On the Generation of Random Multivariate Data

José Camacho

Departamento de Teoría de la Señal, Telemática y Comunicaciones, Universidad de Granada, 18071, Granada, Spain

Abstract

The simulation of multivariate data is often necessary for assessing the performance of multivariate analysis techniques. The random generation of multivariate data when the covariance matrix is completely or partly specified is solved by different methods, from the Cholesky decomposition to some recent alternatives. However, many times the covariance matrix has to be generated also at random, so that the data simulation spans different situations from highly correlated to uncorrelated data. This is the case when assessing a new multivariate analysis technique in Montercarlo experiments. In this paper, we introduce a new algorithm for the generation of random data from covariance matrices of random structure, where the user only decides the data dimension and the level of correlation. We will illustrate the application of this algorithm in several relevant problems in multivariate analysis, namely the selection of the number of Principal Components in Principal Component Analysis, the evaluation of the performance of sparse Partial Least Squares and the calibration of Multivariate Statistical Process Control systems. The algorithm is available as part of the MEDA Toolbox v1.1 [1]

[1] Available at https://github.com/josecamachop/MEDA-Toolbox/releases/tag/v1.1

Key words: Multivariate data, simulation, ADICOV, MEDA Toolbox, Montecarlo

1 Introduction

When a new multivariate technique for data analysis is developed, there is the need to test its performance on a wide set of data structures in Montecarlo experiments. Sometimes the structure of the data is established a priori to a certain extent, like when the data is simulated according to specific analytical instruments [1] or well known industrial processes [2]. In other cases, the technique to be assessed may be applied to data of very different structure. For instance, take the case of the algorithms for the dimensionality selection in PCA or PLS. Such an algorithm needs to be tested on a general perspective, without making assumptions on the data generation procedure, or more specifically on the correlation structure.

While the generation of random data when the covariance matrix is completely or partly specified is solved by different methods [3,4,6], there is no established approach to generate random data from randomly generated covariance matrices to perform Montecarlo experiments. In this paper, we propose such an approach. We will show that the proposed algorithm, named simuleMV, can be very useful in a broad set of problems that are relevant for the research community. In particular, we will illustrate its use in the selection of the number of PCs in PCA using cross-validation, the comparison of performance between sparse and normal versions of PLS and the calibration of Multivariate Statistical Process Control systems for anomaly detection.

The rest of the paper is organized as follows. Section 2 introduces the methods

Email address: josecamacho@ugr.es, tel: +34 958 248898, fax: +34 958 240831 (José Camacho).

for data simulation from a known covariance matrix. Section 3 discusses the problem of data simulation for random covariance. It ends with the definition of the proposal of this paper. Section 4 evaluates this solution and Section 5 presents some application case studies. Section 6 presents the conclusions of the work.

2 Simulation of random data when the covariance is known

Let us define Σ as the covariance matrix for data generation, and $X \sim N(0, I)$ as a random matrix. The goal pursued by the approaches introduced in this section is the generation of a data matrix Y from X but with covariance matrix Σ.

The traditional approach for data generation following this definition is based on the Cholesky Decomposition (CD) of Σ in the product of a lower triangular matrix and its conjugate transpose:

$$\Sigma \underset{CD}{=} CC^T \tag{1}$$

where C is lower triangular. Then, Y can be obtained from C and X:

$$Y = XC^T \tag{2}$$

As a result of this equation, as the sample covariance of X is closer to its population covariance, i.e. the identity matrix I, the sample covariance of Y will be closer to Σ.

The CD approach, however, has a number of drawbacks. First, it can only be applied to non-singular (positive definite) covariance matrices, which is a

severe limitation in chemometric studies. Second, the procedure is limited to X with population covariance I. However, it might be interesting to modify the distribution of the elements in X in order to introduce certain features in Y, like clusters and outliers. Doing so while maintaining a population covariance equal to I is a problem as challenging as the one treated in this paper. Last, and probably the least important, the CD approach provides data matrices which covariance matrix only approximates Σ. Stated otherwise, with the CD we can impose a certain population covariance to Y, but not a sample covariance.

Two recent approaches propose an improved alternative to the CD. On the one hand, Arteaga and Ferrer [3] proposed two methods for data generation with an exact sample covariance. The first method takes as input the complete covariance matrix Σ, while the second method is run from a subset of eigenvalues of Σ. The first method is referred to as *randnm*, and it is the one of interest in this paper. It is based on three steps. First, it performs a Singular Value Decomposition (SVD) of Σ:

$$\Sigma \underset{\boldsymbol{SVD}}{=} VDV^T \tag{3}$$

where, as a result of Σ being a covariance matrix, V contains the eigenvectors of Σ and D is a diagonal matrix containing the eigenvalues.

Then, it generates random matrix X and performs the QR Decomposition on it:

$$X \underset{\boldsymbol{QR}}{=} QR^T \tag{4}$$

where Q is an orthogonal matrix and R an upper triangular matrix.

Finally, matrix Y is obtained as:

$$Y = \sqrt{N-1}QSV^T \tag{5}$$

where N is the number of rows of X and S is the root square element-wise of diagonal matrix D, so that $D = SS$.

On the other hand, Camacho et *al.* [4] presented the ADICOV algorithm, similar to *randnm* in philosophy but of broader application and with the additional property that Y is the least squares approximation of X with sample covariance matrix Σ. A first difference of ADICOV with respect to *randnm* is that X is not internally generated, but an input to the algorithm. This is appropriate to extend the approximation to data sets which are not normally distributed, for instance for X with specific features like outliers and clusters [5]. Another difference is that ADICOV generalizes the approximation to specific subspaces, like those identified with PCA or PLS. It should be noted that extending *randnm* to mimic this features is straightforward.

For the sake of simplicity, we will discuss a simplified version of ADICOV where Y follows Σ in the original space. The first step of ADICOV is an EigenDecomposition of Σ, equivalent to the first step in *randnm*:

$$\Sigma \underset{ED}{=} VDV^T \tag{6}$$

Again, V contains the eigenvectors of Σ and D is a diagonal matrix containing the eigenvalues.

Then, the Polar Decomposition (PD) is performed over matrix XDV^T. This can be done by performing a SVD and retaining the row-wise and column-wise

eigenvectors:

$$XDV^T \underset{\boldsymbol{SVD}}{=} U_p D_p V_p^T \tag{7}$$

$$P_p = U_p V_p^T \tag{8}$$

being P_p the first factor of the PD. Then, matrix Y follows:

$$Y = \sqrt{N-1} P_p S V^T \tag{9}$$

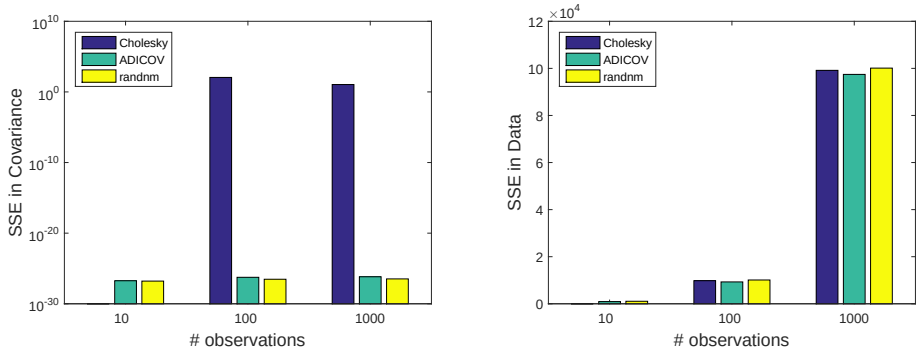
Recall that this is a simplified version of ADICOV. The original version performs several projections between the original space and the subspace of choice.

To understand the differences among the CD, *randnm* and ADICOV, we can perform a number of simple simulations. We generate 100 repetitions of data sets of dimension $n \times 100$, for $n = \{10, 100, 1000\}$, and compare the methods in terms of three figures of merit:

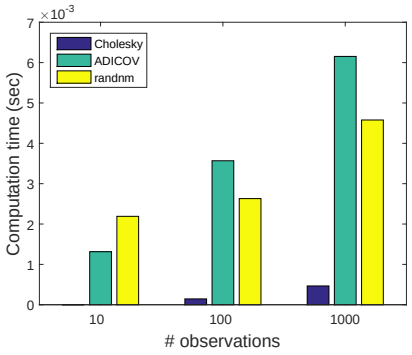
- SSE in Covariance: The sum-of-squares of the difference/error (SSE) between Σ and the covariance of Y
- SSE in Data: The SSE between X and Y
- The computation time

Results are shown in Fig. 1.

Fig. 1(a) presents the SSE in Covariance in logarithm scale. It shows that both ADICOV and *randnm* yield exact solutions, that is, Y has a sample covariance matrix equal to Σ, while CD takes Σ as its population matrix. However, the main limitation of the CD is the fact that when data are not full rank (e.g. for



(a) Covariance (b) Data



(c) Computation time

Fig. 1. Comparison of the Cholesky Decomposition, *randnm* and ADICOV in the simulation of data: Sum-of-squares on error (SSE) in Covariance (a), SSE in Data (b) and computation time in seconds (c). Values shown are average values from 100 repetitions.

10 observations in Fig. 1(a)), CD cannot be even used, while ADICOV and *randnm* still provide exact solutions.

Fig. 1(b) shows the SSE between X and the generated Y for the different methods. ADICOV performs the closest approximation in the quadratic sense, since it was designed specifically for that. Note that this is not necessary an interesting feature, since X is mainly a source of randomness to generate Y and we are not particularly interested on a least squares approximation on X. In any case, differences are minor.

Finally, Fig. 1(c) compares the methods in computation time. The CD takes the fastest computation, and *randnm* is faster than ADICOV for large dimensional data. Still, all methods are very fast.

According to the results, we can conclude that both ADICOV and *randnm* outperform the Cholesky Decomposition for the generation of random data at the expense of an affordable increase of computation time.

3 Simulation of random data when the covariance is unknown

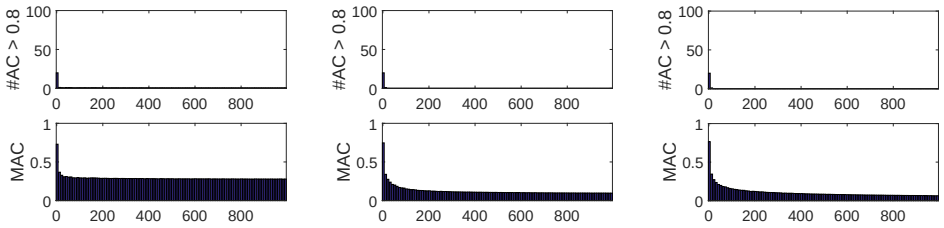
As motivated in the introduction, when simulating random data, there are many situations in which we also need to generate the covariance matrix of the data at random. In such cases, the procedures discussed previously cannot be directly used. Arteaga and Ferrer also studied this problem in [6], providing a solution to compute the covariance matrix by fixing the variances and a subset of the eigenvalues and covariances. Still, this approach does not provide means to draw variances, covariances and eigenvalues to perform Montecarlo experiments, something that deserves future investigation. Here we explore a different approach.

One direct approach to generate the covariance at random is to draw all the coefficients in the covariance matrix as iid samples, e.g. from a normal or a constant distribution. An obvious drawback of this approach is that the resulting matrix may not be semidefinite positive. While this would render the CD useless, the use of the SVD and the ED in *randnm* and ADICOV, respectively, get around this problem, substituting the non-semidefinite positive matrix generated at random by a semidefinite positive counterpart. In the case

of the SVD, only the third factor in the decomposition in eq. (3) is computed, and it is assumed to be equal to the first factor. Thus, if matrix $\tilde{\Sigma} \underset{SVD}{=} UDV^T$ is drawn as iid samples, it is replaced by $\Sigma = VDV^T$. In the case of the ED, matrix $\tilde{\Sigma} \underset{ED}{=} VDV^{-1}$ drawn as iid samples is replaced by $\Sigma = VDV^T$. As a result, this naive approach for random covariance generation can be used with ADICOV or *randnm*, since the SVD in eq. (3) and the ED in eq. (6), respectively, transform the non-semidefinite positive matrix in a suitable one.

Figure 2 shows the result of applying the naive simulation approach using ADICOV and a constant distribution in $[-1, 1]$ to draw the random covariance coefficients. Similar results where obtained for *randnm*. The results include different numbers of variables and observations in the simulated matrix Y. It should be noted that although the data is obtained from a randomly simulated covariance matrix, we assess the results in terms of the distribution of the coefficients in the true–semidefinite positive–correlation matrix, after the ED in ADICOV and a convenient normalization between -1 and 1. In particular, two figures of merit are used: the number of high correlation coefficients, whose absolute value is above 0.8 (number of Absolute Coefficients above 0.8 or $\#AC > 0.8$) and the Mean Absolute Coefficient (MAC). Diagonal correlation coefficients (equal to 1) are not taken into account for both values. Those figures are useful to evaluate whether we are capable or not to simulate data sets with different level of correlation among the variables. In Figure 2, three subfigures are shown for 10, 100 and 1000 observations in Y, respectively. The abscissa shows the number of variables, from 2 to 1000. The figures at the top show the $\#AC > 0.8$ while the figures at the bottom show the MAC.

The figure let us draw several conclusions about the suitability of the naive simulation approach. On the one hand, we cannot obtain high correlation



(a) 10 obs (b) 100 obs (c) 1000 obs

Fig. 2. Evolution of the correlation induced by the naive approach in terms of the number of variables. At the top, number of coefficients with absolute value above 0.8 ($\#AC > 0.8$). At the bottom, mean absolute coefficient (MAC).

coefficients using this approach exception made on a reduced set of dimensions. This is because high correlation coefficients impose strong structural constrains in the covariance matrix. Consider the simple following example: if variable A is highly correlated to variables B and C, B and C need also to be correlated to a certain extent. This short of structural constraints are very unlikely to be met in a random generation of the correlation coefficients like in the naive approach. The result is that most high correlation coefficients are discarded in the ED (or SVD) to obtain a valid, semidefinite positive, covariance matrix, and the naive simulation approach tends to yield data sets of low correlation. On the other hand, the level of correlation depends on the dimension of Y, i.e. the number of observations and variables. This is of course an undesirable property, since we would like to be able to generate different levels of correlation for different dimensions in the resulting matrix, that is, decoupling dimension and correlation level. For that, we need a different simulation approach.

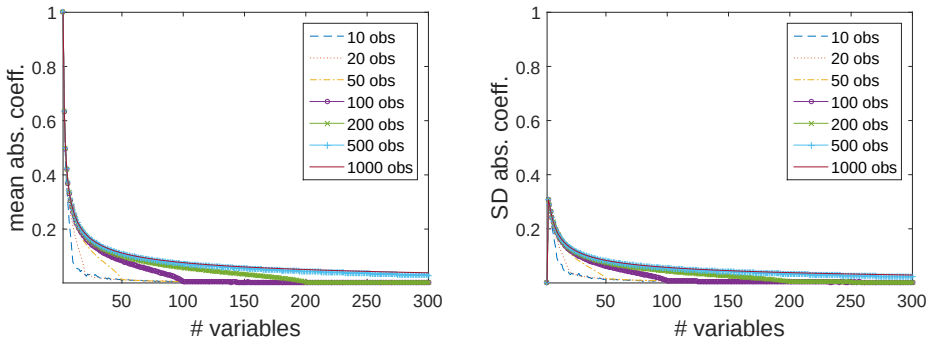
Still, we can make an interesting observation from the results of Figure 2: the level of correlation is mainly influenced by the numbers of observations. Thus, all figures present higher levels of correlations for lower numbers of observa-

tions (leftmost values in the bar plots). This makes sense considering that for a fixed number of variables in Y, we can reduce the rank in this matrix by reducing the number of observations. Imposing a low rank in Y is equivalent to imposing a high the level of correlation of the variables. Therefore, correlation can be made linked to the number of observations. We will use this connection between level of correlation and number of observations in a new simulation procedure.

3.1 Linking correlation level with the number of observations

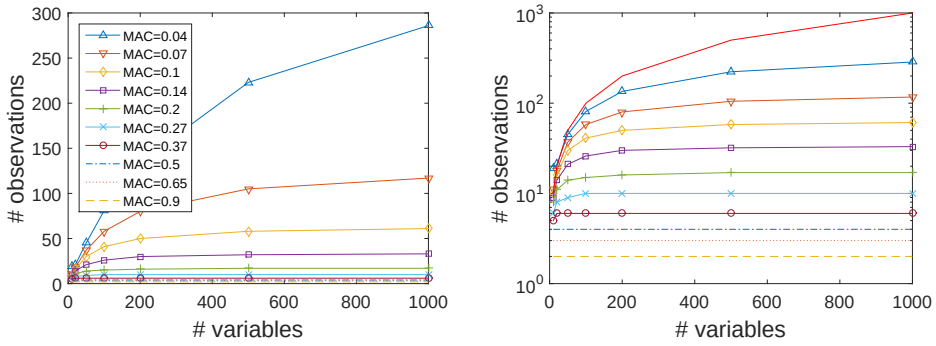
From here onwards we will restrict ourselves to the use of ADICOV for data simulation, though *randnm* could have been used instead. In Figure 3, the average and standard deviation of the absolute correlation coefficients of a correlation matrix drawn from the naive simulation approach are shown for different numbers of observations and variables in Y. The standard deviation behaves very much like the average. Taking advantage of this fact, for every single number of observations we can find a number of variables for which the level of correlation in Y is similar, both in average and dispersion, when generated with the naive approach. Therefore, understanding (modeling) the non-linear relationship among number of observations, number of variables and level of correlation illustrated in Figure 3, we can design an algorithm where the same level of correlation can be applied for different dimensions in Y.

Of course, we want to completely decouple the dimension in Y from the correlation level, so that in the resulting algorithm the user can specify the dimension of Y, N observations and M variables, and the correlation level L. For that, a



(a) (b)

Fig. 3. Mean (a) and standard deviation (b) of the absolute value of the correlation coefficients using the naive simulation approach.



(a) (b)

Fig. 4. Correlation levels (measured as the mean absolute coefficient (MAC) in Σ) in terms of the number of observations and variables: (a) linear scale and (b) logarithmic scale.

straightforward procedure is to perform two consecutive ADICOV runs. The first ADICOV run is performed to generate a data set Z, which number of observations depends on the correlation level desired, taking advantage of the linkage aforementioned between correlation level and number of observations. The covariance matrix of Z, Σ, is then used in the second run to generate Y, now with the number of observations specified by the user. The first step is the challenge we need to address, since the second one is already solved, as discussed in Section 2.

Considering that the covariance matrix Σ is of dimension $M \times M$, and that both M and the level of correlation L are user specified, the first step in the simulation algorithm needs to identify the number of observations in Z to yield Σ with the desired correlation level. Figure 4 shows in both linear and logarithmic scale how this function looks like for different correlation levels, specified as different values of the MAC in the correlation matrix associated to Σ. Note correlation levels can be defined according to different features of the correlation matrix associated to a given covariance matrix. The MAC is just a possible choice we have used in this paper.

Figure 4 shows that the number of observations and the correlation level are inversely related, which means that topmost curves are those for a lower correlation in Σ. Furthermore, we can see that there is a non-linear relationship between the three factors N, M and L. The number of observations in Z for a given correlation level can be approximated by a logarithmic function in the variables. The functions for different levels range from a logarithmic shape to a flat shape, which can be obtained by raising the logarithm to a power inversely related to the correlation level, in particular an exponential decay seems to be a good choice for that. Finally, when the number of variables is very low, e.g. 10, the number of observations is linearly related to the correlation level. Considering all these features, we propose the following model to determine the number of observations in Z, noted as λ, from a user specified correlation level L:

$$\lambda = x_1 \cdot (x_2 - L) \cdot \left(\frac{\log(M)}{\log(x_3)} \right)^{x_4 \, \mathrm{e}^{(-x_5 L)}} \tag{10}$$

where L is defined as an integer in $[1, 10]$, and x_1 to x_5 are model parameters we can fit from the data shown in Figure 4. The correlation level L was defined

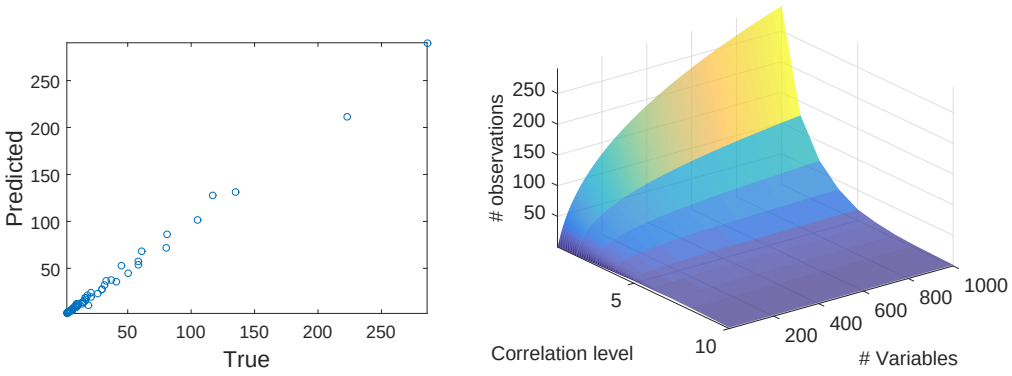


Fig. 5. True Vs Predicted values in the non-linear approximation (a) and model surface (b).

as an integer to simplify the model identification and posterior use, where the ten possible levels correspond to the MAC values in Fig. 4(a). Thus, there is a non-linear relationship between L and the corresponding MAC values that is adequate from a practical point of view, yielding ten levels of correlation that span the interval between no correlation and complete correlation.

We prefer the type of model in eq. (10) to a purely data-based model (e.g. a neural network) with the hope that the log-exp structure observed in the data will work better in extrapolation. After some fitting trials, we decided to use three different models, one for $L = \{1, 2, 3\}$, one for $L = \{4, 5, 6, 7\}$ and one for $L = \{8, 9, 10\}$. This results in a model with 15 parameters we need to fit. In Figure 5(a) the true λ values from Figure 3 are represented against the estimated ones with the model in eq. (10). Using this model, our simulation algorithm can estimate the correct number of observations in Z for given values of L and M. Figure 5(b) shows the resulting model surface.

The *SimuleMV* algorithm is described below. First, if L is set to 0, the identity matrix is used as covariance matrix of Y. Otherwise, the number of observations in Z, λ, is estimated from the fitted model (Figure 5(b)). For brevity, we represented a single model, instead of the ensemble of three local models. Note model parameters are hardcoded to the algorithm, that is, there are actual values for parameters x_i. Then, using a common random generator, we obtain matrix $X_{\lambda,M}$ and run ADICOV from it and using an identity matrix as covariance. This yields matrix Z, with the desired level of correlation. The covariance matrix Σ is then obtained from Z to run a second ADICOV, obtaining matrix Y.

Inputs:

N: number of observations.

M: number of variables.

L: correlation level.

Algorithm:

if $L > 0$

$\lambda = x_1 \cdot (x_2 - L) \cdot \left(\frac{\log(M)}{\log(x_3)}\right)^{x_4 \, \mathrm{e}^{(-x_5 L)}}$

$X_{\lambda,M} \sim N(0_M, I_M)$

$Z = ADICOV(I_M, X_{\lambda,M})$

$\Sigma = \frac{1}{\lambda - 1} \cdot Z^T \cdot Z$

$X_{N,M} \sim N(0_M, I_M)$

$Y = ADICOV(\Sigma, X_{N,M})$

else

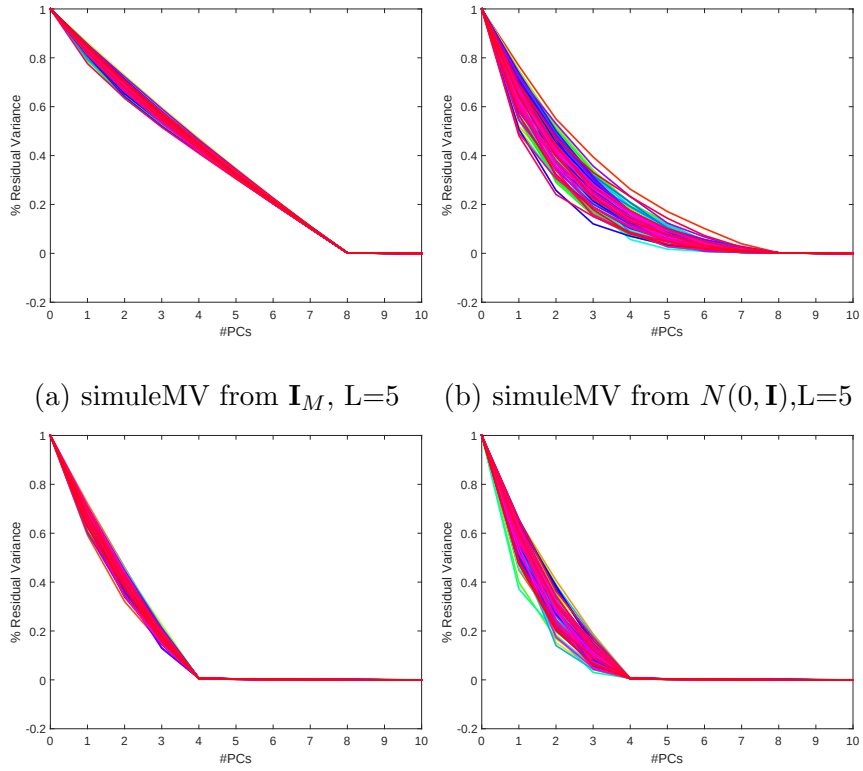
$Y = ADICOV(I_M, X_{N,M})$

One comment is in due regarding the simuleMV implementation. The relationship in equation (10) to obtain a suitable value for λ was identified using the naive approach. However, the simuleMV algorithm does not use this approach, since the first ADICOV call is based on the identity matrix, instead of a random covariance matrix. This has a particular effect on the simulation result, as illustrated in Figure 6, where the explained variability in terms of the number of PCs is shown for one hundred repetitions of the simulation of a 100×10 data set. In Figures 6(a) and 6(c), we show the result of using the

identity matrix in the first ADICOV call of simulaMV. When this is done, the curve of explained variance is highly repetitive and close to a straight line until we reach a point where no more significant PCs are found. This specific variance distribution is the price to pay of using a single parameter, L, to control the correlation in the data, though the repetitive nature is quite convenient. If instead of the identity matrix a random matrix is used, Figures 6(b) and 6(d), the explained variance curves show an exponential decay and they are more variable, reducing the control we have on the result. To see this, compare the results for L=5 and L=7. Using the identity matrix, the curves for different L do not overlap. Thus, we tightly control the result with user parameter L. However, when using a random matrix several of the curves for L=5 and L=7 overlap for the first PCs, which reflects that we can obtain very similar data sets for different values of L–an undesirable result. It should be noted that although simuleMV remains very repetitive in the curve of explained variance, the generated data sets are random, and therefore the generated variability is suitable for Montecarlo experiments. Furthermore, it is always possible to add random noise drawn from $N(0, r \cdot I)$, with r the noise level, in case one desires to avoid rank deficient data matrices.

Figure 6 also reflects that the behaviour of simuleMV can be mimicked using an approach like that in [6] by fixing the distribution of the eigenvalues. The expected benefit would be that we may be able to simulate more complex variance distributions in the eigenvalues and study their effect on the general correlation. The price to pay is that user control would probably require more than parameter L. Thus, we could add more degrees of freedom, but also more complexity of use.

In Fig. 7, an illustrative example of use of *SimuleMV* is shown. The *SimuleMV*



(a) simuleMV from $\mathbf{I}_M$, L=5 (b) simuleMV from $N(0, \mathbf{I})$,L=5

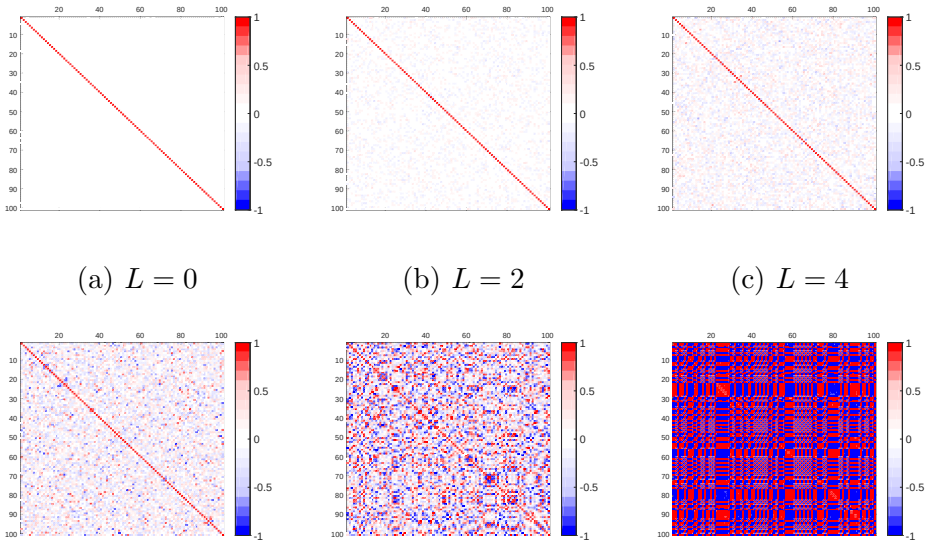
(c) simuleMV from $\mathbf{I}_M$, L=7 (d) simuleMV from $N(0, \mathbf{I})$,L=7

Fig. 6. Percentage of captured variance of the PCs for a 100×10 data.

algorithm is available with the MEDA Toolbox v1.1 at https://github.com/josecamachop/MEDA-Toolbox/releases/tag/v1.1

4 Performance results

The performance of *SimuleMV* is evaluated in Figure 8. It can be seen that *SimuleMV* provides an adequate simulation procedure, that decouples the desired correlation level from the dimension of the simulated matrix. Still, there are some structural limitations, highlighted in light color. For instance, a simulated matrix with dimension 10×100 cannot be full rank–or more accurately, its covariance cannot be full rank. Therefore, we cannot simulate a 10×100 matrix with correlation level 0. Note this is an structural constrain



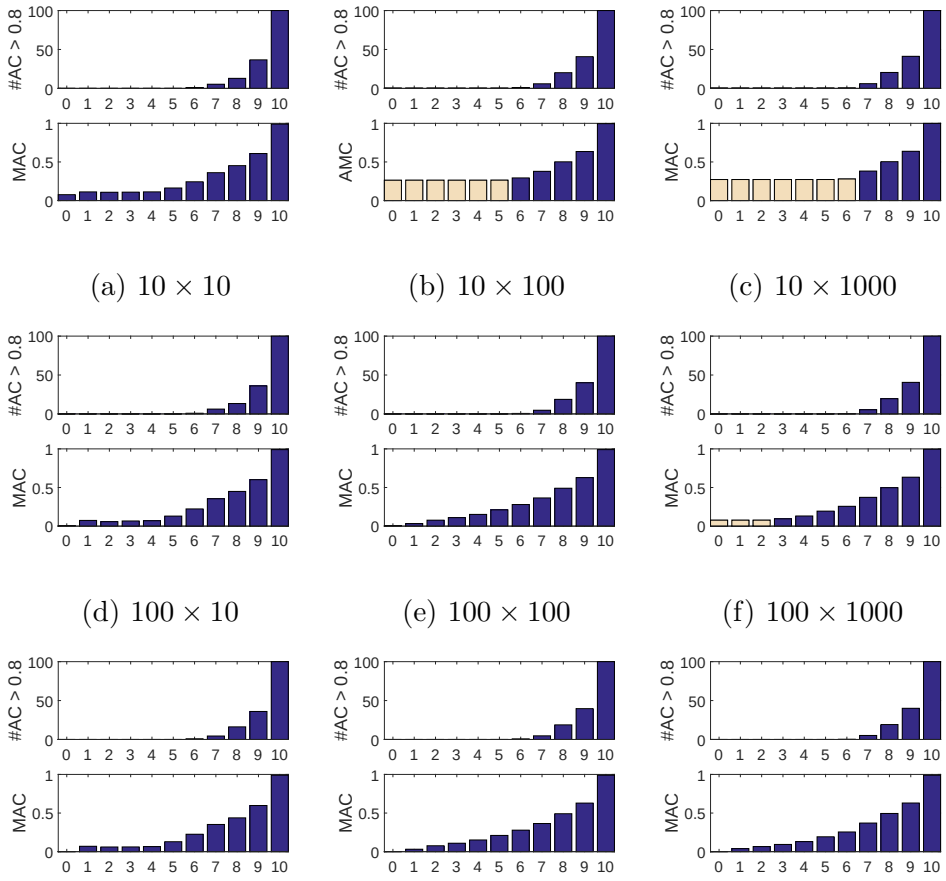
(a) $L = 0$ (b) $L = 2$ (c) $L = 4$

(d) $L = 6$ (e) $L = 8$ (f) $L = 10$

Fig. 7. Correlation maps for different values of L for a 100×100 data set.

that has to be met, and not a limitation of *SimuleMV*. As a matter of fact, the number of observations, in the example 10, provides a minimum value for the attainable level of correlation. Thus, a 10×100 matrix cannot be simulated for correlation levels below 6, since for levels below 6 the number of observations in Z according to our model would be larger than 10. We added a warning message in the algorithm for incompatible input parameters. We can see in Figure 8 that these incompatible parameters are only found for fat data matrices, which cannot be full rank.

Figure 9 shows the results of *SimuleMV* in terms of the number of variables, for comparison with Figure 2 of the naive simulation approach. Exception made on Figure 9(a), which corresponds mainly to unfeasible parameters, the rest of the figures show that *SimuleMV* yields quite stable correlation levels independently of the number of variables or observations. Regarding the independence with the number of variables, we can see that all figures are almost flat. Regarding the independence with the number of observations, we



(a) 10×10 (b) 10×100 (c) 10×1000

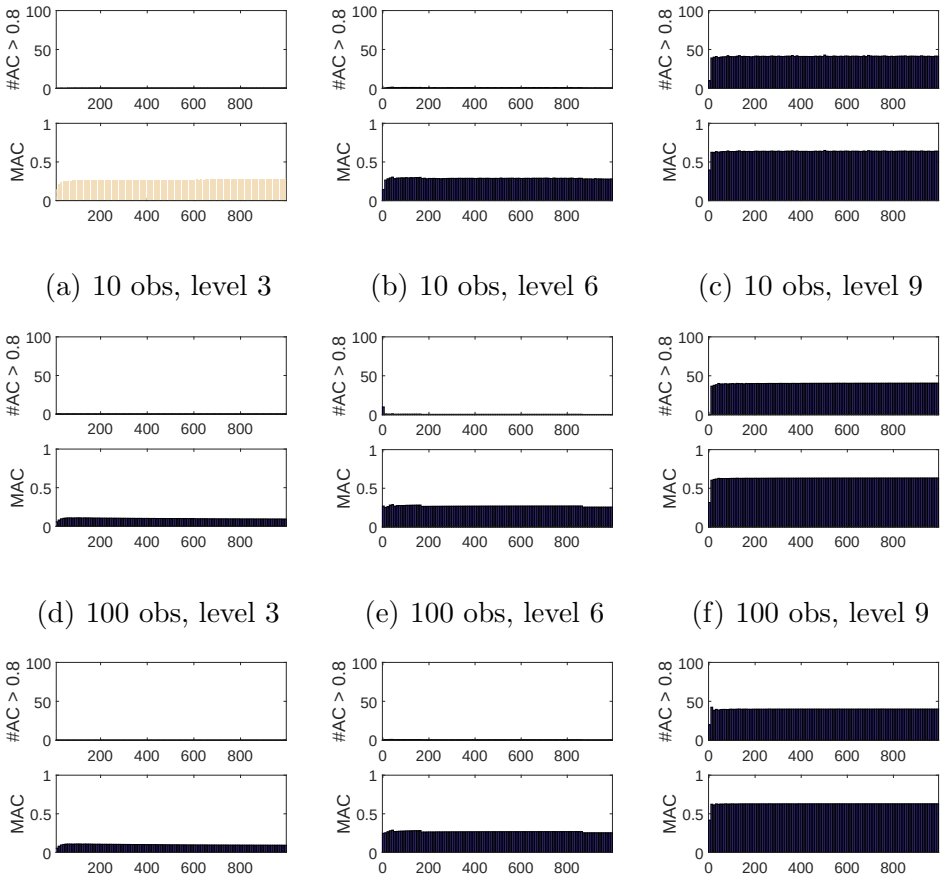
(d) 100×10 (e) 100×100 (f) 100×1000

(g) 1000×10 (h) 1000×100 (i) 1000×1000

Fig. 8. Evolution of the correlation induced by *SimuleMV* in terms of the correlation level. At the top, number of coefficients with absolute value above 0.8 ($\#AC > 0.8$). At the bottom, mean absolute coefficient (MAC).

can see that Figures 9(d) and 9(g), Figures 9(b), 9(e) and 9(h) and Figures 9(c), 9(f) and 9(i) are almost identical. Still, there are some artifacts, but we believe that with the model fitted, *SimuleMV* results are adequate from a practical perspective.

Finally, Figure 10 shows the percentage of variance captured by the first PC in terms of the correlation level. It can be seen that by modifying the correlation level, we explore the complete range of possibilities: from the first PC capturing a 100% of the variance ($L = 10$) to the first PC capturing approximately



(a) 10 obs, level 3 (b) 10 obs, level 6 (c) 10 obs, level 9

(d) 100 obs, level 3 (e) 100 obs, level 6 (f) 100 obs, level 9

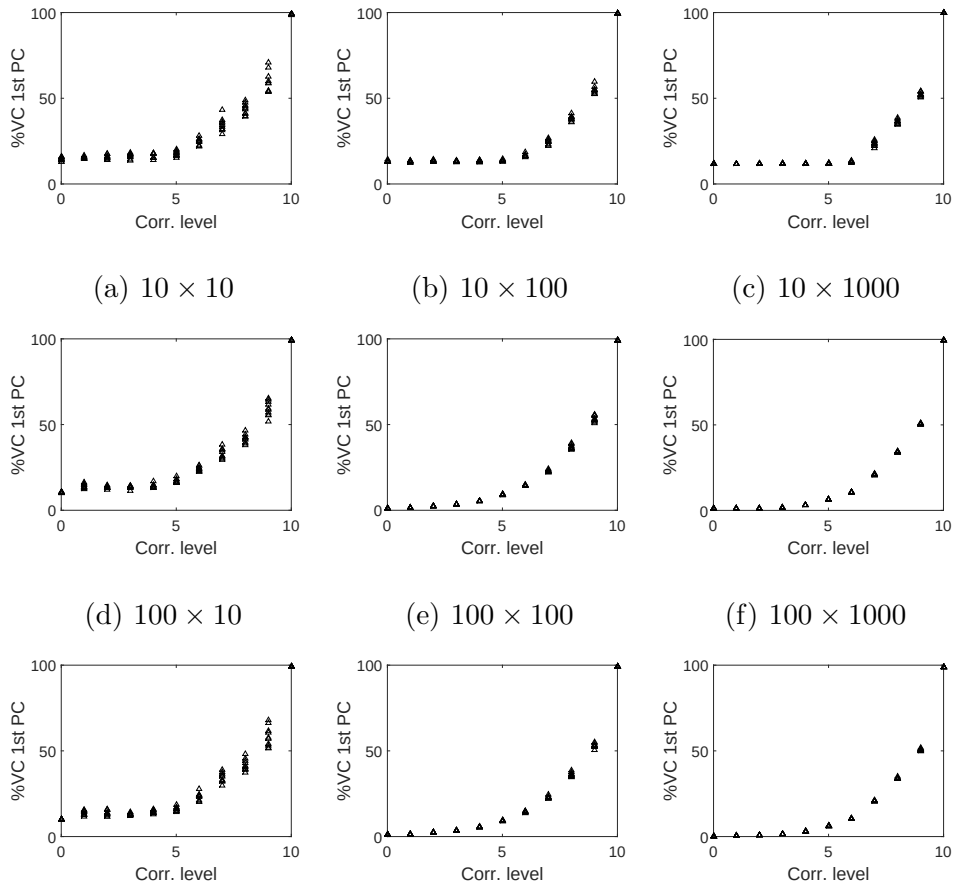
(g) 1000 obs, level 3 (h) 1000 obs, level 6 (i) 1000 obs, level 9

Fig. 9. Evolution of the correlation induced by *SimuleMV* in terms of the number of variables. At the top, number of coefficients with absolute value above 0.8 ($\#AC > 0.8$). At the bottom, mean absolute coefficient (MAC).

$(1/rank(X))\%$ for minimum L value.

5 Examples of application

In the following we discuss three examples of application of *SimuleMV*.



(a) 10×10 (b) 10×100 (c) 10×1000

(d) 100×10 (e) 100×100 (f) 100×1000

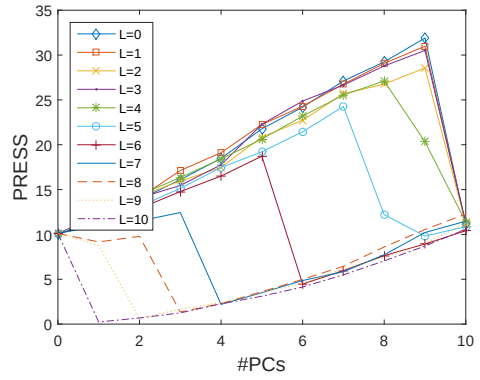
(g) 1000×10 (h) 1000×100 (i) 1000×1000

Fig. 10. Percentage of captured variance of the first PC in terms of the correlation level.

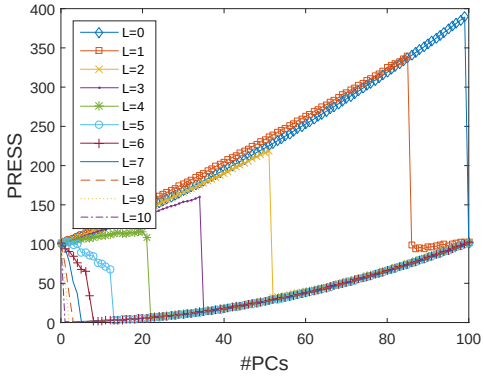
5.1 Cross-validation in PCA

The selection of the number of PCs in PCA in general and using cross-validation (CV) in particular has been a matter of research in chemometrics for several decades [7–9]. A main CV algorithm is the element-wise k-fold (ekf) CV [8,10] and its fitting counterpart, the column-wise k-fold (ckf) [11]. Both ekf and ckf provide an error curve whose minimum identifies the best choice in the number of PCs.

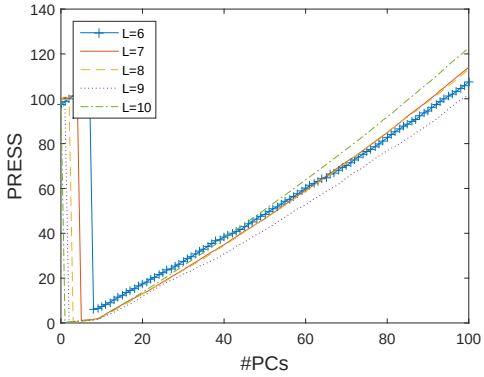
Although a lot of theoretical work has been done on these algorithms and



(a) 10×10



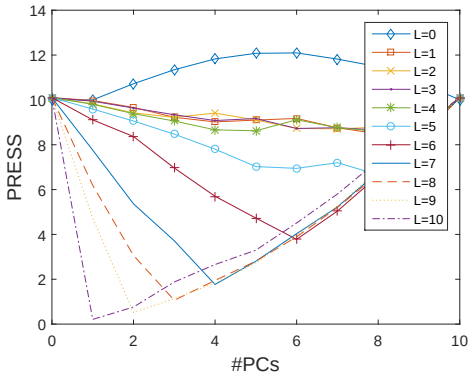
(b) 100×100



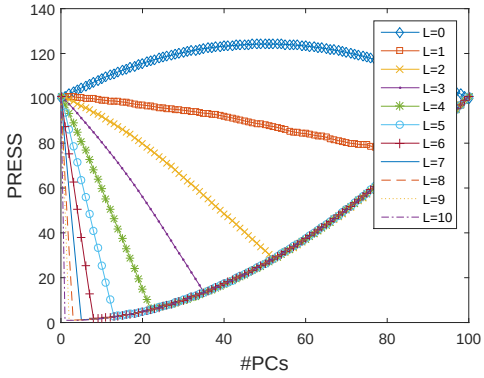
(c) 10×100

Fig. 11. PRESS curves by ekf for different correlation levels.

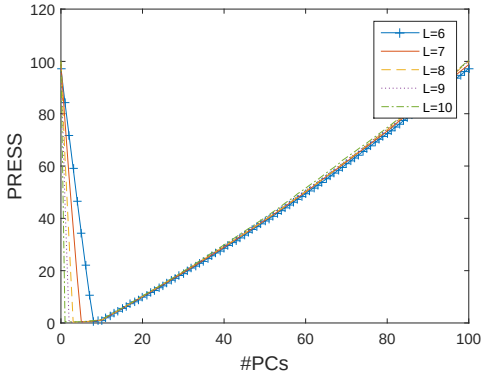
potential variants, see for instance [9,12], they are not well understood yet. We can gain a nice picture of the behavior of the methods using *SimuleMV*. Figures 11 and 12 compare the performance of ekf and ckf for several data dimensions



(a) 10×10



(b) 100×100



(c) 10×100

Fig. 12. PRESS curves by ckf for different correlation levels.

and correlation levels. We used the ekf and ckf implementations in the MEDA Toolbox [13]. We can see that, although the minimums are located in the same numbers of PCs, the ckf provides simpler curves, much easier to understand.

In particular, ckf seems to be more suitable for the automatic selection of PCs, because the curves do not present several local minimums. If we also consider that ckf is much faster, because the observation-wise k-fold operation is skipped, then ckf might be preferred to ekf on a general basis. If we focus now on ckf, Figure 12, we can see that the selected number of PCs is inversely proportional to the correlation level, as expected: high correlation levels lead to a reduced set of high variance components. Also, we can see from the regular shapes, in particular in Figure 12(b), that the profiles are mainly determined by the factors considered in *SimuleMV*: dimension and level of correlation. This agrees with theoretical studies [9,11].

Although this was only an example, the potentiality in the use of *SimuleMV* for CV studies is clear. Following this approach we can investigate further developments of algorithms for dimensionality selection in PCA and other models.

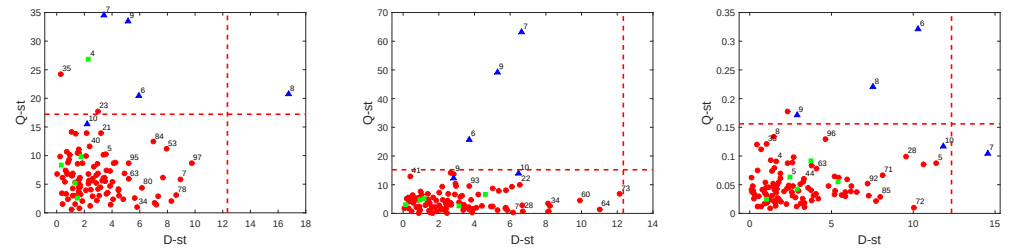
5.2 Multivariate Statistical Process Control

A lot of research in chemometric journals has been devoted to anomaly detection using Multivariate Statistical Process Control (MSPC) [14–18]. There are still many controversial issues, like for instance the dimensionality selection, the use of PCA, PLS, or non-linear counterparts, the procedure for diagnosis, etc. Again, *SimuleMV* can be useful for this type of research. Take the example in Figure 13, where we simulated 100 calibration observations, and normal and anomalous test sets, for different numbers of variables and correlation levels. The calibration data were simulated with the *simuleMV* algorithm. Test data were simulated with ADICOV using the covariance matrix of the calibration

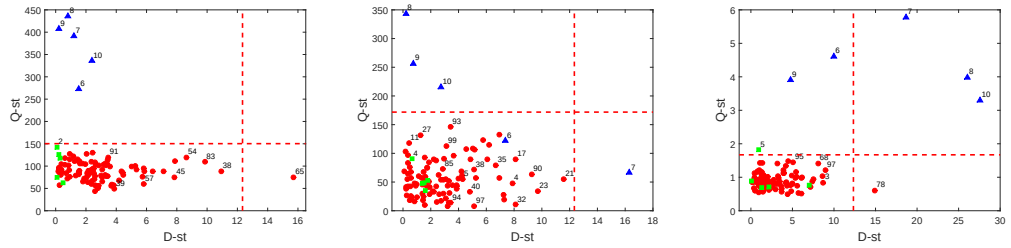
data. Anomalous data were obtained for 3 times the variance of normal data. In all cases, 3 PCs were intentionally selected without a proper justification, in order to see whether this non-justified way of determining the number of PCs affects the performance of MSPC. It can be seen that although we did not follow an appropriate procedure for PCs selection, the MSPC methodology is fairly robust, with a low number of false positives and negatives. The theoretical control limits used [19] work very well provided the number of observations is sufficient. However, if the number of observations is low, lets say 10, we see a different picture, see Figure 14. The D-statistic control limit is overestimated by far, and the 3 PCs model tends to overfit, which can be seen in the fact that normal test data present clearly higher Q-statistics than their calibration counterparts. Again, we can see that the use of *SimuleMV* is useful for understanding the behavior of a MSPC system.

5.3 Comparison of PLS and SPLS

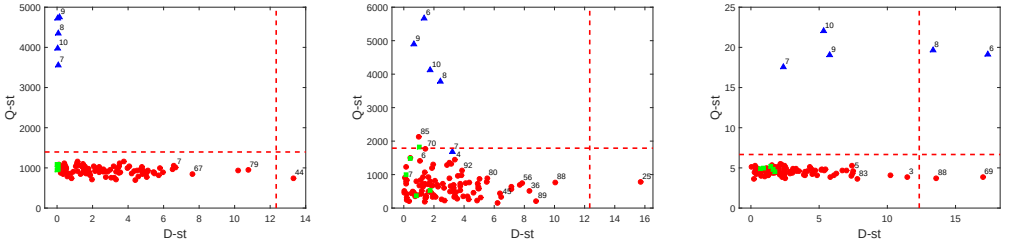
In the last example we investigate the differences between standard PLS and the sparse PLS variant developed in [20], kindly provided by Dr. Ewa Szymanska [21]. One controversial result in SPLS studies is whether it can outperform PLS only in interpretation or also in prediction performance [20,22]. Figure 15 shows a straightforward example to check this. In all cases, we used data of size 20×100 for the X-block, obtained from *SimuleMV*, and a single response variable $\mathbf{y}$ was simulated from the first variable in $\mathbf{X}$ plus a low noise level. Clearly, this is a favorable situation for SPLS, similar to the example used in [23]. Figure 15 compares the medians of the goodness-of-prediction (Q) by double cross-validation with PLS and SPLS models, for 100 repetitions. Ac-



(a) 100 × 10, level 3 (b) 100 × 10, level 6 (c) 100 × 10, level 9



(d) 100 × 100, level 3 (e) 100 × 100, level 6 (f) 100 × 100, level 9



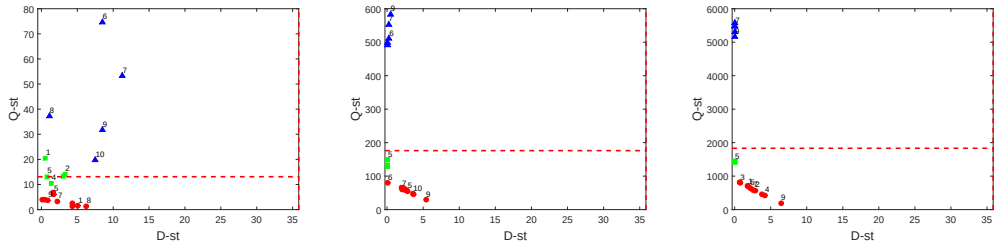
(g) 100 × 1000, level 3 (h) 100 × 1000, level 6 (i) 100 × 1000, level 9

Fig. 13. Experiments on Multivariate Statistical Process Control in PCA with 100 calibration observations. Calibration data is shown as circles in red, normal test data as squares in green and anomalous test data as triangles in blue.

cording to the results, PLS needs of a high correlation level to find adequate models, while SPLS does not have such restriction. Again, *SimuleMV* can be very useful to further investigate the behavior of sparse models.

6 Conclusion

In this paper we present *SimuleMV*, a new algorithm for random data simulation where the user can specify the size of the data to simulate and the



(a) 10×10, level 3 (b) 10×100, level 3 (c) 10×1000, level 3

Fig. 14. Experiments on Multivariate Statistical Process Control in PCA with 10 calibration observations. Calibration data is shown as circles in red, normal test data as squares in green and anomalous test data as triangles in blue.

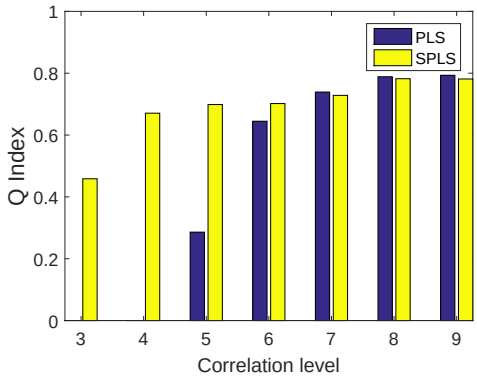


Fig. 15. Q Index of double cross-validation in PLS and SPLS for different correlation levels.

correlation level desired. This algorithm is designed to evaluate the performance of new multivariate analysis techniques. Its application was illustrated in several relevant problems in multivariate analysis, namely the selection of the number of Principal Components in Principal Component Analysis, the evaluation of the performance of sparse Partial Least Squares and the calibration of Multivariate Statistical Process Control systems.

Acknowledgements

This work is partly supported by the Spanish Ministry of Economy and Competitiveness and FEDER funds through project TIN2014-60346-R. The author would like to thank Dr. Edoardo Saccenti and the anonymous reviewers for their useful comments on the paper.

References

[1] P. D. Wentzell, A. C. Tarasuk, Characterization of heteroscedastic measurement noise in the absence of replicates, Analytica Chimica Acta 847 (2014) 16–28.

[2] F. Lei, M. Rotboll, S. B. Jorgensen, A biochemically structured model for saccharomyces cerevisiae, Journal of Biotechnology 88 (3) (2001) 205 – 221.

[3] F. Arteaga, A. Ferrer, How to simulate normal data sets with the desired correlation structure, Chemometrics and Intelligent Laboratory Systems 101 (1) (2010) 38–42.

[4] J. Camacho, P. Padilla, J. Díaz-Verdejo, K. Smith, D. Lovett, Least-squares approximation of a space distribution for a given covariance and latent subspace, Chemometrics and Intelligent Laboratory Systems 105 (2) (2011) 171–180.

[5] J. Camacho, Visualizing Big data with Compressed Score Plots: Approach and research challenges, Chemometrics and Intelligent Laboratory Systems 135 (2014) 110–125.

[6] F. Arteaga, A. Ferrer, Building covariance matrices with the desired structure, Chemometrics and Intelligent Laboratory Systems 127 (2013) 80–88.
URL `http://dx.doi.org/10.1016/j.chemolab.2013.06.003`

[7] S. Wold, Cross-Validatory Estimation of the Number of Components in Factor and Principal Components Models, Technometrics 20 (4) (1978) 397–405.

[8] R. Bro, K. Kjeldahl, A. K. Smilde, H. A. Kiers, Cross-validation of component models: a critical look at current methods., Anal Bioanal Chem. 390 (2008) 1241–1251.

[9] J. Camacho, A. Ferrer, Cross-validation in PCA models with the element-wise k-fold (ekf) algorithm: theoretical aspects, Journal of Chemometrics 26 (7) (2012) 361–373.

[10] J. Camacho, A. Ferrer, Cross-validation in PCA models with the element-wise k -fold (ekf) algorithm : Practical aspects.

[11] E. Saccenti, J. Camacho, On the use of the observation-wise k-fold operation in PCA cross-validation, Journal of Chemometrics 29 (8) (2015) 467–478.

[12] E. Saccenti, J. Camacho, Determining the number of components in principal components analysis: A comparison of statistical, crossvalidation and approximated methods, Chemometrics and Intelligent Laboratory Systems 149 (2015) 99–116.

[13] J. Camacho, A. Pérez-Villegas, R. A. Rodríguez-Gómez, E. Jiménez-Maas, Multivariate exploratory data analysis (meda) toolbox for matlab, Chemometrics and Intelligent Laboratory Systems 143 (2015) 49 – 57.

[14] J. MacGregor, T. Kourti, Statistical process control of multivariate processes, Control Engineering Practice 3 (3) (1995) 403–414.

[15] T. Kourti, P. Nomikos, J. F. MacGregor, Analysis, monitoring and fault diagnosis of batch processes using multiblock and multiway PLS, Journal of Process Control 5 (4) (1995) 277–284.

[16] B. M. Wise, N. L. Ricker, D. F. Veltkamp, B. R. Kowalski, Theoretical basis for

the use of principal component models for monitoring multivariate processes, Process Control and Quality 1 (1) (1990) 41–51.

[17] J. Camacho, J. Picó, Online monitoring of batch processes using multi-phase principal component analysis, Journal of Process Control 16 (10) (2006) 1021–1035.

[18] A. Ferrer, Multivariate Statistical Process Control Based on Principal Component Analysis (MSPC-PCA): Some Reflections and a Case Study in an Autobody Assembly Process, Quality Engineering 19 (4) (2007) 311–325.

[19] P. Nomikos, J. MacGregor, Multivariate Statistical Process Control Charts for Montoting Batch Processes (1995).

[20] K.-A. Le Cao, D. Rossow, C. Robert-Granié, P. Besse, A Sparse PLS for Variable Selection when Integrating Omics data, Statistical Applications in Genetics and Molecular Biology 7 (1) (2008) pp. 35.

[21] E. Szymańska, G. H. Tinnevelt, E. Brodrick, M. Williams, A. N. Davies, H. J. van Manen, L. M. C. Buydens, Increasing conclusiveness of clinical breath analysis by improved baseline correction of multi capillary column - ion mobility spectrometry (MCC-IMS) data, Journal of Pharmaceutical and Biomedical Analysis 127 (2015) 170–175.

[22] R. Calvini, A. Ulrici, J. M. Amigo, Practical comparison of sparse methods for classification of Arabica and Robusta coffee species using near infrared hyperspectral imaging, Chemometrics and Intelligent Laboratory Systems 146 (2015) 503–511.

[23] P. Filzmoser, M. Gschwandtner, V. Todorov, Review of sparse methods in regression and classification with application to chemometrics, Journal of Chemometrics 26 (3-4) (2012) 42–51.