Simulating new multicriteria data from a given data set

Jairo Cugliari[1] and **Antoine Rolland**[2]

Abstract. Several methods have been proposed in the past decades to deal with Multicriteria Decision Aiding (MCDA) problems. Even if the axiomatic foundations of these methods are generally well-known, comparing the different methods or simply analysing the results produced by the methods on real-life problems is arduous as there is a lack of benchmark MCDA datasets in the literature. A solution is therefore to simulate new MCDA dataset examples. But the analysis of real-world examples show that one must deal with data that are lightly linked, and then it is important to take into account dependency between variables when simulating new datasets. We propose in this paper three different approaches to simulate new data based on existing small datasets. We describe these methods, we propose a quality analysis of the results, and we experiment the methods on different examples from the literature.

1 Introduction

There is in Multicriteria Decision Aiding (MCDA) a great number of very specific methods to be proposed, with multiple variants. Testing these methods on several situations, using real datasets, should improve our understanding of advantages and inconveniences of each method, even if an axiomatic analysis has been done. But sometimes in real-world cases, only very few data are available; for example, from an preference learning point of view, the dataset should be so limited that it is too small to be divided into a test subset and a validation subset. Researchers should also desire to have more data to test the proposed methods. Therefore, as already pointed out in [4], there is a need of simulated multicriteria datasets to be used by the MCDA community, either for benchmarking or just to improve the understanding of each method. This paper deals with the simulated datasets issue for MCDA. Our goal is to propose one or several method to simulate new multicriteria data from an existing dataset. Simulated data should be as similar as possible to the initial dataset. Therefore, the proposed methods are supposed to be able to capture the specific structure of the initial dataset (if any), and then generate new data on demand.

Good practices in MCDA point out the fact, among others, that values on criteria should be as statistically independent as possible. Please note that this statistical independence is not the preference independence between criteria (see [7]). It just means that the values taken on different criteria should not be correlated (linearly or not) to avoid redundancy. Each time we mention independence in this paper will refer to statistical indepenence. ut in real life the values taken by an alternative on different criteria are generally not totally independent. Therefore, multicriteria data cannot be well simulated using only statistical independent sampling on each variable. The problem is then to model the correlation between criteria in a plausible way.

[1] Université de Lyon, ERIC Labs, email: Jairo.Cugliari@univ-lyon2.fr
[2] Université de Lyon, ERIC Labs, email: Antoine.Rolland@univ-lyon2.fr

In [4] we introduced a statistical approach to overcome this difficulty using copulas. In the present paper, we first propose in section 2 a new simulation method using Principal Component Analysis (PCA), and we present also the simulation method based on copulas. In section 3, we propose a quality analysis of the simulation results regarding the three methods (copulas, PCA and independent sampling). Last we experiment these methods on real datasets and show that PCA and copulas-based simulators lead to simulated datasets of higher quality than simple independent samples.

2 Generating methods

We describe in this section the three frameworks we use to create simulated MCDA data from available dataset that we describe by a matrix $\mathbf{X}$ with n rows and p columns. Each column represent an criterion (variable). Each row represent an alternative. The first method does not take into account any dependence structure of $\mathbf{X}$ and so simulate values on each criterion independently from others. While this approach can be useful in some case, in real life applications the columns of $\mathbf{X}$ can not be considered totally independent. Therefore, we incorporate this element on the simulation scheme for the second an third method.

2.1 Simulation by independent draws

The first framework uses the inverse of the classical probability integral transform. That is, if a random variable say Y has a distribution function F_Y, then one can simulate realizations of Y in two steps. First, one draws a realization u from a standard uniform law (i.e. with support on the unit interval). Then, the realization y for Y are obtained by doing $y = F_Y^{-1}(u)$, where

$$F_y^{-1}(u) = \inf\{y : F_Y(u) \geq y\} \tag{1}$$

is a generalized inverse of F (since the distribution functions are weakly monotonic and right-continuous).

Our starting point is the set $y_1, \ldots, y_n$ of size n containing realisations of the target variable with an unknown distribution function. Therefore, we estimate F_y by means of the empirical distribution function,

$$\hat{F}_Y(y) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\{y_i \leq y\}}, \tag{2}$$

which is, for each y a simple average of indicators. Graphically, the empirical distribution function is a stepwise function with jumps at the observed realizations $y_1, \ldots, y_n$ an so inducing a restriction to the realization : only observed data points can be obtained with the simulation. In order to relax this constraint we apply a additional smoothing step to obtain from (2) a continuous version using interpolation splines. Then, one need to simulate standard uniform realizations u and apply (1) replacing F_Y by $\hat{F}_Y$.

2.2 Independent draws on latent factors

The independent draws approach is somewhat disappointing when applied to MCDA framework, because the dependence between the alternatives is not considered. In order to overcome with this drawback one may consider a simple transformation of the dataset $\mathbf{X}$ to create latent factors over which the independent hypothesis will be more reasonable. Actually, if the dependence structure of $\mathbf{X}$ is only linear, one may rely on the principal components analysis to extract orthogonal variables which explain the best the variability on the data. Moreover, if $\mathbf{X}$ followed a gaussian distribution this approach would generate independent factors. Our aim is not to push so far the hypothesis but to have a simple approach that will overcome the problems mentioned before.

Concretely, we decompose X using the classical principal component analysis to get a new matrix $\mathbf{F}$. We keep the the barycentre and scale of the original dataset by an appropriate centring and standardization on the columns of $\mathbf{X}$.

Then, we apply the independent draw approach on the columns of $\mathbf{F}$ in order to get a simulated set of latent factors. Using the well known reconstruction formula for PCA, the latent factors are used to get a simulated centred and standardized version of X. Finally, the original barycentre and scales are incorporated to yield on the simulated version of $\mathbf{X}$.

Notice that while many matrix decomposition schemes exists, most of them can not be used because they lack off a reconstruction formula (i.e. Independent Components Analysis).

2.3 Simulation through copula

Simulation through copula on the MCDA framework has been recently proposed by [4]. Following the reference, we describe briefly the procedure without given the technical details.

In a nutshell a copula is a multivariate cumulative distribution function which has all its margins uniformly distributed on the unit interval (see [6] for a more formal presentation of the subject). Intuitively, a probabilistic multivariate structure can then be viewed as the coupling of the marginal behaviour by means of a copula C.

The pair-copula construction has been proposed to avoid some problems that arise on high dimension datasets (large p). Then the multivariate joint density of X is decomposed into a cascade of building blocks called pair-copula. Let f be the joint density of X which is factorized (uniquely up to a relabelling of the elements of X) as

$$f(y_1, \dots, y_n) = f(y_p) f(y_{p-1}|y_n) \dots f(y_1|y_2, \dots, y_p). \quad (3)$$

Then, one can write each of the conditional densities on (3) using the copula recursively which yields on this general expression for a generic element y_i of X given a generic conditioning vector v

$$\begin{aligned} f(y_i|v) =& c_{y_i, v_j|v_{-j}}\left(F(y_i|v_{-j}), F(v_j|v_{-j})\right) \\ & \times f(y_i|v_{-j}). \end{aligned} \quad (4)$$

In last expression we use the notation v_j for the j-th element of v and v_{-j} for all the elements of v but v_j.

Vines copulas have been proposed to classify alternatives factorizations into a structured graphical model. This construction allows highly flexible decompositions of the (possibly high) dimensional distribution of X because each pair-copula can be chosen independently from the others. The iterative decomposition provided by the pair-copula construction is then arranged into a set of linked trees (acyclic connected graph). Two special schemes are usually used: C-vines (canonical vines) and D-vines. In the former one, a dependent variable is identified and chosen to be the root of the tree. In the following tree, the dependence will be computed conditional on this first variable and so on. In the latter scheme, a variable ordering is chosen. Then on the first tree one models the dependence of each of the consecutive pairs of variables. The following tree will model the dependence of the remaining pairs, conditional on the those that were already modelled.

Simulation of copula data (i.e. n-variate data with uniformly distributed marginals) can be done using the probability integral transform. It is convenient to define the h-function

$$h(y|v, \theta) = \frac{\partial^d C_{y, v_j|v_{-j}}(F(y|v_j), F(y|v_{-j}), |\theta)}{\partial F(v_j|v_{-j})}, \quad (5)$$

where θ is a parameter vector associated to the decomposition level. The h-function is the conditional distribution of x given v and we let $h^{-1}(u|v, \theta)$ be its inverse with respect to u, i.e. the inverse of the cumulative conditional distribution. The simulation for the vine is as follows. First sample n uniformly distributed random variables $w_1, w_2, \dots, w_n$. Then use the probability integral transform of the corresponding conditional distribution:

$$\begin{aligned} y_1 &= w_1, \\ y_2 &= F^{-1}(w_2|y_1), \\ y_3 &= F^{-1}(w_3|y_1, y_2), \\ &\dots \\ y_p &= F^{-1}(w_{n-1}|y_1, \dots, y_{p-1}). \end{aligned}$$

At each step, the computation of the inverse conditional distribution is made through the (inverse) h-function.

3 Experiments

3.1 Experiment process

We present in this section the testing process of our experiments. Our goal is to compare the results of simulating data using the three different methods previously stated. For this purpose, we first need to state a quality index of the simulation data.

3.1.1 Quality index

We state that data are correctly simulated if it is not possible to distinguish the real data and the simulated ones. So we need a tool that is able to distinguish two different distributions, such as a statistical multivariate goodness-of-fit test. But as pointed out by McAssey [5], non-parametric goodness-of-fit test that can be used in practice is something very hard to find. We then choose to use the goodness-of-fit test proposed by Szkely and Rizzo [8], based on a geometric approach, and implemented in R. The null hypothesis H_0 is "the two multivariate distributions are the same", versus hypothesis H_1 "the two multivariate distributions are the different". The test returns a p-value : if this p-value is less than a fixed threshold, then the difference between the two distribution is said to be statistically significant, and then the simulated data cannot be considered to have the same distribution as the real ones. The p-value can be seen as a quality index : the greater the p-value, the greater the simulation quality. Of course the p-value will change for each simulated data set. Therefore, the following testing process has been established to compare the different simulation methods:

1. for each real data set, for each simulation method, simulate n new data sets;
2. for each simulated data set, compute the p-value of the goodness-of-fit test comparing real and simulated data sets;
3. plot the boxplot of all the p-values.

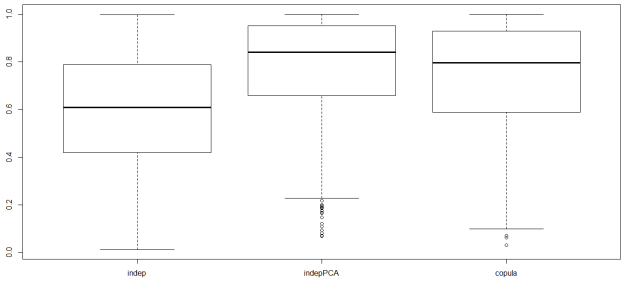


Figure 1: Example of a quality boxplot

On Figure 1, one can observe that real and simulated data can easily be confused, as more than 75% of the simulated data sets have a p-value greater than 0.5 for the confusion test (remember that generally the threshold to reject H_0, equal distribution hypothesis, is a p-value less than 0.05). Other comparison processes have been studied. Especially, we tried to use supervised and unsupervised classification methods to determine whether real and simulated data can be distinguished using machine learning. However, these methods (SVM, k-means, random forest) did not managed to separate real and simulated data when the data have the same margin distribution, which is always the case here by construction.

3.1.2 Comparing different generating processes

In the following, we will use boxplots like the one presented in Figure 1 to compare different simulation processes. If the distribution of p-values obtained by method A is higher than the distribution of p-values obtained by method B, it means that method A gives better simulations than method B, as it should be more difficult to reject the equal distribution hypothesis in case A than in case B. We will compare the three different methods proposed in section 2: "indep" corresponds to independent variables samples, "PCA" corresponds to independent variables samples on the different PCA axes, and "copula" uses copula to learn the links between variables.

3.1.3 Influence factors

Different factors can have an influence on the quality of the data simulation:

1. the number of variables
2. the size of the learning set
3. the strength of the link between variables

The effect of each of these factors will be tested independently. The testing process is the following:

1. for each value of the tested variable, generate 30 different datasets using a multivariate normal distribution with a specified correlation coefficient between the variable (through the Cholewsky matrix). Then exponential (for one variable) and log (for another variable) transformations are used to produce a dataset with a non-linear controlled link between variables. These are the reference datasets.
2. for each reference data set, generate 30 simulated datasets (of the same size that the reference dataset) with each simulation method.
3. Compute the p-value of the goodness-of-fit test comparing reference and simulated datasets.
4. For each method, there is 900 p-values obtained in the same conditions (same number of variables, same size of the learning set, same correlation degree between variables). Then plot the boxplot of the obtained p-values.

3.2 Experiment results

3.2.1 Number of variables

We choose to test the effect of the number of variables on the simulation process by varying the number of variables between 3 and 6. The other parameters are unchanged and have been fixed at 30 for the size of the reference datasets, and 0.5 for the correlation coefficient between criteria. The results are shown in Figure 2. One can see that the effect of changing the number of variables is pretty null, as the four boxplots are very similar. The three methods have almost the same performance whatever the number of variables is.

3.2.2 Size of the learning set

We choose to test the effect of the learning set size on the simulation process by varying the learning set size between 20 and 50 items. The other parameters are unchanged and have been fixed at 4 for the number of variables, and 0.5 for the correlation coefficient between criteria. The results are shown in Figure 3. One can see that the effect of changing the size of the learning set is different for the three methods. The effect is null for the copula method. The effect is small for the PCA method and more important for the independent method : results are worse with an increase of the size of the learning set, certainly because an increase of the size of the datasets has a direct effect on the power of the test, i.e. the capacity if the test to reject H_0 when H_0 is false.

3.2.3 Variables correlations

We choose to test the effect of the correlation degree on the simulation process by varying the correlation degree between 0.1 and 0.9s. The other parameters are unchanged and have been fixed at 4 for the number of variables, and 30 for the size of the learning set. The results are shown in Figure 4. One can observe the differences between the three methods with the correlation degree between the variables. Just remember that a post-treatment has been done to modify the data in order to have a functional (but not linear) link between the variables. The results are very interesting: one can easily see that the three methods have also the same results when the correlation degree is weak, but when the correlation degree is strong, the simulation method based on copula is able to capture the link between variables. The PCA method is also able to capture this link, less powerful in the case of non-linear link. The independent method, as guessed, does not simulate data similar to the initial ones when there is a correlation between variables. Therefore, it is clear on the Figure 4 that the simulation method based on PAC or copula produces simulated data that are more similar to a set of initial data than independent samples.

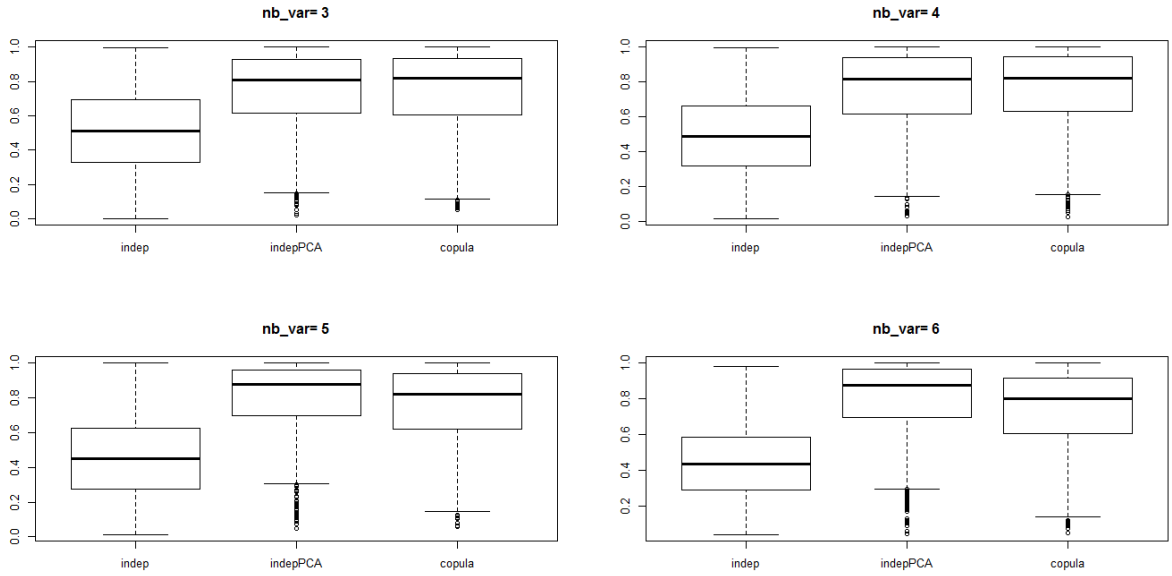


Figure 2: Variation of the quality plots with the number of variables – number of variables=3, 4, 5, 6

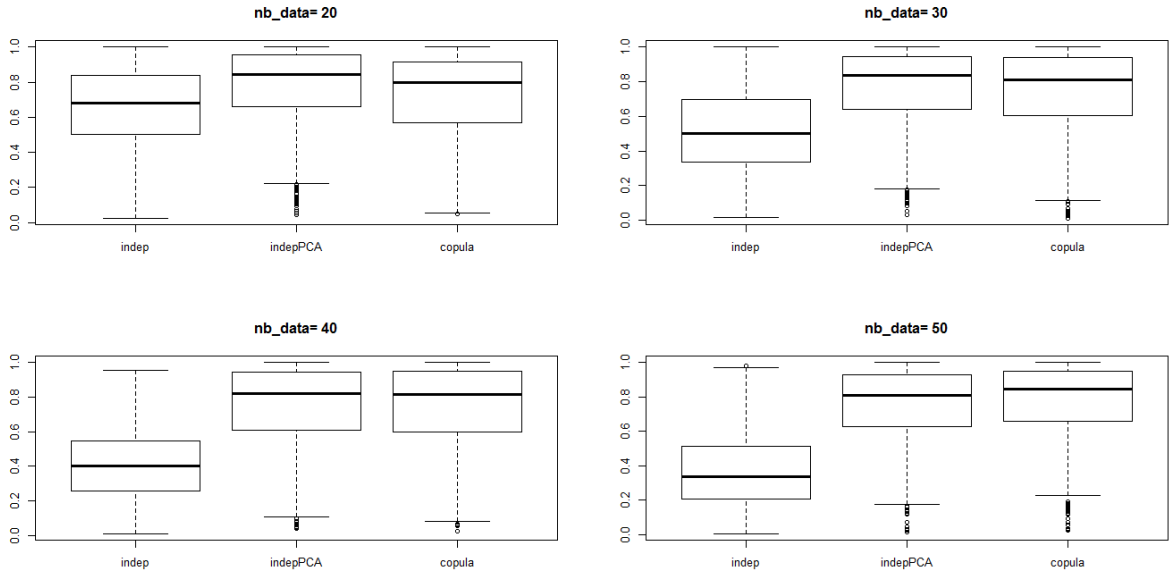


Figure 3: Variation of the quality plots with the size of the learning set – number of learning items= 20, 30, 40, 50

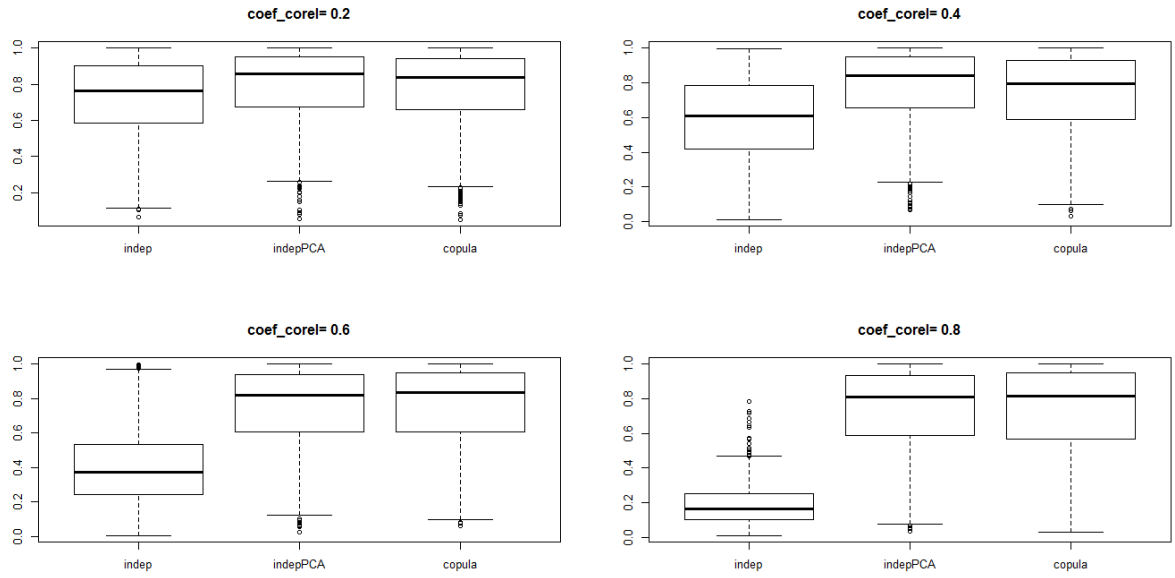


Figure 4: Variation of the quality plots with the correlation degree – cor. coef= 0.2, 0.4, 0.6, 0.8

3.3 Results on real datasets

We propose in this section to test the three different methods on some real datasets from the literature. The first one, proposed in [1], has 4 variables and 29 observations. The variables are almost independent. The second one is introduced in [2], and has 14 observations for 5 criteria. The correlation index between variables are around 0.3 The last one was proposed in [3] and has 27 observations for 4 variables. The correlation coefficient between variables are around 0.7. For each dataset, we produced 500 simulated datasets and then we draw the p-value boxplot as before. Results are shown in Figures 5a, 5b and 5c.

As a result, we can observe that the PCA sample method seems to produce more accurate simulated data, even if the copula sample method leads also to good quality simulated data. For the first dataset, the three different methods are similar, even if the independent sample method do not produce the same quality datasets as the two others. It seems that even if no correlation is observed between the variables in the initial dataset, PCA and copulas sample methods are able to catch a small dependent link and therefore lead to more accurate data generation. For the second data set also the three methods seems to produce sampling data very close to the initial ones. PCA sampling method seems also in this case be the best method, i.e. the method that produces new data that are impossible to discriminate from the initial ones via a goodness-of-fit test. The third case is different, as it is very clear in this case that the independent sampling method is not efficient : Figure 5c shows that most of the sampling produced by the independent methods can be distinguished from the initial dataset, whereas those produced with the copula sampling method, or even better with the PCA sampling method, can be considered as similar to the initial dataset.

These three examples give a good illustration that taking into account dependencies between variables (even if there are not obvious) leads to better simulated data than just independent sampling method. However, it is a surprise for us that on these three examples PCA sampling method seems to produce better results than copula-based method.

4 Conclusion

We proposed two different ways to improve the simulation quality of generated datasets for Multicriteria Decision Process. Both copulas and PCA methods lead to simulated datasets that are more accurate than pure independent samples. Copula method seems to have better results within controlled environment, whereas on the 3 “real” tested datasets PCA method seems to produce better results. However, we strongly encourage practitioners to use one of these two methods to extend the datasets they’re working on and then generate new datasets to test the proposed decision process.

REFERENCES

[1] R. Botreau and A. Rolland. Evaluation multicritère du bien-être animal en ferme : une application des méthodes développées en aide à la décision multicritère. In *8eme congrès de la ROADEF, Clermont-Ferrand*, 2008.

[2] D. Bouyssou, T. Marchant, M. Pirlot, P. Perny, A. Tsoukiàs, and Ph. Vincke. *Evaluation and decision models: a critical perspective*. Kluwer Academic, Dordrecht, 2000.

[3] Th. Briggs, P.L. Kunsch, and B. Mareschal. Nuclear waste management: An application of the multicriteria promethee methods. *European Journal of Operational Research*, 44(1):1 – 10, 1990.

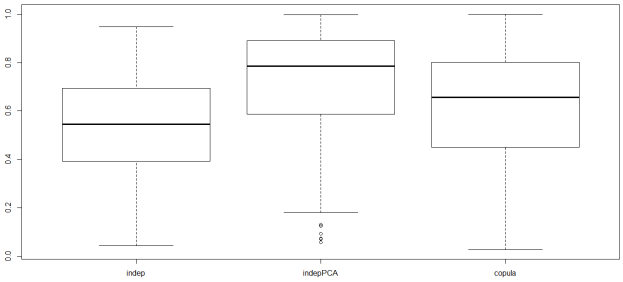
[4] J. Cugliari, A. Rolland, and T.M.T. Tran. On the use of copulas to simulate multicriteria data. In *DA2PL 2014 Workshop (proceedings)*, pages 3–9, 2014.

[5] Michael P. McAssey. An empirical goodness-of-fit test for multivariate distributions. *Journal of Applied Statistics*, 5(40):1120–1131, 2013.

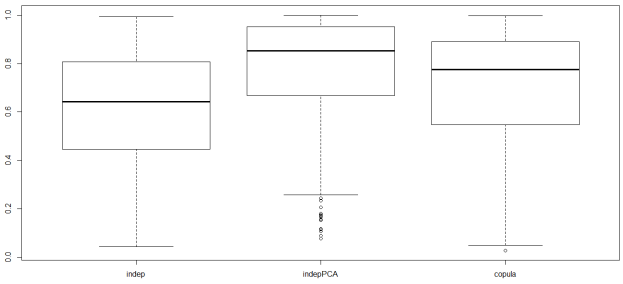
[6] R.B. Nelsen. *An Introduction to Copulas*. Springer, second edition, 2006.

[7] B. Roy. *Multicriteria Methodology for Decision Aiding*. Kluwer Academic Publisher, 1996.

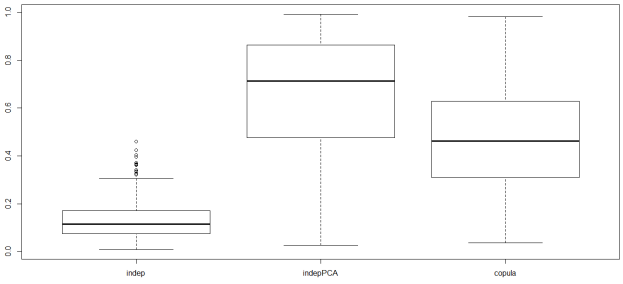
[8] Gabor J. Szkely and Maria L. Rizzo. Testing for equal distributions in high dimension. *InterStat*, (5), 2004.



(a) Case 1: Botreau and Rolland



(b) Case 2: Bouyssou *et al.*



(c) Case 3: Briggs *et al.*

Figure 5: Real datasets