for estimating a hidden dimension (Camastra, 2003), or by taking a value resulting from other requirements, for example, $N = 2$ or $N = 3$, to enable a suitable visualization of the investigated data set. It is worth remembering that the procedure applied earlier for generating an initial solution with the fixed parameter $k = n - N$ creates a solution which does not always have a dimensionality identical to the assumed (as mentioned in the previous section). If a strictly defined dimension of the reduced data set is required, one should adjust the parameter k by repeating the feature selection algorithm with its correctly modified value, or use the initial solution, generated randomly, of the assumed dimension of reduced space. It is also worth mentioning the problem of computational complexity of the procedure worked out here, in particular regarding calculation of the cost function value.

In practice, the computational time for the PSA algorithm increases exponentially with an increase in the sample size. So, despite the heuristic algorithm being the only method available in practice to minimize the stress function S_S or S_R for data sets of large dimensionality and size, its application must, nonetheless, be limited to those cases which are in fact feasible. Therefore, although the number of simulated annealing treads p_{thread} can be fixed at will, it should take the available number of processing units into account. This allows efficient parallel evaluation of the cost function by particular threads. It should also be noted that the subject algorithm, due to its universal character, can be applied to a broad range of problems in statistics and data analysis. An example, from the case of statistical kernel estimators (Kulczycki, 2005; Wand and Jones, 1995), is the introduction of generalization of the basic definition of the estimator of probability distribution density to the following form:

$$f(\hat{y}) = \frac{1}{h^n \sum_{i=1}^{m} w_i} \sum_{i=1}^{m} w_i K\left(\frac{y - y_i}{h}\right). \tag{24}$$

Such a concept allows not only a reduction in the sample size (for removed elements, $w_i = 0$ is assumed), but also alternatively an improvement in the quality of estimation in the reduced space without eliminating any elements from the initial data set. In the former case care should also be taken to normalize the weights after eliminating parts of elements, to fulfill the condition (23). This approach may be used, e.g., in the complete gradient clustering technique (Kulczycki and Charytanowicz, 2010). Weights w_i calculated in the above manner can also be introduced to modified classical methods of data analysis, such as a weighted k-means algorithm (Kerdprasop *et al.*, 2005), or a weighted technique of k-nearest neighbors (Parvin *et al.*, 2010). In the first case, the weights are activated in the procedure for determining centers of clusters. The location of the center of the cluster C_i, denoted by $s_i = [s_{i1}, s_{i2}, s_{iN}]^T$, is updated in every iteration if $\sum_{y_l \in C_i} w_l \neq 0$, according to the formula

$$s_{ij} = \frac{1}{\sum_{y_l \in C_i} w_l} \sum_{y_l \in C_i} w_l y_{lj} \tag{25}$$

for $j = 1, 2, \ldots, N$. In the k-nearest neighbors procedure, however, each distance from neighbors of any element from the learning data set is scaled using the appropriate weight.

4. Numerical verification

The presented methodology underwent detailed numerical testing. It consisted of an analysis of the main functional aspects of the investigated algorithm, in particular the dependence of its efficiency on the values of arbitrarily assumed parameters. Another subject of the numerical analysis was the quality of reduction of the dimension and sample size, also in comparison with alternative solutions available in the literature. Examination was made with both data sets obtained from repositories and the literature, as well as those generated pseudorandomly with various dimensionalities and configurations. The former consist of the data sets, denoted hereinafter as W1, W2, W3, W4 and W5, with the first four taken from the Machine Learning Repository maintained by the Center for Machine Learning and Intelligent Systems, University of California Irvine, on the website (*UC Irvine Machine Learning Repository*, 2013), while the fifth one comes from the authors' own research (Charytanowicz *et al.*, 2010). Thus

- W1: *glass*, contains the results of chemical and visual analysis of glass samples, from 6 weakly separable (Ishibuchi *et al.*, 2001) classes, characterized by 9 attributes;
- W2: *wine*, represents the results of analysis of samples from 3 different producers, making up strongly separable (Cortez *et al.*, 2009) classes, with 13 attributes;

- W3: *Wisconsin Breast Cancer (WBC)*, obtained during oncological examinations (Mangasarian and Wolberg, 1990), has 2 classes and 9 attributes;
- W4: *vehicle*, presents measurements of vehicle shapes taken with a camera (Oliveira and Pedrycz, 2007), grouped into 4 classes, described by 18 attributes;
- W5: *seeds*, lists the dimensions of 7 geometrical properties of 3 species of wheat seeds, obtained using the X-ray technology (Charytanowicz *et al.*, 2010).

For classification purposes the above data sets are divided into learning and testing samples in the proportion 4:1. The latter, data sets R1, R2, R3, R4 and R5, were obtained using a pseudorandom number generator with the normal distribution, with the assumed expected value and the covariance matrix Σ. The tested sets were formed as linear combinations of such components with coefficients $p_k \in (0, 1)$. Their list is presented in Table 1.

For classification purposes, every component of a given distribution is treated as an individual class, while in the case of clustering it is assumed that subsequent clusters contain elements of particular components of linear combinations.

Similarly for the needs of identification of outliers, atypical elements are considered to be components with marginal values of linear combination coefficients. The case of clustering uses the classic procedure of k-means with the assumed number of clusters equal the number of classes appearing in the tested set. Its quality is assessed with the Rand index (Parvin *et al.*, 1971):

$$I_c = \frac{a+b}{\binom{m}{2}} \cdot 100\%, \tag{26}$$

where a and b denote the number of pairs of elements properly assigned to the same and different clusters.

If the results concern the initial space, then the above index specifies itself to $I_{c\mathrm{INIT}}$, whereas $I_{c\mathrm{RED}}$ corresponds to the reduced space. Next, the classification procedure will be realized applying the standard nearest neighbor algorithm. Its results are judged with a natural indicator which is given as a percentage of correctly qualified elements, denoted in the initial space as $I_{k\mathrm{INIT}}$ and in the reduced as $I_{k\mathrm{RED}}$. Finally, an identification of outliers is made by a testing procedure based on statistical kernel estimators (Kulczycki, 2008). The quality of the effects can be appraised in the same way as for classification, where the indicators are termed $I_{o\mathrm{INIT}}$ and $I_{o\mathrm{RED}}$, respectively. All performance indices used in the paper are outlined in Table 2.

The dimension of the reduced space N is arbitrarily assumed following suggestions from the appropriate subject literature (Charytanowicz *et al.*, 2010; Saxena *et al.*, 2010). For pseudorandom data sets its value was selected to ensure a convenient visualization of results. Table 3 presents a summary of these. In every case, the tested procedure was repeated 30 times, following which, for further inference, the mean values and standard deviations of the introduced indexes were used.

Table 3. Dimensions of reduced spaces.

Data set	N	Data set	N
W1	4	R1	1
W2	5	R2	1
W3	4	R3	2
W4	10	R4	2
W5	2	R5	2

Thus, preliminary research concerned the functionality of the investigated procedure, in particular its sensitivity to versions and parameters with arbitrarily assumed values. During the conducted research, the options mentioned below were taken into account:

(i) two versions of generating the initial solution: feature selection (version A) and the linear combination of features (version B); in addition, a randomly obtained initial form of the transformation matrix was also considered, which was, however, dismissed at the preliminary stage, due to its ineffectiveness;

(ii) two possible forms of the cost function, described by the formulas (5) and (4);

(iii) four levels of sensitivity for the stop criterion $\varepsilon_w = \{1, 2, 3, 4\}$, whose range resulted from practical conditioning of the test carried out;

(iv) three possible values for the parameter $T_{\mathrm{scale}} = \{0.01\ 0.1\ 1\}$.

Furthermore, in this phase it was also assumed that $P(1) = 0.7$ and $p_{\mathrm{thread}} = 4$.

In general, it can be ascertained that the results obtained for every option did not differ significantly, therefore (worth stressing) the algorithm worked out is not very sensitive to the selected parameters. In practice, this is highly valuable and considerably increases its applicational potential.

The preferred strategy for generating an initial solution seems to be the version based on feature selection (version A). The form of the cost function, however, does not appear to have a great influence on the quality of the transformation. The difference, while negligible in the sense of the examined indexes, can only be seen following the analysis for $\varepsilon_w = 4$. In problems of identifying outliers and clustering, using the function S_R is recommended, and for classification—the function S_S. However, it is difficult to definitely suggest a value for

Table 1. Characteristics of data sets R1–R5 of pseudorandom nature.

Data set	n	Parameters of distribution components			
		k	p_k	μ_k	Σ_k
R1(m,a,o)	2	1	*o*/2	[0 *a*]	$\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$
		2	*o*/2	[*a a*]	
		3	1-*o*	[*a* 0]	
R2(m,a,o)	3	1	1-*o*	[0 0 0]	$\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}$
		2	*o*/4	[*a* 0 0]	
		3	*o*/4	[*a* 0 -*a*]	
		4	*o*/4	[*a a* 0]	
		5	*o*/4	[0 0 *a*]	
R3(m,a,o)	5	1	*o*/5	[*a* 0 0 0 0]	$[\sigma_{ij}]$ $i,j=1,2,\ldots,n$ $\forall i=j, \sigma_{ij}=1$ $\forall i\neq j, \sigma_{ij}=0$
		2	*o*/5	[0 *a* 0 0 0]	
		3	*o*/5	[0 0 *a* 0 0]	
		4	*o*/5	[0 0 0 *a* 0]	
		5	*o*/5	[0 0 0 0 *a*]	
		6	1-*o*	[*a a a a a*]	
R4(m,a,o)	10	1	*o*/2	[*a* 0 0 0 0 0 0 0 0 0]	$[\sigma_{ij}]$ $i,j=1,2,\ldots,n$ $\forall i=j, \sigma_{ij}=1$ $\forall i\neq j, \sigma_{ij}=0$
		2	*o*/2	[0 0 0 0 0 0 0 0 0 -*a*]	
		3	(1-*o*)/2	[0 0 0 0 0 0 0 0 0 0]	
		4	(1-*o*)/2	[0 0 0 0 0 0 0 0 0 *a*/2]	
R5(m,a,o)	15	1	1-*o*	[5*a* -*a a* -*a a* -*a a* -*a a* -*a a* -*a a* -*a* 5*a*]	$[\sigma_{ij}]$ $i,j=1,2,\ldots,n$ $\forall i=j, \sigma_{ij}=1$ $\forall i\neq j, \sigma_{ij}=0$
		2	*o*/2	[*a* -*a a* -*a a* -*a a* -*a a* -*a a* -*a a* -*a a*]	
		3	*o*/2	[-*a a a* -*a a* -*a a* -*a a* -*a a* -*a a* -*a a* -*a*]	

the coefficient T_{scale}. The most stable in the sense of presented efficiency seems to be the version with $T_{\text{scale}} = 0.1$. Lastly, the most advantageous results were obtained for two versions of the FSA algorithm based on generating an initial solution with selection of adequate features, the sensitivity of the stop criterion $\varepsilon_w = 4$, the coefficient $T_{\text{scale}} = 0.1$, as well as the cost function S_S or S_R. These two versions, called standard in the following part of this paper, were used in later research involving numerical evaluation of the concept presented here.

In the next stage of the experimental research, the dependence of the algorithm's efficiency on the value $P(1)$ was examined. Appropriate tests were carried out considering the cases $P(1) \approx 0$, $P(1) = 0.1, 0.2, \ldots, 0.9$ and also $P(1) \approx 1$. Above all it can be inferred that the FSA algorithm applied is highly efficient with respect to parallel local search (which corresponds to $P(1) \approx 0$), and also primarily a random parallel exploration of the solution space (when $P(1) \approx 1$). In the case of the algorithm used, low probabilities in the interval $[0.1, 0.4]$ are preferred. This results from the employed scheme of generating neighbor solutions, which, together with the procedure of temperature change according to the formula (6), allows "distant jumps" in the solutions space under investigation. It compensates for the influence of the low probability of the acceptance of worse solutions.

The next subject of the research was the presented algorithm for reducing the sample size. Verification of the obtained results consisted in defining the percentage reduction of the sample for particular values of $W = 0, 0.1, \ldots, 2.0$. The dependence of the number of removed elements on the parameter W is not easy to express with an analytical formula. It is, however, possible to continuously adjust the degree of reduction achieved. The value W close to zero implies the removal of a small number of data set elements, while even $W = 2$ results in a reduction greater than 80%. Fixing $W = 1$ usually causes a sample size reduced by about half, because the elements with weights less than average are eliminated.

An increase in the intensity of sample reduction in the case of some data analysis procedures may have a positive influence on the quality of the results in the reduced space. This was mainly observed for the identification of outliers, where $W = 0.9$ produced the best results. In this event, removal from the data set of a significant amount of its elements did not usually lower the difference between typical elements and outliers, but often actually exaggerated it. In clustering and classification, one can use the research to argue that such a high degree of reduction is advisable, as it leads to the elimination of sample elements which are fundamental to the creation of a grouping structure.

Table 2. Performance indices used for evaluating results of the data analysis tasks considered.

Task	Cluster analysis	Classification	Outlier detection
Solution quality indicator and its description	$I_c = \frac{a+b}{\binom{m}{2}} \cdot 100\%$ m: sample size a: number of sample elements properly assigned to the same cluster b: number of sample elements properly assigned to different clusters	$I_k = \frac{a}{m} \cdot 100\%$ m: sample size a: number of sample elements assigned to the right class	$I_o = \frac{a}{m} \cdot 100\%$ m: sample size a: number of sample elements assigned to the right class (outlier/typical)

In conclusion it can be suggested that—where other conditions are not present—for these problems, the compression coefficient should be assumed from the range between 0.1 and 0.2. This often enables the elimination of such elements which, as a result of transformation, undergo significant displacement with respect to the rest, causing an improvement in clustering and classification quality in the reduced space. All suggested values of the algorithm's parameters and recommended options are ultimately summarized in Table 4.

Table 4. Suggested values and options of the algorithm.

Parameter/option	Suggested value
Initial solution generation	Version A
Cost function	S_S for classification S_R otherwise
Sensitivity of stop criterion	$\epsilon_w = 4$
Temperature scaling	$T_{\text{scale}} = 0.1$
Initial probability of worse solution acceptance	$P(1) \in [0.1, 0.4]$
Compression coefficient	$W \cong 1$ for outlier detection $W \in [0.1, 0, 2]$ otherwise

Finally, the effectiveness of the algorithm under investigation was compared with other reference methods. The first verification was made only for the dimension reduction procedure, and then together with the sample size reduction algorithm. Research was carried out using the options given above, whereas for the identification of outliers and clustering the cost function S_R was used, and for classification the function S_S. In the case of dimension reduction, the classic PCA linear algorithm as well as the aforementioned features selection procedure using evolutionary algorithms and the Sammon stress function were used as reference techniques. Research of the latter focused only on the efficiency of classification, for which this strategy was confirmed earlier in the work (Saxena *et al.*, 2010). The results of experiments are shown in Tables 5–7. The reported execution times include both dimensionality reduction and the given data mining task.

In interpreting the results of identifying outliers shown in Table 5, it must be underlined that the proposed method for dimension reduction ensures, in most cases, a greater efficiency of the data analysis procedure used. This is also true when comparing results of this procedure in an unreduced space, as well as when juxtaposing the result with that obtained using principal component algorithm.

Reducing the dimensionality of a data set with the subject algorithm, also in the case of classification (as shown in Table 6), ensures greater average efficiency, both when the reference constitutes the result obtained on the basis of a reduced data set using the principal component algorithm, as well as in the consequent analysis of a features set, chosen with an evolutionary strategy.

For clustering (Table 7), a reduced form of a data set is usually achieved, for which the k-means algorithm produces a result which is more compatible (in the sense of the $I_{c\text{RED}}$ index) with the data structure than for the PCA method.

Table 7. Comparison of results of dimension reduction for clustering.

	W1 *glass*	W2 *wine*	W3 *WBC*	W4 *vehicle*	W5 *seeds*
$\overline{I}_{c\text{INIT}}$	68.23	93.48	66.23	64.18	91.06
$\pm\sigma(I_{c\text{RED}})$	±1.39	±0.68	±0.38	±1.08	±0.53
	Reduction using FSA				
$\overline{I}_{c\text{RED}}$	68.01	92.78	66.17	64.49	89.74
$\pm\sigma(I_{c\text{RED}})$	±0.61	±0.76	±0.31	±0.28	±0.78
$\max(I_{c\text{RED}})$	69.18	94.14	66.69	65.02	90.52
time$[s]$	4.2	2.4	149.3	26.8	18.4
	Reduction using PCA				
$\overline{I}_{c\text{RED}}$	67.71	92.64	66.16	64.16	88.95
$\pm\sigma(I_{c\text{RED}})$	±0.21	±0.88	±0.21	±0.32	±0.55

In the next stage, a complete comparison, similar to the one presented above, was conducted, additionally taking into account the procedure for sample size reduction. According to previous inferences, for the identification of outliers, the compression coefficient W

Table 5. Comparison of results of dimension reduction for identification of outliers.

	R1 (300, 4, 0.1)	R2 (300, 4, 0.1)	R3 (500, 3, 0.1)	R4 (500, 4, 0.1)	R5 (500, 2, 0.1)
$\overline{I}_{o\mathrm{INIT}}$	92.66	91.33	94.4	86.80	90.00
			Reduction using FSA		
$\overline{I}_{o\mathrm{RED}}$	97.33	94.00	93.43	89.04	93.42
$\pm\sigma(I_{o\mathrm{RED}})$	$\pm$0.00	$\pm$1.91	$\pm$0.87	$\pm$1.20	$\pm$0.73
$\max(I_{o\mathrm{RED}})$	97.33	95.33	95.20	91.20	95.20
time$[ms]$	67.7	1169.0	2820.6	28892.6	44405.40
			Reduction using PCA		
$\overline{I}_{o\mathrm{RED}}$	93.33	90.00	92.40	86.80	90.00
time$[ms]$	5.7	15.1	16.9	58.6	98.3

Table 6. Comparison of results of dimension reduction for classification.

	W1 *glass*	W2 *wine*	W3 *WBC*	W4 *vehicle*	W5 *seeds*
$\overline{I}_{k\mathrm{INIT}}$	71.90	74.57	95.88	63.37	90.23
$\pm\sigma(I_{k\mathrm{INIT}})$	$\pm$8.10	$\pm$5.29	$\pm$1.35	$\pm$3.34	$\pm$2.85
			Reduction using FSA		
$\overline{I}_{k\mathrm{RED}}$	69.29	75.14	95.66	63.85	89.29
$\pm\sigma(I_{k\mathrm{RED}})$	$\pm$3.79	$\pm$6.46	$\pm$1.12	$\pm$2.98	$\pm$2.31
$time[s]$	4.5	27.1	10.0	400.2	4.7
			Reduction using PCA		
$\overline{I}_{k\mathrm{RED}}$	58.33	72.00	95.29	62.24	83.09
$\pm\sigma(I_{k\mathrm{RED}})$	$\pm$6.37	$\pm$7.22	$\pm$2.06	$\pm$3.84	$\pm$7.31
			Reduction using evolutionary algorithms		
$\overline{I}_{k\mathrm{RED}}$	64.80	72.82	95.10	60.86	N/A
$\pm\sigma(I_{k\mathrm{RED}})$	$\pm$4.43	$\pm$1.02	$\pm$0.80	$\pm$1.51	

value was fixed at the level 0.9, for classification $W = 0.1$, and for clustering $W = 0.2$. To illustrate, in the latter two, a selected version with more intensive reduction was additionally considered. In order to verify the effectiveness of combining the investigated procedures of dimension and sample size reduction, the possibility was also examined of using clustering together with the principal component algorithm, and the proposed sample size reduction with a similar reduction intensity. Moreover, in the case of identifying outliers, the prospect was also considered of applying the PCA method with the algorithm for condensing data presented by Mitra *et al.* (2002). The results obtained are presented sequentially in Tables 8–10.

The results displayed in Tables 8–10 lead us to conclude that the strategy of sample size reduction is a worthy supplement to the method for dimension reduction based on the FSA algorithm. For identifying outliers, treated data sets saw 30–50% of sample elements removed and, as expected, the quality of identification of outliers improved greatly. In the cases of clustering and classification, the first series of experiments did not apply intensive reduction of the sample size. However, elimination of even a small part of "stray" elements transformed during dimension reduction leads to an increase in the average efficiency of the procedures applied in the reduced space. This is especially obvious in classification with the nearest neighbor algorithm, which is very sensitive to disturbances in the data set structure introduced by the transformation. An increase in the compression coefficient W value, realized in the second stage of the numerical experiments, usually sees the worsening of stability and/or average efficiency of classification and clustering. This mainly concerns the first procedure, where the growth of the value W above 0.5 has decidedly negative implications. In the case of clustering, one can achieve high efficiency of conducted analysis most often in single experiments only, with an additional 20–40% reduction in the sample size, although the improvement was permanent for the data set seeds. Despite the results obtained for sets reduced by the PCA algorithm, and the procedure for sample size reduction proposed here usually being significantly worse than the two-stage method investigated in this paper, a combination of these two algorithms also seems to be a valuable concept in practical applications and is

broadly studied within a separate contribution (Łukasik and Kulczycki, 2013).

In conclusion of this experimental study, let us illustrate the advantages of using the proposed algorithm for additional multidimensional dataset from the UCI Machine Learning Repository. It concerns an image segmentation problem and consists of 2310 instances with 19 attributes each, representing 7 classes of textures. The execution of cluster analysis for PCA-reduced two feature set results in the average accuracy of 79.9%. For the same feature set size, PFSA-based algorithm reaches on average 82.3% of the Rand index value. The obtained cluster structure is illustrated on Fig. 2.

5. Summary and final remarks

The subject of this paper was a complete algorithm for reducing the dimension and sample size, ready to be used in a wide range of exploratory data analysis problems. It constitutes a universal, unsupervised linear transformation of a features space, with the aim of best maintaining distances between sample elements, additionally supplemented by a reduction in significance of those elements whose locations in relation to the rest changed considerably. The foundation for this algorithm is an innovative version of the parallel fast simulated annealing procedure, with the stop criterion based on order statistics, automatic generation of initial temperature and a multidimensional generator of pseudorandom numbers with the Cauchy distribution. The sample size was reduced as a result of calculating weights for particular elements, with the possibility of continuous adjustment of this procedure's intensity, through establishing an appropriate—for an investigated problem—value for the compression coefficient.

Besides presenting the concept of the algorithm and selected applicational properties, diverse numerical experiments were also included. They led to the conclusions regarding the recommended values of parameters which can be fixed within this algorithm, and also allowed its properties to be examined. In addition, during the research the investigated procedure was compared with the principle component method and a technique using evolutionary algorithms. Generally the obtained results of numerical verification, conducted for both pseudorandom and real data sets, confirmed the validity of the proposed concept as well as its many positive properties.

Furthermore, the particular functional components of the procedure presented here can be applied in other tasks of information processing. Thus, the parallel fast simulated annealing algorithm may be used successfully in a wide range of optimization problems, due to its universal structure and relatively intuitive selection of arbitrarily assumed parameters. What is more, the proposed procedure for reducing the sample size can be applied together with other, also nonlinear, strategies for dimension reduction.

Finally, it is worth stressing that, regardless of the calculational complexity of the proposed algorithm, as its execution time is guided by the adaptive stop criterion it corresponds to the difficulty of the problem at hand. The use of linear transformation, which is easy to generalize, and the simple idea of a set of weights, makes it possible to use the investigated method effectively in a wide range of contemporary data analysis problems, also those involving data sets characterized by the influx of new sample elements.

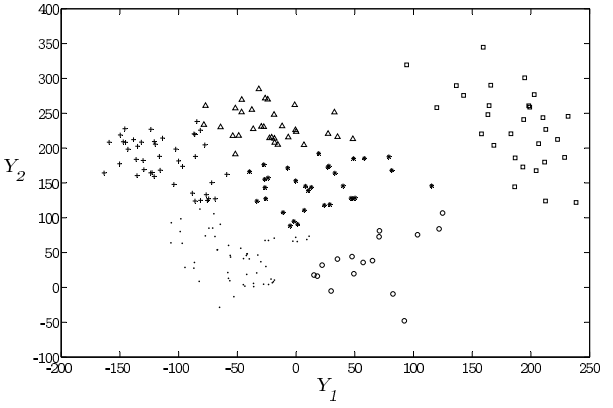


Fig. 2. Result of cluster analysis for the two-dimensional representation of the *segmentation* dataset.

Acknowledgment

The study is co-funded by the European Union from resources of the European Social Fund (project PO KL *Information technologies: Research and their interdisciplinary applications*, agreement UDA-POKL.04.01.01-00-051/10-00).

References

Aarts, E., Korst, J. and van Laarhoven, P. (1997). Simulated annealing, *in* E. Aarts and J. Lenstra (Eds.), *Local Search in Combinatorial Optimization*, Wiley, Chichester, pp. 91–120.

Alba, E. (2005). *Parallel Metaheuristics: A New Class of Algorithms*, Wiley, New York, NY.

Aswani Kumar, C. and Srinivas, S. (2006). Latent semantic indexing using eigenvalue analysis for efficient information retrieval, *International Journal of Applied Mathematics and Computer Science* **16**(4): 551–558.

Aswani Kumar, C. (2009). Analysis of unsupervised dimensionality techniques, *Computer Science and Information Systems* **6**(2): 217–227.

Table 8. Comparison of results of dimension and sample size reduction for identification of outliers.

	R1 (300, 4, 0.1)	R2 (300, 4, 0.1)	R3 (500, 3, 0.1)	R4 (500, 4, 0.1)	R5 (500, 2, 0.1)
	Reduction of dimension using FSA and sample size ($W = 0.9$)				
$\overline{I}_{o\mathrm{RED}}$	98.00	96.71	94.00	90.64	96.41
$\pm\sigma(I_{o\mathrm{RED}})$	$\pm$0.00	$\pm$0.49	$\pm$1.91	$\pm$1.35	$\pm$1.59
$\frac{\overline{m}_{el}}{m} \cdot 100\%$	44.69	46.93	41.91	35.39	38.44
	Reduction of dimension using PCA and sample size (approximate size of reduced sample)				
$\overline{I}_{o\mathrm{RED}}$	94.00	90.00	92.00	86.80	90.00
$\frac{\overline{m}_{el}}{m} \cdot 100\%$	42.33	47.00	46.60	32.20	34.80
	Reduction of dimension using PCA and sample condensation (approximate size of reduced sample)				
$\overline{I}_{o\mathrm{RED}}$	93.33	90.00	93.20	86.00	90.00
$\frac{\overline{m}_{el}}{m} \cdot 100\%$	55.33	54.33	64.20	64.20	62.40

Table 9. Comparison of results of dimension and sample size reduction for clustering.

	W1 *glass*	W2 *wine*	W3 *WBC*	W4 *vehicle*	W5 *seeds*
	Reduction of dimension using FSA and sample size ($W = 0.2$)				
$\overline{I}_{c\mathrm{RED}}$	68.43	92.81	66.29	64.62	89.59
$\pm\sigma(I_{c\mathrm{RED}})$	$\pm$0.62	$\pm$0.76	$\pm$0.62	$\pm$0.24	$\pm$1.57
$\frac{\overline{m}_{el}}{m} \cdot 100\%$	4.31	0.16	5.90	0.83	2.68
	Reduction of dimension using PCA and sample size (approximate size of reduced sample)				
$\overline{I}_{c\mathrm{RED}}$	67.27	92.64	66.16	64.16	88.95
$\pm\sigma(I_{c\mathrm{RED}})$	$\pm$0.62	$\pm$0.76	$\pm$0.62	$\pm$0.24	$\pm$1.57
	Reduction of dimension using FSA and sample size (selected W)				
$\overline{I}_{c\mathrm{RED}}$	68.14	92.84	67.55	64.76	89.69
$\pm\sigma(I_{c\mathrm{RED}})$	$\pm$1.69	$\pm$2.08	$\pm$2.03	$\pm$0.40	$\pm$0.62
W	0.9	0.9	0.5	0.7	0.5
$\frac{\overline{m}_{el}}{m} \cdot 100\%$	44.65	41.14	32.51	21.73	19.38

Azencot, R. (1992). *Simulated Annealing: Parallelization Techniques*, Wiley, New York, NY.

Bartenhagen, C., Klein, H.-U., Ruckert, C., Jiang, X. and Dugas, M. (2010). Comparative study of unsupervised dimension reduction techniques for the visualization of microarray gene expression data, *BMC Bioinformatics* **11**, paper no. 567.

Bartkuté, V. and Sakalauskas, L. (2009). Statistical inferences for termination of Markov type random search algorithms, *Journal of Optimization Theory and Applications* **141**(3): 475–493.

Ben-Ameur, W. (2004). Computing the initial temperature of simulated annealing, *Computational Optimization and Applications* **29**(3): 367–383.

Borg, I. and Groenen, P. (2005). *Modern Multidimensional Scaling. Theory and Applications*, Springer-Verlag, Berlin.

Camastra, F. (2003). Data dimensionality estimation methods: A survey, *Pattern Recognition* **36**(12): 2945–2954.

Charytanowicz, M., Niewczas, J., Kulczycki, P., Kowalski, P., Łukasik, S. and Żak, S. (2010). Complete gradient clustering algorithm for features analysis of x-ray images, *in* E. Piątka and J. Kawa (Eds.), *Information Technologies in Biomedicine*, Vol. 2, Springer-Verlag, Berlin, pp. 15–24.

Cortez, P., Cerdeira, A., Almeida, F., Matos, T. and Reis, J. (2009). Modeling wine preferences by data mining from physicochemical properties, *Decision Support Systems* **47**(4): 547–553.

Cox, T. and Cox, M. (2000). *Multidimensional Scaling*, Chapman and Hall, Boca Raton, FL.

Cunningham, P. (2007). Dimension reduction, *Technical report*, UCD School of Computer Science and Informatics, Dublin.

Czarnowski, I. and Jędrzejowicz, P. (2011). Application of agent-based simulated annealing and tabu search procedures to solving the data reduction problem, *Inter-*

Table 10. Comparison of results of dimension and sample size reduction for classification.

	W1 *glass*	W2 *wine*	W3 *WBC*	W4 *vehicle*	W5 *seeds*
	Reduction of dimension using FSA and sample size ($W = 0.1$)				
$\overline{I}_{k\mathrm{RED}}$	70.48	78.00	95.95	63.96	89.76
$\pm\sigma(I_{k\mathrm{RED}})$	$\pm$7.02	$\pm$4.86	$\pm$1.43	$\pm$2.66	$\pm$3.18
$\frac{\overline{m}_{eL}}{m} \cdot 100\%$	0.92	3.61	0.20	0.30	0.47
	Reduction of dimension using PCA and sample size (approximate size of reduced sample)				
$\overline{I}_{k\mathrm{RED}}$	61.58	69.42	95.80	62.84	84.84
$\pm\sigma(I_{k\mathrm{RED}})$	$\pm$8.90	$\pm$8.08	$\pm$1.46	$\pm$3.79	$\pm$5.91
$\frac{\overline{m}_{eL}}{m} \cdot 100\%$	0.80	6.90	0.20	3.64	1.45
	Reduction of dimension using FSA and sample size (selected W)				
$\overline{I}_{k\mathrm{RED}}$	66.19	70.57	95.32	62.13	88.24
$\pm\sigma(I_{k\mathrm{RED}})$	$\pm$8.46	$\pm$4.75	$\pm$1.48	$\pm$2.63	$\pm$3.95
W	0.5	0.4	0.5	0.4	0.5
$\frac{\overline{m}_{eL}}{m} \cdot 100\%$	17.46	32.59	25.62	8.28	25.38

national Journal of Applied Mathematics and Computer Science **21**(1): 57–68, DOI: 10.2478/v10006-011-0004-3.

David, H. and Nagaraja, H. (2003). *Order Statistics*, Wiley, New York, NY.

Deng, Z., Chung, F.-L. and Wang, S. (2008). FRSDE: Fast reduced set density estimator using minimal enclosing ball approximation, *Pattern Recognition* **41**(4): 1363–1372.

François, D., Wertz, V. and Verleysen, M. (2007). The concentration of fractional distances, *IEEE Transactions on Knowledge and Data Engineering* **19**(7): 873–886.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distribution and the Bayesian restoration in images, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**: 721–741.

Gendreau, M. and Potvin, J.-Y. (2010). *Handbook of Metaheuristics*, Springer, New York, NY.

Han, J. and Kamber, M. (2006). *Data Mining: Concepts and Techniques*, Morgan Kaufmann, San Francisco, CA.

Ingber, L. (1996). Adaptive simulated annealing (ASA): Lessons learned, *Control and Cybernetics* **25**(1): 33–54.

Inza, I., Larranaga, P., Etxeberria, R. and Sierra, B. (2000). Feature subset selection by Bayesian network-based optimization, *Artificial Intelligence* **123**(1–2): 157–184.

Ishibuchi, H., Nakashima, T. and Murata, T. (2001). Three-objective genetics-based machine learning for linguistic rule extraction, *Information Sciences* **136**(1–4): 109–133.

Kerdprasop, K., Kerdprasop, N. and Sattayatham, P. (2005). Weighted k-means for density-biased clustering, *in* A. Tjoa and J. Trujillo (Eds.), *Data Warehousing and Knowledge Discovery,* Lecture Notes in Computer Science, Vol. 3589, Springer-Verlag, Berlin pp. 488–497.

Kulczycki, P. (2005). *Kernel Estimators in System Analysis*, WNT, Warsaw, (in Polish).

Kulczycki, P. (2008). Kernel estimators in industrial applications, *in* B. Prasad (Ed.), *Soft Computing Applications in Industry*, Springer-Verlag, Berlin, pp. 69–91.

Kulczycki, P. and Charytanowicz, M. (2010). A complete gradient clustering algorithm formed with kernel estimators, *International Journal of Applied Mathematics and Computer Science* **20**(1): 123–134, DOI: 10.2478/v10006-010-0009-3.

Kulczycki, P. and Kowalski, P. (2011). Bayes classification of imprecise information of interval type, *Control and Cybernetics* **40**(1): 101–123.

Kulczycki, P. and Łukasik, S. (2014). Reduction of dimension and size of data set by parallel fast simulated annealing, *in* L.T. Koczy, C.R. Pozna, R. Claudiu and J. Kacprzyk (Eds.), *Issues and Challenges of Intelligent Systems and Computational Intelligence*, Springer-Verlag, Berlin, pp. 273–292.

Kuo, Y. (2010). Using simulated annealing to minimize fuel consumption for the time-dependent vehicle routing problem, *Computers & Industrial Engineering* **59**(1): 157–165.

Łukasik, S. and Kulczycki, P. (2011). An algorithm for sample and data dimensionality reduction using fast simulated annealing, *in* J. Tang, I. King, L. Chen and J. Wang (Eds.), *Advanced Data Mining and Applications*, Lecture Notes in Computer Science, Vol. 7120, Springer-Verlag, Berlin, pp. 152–161.

Łukasik, S. and Kulczycki, P. (2013). Using topology preservation measures for multidimensional intelligent data analysis in the reduced feature space, *in* L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. Zadeh and J. Zurada (Eds.), *Artificial Intelligence and Soft Computing,* Lecture Notes in Computer Science, Vol. 7895, Springer-Verlag, Berlin, pp. 184–193.

Maaten, van der, L. (2009). *Feature Extraction from Visual Data*, Ph.D. thesis, Tilburg University, Tilburg.

Mangasarian, O. and Wolberg, W. (1990). Cancer diagnosis via linear programming, *SIAM News* **23**(5): 1–18.

Mitra, P., Murthy, C. and Pal, S. (2002). Density-based multiscale data condensation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(6): 734–747.

Nam, D., Lee, J.-S. and Park, C. (2004). n-dimensional Cauchy neighbor generation for the fast simulated annealing, *IEICE Transactions on Information and Systems* **E87-D**(11): 2499–2502.

Oliveira, J. and Pedrycz, W. (Eds.) (2007). *Advances in Fuzzy Clustering and Its Applications*, Wiley, Chichester.

Pal, S. and Mitra, P. (2004). *Pattern Recognition Algorithms for Data Mining*, Chapman and Hall, Boca Raton, FL.

Parvin, H., Alizadeh, H. and Minati, B. (1971). Objective criteria for the evaluation of clustering methods, *Journal of the American Statistical Association* **66**(336): 846–850.

Parvin, H., Alizadeh, H. and Minati, B. (2010). A modification on k-nearest neighbor classifier, *Global Journal of Computer Science and Technology* **10**(14): 37–41.

Sait, S. and Youssef, H. (2000). *Iterative Computer Algorithms with Applications in Engineering: Solving Combinatorial Optimization Problems*, IEEE Computer Society Press, Los Alamitos, CA.

Sammon, J. (1969). A nonlinear mapping for data structure analysis, *IEEE Transactions on Computers* **18**(5): 401–409.

Saxena, A., Pal, N. and Vora, M. (2010). Evolutionary methods for unsupervised feature selection using Sammon's stress function, *Fuzzy Information and Engineering* **2**(3): 229–247.

Strickert, M., Teichmann, S., Sreenivasulu, N. and Seiffert, U. (2005). DIPPP online self-improving linear map for distance-preserving data analysis, *5th Workshop on Self-Organizing Maps, WSOM'05, Paris, France*, pp. 661–668.

Sumi, S.M., Zaman, M.F. and Hirose, H. (2012). A rainfall forecasting method using machine learning models and its application to the Fukuoka city case, *International Journal of Applied Mathematics and Computer Science* **22**(4): 841–854, DOI: 10.2478/v10006-012-0062-1.

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

Tian, T., Wilcox, R. and James, G. (2010). Data reduction in classification: A simulated annealing based projection method, *Statistical Analysis and Data Mining* **3**(5): 319–331.

UC Irvine Machine Learning Repository (2013). `http://archive.ics.uci.edu/ml/`.

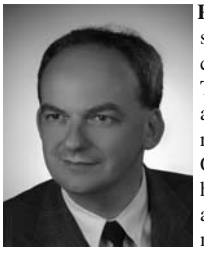
Vanstrum, M. and Starks, S. (1981). An algorithm for optimal linear maps, *Southeastcon Conference, Huntsville, AL, USA*, pp. 106–110.

Wand, M. and Jones, M. (1995). *Kernel Smoothing*, Chapman and Hall, London.

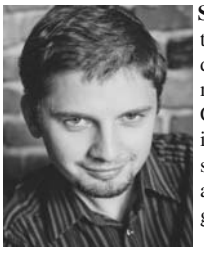
Wilson, D. and Martinez, T. (2000). Reduction techniques for instance-based learning algorithms, *Machine Learning* **38**(3): 257–286.

Xu, R. and Wunsch, D. (2009). *Clustering*, Wiley, Hoboken, NJ.

Zhigljavsky, A. and Žilinskas, A. (2008). *Stochastic Global Optimization*, Springer-Verlag, Berlin.



Piotr Kulczycki is a professor at the Systems Research Institute of the Polish Academy of Sciences and the head of its Center for Information Technology for Data Analysis Methods, as well as at the Cracow University of Technology, where he is the head of the Department of Automatic Control and Information Technology. The field of his scientific activity to date covers applicational aspects of information technology as well as data mining and analysis, mostly connected with the use of modern statistical methods and fuzzy logic in diverse issues of contemporary systems research and control engineering.



Szymon Łukasik is an assistant professor both at the Systems Research Institute of the Polish Academy of Sciences and at the Department of Automatic Control and Information Technology of the Cracow University of Technology. His scientific interests focus on various aspects of data analysis, nature-inspired algorithms, social networks and innovative applications of collective intelligence in distributed knowledge discovery.

Received: 6 April 2013
Revised: 10 November 2013