Basis exchange and learning algorithms for extracting collinear patterns

Leon Bobrowski\textsuperscript{a,b} and Paweł Zabielski\textsuperscript{a}

\textsuperscript{a}Faculty of Computer Science, Białystok University of Technology, Białystok, Poland; \textsuperscript{b}Institute of Biocybernetics and Biomedical Engineering, PAS, Warsaw, Poland

ABSTRACT

Understanding large data sets is one of the most important and challenging problem in modern days. Exploration of genetic data sets composed of high-dimensional feature vectors can be treated as a leading example in this context. A better understanding of large, multivariate data sets can be achieved through exploration and extraction of their structure. Collinear patterns can be an important part of a given data set structure. Collinear (flat) patterns exist in a given set of feature vectors when many of these vectors are located on (or near) some planes in the feature space. Discovered flat patterns can reflect various types of interaction in an explored data set. The presented paper compares basis exchange algorithms with learning algorithms in the task of flat patterns extraction.

1. Introduction

Large data sets might contain very useful information. Models and computational tools for new knowledge extraction from large data sets are constantly being developed within pattern recognition or data mining methods (Duda, Hart, & Stork, 2001; Hand, Smyth, & Mannila, 2001). Along with the rapidly increasing size of databases grows interest related to extraction of knowledge, which can be really useful (Jaseena & Julie, 2014; Julie & Kannan, 2011). However, the discovery of dependencies is not so easy and there are a lot of challenges associated with mining Big Data (Albert, 2013). The contemporary problem of understanding large data sets is crucial for many practical issues (Ian & Eibe, 2005). Better understanding of genetic data sets composed of high-dimensional feature vectors is a leading example in this context (Wong, 2016). Research conducted in that way allows one to prepare for better diagnosis, prognosis, and therapies (Clarke et al., 2008).

We assume that explored data sets are composed of feature vectors which represent particular objects (events, phenomena) in a standardized way in a particular feature space. Data mining tools are used for discovering useful patterns in large data sets (Bobrowski, 2005). These tools let us analyse the data in some automatic way (Sun-Mi &
The term pattern denotes a subset (cluster) of feature vectors characterized by a certain type of regularity (Duda et al., 2001). More specifically, the term collinear (flat) pattern means a large number of feature vectors situated on some planes in feature space. Algorithms presented in this article could find some dependencies in multidimensional data and in results return pattern from given data sets.

Flat patterns can be extracted from large data sets through minimization of a special type of the convex and piecewise linear (CPL) criterion function (Bobrowski, 2005). Basis exchange algorithms allow one to efficiently find the global minimum of the CPL criterion function even in the case of large, multidimensional data sets. Learning algorithms provide another possibility of minimizing CPL functions (Bobrowski & Zabielski, 2016b).

This paper is an extended version of the ACIIDS 2016 paper (Bobrowski & Zabielski, 2016b). The authors describe another additional algorithm and the possibility of discovering collinear patterns based on basis exchange. Furthermore, the comparison of a selected learning algorithm with a basis exchange algorithm in the task of flat patterns discovering is demonstrated. The results of the experimental comparison of these two types of algorithms are discussed.

2. Dual hyperplanes and vertices in the parameter space

Let us consider the data set \( C \) composed of \( m \) feature vectors \( \mathbf{x}_j = [x_{j,1}, \ldots, x_{j,n}]^T \):

\[
C = \{ \mathbf{x}_j \} \quad \text{where} \quad j = 1, \ldots, m. \tag{1}
\]

Feature vectors \( \mathbf{x}_j \) belong to the \( n \)-dimensional feature space \( F[n] \) (\( \mathbf{x}_j \in F[n] \)). Components \( x_{ji} \) of the \( j \)th feature vector \( \mathbf{x}_j \) can be treated as numerical results of \( n \) standardized examinations of the \( j \)th object \( O_j \), where \( x_{ji} \in \mathbb{R} \) or \( x_{ji} \in \{0,1\} \).

Each of \( m \) feature vectors \( \mathbf{x}_j \) from the set \( C \) (1) defines the following dual hyperplane \( h_j \) in the parameter space \( \mathbb{R}^n \) (\( \mathbf{w} \in \mathbb{R}^n \)) (Kushner & Yin, 1997):

\[
(\forall \mathbf{x}_j \in C) \quad h_j = \{ \mathbf{w} : \mathbf{x}_j^T \mathbf{w} = 1 \}, \tag{2}
\]

where \( \mathbf{w} = [w_1, \ldots, w_n]^T \) is the parameter (weight) vector (\( \mathbf{w} \in \mathbb{R}^n \)).

Unit vectors \( \mathbf{e}_i = [0, \ldots, 1, \ldots, 0]^T \) define the below hyperplanes \( h_0^i \) in the \( n \)-dimensional parameter space \( \mathbb{R}^n \) []:

\[
(\forall i \in \{1, \ldots, n\}) \quad h_0^i = \{ \mathbf{w} : \mathbf{e}_i^T \mathbf{w} = 0 \} = \{ \mathbf{w} : w_i = 0 \}. \tag{3}
\]

Consider the set \( S_k \) of \( n \) linearly independent feature vectors \( \mathbf{x}_j \) (1) and unit vectors \( \mathbf{e}_i \):

\[
S_k = \{ \mathbf{x}_j : j \in J_k \} \cup \{ \mathbf{e}_i : i \in I_k \}. \tag{4}
\]

The set \( S_k \) contains \( r_k \) feature vectors \( \mathbf{x}_j \) (\( j \in J_k \)) and \( n - r_k \) unit vectors \( \mathbf{e}_i \) (\( i \in I_k \)).

The \( k \)th vertex \( \mathbf{w}_k = [w_{k,1}, \ldots, w_{k,n}]^T \) in the parameter space \( \mathbb{R}^n \) is described on the basis of the set \( S_k \) (4) as the intersection point of the \( r_k \) hyperplanes \( h_j \) (2) defined by the feature vectors \( \mathbf{x}_j \) (\( j \in J_k \)) and the \( n - r_k \) hyperplanes \( h_0^i \) (3) defined by the unit vectors \( \mathbf{e}_i \) (\( i \in I_k \)). The vertex \( \mathbf{w}_k \) can be defined by the following linear equations:

The vertex \( \mathbf{w}_k \) can be defined by the below set of \( n \) linear equations:

\[
(\forall j \in J_k) \quad \mathbf{w}_k^T \mathbf{x}_j = 1 \tag{5}
\]
and

\[(\forall i \in I_k) \quad w_k^T e_i = 0. \quad (6)\]

Equations (5) and (6) can be represented as the matrix equation:

\[B_k w_k = 1' = [1, \ldots, 1, 0, \ldots, 0]^T, \quad (7)\]

where the square, nonsingular matrix \(B_k\) is the \(k\)th basis linked to the vertex \(w_k\):

\[B_k = [x_{j(1)}, \ldots, x_{j(ik)}, e_{l(ik+1)}, \ldots, e_{l(n)}]^T. \quad (8)\]

The feature vector \(x_{j(l)} (j(l) \in J_k)\) constitutes the \(l\)th row of the matrix \(B_k\). The vertex \(w_k\) can be computed in the below manner (7):

\[w_k = B_k^{-1} 1'. \quad (9)\]

The below three definitions are taken from the work of Bobrowski (2014):

**Definition 1:** The rank \(r_k\) (1 ≤ \(r_k\) ≤ \(n\)) of the \(k\)th vertex \(w_k = [w_{k,1}, \ldots, w_{k,n}]^T \quad (9)\) is defined as the number of the non-zero components \(w_{k,j}\) (\(w_{k,j} \neq 0\)).

**Definition 2:** The degree of degeneration \(d_k\) of the vertex \(w_k\) (9) of the rank \(r_k\) is defined as the number \(d_k = m_k - r_k\), where \(m_k\) is the number of such feature vectors \(x_j\) from the set \(C\) (1), which define the hyperplanes \(h_k^1\) (2) passing through this vertex \((w_k^T x_j = 1)\). The vertex \(w_k\) (9) is degenerated if the degree of degeneration \(d_k\) is greater than zero \((d_k > 0)\).

**Definition 3:** The basis \(B_k\) (8) is regular if and only if the number \(n - r_k\) of the unit vectors \(e_i\) \((i \in I_k\) (6)) in the matrix \(B_k\) is equal to the number of such components \(w_{k,j}\) of the vector \(w_k = [w_{k,1}, \ldots, w_{k,n}]^T \quad (9)\) which are equal to the zero \((w_{k,j} = 0)\).

### 3. Vertexal planes and collinear patterns in feature space

The hyperplane \(H(w, \theta)\) in the feature space \(F[n]\) is defined as follows (Bobrowski, 2005):

\[H(w, \theta) = \{x : w^T x = \theta\}, \quad (10)\]

where \(w\) is the weight vector \((w \in R^n)\) and \(\theta\) is the threshold \((\theta \in R^1)\).

The \(k\)th vertex \(w_k\) (9) of the rank \(r_k\) \((r_k > 1)\) (Definition 1) allows us to define the \((r_k - 1)\)-dimensional vertexal plane \(P_k(x_{j(1)}, \ldots, x_{j(ik)})\) in the feature space \(F[n]\) as the linear combination of \(r_k\) linearly independent feature vectors \(x_{j(i)}\) \((j(i) \in J_k\) (5)) from the set \(S_k\) (4) []:

\[P_k(x_{j(1)}, \ldots, x_{j(ik)}) = \{x \in F[n] : x = \alpha_1 x_{j(1)} + \ldots + \alpha_k x_{j(ik)}\}, \quad (11)\]

where \(j(i) \in J_k\) (5) and the parameters \(\alpha_i\) \((\alpha_i \in R^1)\) satisfy the following condition:

\[\alpha_1 + \ldots + \alpha_k = 1. \quad (12)\]

Two linearly independent vectors \(x_{j(1)}\) and \(x_{j(2)}\) from the set \(C\) (1) support the below straight line \(l(x_{j(1)}, x_{j(2)})\) in the feature space \(F[n]\) \((x \in F[n])\):

\[l(x_{j(1)}, x_{j(2)}) = \{x : x = x_{j(1)} + \alpha(x_{j(2)} - x_{j(1)})\} = \{x : x = (1 - \alpha)x_{j(1)} + \alpha x_{j(2)}\}, \quad (13)\]

where \(\alpha \in R^1\).
The straight line \( l(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) \) (13) can be treated as the vertexical plane \( P_k(\mathbf{x}_{(1)}, \mathbf{x}_{(2)}) \) (11) spanned by two supporting vectors \( \mathbf{x}_{(1)} \) and \( \mathbf{x}_{(2)} \) with \( \alpha_1 = 1 - \alpha \) and \( \alpha_2 = \alpha \). In this case, the regular basis \( B_k \) (8) contains only two basis feature vectors \( \mathbf{x}_{(1)} \) and \( \mathbf{x}_{(2)} \) \( (r_k = 2) \) and \( n - 2 \) unit vectors \( \mathbf{e}_i \) \( (i \in I_k) \). As a result, the vertex \( \mathbf{w}_k = [w_{k1}, \ldots, w_{kn}]^T \) (9) contains only two non-zero components \( w_{k,j} (w_{k,j} \neq 0) \).

**Lemma 1:** The vertexical plane \( P_k(\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(r_k)}) \) (11) based on the vertex \( \mathbf{w}_k \) (9) with the rank \( r_k \) greater than 1 \( (r_k > 1) \) is equal to the hyperplane \( H(\mathbf{w}_k, 1) \) (10) in the \( n \)-dimensional feature space \( F[n] \) defined by this vertex.

**Theorem 1:** The \( j \)th feature vector \( \mathbf{x}_j \) \( (\mathbf{x}_j \in C(1)) \) is located on the vertexical plane \( P_k(\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(r_k)}) \) (11) if and only if the \( j \)th dual hyperplane \( h_j^1 \) (2) passes through the vertex \( \mathbf{w}_k \) (9) of the rank \( r_k \).

The proof of the above lemma and theorem was provided in the article by Bobrowski (2014).

**Definition 4:** The collinear (flat) pattern \( \mathbf{F}_k \) of the dimension \( r_k - 1 \) in the feature space \( F[n] \) is formed by \( m_k \) \( (m_k > r_k) \) feature vectors \( \mathbf{x}_j \) located on the vertexical plane \( P_k(\mathbf{x}_{(1)}, \ldots, \mathbf{x}_{(r_k)}) \) (11). Lemma 1 shows that the flat pattern \( \mathbf{F}_k \) is based on the vertex \( \mathbf{w}_k \) (9):

\[
\mathbf{F}_k = \{ \mathbf{x}_j : \mathbf{w}_k^T \mathbf{x}_j = 1 \}. \tag{14}
\]

The flat pattern \( \mathbf{F}_k \) of the dimension \( r_k - 1 \) is composed of such \( m_k \) feature vectors \( \mathbf{x}_j \) that the dual hyperplanes \( h_j^1 \) (2) pass through the vertex \( \mathbf{w}_k \) (9) of the rank \( r_k \). The \( k \)th vertex \( \mathbf{w}_k = [w_{k1}, \ldots, w_{kn}]^T \) (9) has \( n - r_k \) components \( w_{k,j} \) equal to zero:

\[
(\forall i \in I_k(4)) w_{k,j} = 0. \tag{15}
\]

### 4. CPL collinearity functions

Let us define the collinearity penalty functions \( \varphi_j^c(\mathbf{w}) \) linked to feature vectors \( \mathbf{x}_j \) (1) (Figure 1):

\[
(\forall \mathbf{x}_j \in C(1)),
\]

\[
1 - \varepsilon - \mathbf{x}_j^T \mathbf{w} < 1 - \varepsilon,
\]

\[
\varphi_j^c(\mathbf{w}) = 0 \quad \text{if} \quad 1 - \varepsilon \leq \mathbf{x}_j^T \mathbf{w} \leq 1 + \varepsilon,
\]

\[
\mathbf{x}_j^T \mathbf{w} - 1 + \varepsilon \quad \text{if} \quad \mathbf{x}_j^T \mathbf{w} > 1 - \varepsilon,
\]

where \( \varepsilon \geq 0 \) is the margin \( (\varepsilon \in R^1) \).

**Remark 1:** The collinearity penalty functions \( \varphi_j^c(\mathbf{w}) \) (16) is equal to zero if and only if the distance between point \( \mathbf{w} \) and the dual hyperplane \( h_j \) (2) is not greater than \( \varepsilon / ||\mathbf{x}_j||\):

\[
(\forall \mathbf{x}_j \in C(1)) \quad \text{if} \quad ||\mathbf{x}_j^T \mathbf{w} - 1|| \leq \varepsilon \quad \text{then} \quad \varphi_j^c(\mathbf{w}) = 0. \tag{17}
\]

The \( k \)th collinearity criterion function \( \Phi_k^c(\mathbf{w}) \) is defined here as the sum of the penalty functions \( \varphi_j^c(\mathbf{w}) \) (16) linked to feature vectors \( \mathbf{x}_j \) from the subset \( C_k \) \( (C_k \subset C(1)) \):

\[
\Phi_k^c(\mathbf{w}) = \sum_{j \in A_k} \beta_j \varphi_j^c(\mathbf{w}), \tag{18}
\]
where $J_k = \{j : x_j \in C_k \subseteq C(1)\}$ and the positive parameters $\beta_j (\beta_j > 0)$ can be treated as the prices of particular feature vectors $x_j$. The standard choice of the parameters $\beta_j$ values is one ($\forall j \in \{1, \ldots, m\} \beta_j = 1.0$).

5. The minimal value of the collinearity criterion function

The collinearity criterion function $\Phi^c_k(w)$ (18) like the penalty functions $\psi^c_k(w)$ (16) are $\text{CPL}$ [].

It has been proved that the minimal value $\Phi^c_k(w^*_k)$ of the $\text{CPL}$ criterion function $\Phi^c_k(w)$ (18) can be found in one of the vertices $w_k$ (9): $\exists \exists w^*_k (\forall w) \Phi^c_k(w) \geq \Phi^c_k(w^*_k) \geq 0$. (19)

Let us introduce the below layer $L^{\varepsilon'}(w, \theta)$ in the feature space $F[n]$ which is based on the hyperplane $H(w, \theta)$ (10):

$$L^{\varepsilon'}(w, \theta) = \{x : \theta - \varepsilon' \leq w^T x \leq \theta + \varepsilon'\},$$

(20)

where $\varepsilon' \geq 0$ is the margin ($\varepsilon' \in R^1$).

The layer $L^{\varepsilon'}(w, \theta)$ (20) contains such feature vectors $x$ for which the distance from the hyperplane $H(w, \theta)$ (10) is no greater than $\varepsilon'/||w||$.

**Theorem 2:** If all the feature vectors $x_j$ from the subset $C_k (C_k \subseteq C(1))$ are located inside the layer $L^{\varepsilon'}(w, \theta)$ (20) with $\theta \neq 0$, then the minimal value $\Phi^c_k(w^*_k)$ (19) of the collinearity criterion functions $\Phi^c_k(w)$ (18) is equal to zero for the margin $\varepsilon$ (19) no less than $\varepsilon'/||\theta||$ ($\varepsilon \geq \varepsilon'/||\theta||$).

**Proof:** If the vector $x_j$ from the subset $C_k$ is located inside the layer $L^{\varepsilon'}(w, \theta)$ (23), then

$$\theta - \varepsilon' \leq w^T x_j \leq \theta + \varepsilon'.$$

(21)

Let us suppose that $\theta > 0$. Then

$$1 - \varepsilon'/\theta \leq (w/\theta)^T x_j \leq 1 + \varepsilon'/\theta.$$  

(22)

In accordance with the definition (16) of the collinearity penalty function $\psi^c_j(w)$:

$$\psi^c_{\varepsilon'/\theta}(w/\theta) = 0.$$  

(23)

Because all the feature vectors $x_j$ from the subset $C_k$ are located inside the layer $L^{\varepsilon'}(w, \theta)$
with \( \theta > 0 \), then (18):

\[
\Phi_k^{e/\theta}(\mathbf{w}/\theta) = 0.
\]  

(24)

The last equation means that the minimal value of the collinearity criterion functions \( \Phi_k^e(\mathbf{w}) \) (21) with the margin \( \varepsilon = \varepsilon' / \theta \) is equal to zero in the point \( \mathbf{w}^* = \mathbf{w} / \theta \).

Let us suppose now that \( \theta < 0 \). By dividing both sides of the inequality (25) by the negative number \( \theta \), we get the below result:

\[
1 + \varepsilon' / |\theta| \geq (\mathbf{w}/|\theta|)^T \mathbf{x}_j \geq 1 - \varepsilon' / |\theta|.
\]  

(25)

As a result, we obtain the below equation similar to (24):

\[
\Phi_k^{e/|\theta|}(\mathbf{w}/|\theta|) = 0
\]  

(26)

The minimal value \( \Phi_k^e(\mathbf{w}_k^*) \) (19) of the collinearity criterion function \( \Phi_k^e(\mathbf{w}) \) (18) with \( \varepsilon = \varepsilon' / |\theta| \) is equal to zero. An increase in the margin \( \varepsilon \) in the penalty functions \( \Phi_k^e(\mathbf{w}) \) (16) allows us to preserve the location of all vectors \( \mathbf{x}_j \) in the layer \( \mathbf{L}^e(\mathbf{w}, \theta) \) (20). So, the minimal value \( \Phi_k^e(\mathbf{w}_k^*) \) (19) remains equal to zero.

It can be proved that the minimal value \( \Phi_k^e(\mathbf{w}_k^*) \) (19) of the CPL criterion function \( \Phi_k^e(\mathbf{w}) \) (18) can be found in one of the vertices \( \mathbf{w}_k \) (9) (Simonnard, 1966). So, taking into account equation (26), we infer that such vertex \( \mathbf{w}_k^* \) (9) exists for which \( \Phi_k^{e/\theta}(\mathbf{w}_k^*) = 0 \).

6. Basis exchange algorithm

The basis exchange algorithm was initially proposed and developed as an efficient tool for designing linear classifiers and examining linear separability of large, multidimensional data sets (Bobrowski, 2005). The first version of the basis exchange algorithm was described by Bobrowski and Niemiro (1984) and Bobrowski (1991). The first basis exchange algorithms were aimed at minimizing the perceptron criterion function. The CPL perceptron criterion function links the linear separability concept, which is fundamental in the perceptron theory of neural networks (Duda et al., 2001). The CPL criterion functions were specified later for a variety of goals. Different types of basis exchange algorithms were proposed for particular CPL criterion functions (Bobrowski, 2005).

The basis exchange algorithms are based on the Gauss–Jordan transformation. The famous Simplex algorithm used in the linear programming is also based on this transformation (Simonnard, 1966). Initial procedures of flat patterns extraction and implementation are described in Bobrowski and Zabielski (2016a). A brief description of the basis exchange algorithms aimed at the minimization of the function \( \Phi_k^e(\mathbf{w}) \) (21) is presented below.

Let us assume that the \( k \)th basis \( \mathbf{B}_k \) (8) is the squared matrix composed from \( n \) linearly independent feature vectors \( \mathbf{x}_{j(i)} \) from the subset \( \mathbf{C}_k \) (\( \mathbf{C}_k \subset \mathbf{C} \) (1)):

\[
\mathbf{B}_k = [\mathbf{x}_{j(1)}, \ldots, \mathbf{x}_{j(n)}]^T.
\]  

(27)

The inverse matrix \( \mathbf{B}_k^{-1} \) during the \( k \)th stage can be represented in the below manner:

\[
\mathbf{B}_k^{-1} = [\mathbf{r}_1(k), \ldots, \mathbf{r}_n(k)]^T.
\]  

(28)
The vectors \( \mathbf{x}_{j(0)} \) and \( r_1(k) \) fulfill the below equations:

\[
(\forall i, i' \in \{1, \ldots, n\}) \quad \mathbf{x}_{j(0)}^T r_1(k) = 1
\]  

(29)

and

\[
\text{if } i \neq i', \text{ then } \mathbf{x}_{j(0)}^T r_1(k) = 0.
\]

During the \( k \)th stage of the algorithm, the basis \( \mathbf{B}_k \) (27) is changed into the basis \( \mathbf{B}_{k+1} \). The matrix \( \mathbf{B}_{k+1} \) is created from the matrix \( \mathbf{B}_k \) (27) through replacing of the \( l \)th row \( \mathbf{x}_{j(l)} \) by the new vector \( \mathbf{x}_k \) taken from a given data subset \( \mathbf{C}_k \).

The exchange of the \( l \)th basis vector \( \mathbf{x}_{j(l)} \) by the new vector \( \mathbf{x}_k \) results in the new basis \( \mathbf{B}_{k+1} \) (27) and the new inverse matrix \( \mathbf{B}_{k+1}^{-1} = [r_1(k + 1), \ldots, r_n(k + 1)] \) (28). The Gauss–Jordan transformation allows us to compute efficiently the columns \( r(k + 1) \) of the matrix \( \mathbf{B}_{k+1}^{-1} \) on the basis of the columns \( r(k) \) (1) of the inverse matrix \( \mathbf{B}_k^{-1} \) (28), where \( k = 0, 1, \ldots, K \) (\[]\,\[]\):

\[
r_{k+1}(k + 1) = (1/r(k)\mathbf{x}_k)r_{k}(k)
\]

(30)

and (\( \forall i \neq 1 \))

\[
r_i(k + 1) = r_i(k) - (r_i(k)\mathbf{x}_k)r_{k}(k + 1)
\]

\[
= r_i(k) - (r_i(k)\mathbf{x}_k/r_i(k)\mathbf{x}_k)r_{k}(k).
\]

The basis exchange algorithms are based on the Gauss–Jordan transformation (30). The index \( l \) of the vector \( \mathbf{x}_{j(l)} \) which leaves the basis \( \mathbf{B}_k \) (27) during the \( k \)th stage is determined by the exit criterion of a particular basis exchange algorithm. The entry criterion determines which vector \( \mathbf{x}_k \) from the data subset \( \mathbf{C}_k \) (1) enters the new basis \( \mathbf{B}_{k+1} \) (28). The stop criterion allows us to determine the final stage \( K \) of the basis exchange algorithm. Selected criteria (exit, entry, and stop) should ensure the collinearity function \( \Phi_k^c(\mathbf{w}) \) (18) decreasing during each stage \( k \).

The exchange of the \( k \)th basis \( \mathbf{B}_k \) (27) into the basis \( \mathbf{B}_{k+1} \) causes the move from the vertex \( \mathbf{w}_k \) to \( \mathbf{w}_{k+1} \) (9). After finite number \( K \) of steps, the optimal vertex \( \mathbf{w}_K^* \) (22) is reached this way (Bobrowski, 2005).

### 7. Learning algorithm

The optimal vertex \( \mathbf{w}_k^* \) (9) which constitutes the minimum \( \Phi_k^c(\mathbf{w}_k^*) \) (19) of the collinearity criterion function \( \Phi_k^c(\mathbf{w}) \) (18) can be approximated using learning algorithms (Kushner & Yin, 1997). The Robbins–Monro procedure of the stochastic approximation has been used by us in designing learning algorithm aimed at the criterion function \( \Phi_k^c(\mathbf{w}) \) (18) minimization (Bobrowski & Zabielski, 2016b). For this purpose, it is useful to represent the criterion function \( \Phi_k^c(\mathbf{w}) \) (18) as the regression function \( \Phi_k(\mathbf{w}) \) in the below manner:

\[
\Phi_k(\mathbf{w}) = \sum_{j \in \mathcal{J}_k} p(\mathbf{x}_j) \varphi^c(\mathbf{x}_j; \mathbf{w}),
\]

(31)

where \( \varphi^c(\mathbf{x}_j; \mathbf{w}) = \varphi_j^c(\mathbf{w}) \) (19), and \( p(\mathbf{x}_j) \) is the uniform probability distribution defined on \( m_k \) elements \( \mathbf{x}_j \) of the \( k \)th data subset \( \mathbf{C}_k \) (\( C_k \subset \mathbf{C} \) (1)):

\[
(\forall j \in \mathcal{J}_k) \quad p(\mathbf{x}_j) = 1/m_k.
\]

(32)
The uniform probability distribution \( p(x_j) \) was used to generate the below learning sequence \( \{x(k)\} \) from \( m_k \) elements \( x_j \) of the \( k \)th data subset \( C_k \):
\[
x(1), x(2), \ldots, x(n), \ldots
\]  
(33)
The elements \( x(n) \) of the above learning sequence were generated in each step \( n \) independently in accordance with the probability distribution \( p(x_j) \) (32):
\[
(\forall n = 1, 2, 3, \ldots) \ P\{x(n) = x_j\} = p(x_j) = 1/m_k,
\]  
(34)
where \( P\{x(n) = x_j\} \) is the probability of obtaining the \( j \)th vector \( x_j \) during the \( n \)th learning step.

The Robbins–Monro procedure aimed at minimization of the regression function \( F_k(w) \) (34) can be given in the below manner (Kushner & Yin, 1997):
\[
w(n + 1) = w(n) - \alpha_n \nabla_w \varphi_n(x; w(n)),
\]  
(35)
where \( w(n) \) is the value of the weight vector during the \( n \)th learning step and \( \{\alpha_n\} \) is a sequence of decreasing, positive parameters \( \alpha_n \), e.g. \( \alpha_n = 1/n \) which fulfil the below conditions:
\[
\sum_n \alpha_n = \infty \text{ and } \sum_n \alpha_n^2 < \infty.
\]  
(36)
The Robbins–Monro procedure (35) applied to the regression function \( F_k(w) \) (31) with the penalty functions \( \varphi_n^j(w) \) (16) allows us to specify the below learning algorithm:
\[
(\forall n = 1, 2, \ldots)
\]
if \( w(n) \top x(n) < 1 - \varepsilon \) then \( w(n + 1) = w(n) + \alpha_n x(n) \), and
if \( w(n) \top x(n) > 1 + \varepsilon \) then \( w(n + 1) = w(n) - \alpha_n x(n) \), and
if \( 1 - \varepsilon \leq w(n) \top x(n) \leq 1 + \varepsilon \) then \( w(n + 1) = w(n) \),
where \( x(n) = x_j \) if the feature vector \( x_j \) was generated (34) during the \( n \)th learning step.

The learning algorithm (37) can be treated as a modification of the famous error-correction algorithm used in the Perceptron (Duda et al., 2001).

**Theorem 3:** If the learning sequence \( \{x(n)\} \) (33) is generated in accordance with the stationary probability distribution \( p(x_j) \) (34), then sequence \( \{w(n)\} \) of the weight vectors \( w(n) \) generated with the learning algorithm (37), (36) converges to the optimal vector \( w^* \) (14) which constitutes the minimal value \( F_k^*(w^*_k) \) (19) of the criterion function \( F_k(w) \) (18).

The proof of this theorem can be given using theorems of the stochastic approximation method (Kushner & Yin, 1997).

**8. Experimental results**

The compared basis exchange algorithm (31) and iterative learning algorithm (37) were implemented in C++ and run on an Intel Core i7, 1.73 GHz CPU (turbo boost to 2.93 GHz).

Three synthetic data sets \( D_1, D_2, \) and \( D_3 \) each with different level of outliers, were created for computational experiments. The term outliers means here such additional feature vectors \( x_j \) which were not located on the vertexical line \( L_k(x_{j(1)}, x_{j(2)}) \) (10). The outlier feature vectors \( x_j \) were generated in accordance with the normal distribution
\( N_2(0, I) \) with the unit covariance matrix \( I \) in the case of the two-dimensional feature space \( \mathbb{R}^2 \). The level of outliers was set accordingly: \( D_1 \rightarrow 33\% \), \( D_2 \rightarrow 50\% \), and \( D_3 \rightarrow 66\% \). One of this data set is presented in Figure 2.

Figure 3 shows results of the learning process with the iterative learning algorithm as the distance \( \rho(w(n), w^*_k) \) between the result \( w(n) \) (37) during the \( n \)th step and the optimal solution \( w^*_k \) (19) defined on data sets \( D_k \) with different level of outliers.

**Figure 2.** The two-dimensional learning set \( D_3 \) with added 66\% of outliers.

**Figure 3.** Distance \( \rho(w(n), w^*_k) \) between the result \( w(n) \) during the \( n \)th step and the optimal solution \( w^*_k \) (19) on the two-dimensional learning sets \( D_1 \) (squares), \( D_2 \) (circles), and \( D_3 \) (triangles) with different level of outliers (33\%, 50\%, and 66\%). Experiments were performed using the iterative learning algorithm (37).
As shown in those figures, the iterative algorithm converges after about \( n = 200 \) steps. However, in the case of higher level of outliers, the solution is worse and the distance between the result and the optimal solution \( w^*_k \) (19) is greater. The algorithm ends after \( n = 1000 \) steps in the case of each data set \( D_k \).

As a comparison, Figure 4 presents the results of the basis exchange algorithm, and the plot demonstrates the distance \( \rho(w(n), w^*_k) \) between the result \( w(n) \) (37) during the \( n \)th step and the optimal solution \( w^*_k \) (22) defined on data sets \( D_k \) with different level of outliers.

As shown in those figures, the solution was found in each case for data with different levels of outliers and the distance \( \rho(w(n), w^*_k) \) between the result \( w(n) \) during the \( n \)th step and the optimal solution \( w^*_k \) (19) was equal to zero. At the beginning of the execution (in the first few steps), the basis exchange algorithm gave worse results than the iterative algorithm. But after a few steps and at the end, the basis exchange algorithm is better and gives better results than the iterative algorithm. Moreover, the method of the iterative algorithm takes longer (in case of a number of iteration and execution time) than the basis exchange algorithm.

Next experiments were related to the role of the margin parameter – \( \varepsilon \). Iterative learning algorithm (37) and basis exchange algorithm (30) with different values of the parameter \( \varepsilon \) were tested on data sets shown in Figure 5.

The results of the learning process based on the iterative learning algorithm (37) are presented in Figure 6. The algorithm (37) was stopped when the condition: \( 1 - \varepsilon \leq w(n)^T x(n) \leq 1 + \varepsilon \) was reached.

Those figures have shown that the iterative learning algorithm converges after about 20 steps. The solution is worse and the distance between the result and the defined solution is greater in case of a greater value of the tested parameter \( \varepsilon \). When the margin greater

![Figure 4](image.png)

Figure 4. Distance \( \rho(w(n), w^*_k) \) between the result \( w(n) \) during the \( n \)th step and the optimal solution \( w^*_k \) (19) defined on the two-dimensional learning data sets \( D_1 \) (squares), \( D_2 \) (circles), and \( D_3 \) (triangles) with different levels of outliers. Experiments were performed using the basis exchange algorithm.
than zero ($\varepsilon > 0$) is selected, the algorithm ends faster than in case of algorithms without any margin ($\varepsilon = 0$).

Similarly, Figure 7 presents results of the basis exchange algorithm – the distance $\rho(w(n), w^*_k)$ between the result $w(n)$ during the $n$th step and the optimal solution $w^*_k$ (19) defined on the two-dimensional learning data set $D_4$ with different values of the margin $\varepsilon$. Experiments were performed using an iterative learning algorithm (37).

**Figure 5.** The two-dimensional learning set $D_4$ with the margin $\delta = 0.1$

**Figure 6.** Distance $\rho(w(n), w^*_k)$ between the result $w(n)$ during the $n$th step and the optimal solution $w^*_k$ (19) defined on the two-dimensional learning data set $D_4$ with different values of the margin $\varepsilon$ ($\varepsilon = 0$ represented by squares, $\varepsilon = 0.5 \times \delta$ represented by circles, and $\varepsilon = \delta$ represented by triangles). Experiments were performed using an iterative learning algorithm (37).
The final values of the criterion function $F_k(w)$ (18) for the basis exchange algorithm are equal to zero or are near this value. It gives better results and works faster than the iterative algorithm. The computation ends earlier for the margin $\varepsilon > 0$.

Figures 8–10 show the result dependence on the margin $\varepsilon$ value. With the increase in margin $\varepsilon$ value, more elements from feature vectors are in the solution plane. In case of $\varepsilon = 0.1$, most of them are defined by the result.

![Figure 7. The distance $\rho(w(n), w^*_k)$ between the result $w(n)$ during the $n$th step and the optimal solution $w^*_k$ (19) defined on the two-dimensional learning data set $D_4$ with different values of parameter $\varepsilon$ ($\varepsilon = 0$ represented by squares, $\varepsilon = 0.5 \times \delta$ represented by circles, and $\varepsilon = \delta$ represented by triangles). Experiments were performed using the basis exchange algorithm.](image)

The final values of the criterion function $\Phi_k(w)$ (18) for the basis exchange algorithm are equal to zero or are near this value. It gives better results and works faster than the iterative algorithm. The computation ends earlier for the margin $\varepsilon > 0$.

Figures 8–10 show the result dependence on the margin $\varepsilon$ value. With the increase in margin $\varepsilon$ value, more elements from feature vectors are in the solution plane. In case of $\varepsilon = 0.1$, most of them are defined by the result.

![Figure 8. The result of the data set $D_4$ with the value of parameter $\varepsilon = 0$.](image)
9. Concluding remarks

A better understanding of large, multivariate data sets can be achieved through the identification of collinear (flat) patterns $\mathbf{F}_k$ (14). Extraction of collinear patterns $\mathbf{F}_k$ (17) can be performed through the minimization of the collinear criterion function $\Phi_k^F(\mathbf{w})$ (18) defined on various data subsets $\mathbf{C}_k$ ($\mathbf{C}_k \subset \mathbf{C}$ (1)).

Both the basis exchange algorithm (30) as well as the iterative learning algorithm (37) can be used in the minimization of the collinear criterion function $\Phi_k^F(\mathbf{w})$ (18). Usability of both types of algorithms was tested experimentally. Selected results of these experiments are shown in the figures.

Figure 9. The result of the data set $D_4$ with the value of parameter $\varepsilon = 0.5 \times \delta$.

Figure 10. The result of the data set $D_4$ with the value of parameter $\varepsilon = \delta$. 
Particular attention in the article was paid to the role of the margin $\varepsilon$ (12) in extraction of flat patterns $F_k$ (14). The margin $\varepsilon$ greater than zero ($\varepsilon > 0$) allows us to discover not only collinear patterns $F_k$ (14) as the subsets of points $x_j$ located on vertexical planes $P_k(x_{j(1)}, \ldots, x_{j(r_k)})$ (3), but also flat patterns located in layers $L^\varepsilon(w, \theta)$ (20) of different thicknesses. Noise reduction may be based on such layered aggregation of data subsets $C_k$. Increasing capacity in this area should be achieved by applying the *relaxed linear separability* method of feature subset selection (Bobrowski & Łukaszuk, 2011).

The proposed method of discovering collinear patterns on the basis of the *CPL* criterion functions $\Phi^*_k(w)$ (18) can be treated as a generalization of the methods based on the Hough transformation. The Hough transformation techniques are used in computer vision for detection lines and curves in pictures (Ballard, 1981; Duda & Hart, 1972).

**Disclosure statement**

No potential conflict of interest was reported by the authors.

**Notes on contributors**

*Professor Leon Bobrowski* is the head of the Software Department at the Faculty of Computer Science, Białystok University of Technology. Additionally, he works in the Laboratory of Biomedical Data Analysis at the Institute of Biocybernetics and Biomedical Engineering of the Polish Academy of Sciences in Warsaw. Research interests of Leon Bobrowski include specific methods of data mining, pattern recognition, and medical diagnosis support systems which are based on the minimization of the convex and piecewise linear (CPL) criterion functions defined on data sets. The basis exchange algorithms have been developed and implemented which are similar to linear programming and allow find the minimum of the CPL functions efficiently, even in the case of large, multidimensional data sets. This approach is used for designing medical diagnosis support systems (Hepar), hierarchical neural networks, multivariate decision trees and visualizing transformations. Most recent research topics involve designing prognostic models by using concepts of ranked regression, interval regression and the relaxed linear separability (RLS) method of feature (genes) subsets selection. Teaching experience relates to statistical models and algorithms, multivariate data analysis, decision support systems, and exploratory analysis of large data sets. Professor Leon Bobrowski was co-organizer since 1994 eleventh seminar “Statistics and Clinical Practice” held in Warsaw in the framework of the International Centre of Biocybernetics (ICB) of the Polish Academy of Sciences.

*Paweł Zabielski* was born in 1988 in Białystok (Poland). He was studying at the Faculty of Computer Science at the Białystok University of Technology. He obtained a bachelor degree in Computer Science from the Faculty of Computer Science at the Białystok University of Technology in 2010 and a Master degree in Computer Science in 2011 at the same university, both graduated with a distinction. Paweł Zabielski has been working at the Faculty of Computer Science at the Białystok Technical University since 1st March 2011, at first as a research assistant and from 1st October 2011 as an assistant professor. Research interests of Paweł Zabielski include the methods of processing and analysis of biomedical datasets and applications to medical diagnosis support. He works on developing the methods of knowledge data discovery based on artificial neural network models.

**References**

Albert, B. (2013). *Mining Big Data in real time*. *Informatica*, 37, 15–20.

Ballard, D. H. (1981). Generalizing the *Hough transform* to detect arbitrary shapes. *Pattern Recognition*, 13(2), 111–122.
Bobrowski, L. (1991). Design of piecewise linear classifiers from formal neurons by a basis exchange technique. *Pattern Recognition, 24*(9), 863–870.

Bobrowski, L. (2005). *Data mining based on convex and piecewise linear criterion functions* (in Polish). Bialystok: Technical University of Bialystok.

Bobrowski, L. (2014). Discovering main vertexical planes in a multivariate data space by using CPL functions. In P. Perner (Ed.), *ICDM 2014* (pp. 200–213). Berlin: Springer Verlag.

Bobrowski, L., & Łukaszuk, T. (2011). Relaxed linear separability (RLS) approach to feature (gene) subset selection. In Xuhua Xia (Ed.), *Selected works in bioinformatics* (pp. 103–118). INTECH.

Bobrowski, L., & Niemiro, W. (1984). A method of synthesis of linear discriminant function in the case of nonseparability. *Pattern Recognition, 17*, 205–210.

Bobrowski, L., & Zabielski, P. (2016a). Flat patterns extraction with collinearity models. 9th EUROSIM Congress on Modelling and Simulation, EUROSIM2016, 12–16 September 2016, Oulu Finland, IEEE Conference Publishing Services (CPS).

Bobrowski, L., & Zabielski, P. (2016b). Learning algorithms aimed at collinear patterns. In ACIIDS (2) 2016 (pp. 159–168). Berlin: Springer Verlag.

Clarke, R., Ressom, H. W., Wang, A., Xuan, J., Liu, M. C., Gehan, E. A., & Wang, Y. (2008). The properties of high-dimensional data spaces: Implications for exploring gene and protein expression data. *Nature Reviews Cancer, 8*(1), 37–49.

Duda, O. R., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of Association for Computing Machinery, 15*(1), 11–15.

Duda, O. R., Hart, P. E., & Stork, D. G. (2001). *Pattern classification*. New York, NY: Wiley.

Hand, D., Smyth, P., & Mannila, H. (2001). *Principles of data mining*. Cambridge: MIT Press.

Ian, H. W., & Eibe, F. (2005). *Data mining: Practical machine learning tools and techniques*. San Francisco: Elsevier.

Jaseena, K. U., & Julie, M. D. (2014). Issues challenges and solutions: Big Data mining, pp. 131–140. CS & IT-CSCP 2014.

Julie, M. D., & Kannan, B. (2011). Prediction of key symptoms of learning disabilities in school-age children using rough sets. *International Journal of Computer and Electrical Engineering, 3*(1), 163–169.

Kushner, H. J., & Yin, G. G. (1997). *Stochastic approximation algorithms and applications*. Berlin: Springer Verlag.

Simonnard, M. (1966). *Linear programming*. Englewood Cliffs, NJ: Prentice-Hall.

Sun-Mi, L., & Patricia, A. A. (2003). Bayesian networks for knowledge discovery in large datasets: Basics for nurse researchers. *Journal of Biomedical Informatics, 36*, 389–399.

Wong, K.-C. (Ed.). (2016). *Big data analytics in genomics*. Berlin: Springer Verlag.