VARCLUST: clustering variables using dimensionality reduction

Piotr Sobczyk, Stanislaw Wilczynski, Malgorzata Bogdan, Piotr Graczyk, Julie Josse, Fabien Panloup, Valerie Seegers, Mateusz Staniak

▶ **To cite this version:**

Piotr Sobczyk, Stanislaw Wilczynski, Malgorzata Bogdan, Piotr Graczyk, Julie Josse, et al.. VARCLUST: clustering variables using dimensionality reduction. 2020. hal-03002017v1

HAL Id: hal-03002017

https://hal.science/hal-03002017v1

Preprint submitted on 12 Nov 2020 (v1), last revised 18 Dec 2020 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Statistics and Computing manuscript No.
(will be inserted by the editor)

VARCLUST: clustering variables using dimensionality reduction

Piotr Sobczyk · Stanisław Wilczyński · Małgorzata Bogdan · Piotr Graczyk · Julie Josse · Fabien Panloup · Valérie Seegers · Mateusz Staniak

Received: date / Accepted: date

Abstract VARCLUST algorithm is proposed for clustering variables under the assumption that variables in a given cluster are linear combinations of a small number of hidden latent variables, corrupted by the random noise. The entire clustering task is viewed as the problem of selection of the statistical model, which is defined by the number of clusters, the partition of variables into these clusters and the 'cluster dimensions', i.e. the vector of dimensions of linear subspaces spanning each of the clusters. The "optimal" model is selected using the approximate Bayesian criterion based on the Laplace approximations and using a non-informative uniform prior on the number of clusters. To solve the problem of the search over a huge space of possible models we propose an extension of the *ClustOfVar* algorithm of [29, 7] which was dedicated to subspaces of dimension only 1, and which is similar in structure to the K-centroid algorithm. We provide a complete methodology with theoretical guarantees, extensive numerical experimentations, complete data analyses and implementation. Our algorithm assigns variables to appropriate clusterse based on the consistent Bayesian Information Criterion (BIC), and estimates the dimensionality of each cluster by the PEnalized SEmi-integrated Likelihood Criterion (PESEL) of [24], whose consistency we prove. Additionally, we prove that each iteration of our algorithm leads to an increase of the Laplace approximation to the model posterior probability and provide the criterion for the estimation of the number of clusters. Numerical comparisons with other algorithms show that VARCLUST may outperform some popular machine learning tools for sparse subspace clustering. We also report the results of real data analysis including TCGA breast cancer data and meteorological data, which show that the algorithm can lead to meaningful clustering. The proposed method is implemented in the publicly available `R` package varclust.

Keywords variable clustering · Bayesian approach · k-means · dimensionality reduction · subspace clustering

Piotr Sobczyk
OLX Group, Poland
E-mail: pj.sobczyk@gmail.com

Stanisław Wilczyński
Microsoft Development Center Norway, Oslo
E-mail: sjwilczynski@gmail.com

Małgorzata Bogdan
Institute of Mathematics, University of Wrocław, Poland, and Department of Statistics, Lund University, Sweden
E-mail: malgorzata.bogdan@uwr.edu.pl

Piotr Graczyk
Laboratoire Angevin de Recherche en Mathématiques (LAREMA), Université d'Angers, France
E-mail: piotr.graczyk@univ-angers.fr

Julie Josse
INRIA Montpellier and Ecole Polytechnique in Paris, France
E-mail: julie.josse@polytechnique.edu

Fabien Panloup
Laboratoire Angevin de Recherche en Mathématiques (LAREMA), Université d'Angers, and SIRIC ILIAD, Nantes, Angers, France,
E-mail: fabien.panloup@univ-angers.fr

Valérie Seegers
ICO Angers, France
E-mail: Valerie.Seegers@ico.unicancer.fr

Mateusz Staniak
Institute of Mathematics, University of Wrocław, Poland
E-mail: Mateusz.Staniak@uwr.edu.pl

1 Introduction

Due to the rapid development of measurement and computer technologies, large data bases are nowadays stored and explored in many fields of industry and science. This in turn triggered development of new statistical methodology for acquiring information from such large data.

In large data matrices it usually occurs that many of the variables are strongly correlated and in fact convey a similar message. Principal Components Analysis (PCA) [20,14,15,17] is one of the most popular and powerful methods for data compression. This dimensionality reduction method recovers the low dimensional structures spanning the data. The mathematical hypothesis which is assumed for this procedure is based upon the belief that the denoised data matrix is of a low rank, i.e. that the data matrix $X_{n\times p}$ can be represented as

$$X = M + \mu + E, \tag{1.1}$$

where M is deterministic, $\mathrm{rank}(M) \ll \min(n,p)$, the mean matrix μ is rank one and the matrix E represents a centered normal noise.

Thus, PCA model assumes that all points from the data come from the same low dimensional space, which is often unrealistic. A natural extension of this model is the model of subspace clustering (see [27] and references therein), which assumes that columns of the matrix M can be clustered in groups of low dimensions. From the statistical perspective, in subspace clustering the model (1.1) is applied separately for each cluster and the statistical model for the whole data base is determined by the partition of variables into different clusters and the vector of dimensions (ranks of the corresponding M matrices) for each of the clusters, called here 'cluster dimensions'. From a practical (but also statistical) point of view, the aim of these extensions of PCA-type algorithms is to get a better low dimensional representation of the whole data set, which in turns should provide some better supervised classification algorithms based on these data.

In this paper, we propose in Section 2, VARCLUST as an approximate Bayesian approach to the sparse subspace clustering. Our approach described in Section 3 is based on a novel K-centroids algorithm, made of two steps based on the Laplace approximation. In the first step, given a partition of the variables, we use PESEL [24], a BIC-type estimator, to estimate the dimensions of each cluster. In the second step we perform the partition where the similarity between a given variable and a cluster of variables represented by its linear subspace is measured by the Bayesian Information Criterion in the multiple regression model relating this variable to the set of the subspace's principal components. From a theoretical point of view, we prove in Section 4 the consistency of PESEL, *i.e.* the convergence of the estimator of the cluster dimension towards its true dimension (see Theorem 1). For the VARCLUST itself, we show that our algorithm leads to an increase of the Laplace approximation to the model posterior probability(see Corollary 2). From a numerical point of view, our paper investigates numerous issues in Section 5. The convergence of VARCLUST is empirically checked and some comparisons with other algorithms are provided showing that the VARCLUST algorithm seems to have the ability to retrieve the true model with an higher probability than other popular sparse subspace clustering procedures. Finally, in Section 6, we consider two important applications to breast cancer and meteorological data. Once again, in this part, the aim is twofold: reduction of dimension but also identification of groups of genes/indicators which seem to take action together. In Section 7, the R package `varclust` which uses parallel computing for computational efficiency is presented and its main functionalities are detailed.

2 VARCLUST model

2.1 A low rank model in each cluster

Let $X_{n\times p}$ be the data matrix with p columns $x_{\bullet j} \in \mathbf{R}^n$, $j \in \{1,\dots,p\}$. The clustering of p variables $x_{\bullet j} \in \mathbf{R}^n$ into K clusters consists in considering a column-permuted matrix X' and decomposing

$$X' = \left[X^1|X^2|\dots|X^K\right] \tag{2.1}$$

such that each bloc X^i has dimension $n \times p_i$, with $\sum_{i=1}^K p_i = p$. In this paper we apply to each cluster X^i the model (1.1):

$$X^i = M^i + \mu^i + E^i, \tag{2.2}$$

where M^i is deterministic, $rank(M^i) = k_i \ll min(n,p_i)$, the mean matrix μ^i is rank one and the matrix E^i represents the centered normal noise $N(0,\sigma_i^2 Id)$.

As explained in [24], the form of the rank one matrix μ^i depends on the relation between n and p_i. When $n > p_i$, the n rows of the matrix μ^i are identical, i.e. $\mu^i = \begin{pmatrix} \mathbf{r}^i \\ \vdots \\ \mathbf{r}^i \end{pmatrix}$ where $\mathbf{r}^i = (\mu_1^i,\dots,\mu_{p_i}^i)$.

If $n \leq p_i$, the p_i columns of the matrix μ^i are identical, i.e. $\mu^i = \left(\mathbf{c}^i \dots \mathbf{c}^i\right)$ with $\mathbf{c}^i = (\mu^i_1, \dots, \mu^i_n)^\top$. We point out that our modeling allows some clusters to have $p_i \geq n$, whereas in other clusters p_i maybe smaller than n. This flexibility is one of important advantages of the VARCLUST model.

Next we decompose each matrix M^i, for $i = 1, \dots, K$, as a product

$$M^i = F^i_{n\times k_i} C^i_{k_i \times p_i} \tag{2.3}$$

The columns of $F_{n\times k_i}$ are supposed to be independent and will be called factors (by analogy to PCA).

This model extends the classical model (1.1) for PCA, which assumes that all variables in the data set can be well approximated by a linear combination of just a few hidden "factors", or, in other words, the data belong to a low dimensional linear space. Here we assume that the data comes from a union of such low dimensional subspaces. This means that the variables (columns of the data matrix X) can be divided into clusters X^i, each of which corresponds to variables from one of the subspaces. Thus, we assume that every variable in a single cluster can be expressed as a linear combination of small number of factors (common for every variable in this cluster) plus some noise. This leads to formulas (2.2) and (2.3). Such a representation is clearly not unique. The goal of our analysis is clustering columns in X and in M, such that all coefficients in the matrices $C^1, \dots, C^K$ are different from zero and $\sum_{i=1}^K k_i$ is minimized.

Let us summarize the model that we study. An element of $\mathcal{M}$ is defined by four parameters $(K, \Pi, \mathbf{k}, \mathbb{P}_\theta)$ where:

- K is the number of clusters and $K \leq K_{max}$ for a fixed $K_{max} \ll p$,
- Π is a K-partition of $\{1, \dots, p\}$ encoding a segmentation of variables (columns of the data matrix $X_{n\times p}$) into clusters $X^i_{n\times p_i} =: X_{\Pi_i}$,
- $\mathbf{k} = (k_1, \dots, k_K) \in \{1, \dots, d\}^{\otimes K}$, where d is the maximal dimension of (number of factors in) a cluster. We choose $d \ll n$ and $d \ll p$.
- $\mathbb{P}_\theta$ is the probability law of the data specified by the vectors of parameters $\theta = (\theta_1, \dots, \theta_K)$, with θ_i containing the factor matrix F^i, the coefficient matrix C^i, the rank one mean matrix μ^i and the error variance σ_i^2,

$$\mathbb{P}_\theta(X) = \prod_{i=1}^K \mathbb{P}\left(X_{\Pi_i}|\theta_i\right)$$

and $\mathbb{P}\left(X_{\Pi_i}|\theta_i\right)$ is defined as follows: let $x^i_{\bullet j}$ be the j-th variable in the i-th cluster and let $\mu^i_{\bullet j}$ be the j-th column of the matrix μ^i. The vectors $x^i_{\bullet j}$, $j = 1, \dots, p_i$, are independent conditionally on θ_i and it holds

$$x^i_{\bullet j}|\theta_i = x^i_{\bullet j}|(F^i, C^i, \mu^i, \sigma_i^2) \quad \sim N(F^i c^i_{\bullet j} + \mu^i_{\bullet j}, \sigma_i^2 I_n) \ . \tag{2.4}$$

Note that according to the model (2.4), the vectors $x^i_{\bullet j}|\theta_i$, $j = 1, \dots, k_i$, in the same cluster X^i have the same covariance matrices $\sigma_i^2 I_n$.

2.2 Bayesian approach to subspace clustering

To select a model (number of clusters, variables in a cluster and dimensionality of each cluster), we consider a Bayesian framework. We assume that for any model $\mathcal{M}$ the prior $\pi(\theta)$ is given by

$$\pi(\theta) = \prod_{i=1}^K \pi(\theta_i) \ .$$

Thus, the log-likelihood of the data X given the model $\mathcal{M}$ is given by

$$\begin{aligned}
\ln\left(\mathbb{P}(X|\mathcal{M})\right) &= \ln\left(\int_\Theta \mathbb{P}(X|\theta)\pi(\theta)d\theta\right) \\
&= \ln \prod_{i=1}^K \left(\int_{\Theta_i} \mathbb{P}(X^i|\theta_i)\pi(\theta_i)d\theta_i\right) \\
&= \sum_{i=1}^K \ln\left(\int_{\Theta_i} \mathbb{P}(X^i|\theta_i)\pi(\theta_i)d\theta_i\right) \\
&= \sum_{i=1}^K \ln\left(\mathbb{P}(X^i|\mathcal{M}_i)\right), \qquad (2.5)
\end{aligned}$$

where $\mathcal{M}_i$ is the model for the i-th cluster X^i specified by (2.3) and (2.4).

In our approach we propose an informative prior distribution on $\mathcal{M}$. The reason is that in our case we have, for given K, roughly K^p different segmentations, where p is the number of variables. Moreover, given a maximal cluster dimension $d = d_{max}$, there are d^K different selections of cluster dimensions. Thus, given K, there are approximately $K^p d^K$ different models to compare. This number quickly increases with K and assuming that all models are equally likely we would in fact use a prior on the number of clusters K, which would strongly prefer large values of K. Similar problems were already observed when using BIC to select the multiple regression model based on a data base with many potential predictors. In [5] this problem was solved by using the

modified version of the Bayes Information Criterion (mBIC) with the informative sparsity inducing prior on $\mathcal{M}$. Here we apply the same idea and use an approximately uniform prior on K from the set $K \in \{1, \ldots, K_{max}\}$, which, for every model $\mathcal{M}$ with the number of clusters K, takes the form:

$$\begin{aligned} \pi(\mathcal{M}) &= \frac{C}{K^p d^K} \\ \ln(\pi(\mathcal{M})) &= -p\ln(K) - K\ln(d) + C\ , \end{aligned} \tag{2.6}$$

where C is a proportionality constant, that does not depend on the model under consideration. Using the above formulas and the Bayes formula, we obtain the following Bayesian criterion for the model selection: pick the model (partition Π and cluster dimensions $\mathbf{k}$) such that

$$\begin{aligned} \ln(\mathbb{P}(\mathcal{M}|X)) &= \ln(\mathbb{P}(X|\mathcal{M})) + \ln(\pi(\mathcal{M})) - \ln(\mathbb{P}(X)) \\ &= \sum_{i=1}^{K} \ln \mathbb{P}(X^i|\mathcal{M}_i) - p\ln(K) \\ &\quad - K\ln(d) + C - \ln \mathbb{P}(X)\ . \end{aligned} \tag{2.7}$$

obtains a maximal value. Since $\mathbb{P}(X)$ is the same for all considered models this amounts to selecting the model, which optimizes the following criterion

$$C(\mathcal{M}|X) = \sum_{i=1}^{K} \ln \mathbb{P}(X^i|\mathcal{M}_i) - p\ln(K) - K\ln(d)\ . \tag{2.8}$$

The only quantity left to calculate in the above equation is $\mathbb{P}(X^i|\mathcal{M}_i)$.

3 VARCLUST method

3.1 Selecting the rank in each cluster with the PESEL method

Before presenting the VARCLUST method, let us present shortly the PESEL method, introduced in [24] designed to estimate the number of principal components in PCA. It will be used in the first step of the VARCLUST.

As explained in Section 2 (cf. (2.4)), our model for one cluster can be described by its set of parameters (for simplicity we omit the index of the cluster) $\theta : F \in \mathbb{R}^{n\times k}, c_1, \ldots, c_p$, where $c_i \in \mathbb{R}^{k\times 1}$ (vectors of coefficients), σ^2 and μ. In order to choose the best model we have to consider models with different dimensions, *i.e.* different values of k. The penalized semi-integrated likelihood (PESEL, [24]) criterion is based on the Laplace approximation to the semi-integrated likelihood and in this way it shares some similarities with BIC. The general formulation of PESEL allows for using different prior distributions on F (when $n > p$) or C (when $p > n$). The basic version of PESEL uses the standard gaussian prior for the elements of F or C and has the following formulation, depending on the relation between n and p.

We denote by $(\lambda_j)_{j=1,\ldots,p}$ the non-increasing sequence of eigenvalues of the sample covariance matrix S_n. When $n \le p$ we use the following form of the PESEL

$$\begin{aligned} &\ln(\mathbb{P}(X^i|\mathcal{M}_i)) \approx PESEL(p,k,n) = \\ &-\frac{p}{2}\left[\sum_{j=1}^{k}\ln(\lambda_j) + (n-k)\ln\left(\frac{1}{n-k}\sum_{j=k+1}^{n}\lambda_j\right) + n\ln(2\pi) + n\right] \\ &\quad - \ln(p)\frac{nk - \frac{k(k+1)}{2} + k + n + 1}{2} \end{aligned} \tag{3.1}$$

and when $n > p$ we use the form

$$\begin{aligned} &\ln(\mathbb{P}(X^i|\mathcal{M}_i)) \approx PESEL(n,k,p) = \\ &-\frac{n}{2}\left[\sum_{j=1}^{k}\ln(\lambda_j) + (p-k)\ln\left(\frac{1}{p-k}\sum_{j=k+1}^{p}\lambda_j\right) + p\ln(2\pi) + p\right] \\ &\quad - \ln(n)\frac{pk - \frac{k(k+1)}{2} + k + p + 1}{2}\ . \end{aligned} \tag{3.2}$$

The function of the eigenvalues λ_j of S_n appearing in (3.1),(3.2) and approximating the log-likelihood $\mathbb{P}(X^i|\mathcal{M}_i)$ is called a *PESEL function*. The criterion consists in choosing the value of k maximizing the PESEL function.

When $n > p$, the above form of PESEL coincides with BIC in Probabilistic PCA (see [19]) or the spiked covariance structure model. These models assume that the rows of the X matrix are i.i.d. random vectors. Consistency results for PESEL under these probabilistic assumptions can be found in [2].

In Section 4.1 we will prove consistency of PESEL under a much more general fixed effects model (4.1), which does not assume the equality of laws of rows in X.

3.2 Membership of a variable in a cluster with the BIC criterion

To measure the similarity between l^{th} variable and a subspace corresponding to i^{th} cluster we use the Bayesian Information Criterion. Since the model (2.4) assumes that all elements of the error matrix E^i have the same variance, we can estimate σ_i^2 by MLE

$$\hat{\sigma}_i^2 = \frac{\sum_{\ell\in\Pi_i}\|x_{\bullet\ell} - P_i(x_{\bullet\ell})\|^2}{np_i}\ ,$$

where $P_i(x_{\bullet\ell})$ denotes the orthogonal projection of the column $x_{\bullet\ell}$ on the linear space corresponding

to the i^{th} cluster and next use BIC of the form

$$BIC(l,i) = \frac{1}{2}\left(-\frac{\|x_{\bullet\ell} - P_i(x_{\bullet\ell})\|^2}{\hat{\sigma}_i^2} - \ln n\, k_i\right) \quad . \tag{3.3}$$

As an alternative, one can consider a standard multiple regression BIC, which allows for different variances in different columns of E^i:

$$BIC(l,i) = -n\ln\left(\frac{RSS_{li}}{n}\right) - k_i \ln(n), \tag{3.4}$$

where RSS_{li} is the residual sum of squares from regression of $x_{\bullet\ell}$ on the basis vectors spanning i^{th} cluster.

3.3 VARCLUST algorithm

Initialization and the first step of VARCLUST
Choose randomly a K-partition of $p = p_1^0 + \ldots + p_K^0$ and group randomly $p_1^0, \ldots, p_K^0$ columns of X to form Π^0.
Then, VARCLUST proceeds as follows:

$$\Pi^0 \;\rightarrow\; (\Pi^0, k^0) \;\rightarrow\; (\Pi^1, k^0) \;, \tag{3.5}$$

where k^0 is computed by using PESEL K times, separately to each matrix X_0^i, $i = 1, \ldots, K$. Next, for each matrix X_0^i, PCA is applied to estimate k_i^0 principal factors F_i^1, which represent the basis spanning the subspace supporting i^{th} cluster and the center of the clusters. The next partition Π^1 is obtained by using $BIC(l,i)$ as a measure of similarity between l^{th} variable and i^{th} cluster to allocate each variable to its closest cluster. After the first step of VARCLUST, we get the couple: the partition and the vector of cluster dimensions (Π^1, k^0).

Other schemes of initialization can be consider such as a one-dimensional initialization. Choose randomly K variables which will play the role of one dimensional centers of K clusters . Distribute, by minimizing BIC, the p columns of X to form the first partition Π^1. In this way, after the first step of VARCLUST we again get (Π^1, k^0), where k^0 is the K dimensional all ones vector.

Step $m+1$ of VARCLUST In the sequel we continue by first using PESEL to calculate a new vector of dimensions and next PCA and BIC to obtain the next partition:

$$(\Pi^m, k^{m-1}) \;\rightarrow\; (\Pi^m, k^m) \;\rightarrow\; (\Pi^{m+1}, k^m).$$

4 Theoritical guarentees

In this Section we prove the consistency of PESEL and show that each iteration of VARCLUST asymptotically leads to an increase of the objective function (2.8).

4.1 Consistency of PESEL

In this section we prove that PESEL consistently estimates the rank of the denoised data matrix. The consistency holds when n or p diverges to infinity, while the other dimension remains constant. This result can be applied separately to each cluster X^i, $i \in \{1, \ldots, K\}$, of the full data matrix.

First, we prove the consistency of PESEL (Section 3.1) when p is fixed as $n \to \infty$.

Assumption 1. Assume that the data matrix X is generated according to the following probabilistic model :

$$X_{n\times p} = M_{n\times p} + \mu_{n\times p} + E_{n\times p}, \tag{4.1}$$

where

- for each $n \in \mathbb{N}$, matrices $M_{n\times p}$ and $\mu_{n\times p}$ are deterministic
- $\mu_{n\times p}$ is a rank-one matrix in which all rows are identical, i.e. it represents average variable effect.
- the matrix $M_{n\times p}$ is centered: $\sum_{i=1}^n M_{ij} = 0$ and $\operatorname{rank}(M_{n\times p}) = k_0$ for all $n \geq k_0$
- the elements of matrix $M_{n\times p}$ are bounded: $\sup_{n,i\in(1,\ldots,n),j\in(1,\ldots,p)} |M_{ij}| < \infty$
- there exists the limit: $\lim_{n\to\infty} \frac{1}{n} M_{n\times p}^T M_{n\times p} = L$ and, for all n

$$\left|\frac{1}{n} M_{n\times p}^T M_{n\times p} - L\right| < C\frac{\sqrt{2\ln\ln n}}{\sqrt{n}} \quad , \tag{4.2}$$

where C is some positive constant and $L = UD_{p\times p}U^T$ with

$$D_{p\times p} = \begin{pmatrix} diag[\gamma_i]_{i=1}^{k_0} & 0 \\ 0 & diag[0] \end{pmatrix}$$

with non-increasing $\gamma_i > 0$ and $U^TU = Id_{p\times p}$.
- the noise matrix $E_{n\times p}$ consists of i.i.d. terms $e_{ij} \sim \mathcal{N}(0, \sigma^2)$.

Theorem 1 (Consistency of PESEL)
Assume that the data matrix $X_{n\times p}$ satisfies the Assumption 1. Let $\hat{k}_0(n)$ be the PESEL(p,k,n) estimator of the rank of M.

Then, for p fixed, it holds

$$\mathbb{P}(\exists n_0\ \forall n > n_0 \quad \hat{k}_0(n) = k_0) = 1.$$

Scheme of the Proof.

Let us consider the sample covariance matrix

$$S_n = \frac{(X - \bar{X})^T (X - \bar{X})}{n}.$$

and the population covariance matrix $\Sigma_n = E(S_n)$. The idea of the proof is the following.

Let us denote by $F(n,k)$ the PESEL function in the case when $n > p$. By (3.2), we have

$$F(n,k) = -\frac{n}{2}\Big[\sum_{j=1}^{k}\ln(\lambda_j) + (p-k)\ln\left(\frac{1}{p-k}\sum_{j=k+1}^{p}\lambda_j\right) + p\ln(2\pi) + p\Big] - \ln(n)\frac{pk - \frac{k(k+1)}{2} + k + p + 1}{2}$$

The proof comprises two steps. First, we quantify the difference between eigenvalues of matrices S_n, Σ_n and L. We prove it to be bounded by the matrix norm of their difference, which goes to 0 at the pace $\frac{\sqrt{\ln\ln n}}{\sqrt{n}}$ as n grows to infinity, because of the law of iterated logarith (LIL). We use the most general form of LIL from [21]. Secondly, we use the results from the first step to prove that for sufficiently large n the PESEL function $F(n,k)$ is increasing for $k < k_0$ and decreasing for $k > k_0$. To do this, the crucial Lemma 1 is proven and used. The detailed proof is given in Appendix 8.

Since the version of PESEL for $p >> n$, PESEL(n,k,p), is obtained simply by applying PESEL(p,k,n) to the transposition of X, Theorem 1 implies the consistency of PESEL also in the situation when n is fixed, $p \to \infty$ and the transposition of X satisfies the Assumption 1.

Corollary 1 *Assume that the transposition of the data matrix $X_{n\times p}$ satisfies the Assumption 1. Let $\hat{k}_0(n)$ be the PESEL(n,k,p) estimator of the rank of M.*

Then, for n fixed, it holds

$$\mathbb{P}(\exists p_0\ \forall p > p_0 \quad \hat{k}_0(p) = k_0) = 1.$$

Remark. The above results assume that p or n is fixed. We believe that they hold also in the situation when $\frac{n}{p} \to \infty$ or vice versa. The mathematical proof of this conjecture is an interesting topic for a further research. These theoretical results justify the application of PESEL when $n >> p$ or $p >> n$. Moreover, simulation results reported in [24] illustrate good properties of PESEL also when $p \sim n$. The theoretical analysis of the properties of PESEL when $p/n \to C \neq 0$ remains an interesting topic for further research.

4.2 Convergence of VARCLUST

As noted above in (2.8), the main goal of VARCLUST is identifying the model $\mathcal{M}$ which maximizes, for a given dataset X,

$$\ln(\mathbb{P}(\mathcal{M}|X)) = \sum_{i=1}^{K}\ln\mathbb{P}(X^i|k_i) + \ln(\pi(\mathcal{M}))\ ,$$

where $\ln(\pi(\mathcal{M}))$ depends only on the number of clusters K and the maximal allowable dimension of each cluster d.

Since, given the number of clusters K, the VARCLUST model is specified by the vector of cluster dimensions $k = (k_1, \ldots, k_K)$ and a partition $\Pi = (\Pi_1, \ldots, \Pi_K)$ of p variables into these K clusters, our task reduces to identifying the model for which the following objective function

$$\varphi(\Pi, k)\ := \sum_{i=1}^{K}\ln\mathbb{P}(X^i|k_i)\ , \tag{4.3}$$

obtains a maximum.

Below we will discuss consecutive steps of the VARCLUST Algorithm with respect to the optimization of (4.3). Recall that the $m+1$ step of VARCLUST is

$$(\Pi^m, k^{m-1})\ \to\ (\Pi^m, k^m)\ \to\ (\Pi^{m+1}, k^m),$$

where we first use PESEL to estimate the dimension and next PCA to compute the factors and BIC to allocate variables to a cluster.

1. **PESEL step: choice of cluster dimensions, for a fixed partition of X.**
   First, observe that the dimension of i^{th} cluster in the next $(m+1)^{th}$ step of VARCLUST is obtained as

   $$k_i^m = \underset{k_i\in\{1,\ldots,d\}}{\arg\max}\ PESEL(X^i|k_i)\ .$$

   Thus, denoting by $PESEL$ the PESEL function from (3.1) and (3.2),

   $$\sum_{i=1}^{K} PESEL(X_m^i|k_i^m) \geq \sum_{i=1}^{K} PESEL(X_m^i|k_i^{m-1})\ .$$

   Now, observe that under the standard regularity conditions for the Laplace approximation (see e.g. [4])

   $$\ln\mathbb{P}(X^i|k_i) = PESEL(X^i|k_i) + O_n(1)$$

   when $n \to \infty$ and p_i is fixed and

   $$\ln\mathbb{P}(X^i|k_i) = PESEL(X^i|k_i) + O_{p_i}(1)$$

when $p_i \to \infty$ and n is fixed (see [24]). Thus,

$$\varphi(\Pi, k) = \sum_{i=1}^{K} PESEL(X^i|k_i) + R \ ,$$

where the ratio of R over $\sum_{i=1}^{K} PESEL(X^i|k_i)$ converges to zero in probability, under our asymptotic assumptions.
Therefore, the first step of VARCLUST leads to an increase of $\varphi(\Pi, k)$ up to Laplace approximation, i.e. with a large probability when for all $i \in \{1, \ldots, K\}$, $n >> p_i$ or $p_i >> n$.

2. **PCA and Partition step: choice of a partition, with cluster dimensions k_i^m fixed.** In the second step of the $m+1$-st iteration of VARCLUST, the cluster dimensions k_i^m are fixed, PCA is used to compute the cluster centers F^i and the columns of X are partitioned to different clusters by minimizing the BIC distance from F^i.
Below we assume that the priors $\pi_C(dC)$ and $\pi(d\sigma)$ satisfy classical regularity conditions for Laplace approximation ([4]). Now, let us define the k_i^m–dimensional linear space through the set of respective directions $F^i = (F_1^i, \ldots, F_{k_i}^i)$ with, as a natural prior, the uniform distribution π_F on the compact Grassman manifold F of free k_i-systems of $\mathbb{R}^n$. Moreover, we assume that the respective columns of coefficients $C^i = (C_1^i, \ldots, C_{k_i}^i)$ are independent with a prior distribution π_C on $\mathbb{R}^p$.
It holds

$$\begin{aligned}\log \mathbb{P}(X^i|k_i) &= \log \int_{F\times\sigma} \int_C \mathbb{P}(X^i|F^i, C^i, \sigma_i) \\ &\qquad \pi(dC^i)\pi(d\sigma_i)\pi_F(dF^i) \\ &= \log \int_{F\times\sigma} \prod_{\ell\in\Pi^i} \int \mathbb{P}(X_{\bullet\ell}|F^i, C_{\bullet\ell}) \\ &\qquad \pi_C(dC_{\bullet\ell})\pi(d\sigma_i)\pi_F(dF^i).\end{aligned}$$

When $n \gg k_i$, a Laplace-approximation argument leads to

$$\int \mathbb{P}(X_{\bullet\ell}|F^i, C_{\bullet\ell})\pi_C(dC_{\bullet\ell}) \approx e^{\mathrm{BIC}_\ell|F_i,\sigma_i}$$

where

$$\mathrm{BIC}_\ell|F_i, \sigma_i = \frac{1}{2}\left(-\frac{\|x_{\bullet\ell} - P_i(x_{\bullet\ell})\|^2}{\sigma_i^2} - k_i \ln n\right).$$

Thus, thanks to the Laplace approximation above,

$$\begin{aligned}&\log\mathbb{P}(X^i|k_i) \\ &\quad \approx \log \int_{F^i\times\sigma_i} e^{\sum_{\ell\in\Pi^i} \mathrm{BIC}_\ell|F_i,\sigma_i} \pi(d\sigma_i)\pi_F(dF^i)\end{aligned} \tag{4.4}$$

and

$$\begin{aligned}&\sum_{i=1}^{K} \log \mathbb{P}(X^i|k_i) \\ &\quad \approx \log \int_{F\times\sigma} e^{\sum_{i=1}^{K}\sum_{\ell\in\Pi^i} \mathrm{BIC}_\ell|F_i,\sigma_i} \pi(d\sigma)\pi_F(dF) \ .\end{aligned} \tag{4.5}$$

Now, by Laplace approximation, when $p_i >> k_i$, the right-hand side of (4.5) can be approximated by

$$\psi(\Pi|k) - \sum_{i=1}^{K} \frac{\dim F_i + 1}{2} \ln n, \tag{4.6}$$

where we denote

$$\psi(\Pi|k) = max_{(F,\sigma)}\xi(\Pi, F, \sigma|k), \tag{4.7}$$

$$\xi(\Pi, F, \sigma|k) = \sum_{i=1}^{K}\sum_{\ell\in\Pi^i}\left(-\frac{\|x_{\bullet\ell} - P_i(x_{\bullet\ell})\|^2}{\sigma_i^2} - \ln n\, k_i\right) \ . \tag{4.8}$$

Now, the term $\ln n \sum_{i=1}^{K} \frac{\dim F_i+1}{2}$ in (4.6) is the same for each Π, so increasing (4.5) is equivalent to increasing $\psi(\Pi|k)$.
Now, due to the well known Eckhart-Young theorem, for each $i \in \{1, \ldots, K\}$, the first k_i principal components of X^i form the basis for the linear space "closest" to X^i, i.e. the PCA part of VARCLUST allows to obtain F^m and σ^m, such that

$$(F^m, \sigma^m|\Pi^m, k^m) = argmax_{F,\sigma}\xi(\Pi^m, F, \sigma|k^m) \ .$$

Thus $\psi(\Pi^m|k^m) = \xi(\Pi^m, F^m, \sigma^m|k^m)$.
Finally, in the Partition (BIC) step of the algorithm the partition Π^{m+1} is selected such that

$$\Pi^{m+1}|E^m, \sigma^m, k^m = argmax_\Pi \xi(\Pi, E^m, \sigma^m|k^m) \ .$$

In the result it holds that

$$\psi(\Pi^{m+1}|k^m) \geq \psi(\Pi^m|k^m)$$

and consequently,

$$\varphi(\Pi^{m+1}, k^m) \geq \varphi(\Pi^m, k^m) \ ,$$

with a large probability if only $k_i << min(n, p_i)$ for all $i \in \{1, \ldots, K\}$.

The combination of results for both steps of the algorithm implies

Corollary 2 *In the VARCLUST algorithm, the objective function $\varphi(\Pi^{m+1}, k^m)$ increases with m with a large probability if for all $i \in \{1, \ldots, K\}$, $k_i << min(n, p_i)$ and one of the following two conditions holds: $n >> p_i$ or $p_i >> n$.*

Remark 1 The above reasoning illustrates that both steps of VARCLUST asymptotically lead to an increase of the same objective function. The formula (4.8) suggests that this function is bounded, which implies that VARCLUST converges with a large probability. In Figure 7 we illustrate the convergence of VARCLUST based on the more general version of BIC (3.4) and a rather systematic increase of the mBIC approximation to the model posterior probability

$$mBIC(K, \Pi, k) = \sum_{i=1}^{K} PESEL(X_m^i | k_i^m) - p \ln K - K \ln d$$

in consecutive iterations of the algorithm.

5 Simulation study

In this section, we present the results of simulation study, in which we compare VARCLUST with other methods of variable clustering. To assess the performance of the procedures we measure their effectiveness and execution time. We also use VARCLUST to estimate the number of clusters in the data set. In all simulations we use VARCLUST based on the more general version of BIC (3.4).

5.1 Clustering methods

In our simulation study we compare the following methods:

1. Sparse Subspace Clustering (SSC, [11])
2. Low Rank Subspace Clustering (LRSC, [28])
3. VARCLUST with multiple random initializations. In the final step, the initialization with the highest mBIC is chosen.
4. VARCLUST with initialization by the result of SSC (VARCLUST$_{aSSC}$)
5. ClustOfVar (COV, [29], [8])

The first two methods are based on spectral clustering and detailed description can be found in the given references. For the third considered procedure we use the one-dimensional random initialization. This means that we sample without replacement K variables which are used as one dimensional centers of K clusters. The fourth method takes advantage of the possibility to provide the initial segmentation before the start of the VARCLUST procedure. It accelerates the method, because then there is no need to run it many times with different initializations. We build the centers by using the second step of VARCLUST (PESEL and PCA) for a given segmentation. In this case we use the assignment of the variables returned by SSC. Finally, we compare mentioned procedures with COV, which VARCLUST is an extended version of. COV also exploits k-means method. Initial clusters' centers are chosen uniformly at random from the data. Unlike in VARCLUST the center of a cluster is always one variable. The similarity measure is squared Pearson correlation coefficient. After assignment of variables, for every cluster PCA is performed to find the first principal component and make it a new cluster center. VARCLUST aims at overcoming the weaknesses of COV. Rarely in applications the subspace is generated by only one factor and by estimating the dimensionality of each cluster VARCLUST can better reflect the true underlying structure.

5.2 Synthetic data generation

To generate synthetic data to compare the methods from the previous section we use two generation procedures detailed in algorithms 1 and 2. Later we refer to them as modes. Factors spanning the subspaces in the first mode are shared between clusters, whereas in the second mode subspaces are independent. As an input to both procedures we use: n - number of individuals, SNR - signal to noise ratio, K - number of clusters, p - number of variables, d - maximal dimension of a subspace. SNR is the ratio of the power of signal to the power of noise, i.e., $SNR = \frac{\sigma^2}{\sigma_\varepsilon^2}$ the ratio of variance of the signal to the variance of noise.

Algorithm 1 Data generation with shared factors

Require: n, SNR, K, p, d
  Number of factors $m \leftarrow K\frac{d}{2}$
  Factors $F = (f_1, \ldots, f_m)$ are generated independently from the multivariate standard normal distribution and then F is scaled to have columns with mean 0 and standard deviation 1
  Draw subspaces dimension $d_1, \ldots d_K$ uniformly from $\{1, \ldots, d\}$
  for $i = 1, \ldots, K$ **do**
    Draw i-th subspace basis as sample of size d_i uniformly from columns of F as F^i
    Draw matrix of coefficients C_i from $\mathcal{U}(0.1, 1) \cdot sgn(\mathcal{U}(-1, 1))$
    Variables in the i-th subspace are $X^i \leftarrow F^i C_i$
  end for
  Scale matrix $X = (X_1, \ldots, X_K)$ to have columns with unit variance
  return $X + Z$ where $Z \sim \mathcal{N}(0, \frac{1}{SNR} I_n)$

Algorithm 2 Data generation with independent subspaces

Require: n, SNR, K, p, d
Draw subspaces' dimension $d_1, \dots d_K$ uniformly from $\{1, \dots, d\}$
for $i = 1, \dots, K$ **do**
Draw i-th subspace basis F^i as sample of size d_i from multivariate standard normal distribution
Draw matrix of coefficients C_i from $\mathcal{U}(0.1, 1) \cdot sgn(\mathcal{U}(-1, 1))$
Variables in i-th subspace are $X^i \leftarrow F^i C_i$
end for
Scale matrix $X = (X_1, \dots, X_K)$ to have columns with unit variance
return $X + Z$ where $Z \sim \mathcal{N}(0, \frac{1}{SNR} I_n)$

5.3 Measures of effectiveness

To compare clustering produced by our methods we use three measures of effectiveness.

1. Adjusted Rand Index - one of the most popular measures. Let A, B be the partitions that we compare (one of them should be true partition). Let a, b, c, d denote respectively the number of pairs of points from data set that are in the same cluster both in A and B, that are in the same cluster in A but in different clusters in B, that are in the same cluster in B but in different clusters in A and that are in the different clusters both in A and B. Note that the total number of pairs is $\binom{p}{2}$. Then

$$ARI = \frac{\binom{p}{2}(a+d) - [(a+b)(a+c)+(b+d)(c+d)]}{\binom{p}{2}^2 - [(a+b)(a+c)+(b+d)(c+d)]}$$

The maximum value of ARI is 1 and when we assume that every clustering is equally probable its expected value is 0. For details check [16].
The next two measures are taken from [26]. Let $X = (x_1, \dots x_p)$ be the data set, A be a partition into clusters $A_1, \dots A_n$ (true partition) and B be a partition into clusters $B_1, \dots, B_m$.
2. Integration - for the cluster A_j it is given by formula

$$Int(A_j) = \frac{max_{k=1,\dots,m} \#\{i \in \{1, \dots p\} : X^i \in A_j \wedge X^i \in B_k\}}{\#A_j}$$

Cluster B_k for which the maximum is reached is called integrating cluster of A_j. Integration can be interpreted as the percentage of data points from given cluster of true partition that are in the same cluster in partition B. For the whole clustering

$$Int(A, B) = \frac{1}{n} \sum_{j=1}^{n} Int(A_j)$$

3. Acontamination - for cluster A_j it is given by formula

$$Acont(A_j) = \frac{\#\{i \in \{1, \dots p\} : X^i \in A_j \wedge X^i \in B_k\}}{\#B_k}$$

where B_k is integrating cluster for A_j. Idea of acontamination is complementary to integration. It can be interpreted as the percentage of the data in the integrating cluster B_k are from A_j. For the whole clustering

$$Acont(A, B) = \frac{1}{n} \sum_{j=1}^{n} Acont(A_j)$$

Note that the bigger ARI, integration and acontamination are, the better is the clustering. For all three indices the maximal value is 1.

5.4 Simulation study results

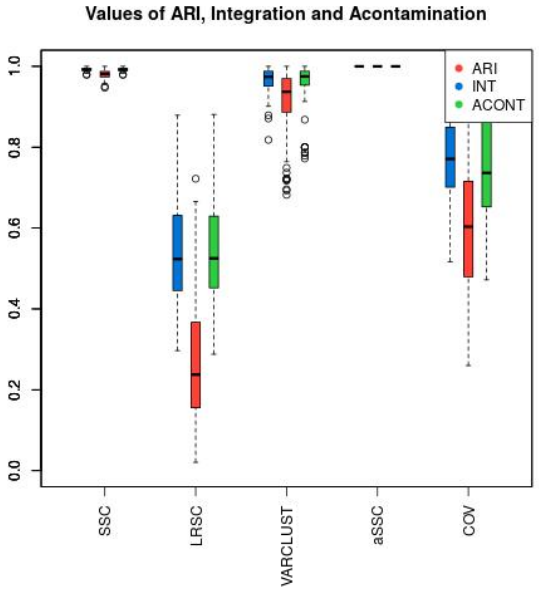
In this section we present the outcome of the simulation study. We generate the synthetic data 100 times. We plot multiple boxplots to compare clusterings of different methods. By default the number of runs (random initializations) is set to $n_{init} = 30$ and the maximal number of iterations within the k-means loop is set to $n_{iter} = 30$. Other parameters used in given simulation are written above the plots. They include parameters from data generation algorithms (1, 2) as well as *mode* indicating which of them was used.

5.4.1 Generation method

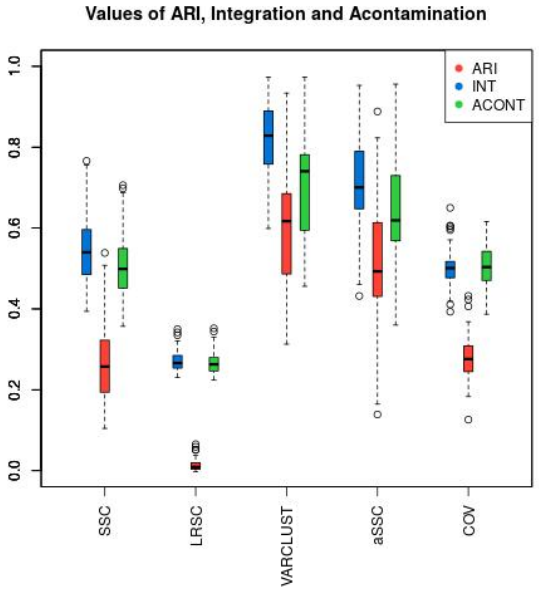
In this section we compare the methods with respect to the parameter *mode*, which takes the value *shared* (data generated using 1), if the subspaces may share the factors, and the value *not_shared* (data generated using 2) otherwise (Figure 1). When the factors are not shared, SSC and VARCLUST provide almost perfect clustering. We can see that in case of shared factors the task is more complex. All the methods give worse results in that case. However, VARCLUST and VARCLUST$_{aSSC}$ outperform all the other procedures and supply acceptable clustering in contrast to SSC, LRSC and COV. The reason for that is the mathematical formulation of SSC and LRSC - they assume that the subspaces are independent and do not have common factors in their bases.

Fig. 1: Comparison with respect to the data generation method. Simulation parameters: $n = 100$, $p = 800$, $K = 5$, $d = 3$, $SNR = 1$.

(a) factors not shared



(b) shared factors



5.4.2 Number of variables

In this section we compare the methods with respect to the number of variables (Figure 2). When the number of features increases, VARCLUST tends to produce better clustering. For our method this is an expected effect because when the number of clusters and subspace dimension stay the same we provide more information about the cluster's structure with every additional predictor. Moreover, PESEL from (3.1) gives a better approximation of the cluster's dimensionality and the task of finding the real model becomes easier. However, for COV, LRSC, SSC this does not hold as the results are nearly identical.

5.4.3 Maximal dimension of subspace

We also check what happens when the number of parameters in the model of VARCLUST increases. In Figure 3, in the first column, we compare the methods with respect to the maximal dimension of a subspace ($d = 3, 5, 7$). However, in real-world clustering problems it is common that it is not known. Therefore, in the second column, we check the performance of VARCLUST and VARCLUST$_{aSSC}$ when the given maximal dimension as a parameter is twice as large as maximal dimension used to generate the data.

Looking at the first column, we can see that the effectiveness of VARCLUST grows slightly when the maximal dimension increases. However, this effect is not as noticeable as for SSC. It may seem unexpected for VARCLUST but variables from subspaces of higher dimensions are easier to distinguish because their bases have more independent factors. In the second column, the effectiveness of the methods is very similar to the first column except for $d = 3$, where the difference is not negligible. Nonetheless, these results indicate that thanks to PESEL, VARCLUST performs well in terms of estimating the dimensions of the subspaces.

5.4.4 Number of clusters

The number of the parameters in the model for VARCLUST grows significantly with the number of clusters in the data set. In Figure 4 we can see that for VARCLUST the effectiveness of the clustering diminishes when the number of clusters increases. The reason is the larger number of parameters in our model to estimate. The opposite effect holds for LRSC, SSC and COV, although it is not very apparent.

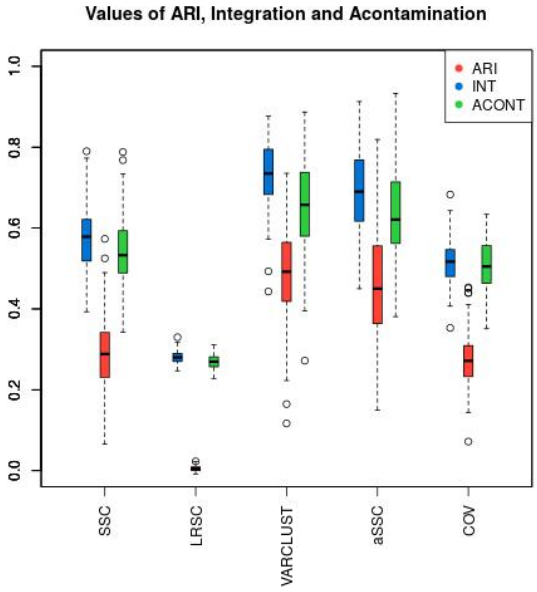
5.4.5 Signal to noise ratio

One of the most important characteristics of the data set is signal to noise ratio (SNR). Of course, the problem of clustering is much more difficult when SNR is small because the corruption caused by noise dominates the data. However, it is not uncommon in practice to find data for which $SNR < 1$.

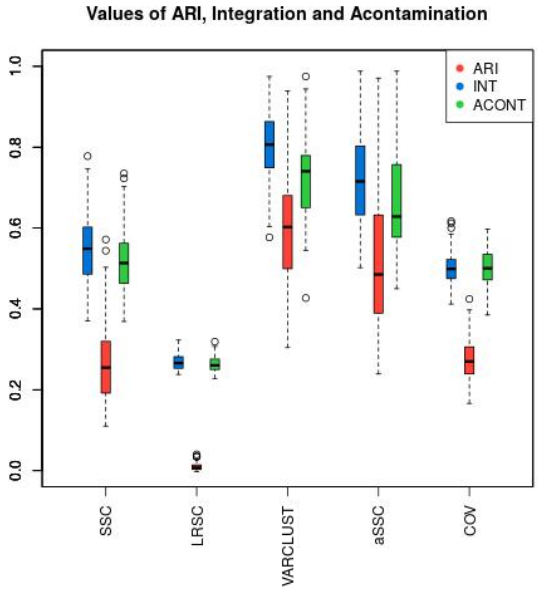
In Figure 5 we compare our methods with respect to SNR. For $SNR = 0.5$, VARCLUST supplies a decent clustering. In contrary, SSC and

Fig. 2: Comparison with respect to the number of variables. Simulation parameters: $n = 100$, $K = 5$, $d = 3$, $SNR = 1$, *mode* : *shared*.

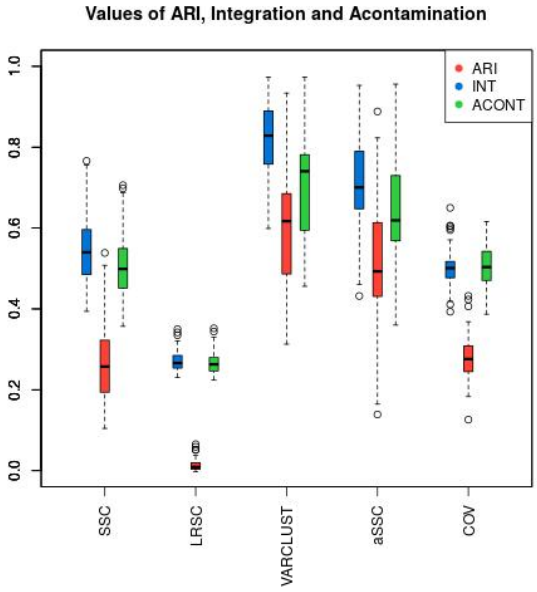
(a) $p = 300$



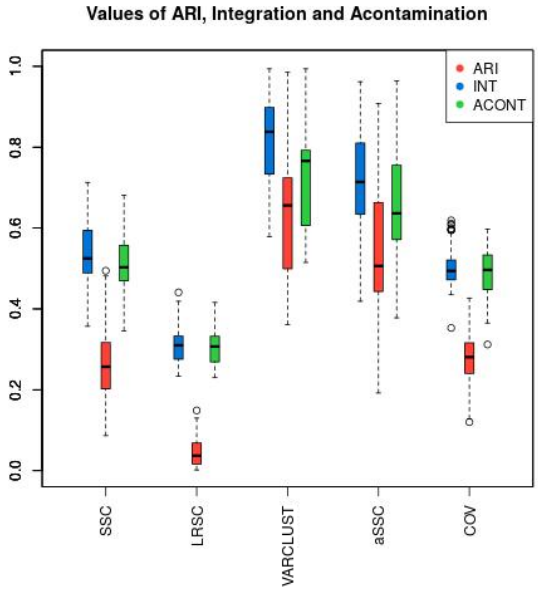
(b) $p = 600$



(c) $p = 800$



(d) $p = 1500$



LRSC perform poorly. All methods give better results when SNR increases, however for SSC this effect is the most noticeable. For $SNR \geq 1$, SSC produces perfect or almost perfect clustering while VARCLUST performs slightly worse.

5.4.6 Estimation of the number of clusters

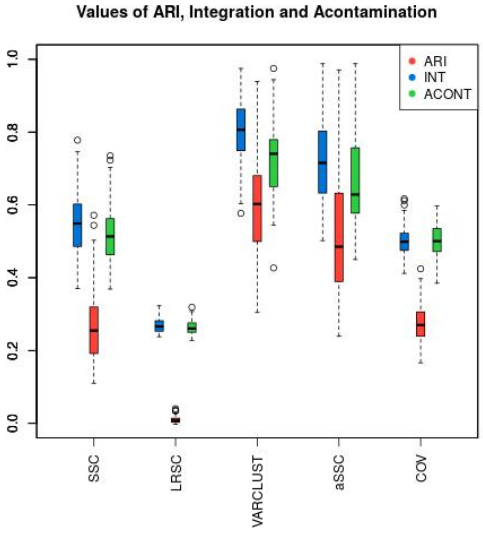
Thanks to mBIC, VARCLUST can be used for automatic setection of the number of clusters. We generate the data set with given parameters 100 times and check how often each number of clusters from range $\left[K - \frac{K}{2}, K + \frac{K}{2}\right]$ is chosen (Figure 6). We see that for $K = 5$ the correct number of clusters was chosen most times. However, when the number of clusters increases, the clustering task becomes more difficult, the number of parameters in the model grows and VARCLUST tends to underestimate the number of clusters.

5.4.7 Number of iterations

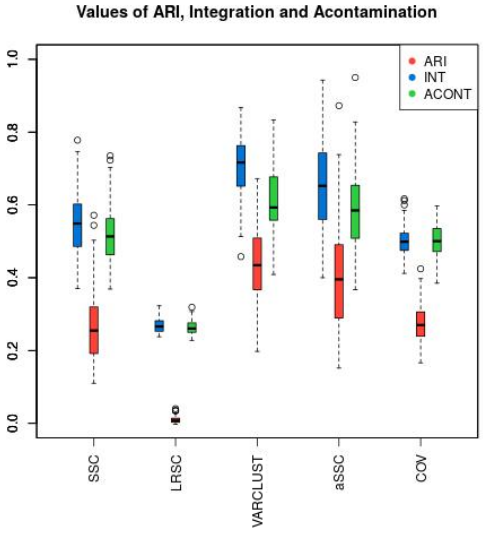
In this section we investigate convergence of mBIC within k-means loop for four different initializa-

Fig. 3: Comparison with respect to the number of variables. Simulation parameters: $n = 100$, $p = 600$, $K = 5$, $SNR = 1$, $mode$: $shared$. In the left column the maximal dimension passed to VARCLUST was equal to d, in the right we passed $2d$.

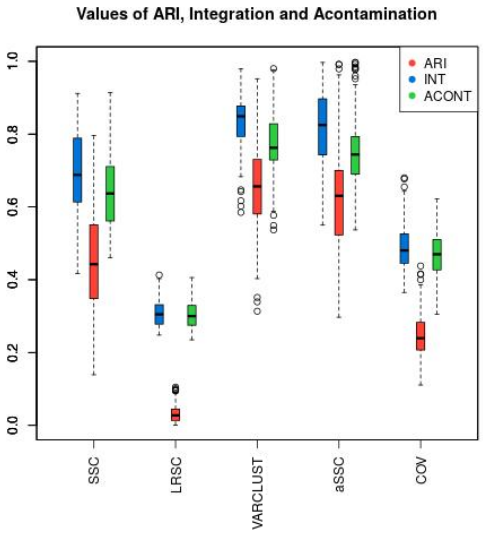
(a) $d = 3$



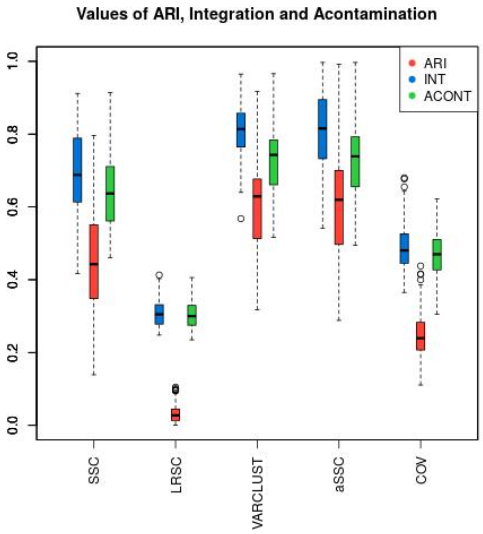
(b) $d = 3$



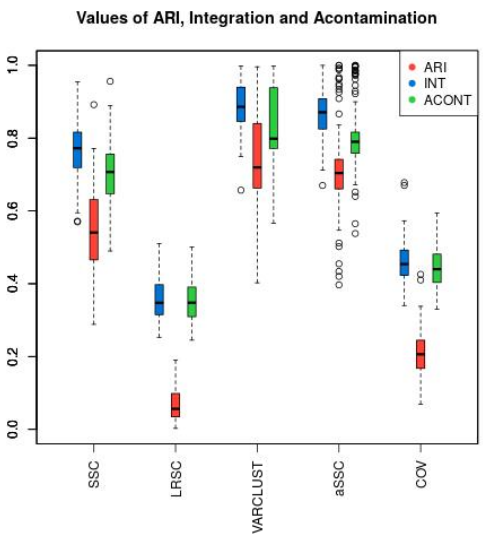
(c) $d = 5$



(d) $d = 5$



(e) $d = 7$



(f) $d = 7$

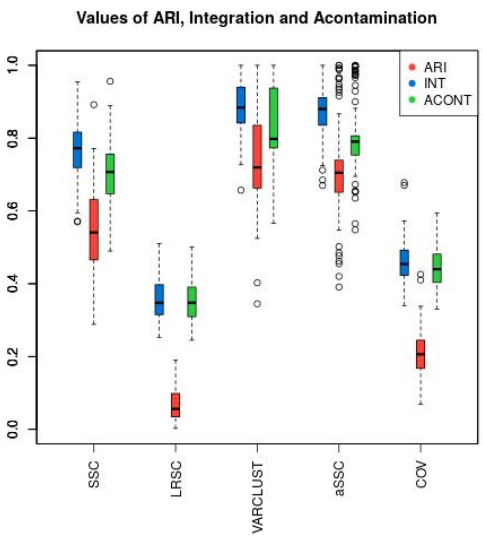
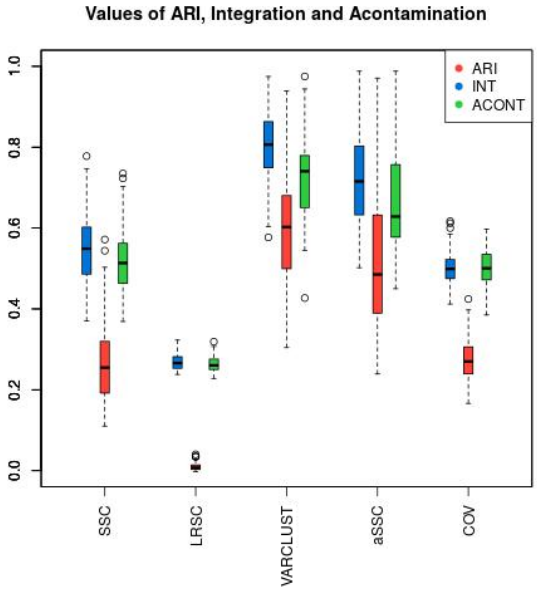
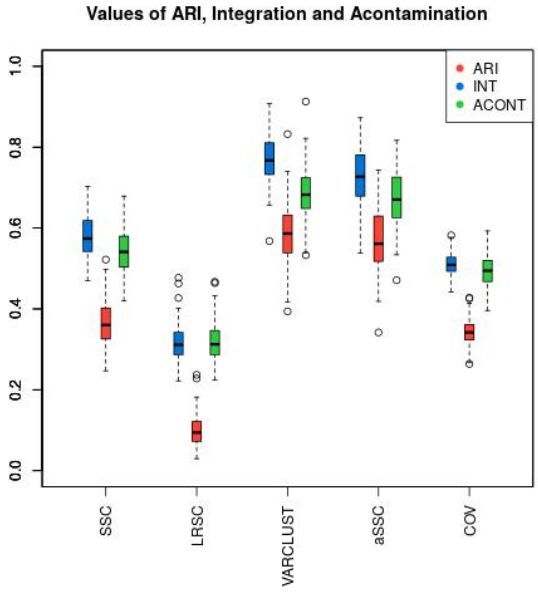


Fig. 4: Comparison with respect to the number of clusters. Simulation parameters: $n = 100$, $p = 600$, $d = 3$, $SNR = 1$, $mode$: *not shared*.

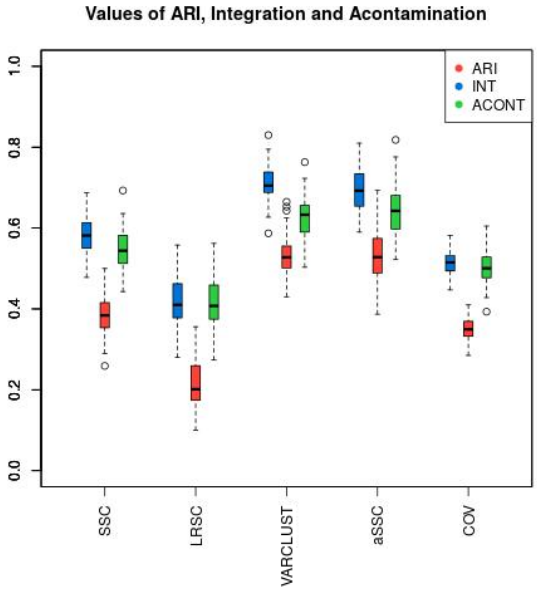
(a) $K = 5$



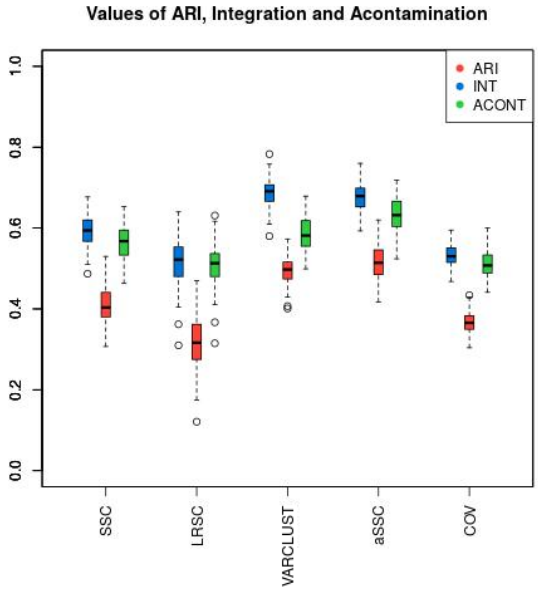
(b) $K = 10$



(c) $K = 15$



(d) $K = 20$



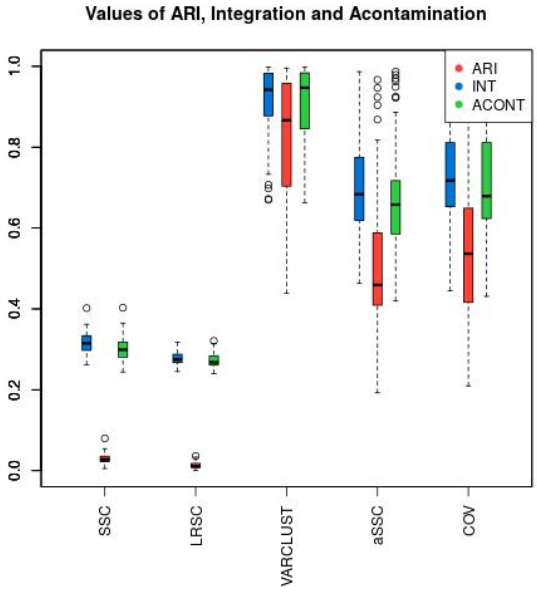
tions (Figure 7). We can see that it is quite fast: in most cases it needed no more than 20 iterations of the k-means loop. We can also notice that the size of the data set (in this case the number of variables) has only small impact on the number of iterations needed till convergence. However, the results in Figure 7 show that multiple random initializations in our algorithm are required to get satisfying results - the value of mBIC criterion varies a lot between different initializations.

5.4.8 Execution time

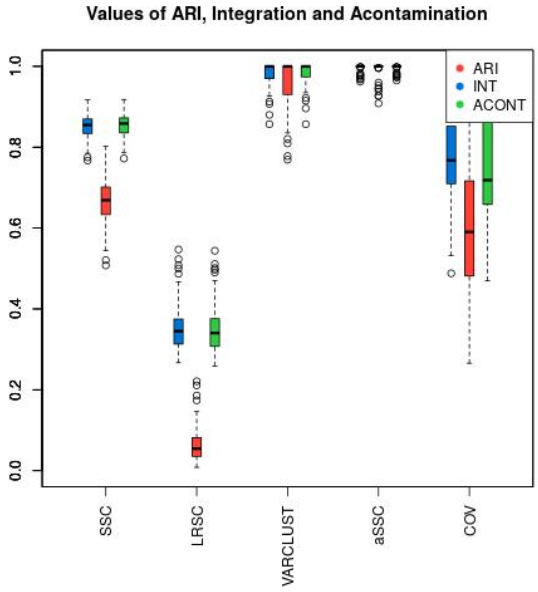
In this section we compare the execution times of compared methods. They were obtained on the machine with *Intel(R) Core(TM) i7-4790 CPU 3.60GHz, 8 GB RAM*. The results are in Figure 8. For the left plot $K = 5$ and for the right one $p = 600$. On the plots for both VARCLUST and COV we used only one random initialization. Therefore, we note that for $n_{init} = 30$ the execution time of VARCLUST will be larger. However, not by exact factor of n_{init} thanks to parallel im-

Fig. 5: Comparison with respect to the signal to noise ratio. Simulation parameters: $n = 100$, $p = 600$, $K = 5$, $d = 3$, *mode* : *not shared*.

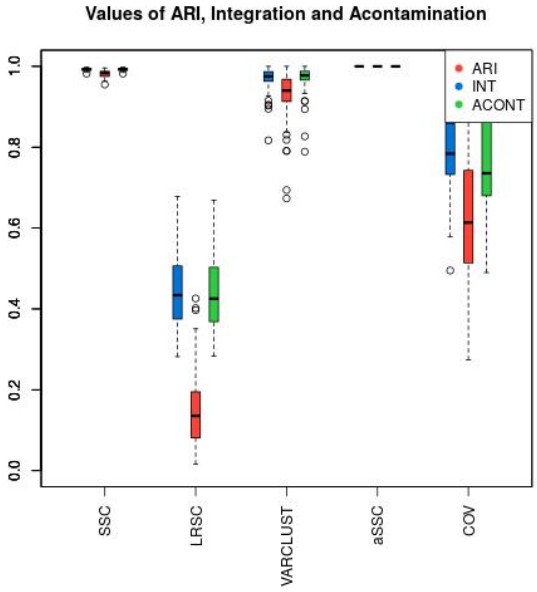
(a) $SNR = 0.5$



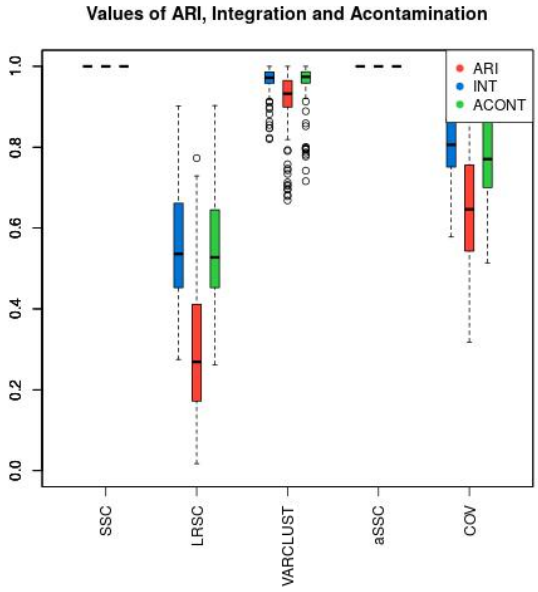
(b) $SNR = 0.75$



(c) $SNR = 1$



(d) $SNR = 2$



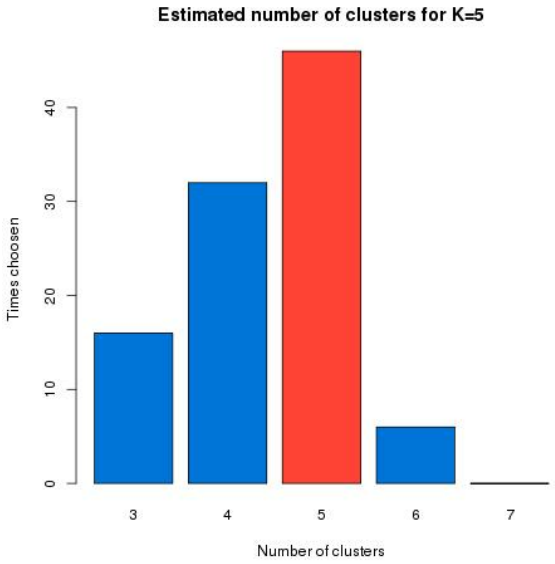
plementation in [25]. Nonetheless, VARCLUST is the most computationally complex of these methods. We can see that COV and SSC do not take longer for bigger number of clusters when the opposite holds for VARCLUST and LRSC. What is more, when the number of variables increases, the execution time of SSC grows much more rapidly than time of one run of VARCLUST. Therefore, for bigger data sets it is possible to test more random initializations of VARCLUST in the same time as computation of SSC. Furthermore, running VARCLUST with segmentation returned by SSC (enhancing the clustering) is not much more time consuming than SSC itself.

5.4.9 Discussion of the results

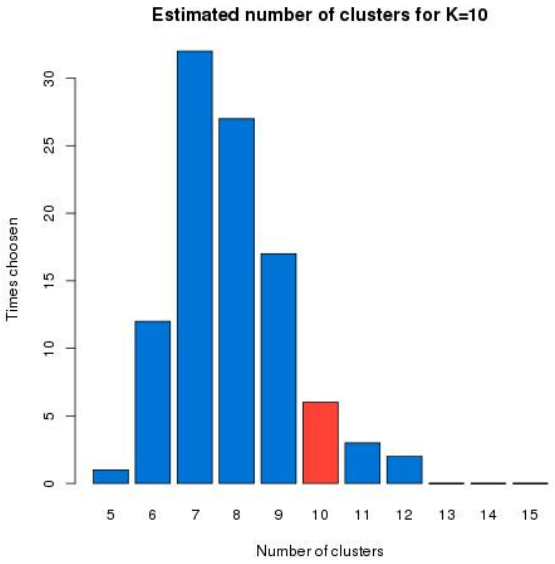
The simulation results prove that VARCLUST is an appropriate method for variable clustering. As one of the very few approaches, it is adapted to the data dominated by noise. One of its biggest advantages is a possibility to recognize subspaces which share factors. It is also quite robust to in-

Fig. 6: Estimation of the number of clusters. Simulation parameters: $n = 100,\ p = 600,\ d = 3,\ SNR = 1\ mode : not\ shared.$

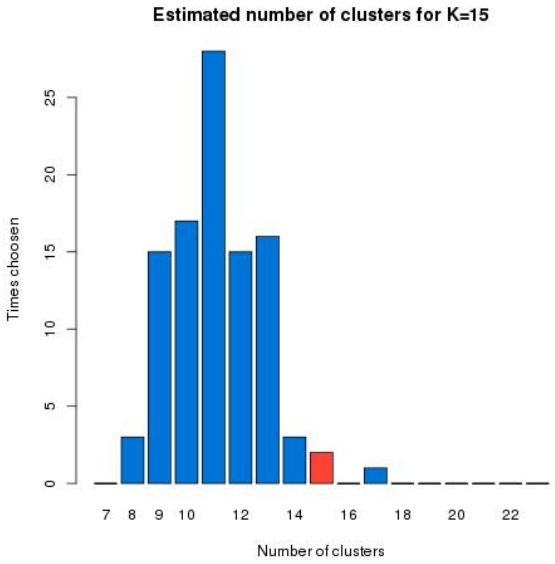
(a) $K = 5$



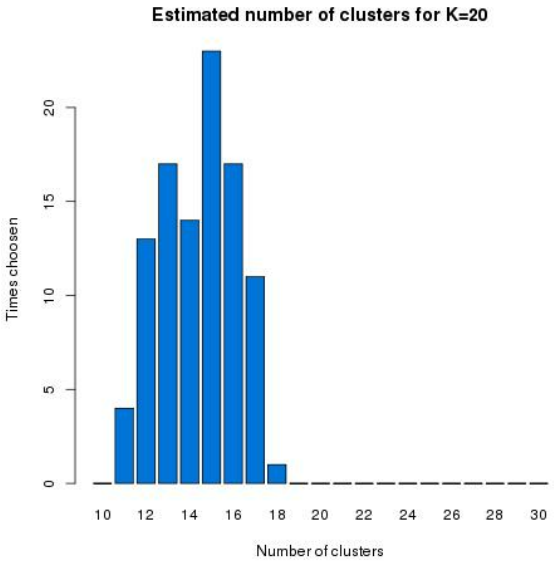
(b) $K = 10$



(c) $K = 15$



(d) $K = 20$



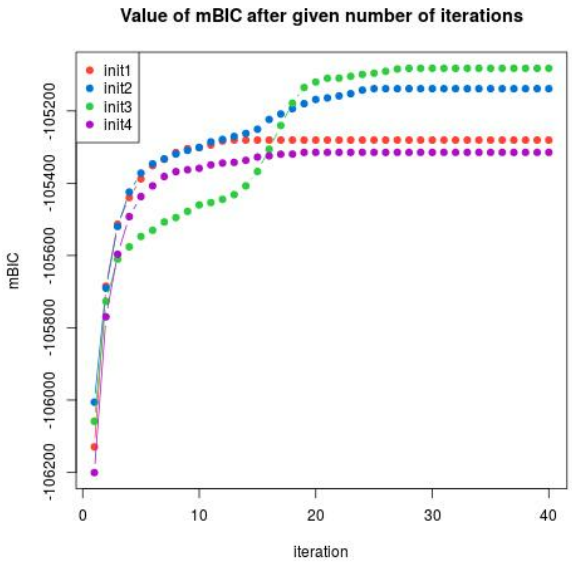
crease in the maximal dimension of a subspace. Furthermore, it can be used to detect the number of clusters in the data set. Last but not least, in every setting of the parameters used in our simulation, VARCLUST outperformed LRSC and COV and did better or as well as SSC. The main disadvantage of VARCLUST is its computational complexity. Therefore, to reduce the execution time one can provide custom initialization as in VARCLUST$_{aSSC}$. This method in all cases provided better results than SSC, so our algorithm can also be used to enhance the clustering results of the other methods. The other disadvantage of VARCLUST is a problem with the choice of the parameters n_{init} or n_{iter}. Unfortunately, when data size increases, in order to get acceptable clustering we have to increase at least one of these two values. However, it is worth mentioning that in case of parameters used in out tests $n_{init} = 30$ and the maximal number of iterations equal to 30 on a machine with 8 cores the execution time of VARCLUST is comparable with execution time of SSC.

6 Applications to real data analysis

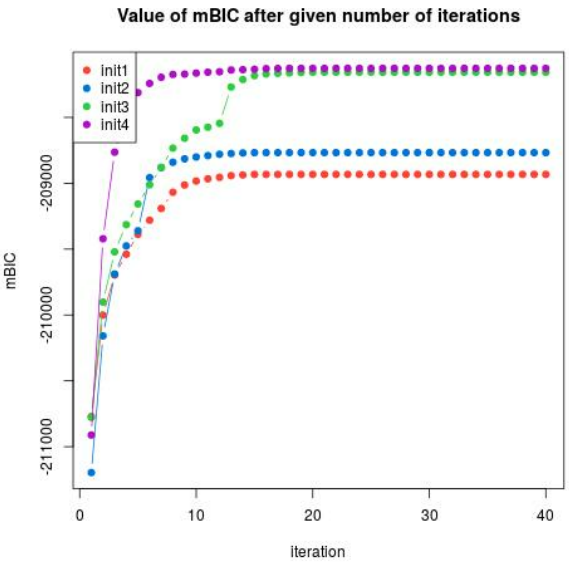
In this section we apply VARCLUST to two different data sets and show that our algorithm can

Fig. 7: mBIC with respect to the number of iterations for 4 different initializations. Simulation parameters: $n = 100$, $K = 5$, $d = 3$, $SNR = 1$ $mode : shared$.

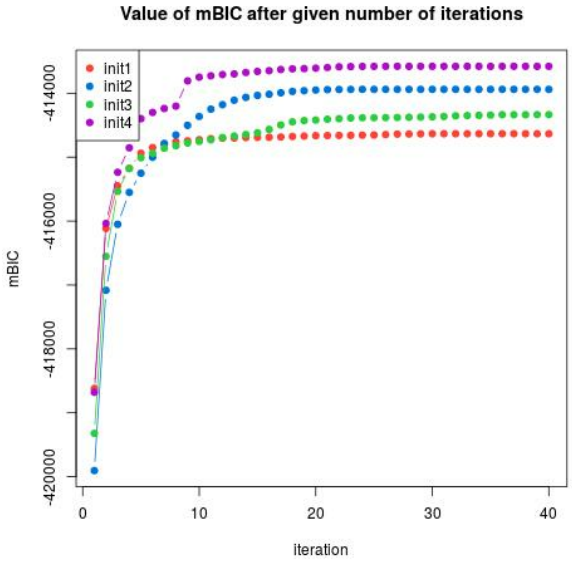
(a) $p = 750$



(b) $p = 1500$



(c) $p = 3000$



produce meaningful, interpretable clustering and dimensionality reduction.

6.1 Meteorological data

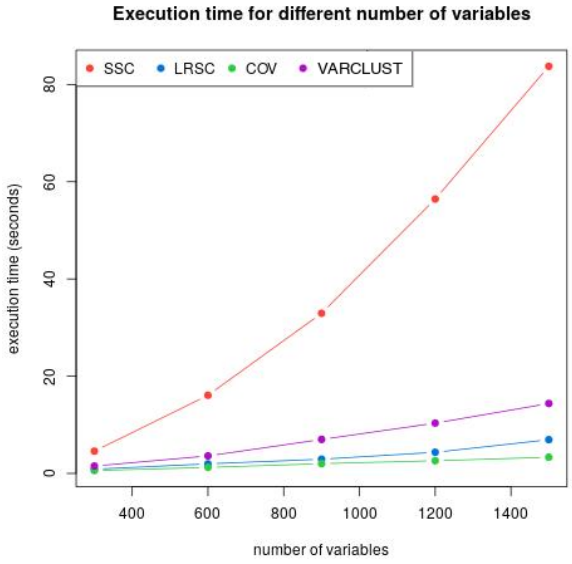
First, we will analyze air pollution data from Kraków, Poland [1]. This example will also serve as a short introduction to the `varclust` R package.

6.1.1 About the data

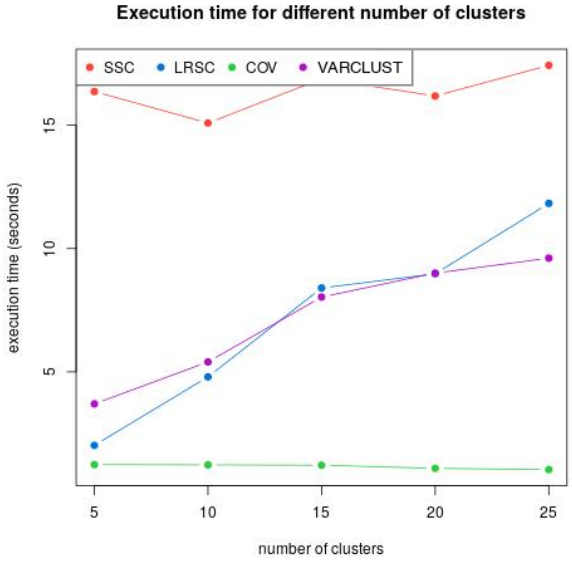
Krakow is one of the most polluted cities in Poland and even in the world. This issue has gained enough recognition to inspire several grass-root initiatives that aim to monitor air quality and inform citizens about health risks. *Airly* project created a huge network of air quality sensors which were deployed across the city. Information gathered by the network is accessible via the `map.airly.eu` website. Each of 56 sensors measures temperature, pressure, humidity and levels of particulate matters PM1, PM2.5 and PM10 (number corresponds to the mean diameter). This way, air quality is described by 336 variables. Measurements are done on an hourly basis.

Fig. 8: Comparison of the execution time of the methods with respect to p and K. Simulation parameters:$n = 100$, $d = 3$, $SNR = 1$ $mode : shared$.

(a) With respect to the number of variables



(b) With respect to the number of clusters



Here, we used data from one month. We chose March, because in this month the number of missing values is the smallest. First, we removed non-numerical variables from the data set. We remove columns with a high percentage (over 50%) of missing values and impute the other by the mean. We used two versions of the data set: `march_less` data frame containing hourly measurements (in this case number of observations is greater than number of variables) and `march_daily` containing averaged daily measurements (which satisfies the $p \gg n$ assumption). Results for both versions are consistent. The dimensions of the data are 577×263 and 25×263, respectively. Both data sets along with R code and results are available on `https://github.com/mstaniak/varclust_example`

6.1.2 Clustering based on random initialization

When the number of clusters is not known, we can use the `mlcc.bic` function which finds a clustering of variables with an estimated number of clusters and also returns factors that span each cluster. A minimal call to `mlcc.bic` function requires just the name of a data frame in which the data are stored.

```
varclust_minimal <-
mlcc.bic(march_less, greedy = F)
```

The returned object is a list containing the resulting segmentation of variables (`segmentation` element), a list with matrices of factors for each cluster, mBIC for the chosen model, list describing dimensionality of each cluster and models fitted in other iterations of the algorithm (non-optimal models). By default, at most 30 iterations of the greedy algorithm are used to pick a model. Also by default it is assumed that the number of clusters is between 1 and 10, and the maximum dimension of a single cluster is 4. These parameters can be tweaked. Based on comparison of mBIC values for clustering results with different maximum dimensions, we selected 6 as the maximum dimension.

```
varclust_clusters =
mlcc.bic(march_less, greedy = TRUE,
flat.prior = TRUE, max.dim = 6)
```

To minimize the impact of random initialization, we can run the algorithm many times and select best clustering based on the value of mBIC criterion. We present results for one of clusterings obtained this way.

We can see that variables describing temperature, humidity and pressure were grouped in four clusters (with pressure divided into two clusters and homogenous clusters for humidity and temperature related variables), while variables that describe levels of particulate matters are spread among different clusters that do not describe simply one size of particulate matter (1, 2.5 or 10), which may imply that measurements are in a sense non-homogenous. In Figure 9 we show how these clusters are related to geographical locations.

6.1.3 Clustering based on SSC algorithm

The `mlcc.bic` function performs clustering based on a random initial segmentation. When the number of clusters is known or can be safely assumed,

we can use the `mlcc.reps` function, which can start from given initial segmentations or a random segmentation. We will show how to initialize the clustering algorithm with a fixed grouping. For illustration, we will use results of Sparse Subspace Clustering (SSC) algorithm. SSC is implemented in a Matlab package maintained by Ehsan Elhamifar [11]. As of now, no R implementation of SSC is available. We store resulting segmentations for numbers of clusters from 1 to 20 in vectors called `clx`, where x is the number of clusters. Now the calls to `mlcc.reps` function should look like the following example.

```
vclust10 <- mlcc.reps(march_less,
        numb.clusters=10,max.iter=50,
        initial.segmentations=list(cl10))
```

The result is a list with a number of clusters (`segmentation`), calculated mBIC and a list of factors spanning each of the clusters. For both initialization methods, variability of results regarding the number of clusters diminished by increasing the `numb.runs` argument to `mlcc.bic` and `mlcc.reps` functions which control the number of runs of the k-means algorithm.

6.1.4 Conclusions

We applied VARCLUST algorithm to data describing air quality in Kraków. We were able to reduce the dimensionality of the data significantly. It turns out that for each characteristics: temperature, humidity and the pressure, measurements made in 56 locations can be well represented by a low dimensional projection found by Varclust. Additionally, variables describing different particulate matter levels can be clustered into geographically meaningful groups, clearly separating the center and a few bordering regions. If we were to use these measurements as explanatory variables in a model describing for example effects of air pollution on health, factors that span clusters could be used instead as predictors, allowing for a significant dimension reduction.

The results of the clustering are random by default. Increasing the number of runs of k-means algorithm and maximum number of iterations of the algorithm stabilize the results. Increasing these parameters also increases the computation time. Another way to remove randomness is to select an initial clustering using another method. In the examples, clustering based on SSC algorithm was used.

The `mlcc.bic` function performs greedy search by default, meaning that the search stops after first decrease in mBIC score occurs. On the one hand, this might lead to suboptimal choice of number of clusters, so setting `greedy` argument to `FALSE` might be helpful, but on the other hand, the criterion may become unstable for some larger numbers of clusters.

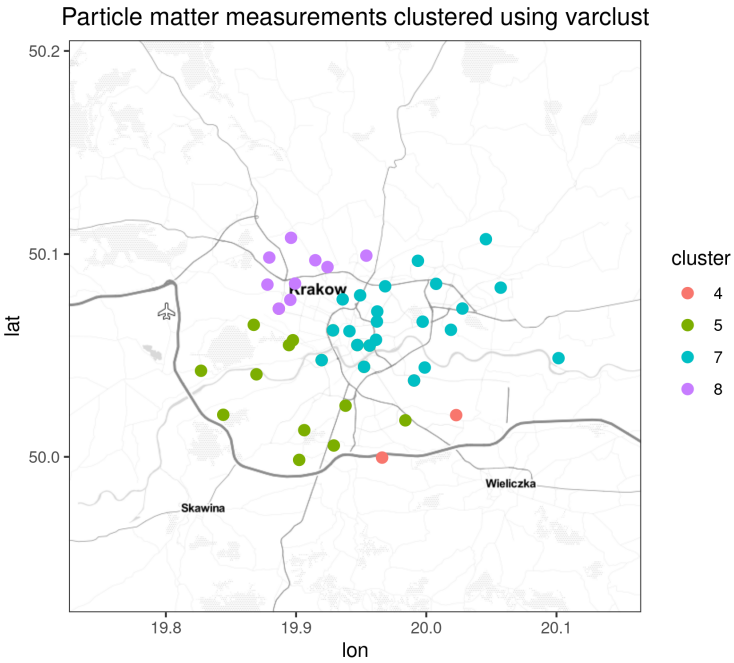


Fig. 9: Clusters of variables describing particulate matter levels on a map of Krakow. Without any prior knowledge on spatial structure, VARCLUST groups variables corresponding to sensors located near each other.

6.2 TCGA Breast Cancer Data

In the next subsection, the VARCLUST clustering method is applied on large open-source

data generated by The Cancer Genome Atlas (TCGA) Research Network, available on `http://cancergenome.nih.gov/`. TCGA has profiled and analyzed large numbers of human tumours to discover molecular aberrations at the DNA, RNA, protein, and epigenetic levels. In this analysis, we focus on the Breast Cancer cohort, made up of all patients reviewed by the TCGA Research Network, including all stages and all anatomopathological characteristics of the primary breast cancer disease, as in [6].

The genetic informations in tumoral tissues DNA that are involved in gene expression are measured from messenger RNA (mRNA) sequencing. The analysed data set is composed of $p = 60488$ mRNA transcripts for $n = 1208$ patients.

For this data set, our objective is twofold. First, from a machine learning point of view, we hope that this clustering procedure will provide a sufficiently efficient dimension reduction in order to improve the forecasting issues related to the cancer, for instance the prediction of the reaction of patients to a given treatment or the life expectancy in terms of the transcriptomic diagnostic.
Second, from a biological point of view, the clusters of gene expression might be interpreted as distinct biological processes. Then, a way of measuring the quality of the VARCLUST method is to compare the composition of the selected clusters with some biological pathways classification (see Figure 10). More precisely, the goal is to check if the clusters constructed by VARCLUST correspond to already known biological pathways (Gene Ontology, [12]).

6.2.1 Data extraction and gene annotations

This ontological classification aims at doing a census of all described biological pathways. To grasp the subtleties inherent to biology, it is important to keep in mind that one gene may be involved in several biological pathways and that most of biological pathways are slot or associated with each other. The number of terms on per Biological process ontology was 29687 in January 2019 while the number of protein coding genes is around 20000. Therefore, one cannot consider each identified biological process as independent characteristic.

The RNASeq raw counts were extracted from the TCGA data portal. The scaling normalization and log transformation ([23]) were computed using voom function ([18]) from limma package version 3.38.3 ([22]). The gene annotation was realised with biomaRt package version 2.38.0 ([9], [10]).

The enrichment process aims to retrieve a functional profile of a given set of genes in order to better understand the underlying biological processes. Therefore, we compare the input gene set (i.e, the genes in each cluster) to each of the terms in the gene ontology. A statistical test can be performed for each bin to see if it is enriched for the input genes. It should be mentioned that all genes in the input genes may not be retrieved in the Gene Ontology Biological Process and conversely, all genes in the Biological Process may not be present in the input gene set. To perform the GO enrichment analysis, we used GoFuncR package [13] version 1.2.0. Only Biological Processes identified with Family-wise Error Rate p-value < 0.05 were reviewed. Data processing and annotation enrichment were performed using R software version 3.5.2.

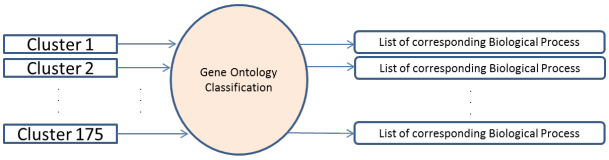


Fig. 10: Bioinformatic annotation process for each cluster identified by VARCLUST

6.2.2 Evolution of the mBIC and clusters strucure

The number of clusters to test was fixed to 50, 100, 150, 175, 200, 225, 250. The maximal subspace dimension was fixed to 8, the number of runs was 40, and the maximal number of iterations of the algorithm was 30.

As illustrated in the Figure 12, the mBICs remain stable from the 35th iteration. The mBIC is not *a.s.* increasing between 50 and 250 clusters sets. The mBIC for $K = 175$ and $K = 250$ clusters sets were close. The proportion of clusters with only one principal component is also higher for $K = 175$ and $K = 250$ clusters sets.

6.2.3 Biological specificity of clusters

In this subsection, we focus on some biological interpretations in the case: $K = 175$ clusters.
In order to illustrate the correspondance between the genes clustering and the biological annotations in Gene Ontology, we have selected one cluster with only one Gene Ontology Biological Process (Cluster number 3) and one cluster with two Gene Ontology Biological processes (Cluster number 88). We keep this numbering notation in the sequel.
Among the 98 genes in Cluster 3, 70 (71.4%, called "Specific Genes") were reported in the GO Biological process named *calcium-independent cell-cell adhesion via plasma membrane, cell-adhesion*

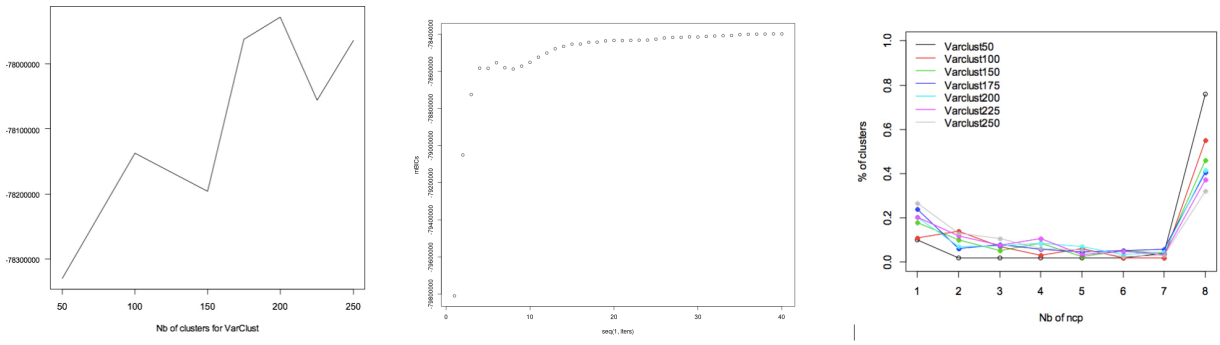


Fig. 11: Left: evolution of the mBIC with the number of clusters; middle: evolution of the mBIC with the number of iterations; right: number of principal components in clusters in terms of K.

molecules (*GO* : 0016338). The number of principal components in this cluster was 8 (which may indicate that one Biological process has to be modeled using many components). Among the 441 genes in Cluster 88, 288 (65.3%) were reported in the GO Biological processes named *small molecule metabolic process* (*GO* : 0044281) and *cell-substrate adhesion* (*GO* : 0031589). The number of principal components in this cluster was also 8.

To investigate whether the *specific* genes, *i.e.* involved in the GO biological process are well separated from *unspecific* genes (not involved in the GO biological process), we computed two standard PCAs in Clusters 3 and 88 separetely. As shown in Figure 12, the separation is well done.

7 VARCLUST package

The package [25] is an R package that implements VARCLUST algorithm. To install it, run `install.packages("varclust")` in R console. The main function is called mlcc.bic and it provides estimation of:

- Number of clusters K
- Clusters dimensions $\mathbf{k}$
- Variables segmentation Π

These estimators minimize modified BIC described in Section 2.

For the whole documentation use `?mlcc.bic`. Apart from running VARCLUST algorithm using random initializations, the package allows for a hot start specified by the user.

Information about all parameters can be found in the package documentation. Let us just point out few most important from practical point of view.

- If possible one should use multiple cores computation to speed up the algorithm. By default all but one cores are used. User can override this with numb.cores parameter
- To avoid algorithm getting stuck in the local minimum one should run it with random initialization multiple times (see parameter numb.runs). Default value is 20. We advice to use as many runs as possible (100 or even more).
- We recommend doing a hot-start initialization with some non-random segmentation. Such a segmentation could be result of some expert knowledge or different clustering method e.g. SSC. We explore this option in simulation studies.
- Parameter max.dim should reflect how large dimensions of clusters are expected to be. Default value is 4.

Acknowledgements M. Bogdan, P. Graczyk, F. Panloup and S. Wilczyński thank the AAP MIR 2018-2020 (University of Angers) for its support.

P. Graczyk and F. Panloup are grateful to SIRIC ILIAD program (supported by the French National Cancer Institute national (INCa), the Ministry of Health and the Institute for Health and Medical Research) and to PANORisk program of the Région Pays de la Loire for their support.

F. Panloup is also supported by the *Institut de Cancérologie de l'Ouest.*

8 Appendix. Proof of the PESEL Consistency Theorem

In the following we shall denote the sample covariance matrix

$$S_n = \frac{(X - \bar{X})^T (X - \bar{X})}{n},$$

the covariance matrix

$$\Sigma_n = E\left(S_n\right) = \frac{M_{n\times p}^T M_{n\times p}}{n} + \frac{n-1}{n}\sigma^2 Id$$

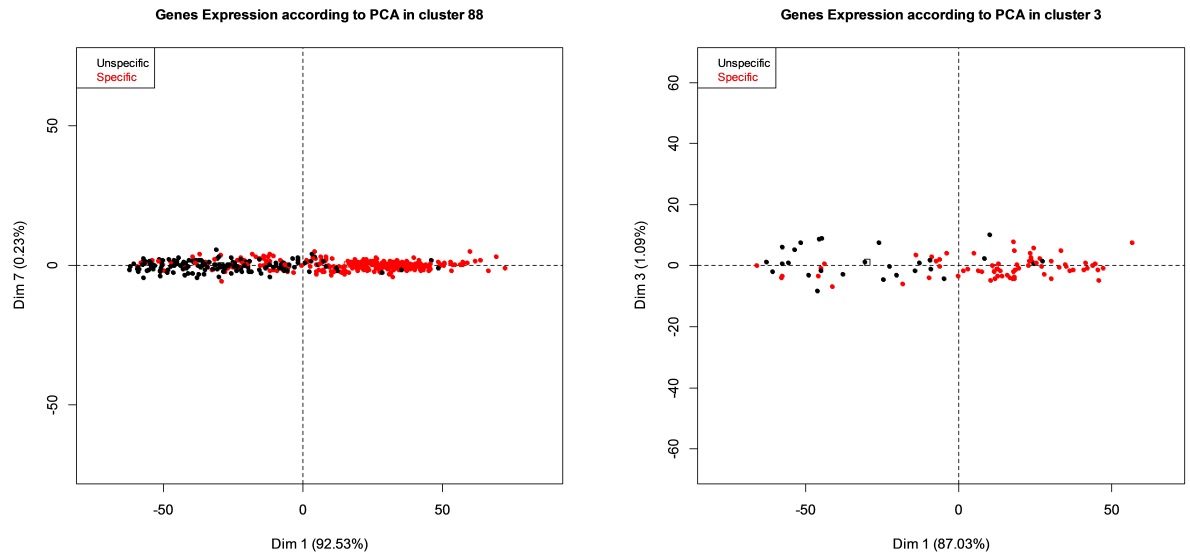


Fig. 12: Repartition of specific (red color) and unspecific genes (black color) according to a standard PCA.

and the heterogeneous PESEL function $F(n,k)$

$$F(n,k) = \underbrace{-\frac{n}{2}\left[\sum_{j=1}^{k}\ln(\lambda_j) + (p-k)\ln\left(\frac{1}{p-k}\sum_{j=k+1}^{p}\lambda_j\right) + p\ln(2\pi) + p\right]}_{G(k)} \underbrace{-\ln(n)\frac{pk - \frac{k(k+1)}{2} + k + p + 1}{2}}_{P(n,k)} \quad (8.1)$$

Proposition 1 *Let E have i.i.d. entries with a normal distribution $\mathcal{N}(0,\sigma^2)$. There exists a constant $C > 1$ such that almost surely,*

$$\exists_{n_0}\forall_{n\geq n_0} \quad \|\frac{1}{n}(E-\bar{E})^T(E-\bar{E}) - \sigma^2 Id\| \leq C\frac{\sqrt{2\ln\ln n}}{\sqrt{n}}$$

Proof. It is a simple corollary of LLN and LIL. The term jk of $\frac{1}{n}(E-\bar{E})^T(E-\bar{E})$ equals

$$\frac{1}{n}(E_{\bullet j} - \bar{E}_{\bullet j}\mathbf{1})^T(E_{\bullet k} - \bar{E}_{\bullet k}\mathbf{1}) = \frac{1}{n}\sum_{i=1}^{n} E_{ij}E_{ik} - \bar{E}_{\bullet j}\bar{E}_{\bullet k}.$$

An upper bound of convergence of $\frac{1}{n}\sum_{i=1}^{n} E_{ij}E_{ik}$ to $\sigma^2\delta_{jk}$ is $\frac{\sqrt{2\ln\ln n}}{\sqrt{n}}$. It is easy to show that an upper bound of convergence of $\bar{E}_{\bullet j}\bar{E}_{\bullet k}$ to 0 is $(\frac{\sqrt{2\ln\ln n}}{\sqrt{n}})^2 \leq \frac{\sqrt{2\ln\ln n}}{\sqrt{n}}$. □

Proposition 2 *Let E have i.i.d. entries with a normal law $\mathcal{N}(0,\sigma^2)$. There exists a constant $C > 1$ such that almost surely,*

$$\exists_{n_0}\ \forall_{n\geq n_0} \quad \|\frac{1}{n}(X-\bar{X})^T(X-\bar{X}) - (L+\sigma^2 Id)\| \leq C\frac{\sqrt{2\ln\ln n}}{\sqrt{n}} \quad (8.2)$$

Proof. It is easy to check that $X - \bar{X} = M + E - \bar{E}$. We write

$$\frac{1}{n}(X-\bar{X})^T(X-\bar{X}) = \frac{1}{n}M^TM + \frac{1}{n}(E-\bar{E})^T(E-\bar{E}) + \frac{1}{n}(M^TE + E^TM) - \frac{1}{n}(M^T\bar{E} + \bar{E}^TM)$$

To the first two terms we apply, respectively, the hypothesis (4.2) and the Proposition 1.

To prove the right pace of convergence of the third term $\frac{1}{n}(M^TE + E^TM)$ we consider every term $(M^TE)_{ij} = \langle M_{\bullet i}, E_{\bullet j}\rangle$ for which we use a generalized version of Law of Iterated Logarithm from [21]. Its assumptions are trivially met for random variables

$$M_{li}E_{lj} \sim \mathcal{N}(0, M_{li}^2\sigma^2)$$

as they are Gaussian and $\frac{B_{n+1}}{B_n} = \frac{n+1}{n} \to 1$, where B_n is defined as $B_n = \sum_l M_{li}^2\sigma^2$. Then, by [21], the following holds

$$\limsup_{n\to\infty} \frac{\sum_l M_{li}E_{lj}}{\sqrt{2B_n\log\log B_n}} = 1 \quad a.s.$$

The fourth term $\frac{1}{n}(M^T\bar{E} + \bar{E}^TM)$ is treated using Cauchy-Schwarz inequality:

$$|(\frac{1}{n}M^T\bar{E})_{ij}| = \frac{1}{n}|\langle M_{\bullet i}, \bar{E}_{\bullet j}\rangle| \leq \frac{1}{n}\|M_{\bullet i}\|\|\bar{E}_{\bullet j}\| = \frac{1}{n}\|M_{\bullet i}\|\sqrt{n\ \overline{E_{\bullet j}}^2} = \frac{1}{\sqrt{n}}\|M_{\bullet i}\||\overline{E_{\bullet j}}|.$$

By LIL, $|\overline{E_{\bullet j}}| \leq C\frac{\sqrt{2\ln\ln n}}{\sqrt{n}}$. The square of the first term $(\frac{1}{\sqrt{n}}\|M_{\bullet i}\|)^2$ converges to a finite limit by the assumption (4.2). □

Lemma 1 *There exists $C' > 0$ such that almost surely,*

$$\exists n_0\ \forall n \geq n_0 \quad \|\lambda(S) - \lambda(\Sigma)\|_\infty \leq C'\frac{\sqrt{2\ln\ln n}}{\sqrt{n}}, \tag{8.3}$$

where **S** *is sample covariance matrix for data drawn according to model* (4.1)*, Σ is its expected value and function $\lambda(\cdot)$ returns sequence of eigenvalues.*

Proof.
Observe that

$$\left\|\frac{(X-\overline{X})^T(X-\bar{X})}{n} - \Sigma\right\|_\infty \leq \|\frac{1}{n}(X-\bar{X})^T(X-\bar{X}) - (L+\sigma^2 Id)\| + \|(L+\sigma^2 Id) - \Sigma\|$$

We apply Proposition 2 to the first term and the assumption (4.2) to the second one.
Inequality (8.3) holds because (8.2) holds and, by Theorem A.46(A.7.3) from [3], when A, B are symmetric, it holds

$$\max_k |\lambda_k(A) - \lambda_k(B)| \leq \|A - B\|,$$

where function $\lambda_k(\cdot)$ denotes the k^{th} eigenvalue in the non-increasing order. □

Proof of Theorem 1.

Let $\epsilon_n = \max^i |\lambda_i(S_n) - \lambda_i(L)|$. From Lemma 1 we have $\lim_n \epsilon_n = 0$ almost surely, so for $k \leq k_0 - 1$, for almost all samplings, there exists n_0 such that if $n \geq n_0$,

$$\epsilon_n < \sigma^2 \text{ and } \epsilon_n < \frac{1}{4}\min_{k\leq k_0-1} c_k(\gamma),$$

where $c_k(\gamma) = \gamma_{k+1} - \frac{\sum_{k+2}^p \gamma_i}{p-k-1} > 0$.
We study the sequence of non-penalty terms $G(k)$ (see (8.1)). For simplicity, from now on, we use notation $\lambda_j = \lambda_j(S_n)$. We consider $G(k) - G(k+1)$ thus getting rid of the minus sign.

$$\begin{aligned} G(k) - G(k+1) = \\ &= \ln\lambda_{k+1} + (p-k-1)\ln\frac{\sum_{k+2}^p \lambda_j}{p-k-1} \\ &- (p-k)\ln\frac{\sum_{k+1}^p \lambda_j}{p-k} \\ &= \ln\lambda_{k+1} - \ln\frac{\sum_{k+2}^p \lambda_j}{p-k-1} \\ &+ (p-k)\left[\ln\frac{\sum_{k+2}^p \lambda_j}{p-k-1} - \ln\frac{\sum_{k+1}^p \lambda_j}{p-k}\right] \end{aligned}$$

Let us now denote $a = \lambda_{k+1}$ and $b = \frac{\sum_{k+2}^p \lambda_j}{p-k-1}$. Then the above becomes:

$$\ln a - \ln b + (p-k)\left[\ln b - \ln\frac{b(p-k-1)+a}{p-k}\right]$$

Case $k \leq k_0 - 1$.
We will use notation as above and exploit concavity of ln function by taking Taylor expansion at point x_0

$$f(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{f''(x^\star)}{2}(x-x_0)^2,$$

where $x^\star \in (x, x_0)$.
Let $x_0 = \theta x_1 + (1-\theta)x_2$ and $x = x_1$. Then

$$f(x_1) = f(x_0) + f'(x_0)(1-\theta)(x_1 - x_2) + \frac{f''(x_1^\star)}{2}(1-\theta)^2(x_1-x_2)^2.$$

Similarly, we take $x = x_2$, multiply both equations by θ and $1-\theta$ respectively and sum them up. We end up with the formula

$$\theta f(x_1) + (1-\theta)f(x_2) = f(x_0) + \theta(1-\theta)(x_2-x_1)^2\left[\frac{f''(x_1^\star)}{2}(1-\theta) + \frac{f''(x_2^\star)}{2}\theta\right].$$

In our case $f''(x) = -\frac{1}{x^2}$, which means that $\frac{f''(x_i^\star)}{2} < \frac{f''(x_2)}{2}$ because $x_1^\star \in (x_1, x_0) < x_2$ and $x_2^\star \in (x_0, x_2) < x_2$. This yields

$$\begin{aligned} &\theta f(x_1) + (1-\theta)f(x_2) - f(x_0) = \\ &\theta(1-\theta)(x_2-x_1)^2\left[\frac{f''(x_1^\star)}{2}(1-\theta) + \frac{f''(x_2^\star)}{2}\theta\right] \\ &< \theta(1-\theta)(x_2-x_1)^2\frac{f''(x_2)}{2} \end{aligned} \tag{8.4}$$

Now, going back to $G(k)$, we set

$$x_1 = b = \frac{\sum_{k+2}^p \lambda_j}{p-k-1}, \quad x_2 = a = \lambda_{k+1}, \quad \theta = 1 - \frac{1}{p-k}. \tag{8.5}$$

By multiplying both sides of (8.4) by $p-k$ we get

$$\begin{aligned}&(p-k-1)\ln\left(\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}\right)+\ln(\lambda_{k+1})\\&-(p-k)\ln\left((1-\frac{1}{p-k})\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}+\frac{1}{p-k}\lambda_{k+1}\right)\\&<-\left(1-\frac{1}{p-k}\right)\left(\lambda_{k+1}-\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}\right)^2\frac{1}{2\lambda_{k+1}^2}\end{aligned}$$

So, using $k+1\le k_0$ in the last inequality, we get

$$\begin{aligned}&G(k+1)-G(k)>\\&\qquad\left(1-\frac{1}{p-k}\right)\left(\lambda_{k+1}-\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}\right)^2\frac{1}{2\lambda_{k+1}^2}\\&\qquad=\frac{p-k-1}{p-k}\left(\lambda_{k+1}-\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}\right)^2\frac{1}{2\lambda_{k+1}^2}\\&\qquad>\frac{p-k_0-1}{p-k_0}\left(\lambda_{k+1}-\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}\right)^2\frac{1}{2\lambda_1^2}\end{aligned}$$

From Lemma 1, $\lambda_i\in[\gamma_i+\sigma^2-\epsilon_n,\gamma_i+\sigma^2+\epsilon_n]$, where ϵ_n goes to 0 and

$$\begin{aligned}&\left(\lambda_{k+1}-\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}\right)\ge\\&\ge\gamma_{k+1}+\sigma^2-\epsilon_n-\frac{\sum_{k+2}^{p}(\gamma_i+\sigma^2+\epsilon_n)}{p-k-1}\\&=c_k(\gamma)-2\epsilon_n\ge\min_{k\le k_0-1}c_k(\gamma)-2\epsilon_n>0\end{aligned}$$

for some constants $c_k(\gamma)$. Thus

$$\begin{aligned}&G(k+1)-G(k)\\&>\frac{p-k_0-1}{p-k_0}(\min_{k\le k_0-1}c_k(\gamma)-2\epsilon_n)^2\frac{1}{2(\gamma_1+\sigma^2+\epsilon_n)^2}\\&>\frac{C'}{2}\min_{k\le k_0-1}c_k(\gamma)>C>0\end{aligned}$$

where C, C' are constants independent of k and n. It follows that for n large enough

$$\begin{aligned}&\frac{n}{2}[G(k+1)-G(k)]\\&\qquad\ge\frac{n}{2}C\\&\qquad\gg\frac{\ln n}{2}(p-k)\\&\qquad=P(n,k+1)-P(n,k).\end{aligned}$$

This implies that the PESEL function $F(n,k)=\frac{n}{2}G(k)-P(n,k)$ is strictly increasing for $k\le k_0$.

Case $k\ge k_0$. By Lemma 1 we have that, for almost all samplings, there exists n_0 such that if $n\ge n_0$,

$$\epsilon_n\le C\frac{\sqrt{2\ln\ln n}}{\sqrt{n}}\text{ and }\epsilon_n<\frac{1}{2}\sigma^2.$$

We apply the formula (8.4) and as before, we use the notations (8.5). It yields

$$\begin{aligned}G(k+1)-G(k)&\le\left(1-\frac{1}{p-k}\right)\left(\lambda_{k+1}-\frac{\sum_{k+2}^{p}\lambda_j}{p-k-1}\right)^2\frac{1}{2b^2}\\&\le(\lambda_{k+1}-b)^2\frac{1}{2b^2}\\&\le(|\lambda_{k+1}-\sigma^2|+|\sigma^2-b|)^2\frac{1}{2b^2}\\&\le(|\lambda_{k+1}-\sigma^2|+\frac{\sum_{k+2}^{p}|\sigma^2-\lambda_j|}{p-k-1})^2\frac{1}{2b^2}\\&\le4\epsilon_n^2\frac{1}{2(\sigma^2-\epsilon_n)^2}\le C^2\frac{2\ln\ln n}{n}\frac{4}{2\sigma^4}\\&=C'\frac{\ln\ln n}{n}\end{aligned}$$

and consequently

$$\frac{n}{2}\left[G(k+1)-G(k)\right]\le C''\ln\ln n$$

Recall that the PESEL function equals $F(n,k)=\frac{n}{2}G(k)-P(n,k)$. The increase of $\frac{n}{2}G(k)$ is smaller than the rate $\ln\ln n$, while the increase of penalty $P(n,k+1)-P(n,k)=\frac{\ln n}{2}(p-k)$ is of rate $\ln n$. Consequently, there exists n_1 such that for $n>n_1$, the PESEL function is strictly decreasing for $k\ge k_0$ with probability 1.

We saw in the first part of the proof that the PESEL function $F(n,k)$ is strictly increasing for $k\le k_0$, for n big enough. It implies that with probability 1, there exists n_2 such that for $n>n_2$ we have $\hat{k}_0(n)=k_0$. □

References

1. Airly: Airly sp. z.o.o., Air quality data from extensive network of sensors (version 2) (2017). Retrieved from `https://www.kaggle.com/datascienceairly/air-quality-data-from-extensive-network-of-sensors`
2. Bai, Z., Choi, K., Fujikoshi, Y.: Consistency of AIC and BIC in estimating the number of significant components in high-dimensional principal component analysis. Ann. Statist. **46**(3), 1050–1076 (2018)
3. Bai, Z., Silverstein, J.W.: Spectral Analysis of Large Dimensional Random Matrices. New York, NY: Springer (2010)
4. Bhat, H.S., Kumar, N.: On the derivation of the bayesian information criterion. School of Natural Sciences, University of California (2010)
5. Bogdan, M., Ghosh, J.K., R.W., D.: Modifying the schwarz bayesian information criterion to locate multipleinteracting quantitative trait loci. Genetics **167**, 989–999 (2004)

6. Cancer Genome Atlas Network: Comprehensive molecular portraits of human breast tumours. Nature **490**(7418), 61–70 (2012)
7. Chavent, M., Kuentz, V., Liquet, B., Saracco, L.: ClustOfVar: An R Package for the Clustering of Variables. ArXiv e-prints (2011)
8. Chavent, M., Kuentz, V., Liquet, B., Saracco, L.: Clustofvar: An r package for the clustering of variables. J. Stat. Softw **50** (2011). DOI 10.18637/jss.v050.i13
9. Durinck, S., Moreau, Y., Kasprzyk, A., Davis, S., De Moor, B., Brazma, A., Huber, W.: BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. Bioinformatics **21**(16), 3439–3440 (2005)
10. Durinck, S., Spellman, P.T., Birney, E., Huber, W.: Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Nat Protoc **4**(8), 1184–1191 (2009)
11. Elhamifar, E., Vidal, R.: Sparse subspace clustering. In: In CVPR (2009)
12. Gene Ontology Consortium: The Gene Ontology (GO) database and informatics resource. Nucleic Acids Res. **32**(Database issue), D258–261 (2004)
13. Grote, S.: GOfuncR: Gene ontology enrichment using FUNC (2018). R package version 1.2.0
14. Hotelling, H.: Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology **24**, 417–441, 498–520 (1933)
15. Hotelling, H.: Relations between two sets of variates. Biometrika **28**, 321–377 (1936)
16. Hubert, L., Arabie, P.: Comparing partitions. Journal of classification **2**(1), 193–218 (1985)
17. Jolliffe, I.: Principal Component Analysis. Springer Series in Statistics. Springer (2002)
18. Law, C.W., Chen, Y., Shi, W., Smyth, G.K.: voom: Precision weights unlock linear model analysis tools for RNA-seq read counts. Genome Biol. **15**(2), R29 (2014)
19. Minka, T.P.: Automatic choice of dimensionality for pca. NIPS **13**, 514 (2000)
20. Pearson, K.: On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine **2**(11), 559–572 (1901)
21. Petrov, P., Petrov, V.: Limit Theorems of Probability Theory: Sequences of Independent Random Variables. Oxford science publications. Clarendon Press (1995). URL `https://books.google.pl/books?id=4LkdSaI4xXMC`
22. Ritchie, M.E., Phipson, B., Wu, D., Hu, Y., Law, C.W., Shi, W., Smyth, G.K.: limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res. **43**(7), e47 (2015)
23. Robinson, M.D., Oshlack, A.: A scaling normalization method for differential expression analysis of RNA-seq data. Genome Biol. **11**(3), R25 (2010)
24. Sobczyk, P., Bogdan, M., Josse, J.: Bayesian dimensionality reduction with pca using penalized semi-integrated likelihood. Journal of Computational and Graphical Statistics **26**(4), 826–839 (2017). DOI 10.1080/10618600.2017.1340302. URL `https://doi.org/10.1080/10618600.2017.1340302`
25. Sobczyk, P., Wilczyński, S., Josse, J., Bogdan, M.: varclust: Variables Clustering (2017). URL `https://github.com/psobczyk/varclust`. R package version 0.9.4
26. Sołtys, M.: Metody analizy skupień. Master's thesis, Wrocław University of Technology (2010)
27. Vidal, R.: Subspace clustering. Signal Processing Magazine **28**, 52–68 (2011)
28. Vidal, R., Favaro, P.: Low rank subspace clustering (lrsc). Pattern Recognition Letters **43**(0), 47 – 61 (2014). {ICPR2012} Awarded Papers
29. Vigneau, E., Qannari, E.M.: Clustering of variables around latent components. Communications in Statistics-Simulation and Computation **32**(4), 1131–1150 (2003)