KYBERNETIKA — VOLUME 34 (1998), NUMBER 5, PAGES 515 - 534

DETECTION OF INFLUENTIAL POINTS BY CONVEX HULL VOLUME MINIMIZATION[1]

PETR TICHAVSKÝ AND PAVEL BOČEK

A method of geometrical characterization of multidimensional data sets, including construction of the convex hull of the data and calculation of the volume of the convex hull, is described. This technique, together with the concept of minimum convex hull volume, can be used for detection of influential points or outliers in multiple linear regression. An approximation to the true concept is achieved by ordering the data into a linear sequence such that the volume of the convex hull of the first n terms in the sequence grows as slowly as possible with n. The performance of the method is demonstrated on four well known data sets. The average computational complexity needed for the ordering is estimated by $O(N^{2+(p-1)/(p+1)})$ for large N, where N is the number of observations and p is the data dimension, i. e. the number of predictors plus 1.

1. INTRODUCTION

The purpose of the paper is to describe a method of geometrical characterization of multidimensional data sets by means of convex hulls and convex hull volumes. Recall that the convex hull of a set Y in the Euclidean space, denoted $\mathrm{conv}(Y)$, is the smallest convex set, in the sense of inclusion, that is spanned by Y. For a finite data set, the convex hull is a convex polytope with vertices, edges, and faces making up its boundary. The method is applied to the problem of detection of influential points or outliers in fitting the data by a linear model.

The need for an alternative way of characterization of data sets arises when the data length N is close to or smaller than $5p$, where p is the data dimension; it is a generally accepted fact that the standard statistical tools loose their capability to efficiently describe the data for such small N. The data geometry just represents more information that can be used in data analysis.

Geometrical properties of multivariate data sets, namely different kinds of ordering, have been studied recently by Barnett [2]. The idea of constructing the convex hull of the data has been introduced by Bebbington [3] in *convex peeling*; the convex peeling is an analogy to the trimmed mean in multiple dimensions.

[1]Supported by the Grant Agency of the Academy of Sciences of the Czech Republic through grant K 1075601.

The linear modeling problem can be formulated as follows: given N points $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^T$, $i = 1, \ldots, N$ in a p-dimensional space, find a hyperplane (i.e. an affine subspace of dimension $p-1$) that best fits (in some sense) some large subset of the given data. The data that excesively deviate from the model are called influential points or outliers. The main task addressed here is the detection of multiple outliers.

A closely related problem is that of *linear regression*, where one component of the data is taken as a dependent variable and the other components are called predictors. Examples of robust methods of the linear regression are the Least Median of Squares (LMS) and the Least Trimmed Squares (LTS) by Rousseeuw [19]. Efficient algorithms for calculating these estimators have been proposed by Boček and Lachout [4] and Hawkins [13], respectively. The breakdown point of these methods may reach almost 50 %, which means that the estimators are insensitive to the presence of up to almost one half of arbitrarily severe outliers in the data.

In cases when all components of data are equivalent and none of them is preferred to the others, different criteria and methods for fitting the data by a linear model have been proposed.

Rousseeuw [20] uses the minimum volume ellipsoid (MVE) covering a given number, at least one half, of the observations, to construct robust estimators. These estimators achieve the maximum possible breakdown point, almost 50 %, again. Since finding the exact MVE is extremely computationally demanding, several algorithms have been suggested for approximating the MVE, see e. g. Rousseeuw and Leroy [21], Rousseeuw and van Zomeren [22], and Hadi [12]. All these modifications, except for the algorithm by Hadi, are probabilistic, based on random drawing of p-member subsamples of the data, similar to the *elemental set* approach, treated by Hawkins et al. [14].

The algorithm proposed by Hadi is finite computationally feasible, having complexity of $O(N^2 \log N)$. It starts with finding an initial p-member subset of the data, which most likely does not contain outliers, and proceeds inductively by sequentially increasing the basic subset until some stopping criterion is met. Increasing the size of the basic subset is done by minimizing the Mahalanobis distance of the point $\boldsymbol{x}_i$ from the centre $\boldsymbol{c}_b$, relative to the measure of dispersion $\boldsymbol{S}_b$,

$$D_i(\boldsymbol{c}_b, \boldsymbol{S}_b) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{c}_b)^T \boldsymbol{S}_b^{-1} (\boldsymbol{x}_i - \boldsymbol{c}_b)}$$

where $\boldsymbol{c}_b$ and $\boldsymbol{S}_b$ are the mean and the covariance of the basic set. Although the algorithm seems to perform well in outlier detection, its interpretation is not straightforward.

The goal of this paper is to present a procedure for the outlier detection, which would have a transparent geometric interpretation, equivalent roles for all components of the data and which would be computationally feasible. The basic idea is to replace the concept of the MVE by the concept of the Minimum Convex Hull Volume (MCHV). The novel task is to find the subset of the given data that contains a given number (at least one half) of the points, whose convex hull has the minimum possible volume.

The concept of the MCHV is justified by the idea that the data satisfying the linear model should lie in the same hyperplane and hence they should span the convex

hull with zero volume. It follows that the data points that excessively increase the volume of the convex hull can be considered as outliers.

The determination of the MCHV has, however, at least the same computational complexity as the determination of the MVE. Hence, we suggest the following approximation, called simply CHV (Convex Hull Volume) method. Like the algorithm by Hadi, the procedure is based on sequential increasing of some basic set, which probably does not contain outliers at first. At each step, the point is appended to the sequence, that increases the convex hull by as little as possible. In this way, the whole data set is ordered into a sequence according to the order of entry of the points into the basic set. In addition, each point is characterized by the volume of the convex hull at the time instant of its inclusion. The possible outliers conclude the sequence and increase the convex hull volume significantly.

It is easy to see that scaling of the data along coordinate axes causes proportional changes of volumes of convex hulls of the data, but it does not affect positions of individual points relative to convex hulls of other data. It follows that both MCHV and CHV as methods of outlier detection are scale invariant.

Practical implementation of the algorithm CHV is closely related to the well known task for finding a convex hull of the given finite data set. Usually, this problem means determining the points that are vertices in the convex hull, possibly together with some ordering and adjacency information. The CHV algorithm requires further evaluation of the volumes of convex hulls, which is done by decomposition of the hull to simpler polytopes with $p+1$ vertices, so called simplexes. As a byproduct, the output of the algorithm includes a complete list of the faces of $\mathrm{conv}(X)$. The list of all vertices of $\mathrm{conv}(X)$ can be obtained simply as a union of the vertices of the faces.

In the literature two main algorithms have been recently proposed for solving the convex hull problem in an arbitrary dimension: the *gift-wrapping* method of Chand and Kapur [7] and the *beneath-beyond* method due to Seidel [25]. For a more general introduction see the books of Edelsbrunner [9] and Preparata and Shamos [17]. The algorithm described in this paper is a modification of the simpler gift-wrapping method. It is the subject of Section 2. In subsection 2.8 the computational load of the procedure is treated. The computational complexity is estimated at $O(N^{2+(p-1)/(p+1)})$ operations for large N using the assumption that the data are drawn uniformly and independently from a ball.

The selection of a suitable initial p-member subset of the data is hard problem, in general. If the number $N < 5p$, this problem appears to be almost as hard as the whole outlier detection. The main criterion for comparing different CHV runs with different initial sets is the volume of the convex hull of the first k points. Another criterion, showing how "flat" the initial set of k points (denoted X_k) is, is the ratio of the convex hull volume and the volume of the of the min–max parallelepiped

$$\prod_{\ell=1}^{p} \left[\max_{i \in X_k} x_{i\ell} - \min_{i \in X_k} x_{i\ell} \right].$$

A number of partially useful methods of constructing a suitable initial set is given in Section 3. Often, a successful approximation to the MCHV is found in a single run

of the CHV, as it is shown on numerical examples in Section 4. The performance of the CHV in outlier detection is compared to those of the LMS method.

2. ALGORITHM CHV

In this section, the structure of algorithm CHV is described in more details. For this purpose we use some terms known in linear algebra that are related to p-dimensional Euclidean or linear space, such as linear and affine subspace, dimension, scalar product, orthogonal and orthonormal basis, convex set, etc. We do not give exact definitions of these terms here for the sake of brevity, but refer the reader to a good introductory book into linear algebra.

The derivation of substantial part of the algorithm is based on the following assumption:

Assumption 1. Each $(p+1)$-tuple of a given set of points is affinely independent, i.e. it does not lie in one hyperplane (affine subspace of dimension $p-1$).

The assumption excludes some degenerate cases. It is very often fulfilled by real data sets. In subsection 2.7 it is shown how the assumption can be overcome by the conceptual perturbation technique of Edelsbrunner and Mücke [10].

The rest of the section is organized as follows: In the first subsection we introduce some necessary geometrical terms. The subsections 2.2 – 2.5 are devoted to solutions of subproblems, which represent the substantial parts of the algorithm. The frame of the CHV is a subject of subsection 2.6. In subsection 2.7 the algorithm complexity is discussed.

2.1. Definitions and notations

In this subsection we define the necessary geometrical terms required in the sequel. At the same time we specify representation of the terms in computer memory that is used for a practical implementation of the algorithm.

Point. Each point of the given set is represented by an integer denoting its order in the input list of the data. Points will be denoted by small latin letters, i, j, k, etc. We shall distinguish between a point, say i, and the coordinate vector $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^T$ that determines the position of this point in the space. Assume for simplicity that the given set of points is $X = \{1, 2, \ldots, N\}$.

Convex Hull of a set X of points is denoted $\mathrm{conv}(X)$. As mentioned above, for a finite set X it is a convex polytope. In our implementation, the convex hull is represented by a list of faces, see below.

Simplex is a convex hull of a $(p+1)$-member set of affinely independent points.

Vertex in a convex polytope P is a point that lies in the boundary of the polytope, so that is not a convex combination of two distinct points from P. If $P = \mathrm{conv}(X)$, it follows that each vertex of P is also a member of X.

Face of a convex polytope is a convex hull of at least p vertices of the polytope, that lies in one hyperplane and is a subset of a boundary of the polytope. It means that the whole polytope lies in one halfspace determined by the hyperplane. From Assumption 1 it easily follows that each face of conv(X) has exactly p vertices.

Also, in computer implementation of the CHV each face of the polytope can be represented by ordered p-tuple of points, e.g. $s = (i_1, \ldots, i_p)$, where $i_1 < \ldots < i_p$ or, like e.g. in PASCAL, by a (p-member) set of points.

Vector of Outer Normal of a face of a convex polytope is the unit normal vector $\boldsymbol{n}$ of the hyperplane that contains the face, and its orientation is directed out from the polytope. The last condition means that the scalar product $(\boldsymbol{n}, \boldsymbol{x}_a - \boldsymbol{x}_f)$ is negative for any points f from the face and a from the polytope, not lying in the face, where $\boldsymbol{x}_a$ and $\boldsymbol{x}_f$ are the coordinate vectors of a and f, respectively.

Adjacent Faces of a convex polytope are two faces that differ only in one vertex. It is easy to see that the remaining $p - 1$ vertices are common to exactly these two faces, unless the polytope is degenerate, $N \leq p$. The structure of the polytope boundary can be described by a face adjacence graph, where each face is represented by a node and adjacent faces are connected by edges. Assumption 1 implies that each node has exactly p neighbors.

Area of a face is the $(p-1)$-dimensional volume of the face. It is determined (recursively, after all) as a volume of a simplex in the $(p-1)$-dimensional space.

Volume of a Simplex. Obviously, any p-tuple of the vertices of the simplex, say $(i_1, \ldots, i_p)$, is a face of the simplex. The orthogonal distance of the remaining vertex, say h, from the face is called height of the simplex. It can be calculated as absolute value of the scalar product $(\boldsymbol{n}, \boldsymbol{x}_h - \boldsymbol{x}_s)$, where $\boldsymbol{n}$ is a unit normal vector of the face and $\boldsymbol{x}_s$ is an arbitrary vertex of the face. The volume of the simplex is equal to the product of the area of the face and the height, divided by the dimension of the space, p.

Volume of a Polytope is determined as a sum of volumes of a set of simplexes, that have pairwise disjunct interiors and their union is the whole polytope. The volume of the hull, as expected, does not depend on the particular decomposition of the polytope to individual simplexes.

2.2. Calculating volume of a simplex

In addition to calculating the volume of a simplex, the algorithm described in this subsection also computes the area and the vector of the outer normal of a face of a convex polytope.

Let $i_1, \ldots, i_{p+1}$ be $p+1$ affinely independent points in the space. If the intention is to calculate the area and the normal vector of the face $s = (i_1, \ldots, i_p)$, an arbitrary

point from the polytope, not lying in the face, can be taken for i_{p+1}. For simplicity of the notation, we shall denote the coordinate vectors of the points by $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{p+1}$.

The algorithm proceeds in p steps. Gramm-Schmidt's orthogonalization process is used to generate a sequence of unit vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_p$ so that $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k)$ is an orthonormal basis of the linear space spanned by $\boldsymbol{x}_2 - \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{k+1} - \boldsymbol{x}_1$ for $k = 1, \ldots, p$. At the same time a real sequence $\alpha_1, \ldots, \alpha_p$ is calculated, where α_k is the k-dimensional volume of the convex hull of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{k+1}$. Then, α_p represents the volume of the simplex spanned by all the points, α_{p-1} is the area of the face s and the unit vector of the outer normal of the face is given by $\boldsymbol{n} = -\boldsymbol{v}_p$.

Algorithm 2.2

Initial step

$$\boldsymbol{v}_1' := \boldsymbol{x}_2 - \boldsymbol{x}_1 \qquad \alpha_1 := \|\boldsymbol{v}_1'\| \qquad \boldsymbol{v}_1 := \boldsymbol{v}_1'/\|\boldsymbol{v}_1'\|;$$

Iteration: for $k = 2, \ldots, p$ **do**

$$\begin{aligned} \boldsymbol{v}_k' &:= (\boldsymbol{x}_{k+1} - \boldsymbol{x}_1) - \sum_{j=1}^{k-1} (\boldsymbol{x}_{k+1} - \boldsymbol{x}_1, \boldsymbol{v}_j)\, \boldsymbol{v}_j, \\ \alpha_k &:= \alpha_{k-1}\, \|\boldsymbol{v}_k'\|/k, \qquad \boldsymbol{v}_k := \boldsymbol{v}_k'/\|\boldsymbol{v}_k'\| \end{aligned}$$

endfor

It is easy to see that the length of $\boldsymbol{v}_k'$ is equal to the orthogonal distance of the point $\boldsymbol{x}_{k+1}$ from the affine space spanned by $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$ for $k = 1, \ldots, p$.

2.3. Finding one face of a convex hull

In this subsection the following problem is solved. Let X be a finite set of points containing at least $p+1$ members and let $i_1 \in X$ be a vertex of $\mathrm{conv}(X)$. The task is to find at least one face of $\mathrm{conv}(X)$ that contains the point i_1, say $s = (i_1, \ldots, i_p)$. In the sequel the notations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_p$ and $\boldsymbol{x}_i$ is used for coordinate vectors of the points $i_1, \ldots, i_p$ and for $i \in X$, respectively.

As a byproduct of the algorithm, the same sequence $\{\boldsymbol{v}_k\}$ as in Algorithm 2.2 is generated; also the sequence of volumes $\{\alpha_k\}$ can be calculated at the same time.

The point i_2 is selected by maximizing the angle φ_i between the vectors $\boldsymbol{x}_i - \boldsymbol{x}_1$ and $\boldsymbol{x}_0 - \boldsymbol{x}_1$, where $\boldsymbol{x}_0$ is the coordinate vector of an arbitrarily chosen point i_0 in X, $i_0 \neq i_1$. The situation is shown in Figure 1. The cosine of this angle, which is to be minimized, is calculated as a scalar product of the corresponding unit vectors.

For $k > 2$ the point i_k is chosen similarly, but instead of the differences $\boldsymbol{x}_i - \boldsymbol{x}_1$, $i \in X$ the calculation proceeds with orthogonal projections of these vectors (denoted below as $\boldsymbol{y}_i$) into orthogonal complement of the linear space spanned by $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{k-2}$. This projection has the property that the images of the up-to-time found points, $i_1, \ldots, i_{k-1}$, coincide. The projection of the vector $\boldsymbol{x}_0 - \boldsymbol{x}_1$ is denoted by $\boldsymbol{w}_0$. Finally, the normal vector of the face is calculated in the manner used in Algorithm 2.2.

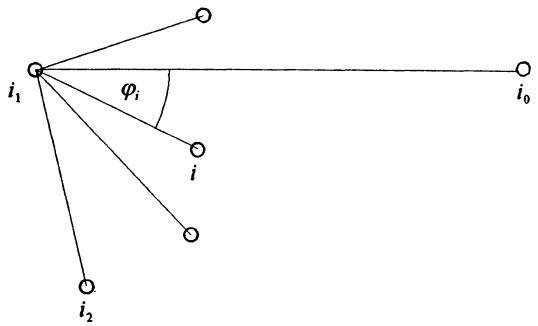

Fig. 1. Illustration of finding the point i_2 in Algorithm 2.3.

Algorithm 2.3

Initial step

$$\begin{array}{lllll} \textbf{Select} & i_0 & \in & X - \{i_1\} & \text{(arbitrarily)} \\ \textbf{Put} & \boldsymbol{w}_0 & := & \boldsymbol{x}_0 - \boldsymbol{x}_1 & \boldsymbol{w} := \boldsymbol{w}_0 / \|\boldsymbol{w}_0\| \\ & i_2 & := & \underset{i \in X - \{i_0, i_1\}}{\arg\min} \dfrac{(\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{w})}{\|\boldsymbol{x}_i - \boldsymbol{x}_1\|} & \\ & \boldsymbol{v}_1' & := & \boldsymbol{x}_2 - \boldsymbol{x}_1 & \boldsymbol{v}_1 := \boldsymbol{v}_1' / \|\boldsymbol{v}_1'\|; \end{array}$$

Iteration: if $p \geq 3$ then for $k = 3, \ldots, p$ do

$$\begin{array}{rcl} \boldsymbol{w}_0 & := & \boldsymbol{w}_0 - (\boldsymbol{w}_0, \boldsymbol{v}_{k-2})\, \boldsymbol{v}_{k-2} \qquad \boldsymbol{w} := \boldsymbol{w}_0 / \|\boldsymbol{w}_0\| \\ i_k & := & \underset{i \in X - \{i_0, \ldots, i_{k-1}\}}{\arg\min} \dfrac{(\boldsymbol{y}_i, \boldsymbol{w})}{\|\boldsymbol{y}_i\|}, \end{array}$$

where $$\boldsymbol{y}_i := (\boldsymbol{x}_i - \boldsymbol{x}_1) - \sum_{j=1}^{k-2} (\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{v}_j)\, \boldsymbol{v}_j, \quad \text{for } i \in X$$

$$\begin{array}{rcl} \boldsymbol{v}_{k-1}' & := & (\boldsymbol{x}_k - \boldsymbol{x}_1) - \sum_{j=1}^{k-2} (\boldsymbol{x}_k - \boldsymbol{x}_1, \boldsymbol{v}_j)\, \boldsymbol{v}_j, \\ \boldsymbol{v}_{k-1} & := & \boldsymbol{v}_{k-1}' / \|\boldsymbol{v}_{k-1}'\| \end{array}$$

endfor

Concluding step

$$\boldsymbol{w}_0 := \boldsymbol{w}_0 - (\boldsymbol{w}_0, \boldsymbol{v}_{p-1})\, \boldsymbol{v}_{p-1} \qquad \boldsymbol{n} := -\boldsymbol{w}_0 / \|\boldsymbol{w}_0\|$$

2.4. Finding an adjacent face

Assume that $s = (i_1, \ldots, i_p)$ is a face of a convex hull of the set X. It is already stated above that each face has p adjacent neighbors, each of which is obtained from s by interchanging one vertex by another, proper vertex of $\mathrm{conv}(X)$. Without any loss of generality we shall assume that the task is to find an adjacent face of $\mathrm{conv}(X)$, that differs from s in the last vertex, i_p, (otherwise first we change the order of the vertices in s).

The first step of the algorithm consists in finding an orthogonal projection operator into a two-dimensional plane, in which the images of the points $i_1, \ldots, i_{p-1}$ coincide. It is easy to see that the orthogonal projection is determined by two vectors: the vector of the unit normal of the face s, denoted $\boldsymbol{n}$, and a vector $\boldsymbol{v}$, that is equal to $\boldsymbol{v}_{p-1}$ obtained in Algorithm 2.2. It is obvious that the vector $\boldsymbol{v}$ depends on the vertex of s, which should be replaced.

The image of any point $i \in X$ has the coordinate vector

$$\boldsymbol{P}(i) = (\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{n})\,\boldsymbol{n} + (\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{v})\,\boldsymbol{v}$$

The new point i'_p that should replace i_p in the adjacent face is determined as the one that maximizes the angle β_i between the vectors $\boldsymbol{P}(i)$ and $\boldsymbol{v}$, $i \in X - \{i_1, \ldots, i_p\}$, see Figure 2.

It can be easily seen that cotangens of this angle, which is to be minimized, is equal to the ratio of the scalar products $-(\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{v})/(\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{n})$. Note that the product $(\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{n})$ is different from zero due to the fact that none other point from X lies in the hyperplane containing the face s with respect of Assumption 1.

Algorithm 2.4

1. **Find** the vectors $\boldsymbol{n}$ and $\boldsymbol{v} = \boldsymbol{v}_{p-1}$ according to Algorithm 2.2

2. **Find** the vertex i'_p as

$$i'_p := \operatorname*{arg\,max}_{i \in X - \{i_1, \ldots, i_p\}} \frac{(\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{v})}{(\boldsymbol{x}_i - \boldsymbol{x}_1, \boldsymbol{n})}$$

Note that the normal vector of the rotated face can be easily calculated as a linear combination of the vectors $\boldsymbol{v}$ and $\boldsymbol{n}$, more exactly by normalizing the vector

$$\boldsymbol{n}'' = -(\boldsymbol{x}'_p - \boldsymbol{x}_1, \boldsymbol{v})\,\boldsymbol{n} + (\boldsymbol{x}'_p - \boldsymbol{x}_1, \boldsymbol{n})\,\boldsymbol{v},$$

where $\boldsymbol{x}'_p$ is the coordinate vector of i'_p. It is obvious from Figure 2. Also, the area of the new face can be calculated in a less complex way than by calling a new the procedure in Algorithm 2.4.

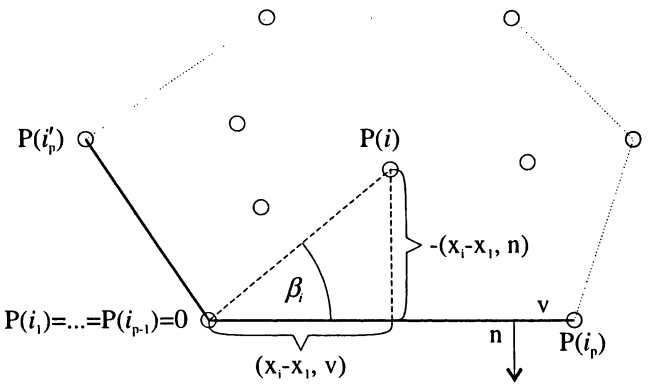

Fig. 2. Illustration of finding the adjacent face in Algorithm 2.4.

2.5. Walk through the whole adjacence graph

In this subsection we show how the Algorithms 2.3 and 2.4 can be used to generate a list of all faces of the polytope, which have one common vertex. The proposed approach is a variant of the so called width-first-search in the face adjacence graph. An easy modification of this algorithm allows to find *all* faces of the polytope: this is the subject of the "gift-wrapping" method, [7].

Below, the list of faces is referred as a sequence $L(1), \ldots, L(N_L)$ of sets of vertices of the faces, for simplicity. In a real computer implementation of the algorithm, however, it is convenient to organize the list in some finer data structure, which allows fast operations of search in the list and insertion of a new face.

For this purpose we consider a natural ordering of the faces in the list, e.g. lexicographic. If the list is stored as ordered and contains n items, then the search in the list can be performed in $O(\log n)$ operations, while inserting of one face requires $O(n)$ operations. Even more efficient data structures include binary trees and 2–3 trees [1], where the complexity of *both* of the operations is reduced to $O(\log n)$. A linear sequence $\{L(i)\}$ is used anyway, too, but has the meaning of pointers to faces in the list.

Further our algorithm employs an array $\ell(\cdot)$ of sets of points which is needed for storing the auxiliary information, which vertices of the sets $L(\cdot)$ remain to be considered for rotations later.

Let i_0 be the fixed vertex on the polytope boundary. The index of a currently processed face is denoted c.

Algorithm 2.5

Initialization: Use Algorithm 2.3 to find a face s that contains the point i_0.
Put $n := 1, \quad c := 1, \quad L(1) := s, \quad \ell(1) := s - \{i_0\}$.

While $n \geq c$ **do**

If $\ell(c) \neq \{\}$ **then for all** $k \in \ell(c)$ **do**

Rotate the face $L(c)$ by replacing the vertex k by a new one, k', according to Algorithm 2.4 and denote the new face as s.

Search for s in the currently existing list of faces. (It is possible to restrict the search to $\{L(c+1), \ldots, L(n)\}$.)

If s is a member of the list, i.e. $s = L(j)$ for some j, **then**

put $\ell(j) := \ell(j) - \{k'\}$

else add s into the current list,

$$n := n+1, \qquad L(n) := s, \qquad \ell(n) := s - \{k', i_0\}$$

endif

endfor

Put $c := c+1$

endwhile

The list of faces $\{L(\cdot)\}$ generated by Algorithm 2.5 has the property that any rotation of any face in the list, which is obtained by replacing a vertex $i \neq i_0$, leads to a face which is also a member of the list. Since the graph of adjacent faces is continuous (and finite), the list consists of all faces of the polytope which contain i_0 as required. Note that the list of *all* faces of the polytope could be obtained if the condition $i \neq i_0$ is relaxed.

2.6. The main algorithm CHV

Let X be a finite set of points satisfying Assumption 1 and let $i_1, \ldots, i_p$ be a p-member user selected subset of X. The task is to order all the points in X into a sequence $i_1, \ldots, i_N$, so that

$$\mathrm{vol}[\mathrm{conv}(X_{n+1})] = \min_{i \in X - X_n} \mathrm{vol}[\mathrm{conv}(X_n \cup \{i\})],$$

where $X_n = \{i_1, \ldots, i_n\}$, $n = p, \ldots, N-1$. (Note that the case when the starting set has more than p members can be solved similarly: an arbitrary p-member subset of the starting set is chosen and at the beginning of the procedure, first the remaining points from the starting set are appended to the sequence.)

Now, we show, how the volume of the polytopes $\mathrm{conv}(X_n \cup \{i\})$ can be calculated.

Let s be a face of $\mathrm{conv}(X_n)$, having the vector of the outer normal $\boldsymbol{n}_s$ and let $\boldsymbol{x}_s$ be coordinate vector of one (arbitrary) vertex of s. We say that the face s *looks* at the point i if

$$\omega_{s,i} \stackrel{\Delta}{=} (\boldsymbol{x}_i - \boldsymbol{x}_s, \boldsymbol{n}_s) > 0.$$

In the opposite case we say that the face s *does not see* the point i. The term $\omega_{s,i}$ has the meaning of the height of the simplex $\mathrm{conv}(s \cup \{i\})$ and thus it does not depend on the choice of the vertex $\boldsymbol{x}_s$ in the face. Assumption 1 assures that $\omega_{s,i}$ cannot be zero.

It is claimed that the polytope $\text{conv}(X_n \cup \{i\})$ can be written as a union of $\text{conv}(X_n)$ and all simplexes $\text{conv}(s \cup \{i\})$ where s looks at i. Recall from subsection 2.1 that the volume of the simplex $\text{conv}(s \cup \{i\})$ is equal to $\psi_s \omega_{s,i}/p$, where ψ_s is the area of the face s. Next, it is easy to see that the simplexes and $\text{conv}(X_n)$ have pairwise disjunct interiors. Collecting these facts results in a method of calculating volumes of convex polytopes.

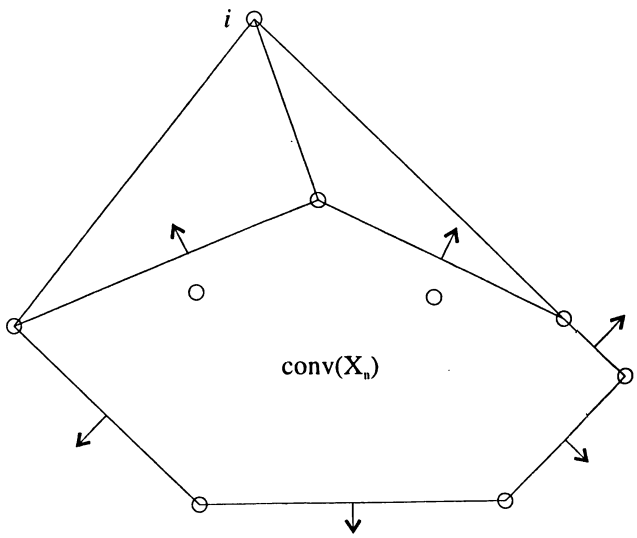

Fig. 3. Illustration of finding the convex hull of $X_n \cup \{i\}$.

Now, take note about the faces of the extended polytope $\text{conv}(X_{n+1})$=$\text{conv}(X_n \cup \{i_{n+1}\})$. They can be divided into two groups. The faces in the former group have common vertex i_{n+1}, the latter group contains exactly the faces of $\text{conv}(X_n)$ that do not see the point i_{n+1}. It is obvious from Figure 3. Using this fact and Algorithm 2.5, the list of all faces of $\text{conv}(X_n)$ can be recursively updated.

Let us introduce the notation

$$\gamma_n(i) = \text{vol}[\text{conv}(X_n \cup \{i\})] \qquad i \in X, \quad n = 1, \ldots, N$$

In computer implementation, each point is characterized by one scalar $\gamma(i)$, which is updated at each step, until it is appended to the sequence. Hence we shall skip the subscript n of $\gamma(i)$ in the algorithm. The output sequence of volumes is $\gamma(i_n) = \text{vol}[\text{conv}(X_n)]$, $n = 1, \ldots, N$. Obviously $\gamma(i_1) = \ldots = \gamma(i_p) = 0$ for the initial p points in the sequence because the convex hull of less than $p+1$ points has a zero volume.

The list of faces of $\text{conv}(X_n)$ considered in this subsection is somewhat different from the list in Algorithm 2.5. It need not be stored in a special data structure, but it should contain more information about the faces. Each face s is characterized by a triad $(i_s, \psi_s, \boldsymbol{n}_s)$ of one (arbitrary) vertex (its coordinate vector is denoted below as $\boldsymbol{x}_s$), the area, and the unit vector of the outer normal, respectively. At

the beginning, the list of faces is initialized by two items, both determined by the chosen subset of points, but differing in orientation of the normal vector.

The main procedure CHV is summarized as follows:

Algorithm 2.6

Initial step

Select (randomly, systematically or using an apriori information) an initial p-tuple $X_p = \{x_1, \ldots, x_p\}$ and denote it s.

Calculate ψ_s and n_s for the face s using Algorithm 2.2 and an arbitrary point $i_{p+1} \in X - s$. Initialize the list of faces by the items (i_1, ψ_s, n_s) and $(i_1, \psi_s, -n_s)$.

Put $\gamma(i) := 0$ for $i \in X_p$
$\gamma(i) := \psi_s \, |(x_i - x_1, n_s)|/p$ otherwise.

Iteration: for $n = p, \ldots, N-1$ **do**

Find $i_{n+1} := \underset{i \in X - X_n}{\operatorname{argmin}} \gamma(i)$ and **append** it to the resultant sequence.

Put $\Delta := \gamma(i_{n+1}) - \gamma(i_n)$

For all $i \in X - X_{n+1}$ **put** $\gamma(i) := \gamma(i) + \Delta$

For all faces $s = s(i_s, \psi_s, n_s)$ of $\operatorname{conv}(X_n)$ so that $(x_{n+1} - x_s, n_s) > 0$ **do**:

for all $i \in X - X_{n+1}$ calculate $\omega_{s,i} := (x_i - x_s, n_s)$.

If $\omega_{s,i} > 0$, then **put** $\gamma(i) := \gamma(i) - \psi_s \, \omega_{s,i}/p$

endfor

delete s from the list of faces.

endfor

Find all faces of $\operatorname{conv}(X_{n+1})$ containing i_{n+1} using Algorithm 2.5 and **add** them into the current list of faces. (After completing this step the list should contain all faces of $\operatorname{conv}(X_{n+1})$).

For each such a face $s = s(i_{n+1}, \psi_s, n_s)$ **do**:

for all $i \in X - X_{n+1}$ calculate $\omega_{s,i} := (x_i - x_{n+1}, n_s)$.

If $\omega_{s,i} > 0$ then **put** $\gamma(i) := \gamma(i) + \psi_s \, \omega_{s,i}/p$

endfor

endfor

endfor

Note that in general, each rotation (application of Algorithm 2.2) included in Algorithm 2.5 requires $O(N)$ operations. In Algorithm 2.6, however, it is possible to perform each call of the procedure more efficiently, as explained below.

Assume that in n–th step a face s of $\operatorname{conv}(X_{n+1})$ containing i_{n+1} should be rotated by replacing a point $i \neq i_{n+1}$ by a point i'. While in the original procedure the point i' is searched in the whole set $X_n - s$, here the search can be restricted to a smaller set, $\mathcal{N}_n - s$, where $\mathcal{N}_n$ is a set of vertices of $\operatorname{conv}(X_{n+1})$ adjacent to the

point i_{n+1}. More exactly, $\mathcal{N}_n$ is a set of all points $j \in X_n$ so that the connecting line $\text{conv}\{j, i_{n+1}\}$ lies on the boundary of $\text{conv}(X_{n+1})$; we say that (j, i_{n+1}) is an edge of the polytope. It can be easily seen that the set $\mathcal{N}_n$ can be obtained as the intersection

$$\mathcal{N}_n = \mathcal{P}_n \cap \mathcal{Q}_n,$$

where $\mathcal{P}_n$ is the union of vertices of all faces of $\text{conv}(X_n)$ that see i_{n+1}, and $\mathcal{Q}_n$ is the union of vertices of all faces of the polytope that do not look at i_{n+1}.

2.7. Coping with degenerate point sets

It has been shown recently by Edelsbrunner and Mücke [10] that the point sets that are not in the prescribed position can be (conceptually) perturbed to satisfy the general position assumption (here Assumption 1).

In the CHV algorithm (2.6), the event that Assumption 1 is not fulfilled is indicated by the result $\omega_{s,i} = 0$ for some face s and point i. The problem is solved by a small perturbation of one of coordinates of the point i immediately when the case occurs. The perturbation of the coordinate should lie somewhere between the precision of the processed data and the computer precision limit.

If such a shift can be performed, usually the involved loss of calculation precision is negligible. Otherwise we propose to assign two coordinate vectors to each point. The former coordinate vector is possibly perturbed and used for finding the adjacent faces, and the later vector is unperturbed and used for calculating the volumes of the simplexes.

2.8. Algorithm complexity

It is obvious that the algorithm complexity strongly depends on the number N_L of faces of the convex hull of the given data. In each of N steps, the CHV algorithm (i) updates the volumes $\gamma(i)$, $i = 1, \ldots, N$ in $O(N \cdot N_L)$ operations and (ii) updates the list of the faces of $\text{conv}(X_n)$. The update of the list of the faces is performed in $O(N \cdot N_L)$ operations, also, because there is $O(N_L)$ rotations and each of them requires $O(N)$ operations. We conclude that the algorithm complexity is $O(N^2 \cdot N_L)$.

It remains to estimate the number N_L. We start with the worst case. In 1971, P. McMullen [15] proved that the maximum number of faces of a polytope with N vertices in a p-dimensional space is

$$N_{L(\max)} = \begin{cases} \binom{N-m}{m} \frac{N}{N-m} & \text{for even } p, \quad p = 2m \\ 2\binom{N-m-1}{m} & \text{for odd } p, \quad p = 2m+1 \end{cases}$$

It follows that $N_{L(\max)} = O(N^{\lfloor p/2 \rfloor})$, where the brackets denote the (lower) integer part.

In spite of this unfavourable result, it appears that in most of the practical examples the encountered N_L is not so large; usually it is much smaller than $N_{L(\max)}$, see Example 1–4 in Section 4. In the sequel we explain this observation and present an alternative method for estimating N_L for given N and p.

Assume that the N points are chosen independently and uniformly at random from a p-dimensional ball. Then, the number of the faces of the convex hull of the point is a random variable, depending on the positions of the points. Consider the expected value of this variable, denoted $N_{L(\text{mean})}$, and study how the number depends on N and p.

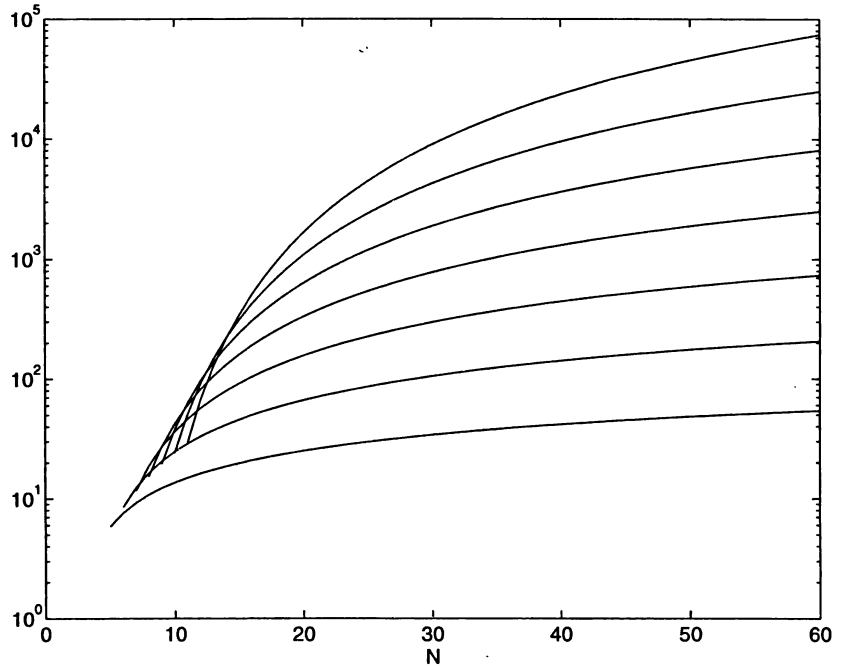

Fig. 4. Expected number of faces of a convex hull of N points, independently and uniformly drawn from a ball in the p-dimensional Euclidean space, as a function of N for $p = 3, \ldots, 9$ (from the bottom up), respectively.

An asymptotic expression for $N_{L(\text{mean})}$ for $N \to \infty$ has been derived first by H. Raynaud [18],

$$N_{L(\text{mean})} \approx 2 \frac{B_{p^2-1}}{B_{p^2}} \frac{\Gamma(\frac{p^2+1}{p+1})}{(p+1)!} \left[(p+1)\frac{B_p}{B_{p-1}}\right]^{\frac{p^2+1}{p+1}} N^{\frac{p-1}{p+1}}$$

where B_r is the volume of the unit ball in the r-dimensional space,

$$B_r = \frac{2}{r} \frac{\pi^{r/2}}{\Gamma(\frac{r}{2})}$$

In short, the result says that $N_{L(\text{mean})} = O(N^{(p-1)/(p+1)})$ for $N \to \infty$. It is in accordance with the idea that for large N a significant portion of the points lies in the interion of the convex hull.

The *exact* analytic expression for $N_{L(\text{mean})}$ for general N and p have been derived by C. Buchta and J. Müler [6]. Since the general formula for $N_{L(\text{mean})}$ is complex, we present only numerical results, plotted in Figure 4, for $p = 3, \ldots, 9$ and $N = p+2, \ldots, 60$.

It is worth noting that the expected number of the faces remains the same if the points are chosen at random, uniformly and independently, from a p-dimensional ellipsoid, instead of the ball. Alternatively, it may be assumed that the points are drawn at random with a p-dimensional Gaussian distribution. An expression for the expected number of the faces for the last case is due to Efron [11], but it involves calculation of complex multiple integrals. However, it is possible to guess that the expected number in this case is smaller than $N_{L(\text{mean})}$, because the points will be more concentrated near the center of the distribution, in the interior of the convex hull, and thus not contribute to the number of the faces.

3. APPROXIMATION TO THE MCHV

Consider the MCHV problem, minimization of the volume of convex hull of K from N observations, where $N/2 < K < N$. A satisfactory approximate solution to the problem is given by the CHV algorithm if the Kth volume $\gamma(K)$ is small compared to $\gamma(N)$ and small compared to volume of the corresponding min–max parallelpiped. It is a hard task to find such a solution in general, it may require multiple run of the CHV with various initial sets.

In this section we provide two ad hoc methods for finding the initial set that possibly allow to obtain a satisfactory solution to MCHV problem in a single run of the CHV.

The first method is due to Hadi [12], who solved the same problem (finding a p-member outlier-free subset of the data) for his algorithm by ordering the observations in ascending order according to the criterion $D_i(\boldsymbol{c}_b, \boldsymbol{S}_b)$ in Introduction, where $\boldsymbol{c}_b$ and $\boldsymbol{S}_b$ are replaced by some robust location and covariance matrix estimators. Then, the initial basic set is given by the first p points in the sequence.

Beside this, we propose still another simple procedure, which seeks for the starting set in the main cluster of the data.

Procedure 2. For each coordinate i, $i = 1, \ldots, p$, order the measurements with respect to the ith coordinates and assign to each point $k \in X$ its "distance" from the median according to

$$\kappa_{ik} = |o_i(k) - (N+1)/2|,$$

where $o_i(k)$ is the order of the point k in the sequence ordered with respect to the ith coordinates and N is the number of the data. For example, if N is odd, the middle point in the ordered sequence has the distance $\kappa_{ik} = 0$, its neighbors have $\kappa_{ik} = 1$, etc.

Then, the starting set X_p is selected by minimizing the criterion

$$\mathcal{V}(k) = \max_{i=1,\ldots,p} \kappa_{ik}.$$

The latter procedure is used in the following section.

4. NUMERICAL EXAMPLES

The algorithm CHV has been implemented on a PC, the source program is written in PASCAL. Bellow we list the computer results obtained for four very well known data sets, examined e. g. in [21]. For comparison, we calculated the LMS estimators, also, using the computational procedure developed by Boček and Lachout [4]. Then, the ordering of the points according to their absolute residuals with respect to the LMS linear model is considered.

Example 1: COLEMAN. (Sociometrical information on 20 schools from the Mid–Atlantic and New England States, [16]). The data have the dimension $p = 6$, and the length $N = 20$.

Procedure 2 gives the starting set $\mathcal{S}_1 = \{5, 8, 9, 14, 16, 18\}$. It has been found that the polytope conv(X) has 246 faces, the points no. 8 and 14 lie in the interior of the polytope, and the rest measurements are its vertices. The volume of conv(X) is 0.056 % of the volume of the min–max parallelpiped. The CHV output sequence starts with $\mathcal{S}_1$ and continues with $\{3, 6, 12, 4, 13, 7, 20, 19, 1, 17, 10, 11, 15, 2\}$. However, it appeared that starting set $\mathcal{S}_1$ is not optimum. For some other starting sets, e. g. for $\mathcal{S}_2 = \{6, 7, 9, 11, 15, 16\}$, the point no. 18 (member of $\mathcal{S}_1$) terminates the output sequences and thus can be considered for an outlier. Next, the LMS linear regression model, obtained by minimizing the $k = 13$th residual, results

$$\hat{y} = -0.3336\, x_1 + 0.0554\, x_2 + 0.6233\, x_3 + 0.9091\, x_4 - 2.0618\, x_5 + 21.8439$$

with

$$\hat{r}_{(13)} = 0.2926.$$

Again, the point no. 18 has the largest residual with respect to the model. All these facts show that Procedure 2 "failed" in this example. The sequences resulting from CHV$(\mathcal{S}_1)$, CHV$(\mathcal{S}_2)$ and from the LMS algorithm are listed in Table 1 (a). We conclude that only the points no. 18 and 3 can be considered for outliers.

Example 2: STACKLOSS. (The operation of a plant for the oxidation of ammonia to nitric acid, [5]), $p = 4$, $N = 21$.

Procedure 2 gives the starting set $\mathcal{S}_1 = \{5, 6, 9, 20\}$. The convex hull of the data has 66 faces, the points no. 5, 6 and 20 lie in the interior of the polytope, and the rest ones are the vertices. The volume of conv(X) is 2.33 % of the volume of the min–max parallelpiped. Note that since the data are integers, many 5-tuples of them lie in the same hyperplane and thus break Assumption 1, so that the perturbation technique have had to be used.

The LMS linear regression model is based on minimizing the $k = 12$th residual and gives

$$\hat{y} = 0.75\,x_1 + 0.4043\,x_2 - 0.0213\,x_3 - 35.4149$$

with

$$\hat{r}_{(12)} = 0.5319.$$

The results are listed in Table 1(b). The points no. 1, 2, 3, 4 and 21 seem to be outliers with respect to both of the criteria, CHV and LMS.

Table 1. The concluding part of the CHV and the LMS sequences with the corresponding convex hull volumes and the residuals, respectively.
(a) COLEMAN (b) STACKLOSS (c) SALINITY (d) WOOD.

n	CHV($\mathcal{S}_1$)		CHV($\mathcal{S}_2$)		LMS	
	i_n	$\gamma(i_n)$	i_n	$\gamma(i_n)$	i_n	res.
⋮	⋮	⋮	⋮	⋮	⋮	⋮
12	7	57.5	14	23.2	5	0.293
13	20	62.6	5	33.1	19	0.293
14	19	79.4	8	43.2	1	−1.064
15	1	117.8	13	61.3	4	−1.421
16	17	183.0	12	97.0	10	1.668
17	10	275.0	17	146.2	12	1.929
18	11	441.1	10	225.9	17	−2.570
19	15	545.3	3	377.3	3	− 4.735
20	2	715.4	18	715.4	18	6.467

(a)

n	CHV		LMS	
	i_n	$\gamma(i_n)$	i_n	res.
⋮	⋮	⋮	⋮	⋮
13	17	374.1	8	1.192
14	7	483.2	14	−1.787
15	8	545.0	20	2.075
16	13	693.3	13	−2.617
17	21	1291	2	3.372
18	4	2319	3	7.974
19	3	3636	21	−8.234
20	1	4436	1	8.394
21	2	5128	4	9.064

(b)

n	CHV		LMS	
	i_n	$\gamma(i_n)$	i_n	res.
⋮	⋮	⋮	⋮	⋮
20	13	73.09	9	1.73
21	28	80.80	1	−1.87
22	6	95.10	11	2.00
23	4	111.3	10	2.16
24	17	135.6	8	−2.74
25	3	159.4	24	4.21
26	15	197.9	23	5.02
27	5	305.8	5	5.33
28	16	500.9	16	13.82

(c)

n	CHV		LMS	
	i_n	$\gamma(i_n)$	i_n	res.
⋮	⋮	⋮	⋮	⋮
12	11	$4.68 \cdot 10^{-10}$	2	0.0042
13	14	$1.08 \cdot 10^{-9}$	13	0.0042
14	12	$1.70 \cdot 10^{-9}$	1	0.0134
15	5	$2.72 \cdot 10^{-9}$	7	0.0154
16	7	$4.25 \cdot 10^{-9}$	5	0.0178
17	4	$2.73 \cdot 10^{-8}$	4	−0.1887
18	6	$3.54 \cdot 10^{-8}$	8	−0.2152
19	8	$4.24 \cdot 10^{-8}$	6	−0.2187
20	19	$4.94 \cdot 10^{-8}$	19	−0.2530

(d)

Example 3: SALINITY. (Measurements of water salinity, [23]), $p = 4$, $N = 28$.

Procedure 2 gives the starting set $\mathcal{S}_1 = \{12, 18, 20, 25\}$. The convex hull of the data has 103 faces, the points no. 12, 25 and 26 lie in the interior of the polytope, and the rest ones are the vertices. The volume of $\mathrm{conv}(X)$ is 6.84 % of the volume of the min–max parallelpiped.

The LMS linear regression model is based on minimizing the $k = 16$th residual and gives

$$\hat{y} = 0.3618\, x_1 - 0.0863\, x_2 - 1.3267\, x_3 + 37.3671$$

with

$$\hat{r}_{(16)} = 0.3146.$$

The results are listed in Table 1 (c). Only the points 5 and 16 seem to be far deteriorated from the linear model.

Example 4: WOOD. (Anatomical factors on wood specific gravity; the data are taken from [8] and modified by Rousseeuw [19], who planted four outliers at cases 4, 6, 8 and 19), $p = 6$, $N = 20$.

Procedure 2 gives the starting set $\mathcal{S}_1 = \{2, 5, 9, 11, 14, 15\}$. The convex hull of the data has 346 faces, all the 20 points have been found on the boundary of the polytope. The volume of the convex hull is approximately 19 % of the volume of the min–max parallelpiped.

The LMS estimation, consisting in minimization of the 13th absolute residual with respect to the regression model, gives

$$\hat{y} = 0.2355x_1 + 0.0460x_2 - 0.5747x_3 - 0.3668x_4 + 0.6264x_5 + 0.3185,$$

where

$$\hat{r}_{(13)} = 0.0042.$$

The results are listed in Table 1 (d). We note that all of the four outliers, 4, 6, 8, and 19, were detected by both of the criteria.

5. CONCLUSIONS

The CHV algorithm is a novel heuristic method for outlier detection in the multivariate linear regression, based on geometry of data sets. The method may also be regarded as a special case of a clustering algorithm. The average computational load of the procedure for large data samples is $O(N^{2+(p-1)/(p+1)})$.

The numerical examples show that the CHV method detects the same outliers as the LMS in the examined cases, but it assigned different weights to the outliers.

ACKNOWLEDGEMENT

The authors wish to thank Doc. Dr. J. Á. Víšek for his helpful suggestions.

(Received February 7, 1997.)

REFERENCES

[1] A. V. Aho, J. E. Hopcroft and J. D. Ullman: The Design and Analysis of Computer Algorithms. Addison–Wesley, Reading, MA 1974.

[2] V. Barnett: The ordering of multivariate data. J. Roy. Statist. Soc. Ser. A *139* (1976), 318–344.

[3] A. C. Bebbington: A method of bivariate trimming for robust estimation of the correlation coefficient. Appl. Statist. *27* (1978), 221–226.

[4] P. Boček and P. Lachout: Linear programming approach to LMS–estimation. Comput. Statist. Data Anal. *19* (1995), 129–134.

[5] K. A. Brownlee: Statistical Theory and Methodology in Science and Engineering. Second edition. Wiley, New York 1965.

[6] C. Buchta and J. Müler: Random polytopes in a ball. J. Appl. Probab. *21* (1984), 753–762.

[7] D. R. Chand and S. S. Kapur: An algorithm for convex polytopes. J. Assoc. Computer Mach. *17* (1970), 78–86.

[8] N. R. Draper and H. Smith: Applied Regression Analysis. Wiley, New York 1966.

[9] H. Edelsbrunner: Algorithms in Combinatorical Geometry. Springer–Verlag, Heidelberg 1987.

[10] H. Edelsbrunner and Mücke: Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. ACM Trans. Graphics *9* (1990), 66–104.

[11] B. Efron: The convex hull of a random set of points. Biometrika *52* (1965), 331–345.

[12] A. S. Hadi: Identifying multiple outliers in multivariate data. J. Roy. Statist. Soc. Ser. B *54* (1992), 761–771.

[13] D. M. Hawkins: The feasible solution algorithm for least trimmed squares regression. Computat. Statist. Data Anal. *17* (1994), 185–196.

[14] D. M. Hawkins, D. Bradu and G. V. Kass: Location of several outliers in multiple–regression data using elemental sets. Technometrics *26* (1984), 197–208.

[15] P. McMullen: The maximum numbers of faces of convex polytope. Mathematika *17* (1971), 179–184.

[16] F. Mosteller and J. W. Tukey: Data Analysis and Regression. Addison–Wesley, Reading, MA 1977.

[17] F. P. Preparata and M. I. Shamos: Computational Geometry — An Introduction. Springer–Verlag, New York 1985.

[18] H. Raynaud: Sur l'enveloppe convexe des nuages de points aléatoires dans R^n. J. Appl. Probab. *7* (1970), 35–48.

[19] P. J. Rousseeuw: Least median of squares regression. J. Amer. Statist. Assoc. *79* (1984), 871–880.

[20] P. J. Rousseeuw: Multivariate estimation with high breakdownpoint. In: Mathematical Statistics and Applications (W. Grossmann, G. Pflug, I. Vincze and W. Wertz, eds.). Reidel, Dordrecht 1985, vol. B, pp. 283–297.

[21] P. J. Rousseeuw and A. M. Leroy: Robust Regression and Outlier Detection. Wiley, New York 1987.

[22] P. J. Rousseeuw and B. C. van Zomeren: Unmasking multivariate outliers and leverage points (with comments). J. Amer. Statist. Assoc. *85* (1990), 633–651.

[23] D. Ruppert and R. J. Carroll: Trimmed least squares estimation in the linear model. J. Amer. Statist. Assoc. *75* (1980), 828–835.

[24] R. Schneider: Discrete aspects of stochastic geometry. In: Handbook of Discrete and Computational Geometry (J. Goodman and J. O'Rourke, eds.). CRC Press, Boca Raton 1997.

[25] R. Seidel: Constructing higher–dimensional convex hull at logarithmic cost per face. In: Proc. 18th Annual ACM Symposium on Theory Computing, Association for Computing Machinery, New York 1986, pp. 404–413.

Ing. Petr Tichavský, CSc. and Mgr. Pavel Boček, Institute of Information Theory and Automation – Academy of Sciences of the Czech Republic, Pod vodárenskou věží 4, 182 08 Praha 8. Czech Republic.

e-mails: tichavsk@utia.cas.cz, bocek@utia.cas.cz