AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ

KATEDRA INFORMATYKI STOSOWANEJ

Praca dyplomowa magisterska

Niespójności niekompletnych macierzy porównań parami
Inconsistency of incomplete pairwise comparisons matrices

Autor: *Dawid Talaga*
Kierunek studiów: *Informatyka*
Opiekun pracy: *dr hab. Konrad Kułakowski*

Kraków, 2018

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystycznego wykonania albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.", a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.): „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej «sądem koleżeńskim».", oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Serdecznie dziękuję Promotorowi Panu dr. hab. Konradowi Kułakowskiemu za zainteresowanie mnie tematem PC oraz życzliwe wsparcie.
Dziękuję moim Rodzicom, dzięki którym mogłem rozwijać się i studiować oraz bliskim mi osobom, które dodawały mi wiary w moje siły.

Contents

1. Introduction

1.1. Pairwise Comparisons method

People have made decisions for ages. Some of them are very simple and come easily but others, more complicated, require deeper analysis. It happens when there are many compared objects, which are complex and the selection criterion is hard to measure clearly. Fortunately, the development of mathematics brought an interesting tool - *The Pairwise Comparisons (PC) Method*. The first case of using the method (in a very simple version) is the election system described by *Ramond Llull* (Colomer, 2013) in the thirteenth century. Its rules were based on the fact that the candidates were pairwise compared with each other and the winner was the one who won in the largest number of direct comparisons. The method was reinvented in the eighteenth century by *Condorcet* and *Bord* (Kułakowski, 2016) as they proposed it in their voting system. In the twentieth century, the method found the application in the theory of social choice, the main representatives of which were the Nobel prize winners *Keneth Arrow* (Arrow, 1950) and *Amartya Sen* (Sen, 1977). The current shape of the method was influenced by the changes introduced by *Fechner* [Fechner 1966] and then refined by Thrustone (Thurstone, 1994). However, the breakthrough was the introduction to the method The Analytic Hierarchy Process (AHP) by *Saaty* (Saaty, 2008), which allowed to compare many more complex objects and create a hierarchical structure.The main aim of this paper is to check which method of calculating the inconsistency is the best in this case. I order to do it, a series of tests was carried out on various known inconsistency indexes, taking into account many different parameters: the matrix size, the amount of missing data, a level of inconsistency. The results of the research are included below.

The PC method is based on the assumption that it is not worth comparing all objects at the same time. It is better to compare them in pairs and then gather the results together. Such pairwise comparisons are much more intuitive and natural for a human being. How can one be sure that these judgments are consistent? Or what to do if some comparisons are missing? In such a case, is it worth taking the *PC method* at all?

The answer to the first question is the concept of inconsistency introduced into the method. This paper tries to answer the next two questions - meaning to examine whether available methods for determining inconsistencies give reliable results when a part of the comparisons are missing.

1.2. The aim of the work

The main aim of this paper is to check which method of calculating the inconsistency is the best when some comparisons are missing. I order to do it, a series of tests was carried out on various known inconsistency indexes, taking into account many different parameters: the matrix size, the amount of missing data, a level of inconsistency. The tests were implemented in R language, which properly fit into numerical calculations.

It was not know from the beginning what will be the result of this work. It was considered that one or more existing inconsistency indexes turn out appropriate also for incomplete matrixes or the tests show that the inconsistency can not be calculated by these methods. The results of the research are included below.

1.3. Contents of the work

The work consists of the theoretical part and the description of conducted experiments and their results. It gives together six chapters.

The Second section shows the Pairwise Comparisons method and the problem of inconsistency. Understanding the basics is necessary to go to further chapters.

In the third section sixteen available methods for calculating the inconsistency for the PC matrixes are presented.

The next chapter shows ideas which allows to perform tests. These are modifications of existing inconsistency indexes which enables to adjust their to incomplete matrixes and the algorithm which tests the quality of modified indexes.

The results of experiments are presented and discussed in the fifth section.

The last chapter contains summary, conclusions and ideas for future researches.

2. Pairwise Comparisons method

2.1. PC matrix

The *PC method* is used to choose the best alternative from a set of concepts. However, this goal is achieved by comparing in pairs. A numerical value is assigned to each pair. It not only determines which alternative is preferred but also informs about the intensity of this preference. In this way, the finite set of concepts $C = \{c_1, \ldots, c_n\}$ is transformed into a *PC matrix* $M = (m_{ij})$, where $m_{i,j} \in R$ and $i, j \in \{1, \ldots, n\}$. The *PC matrix* for n concepts is following:

$$M = \begin{pmatrix} 1 & m_{12} & \ldots & m_{1n} \\ m_{21} & 1 & \ldots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \ldots & 1 \end{pmatrix}$$

It is worth noting that the values m_{ij} and m_{ji} represent the same pair. Therefore, one should expect that $m_{ji} = \frac{1}{m_{ij}}$. If

$$\forall i, j \in \{1, \ldots, n\} : m_{ij} = \frac{1}{m_{ji}}, \tag{2.1}$$

the matrix is called a *reciprocal*.

2.2. Weight vector

The PC matrix is the basis for calculating the method. It is used in a function $\mu : C \to R$ that assigns a positive real number to each alternative in the set C. The vector

$$\mu = [\mu(c_1), \ldots, \mu(c_n)]$$

formed in this way is called *the weight vector* or *the priority vector* (see Rys. 2.1).It informs which alternative has won.

There are many ways to calculate the vector μ, among the popular ones are the method using the matrix's eigenvalues or the method based on geometric means (Saaty et al., 1998).

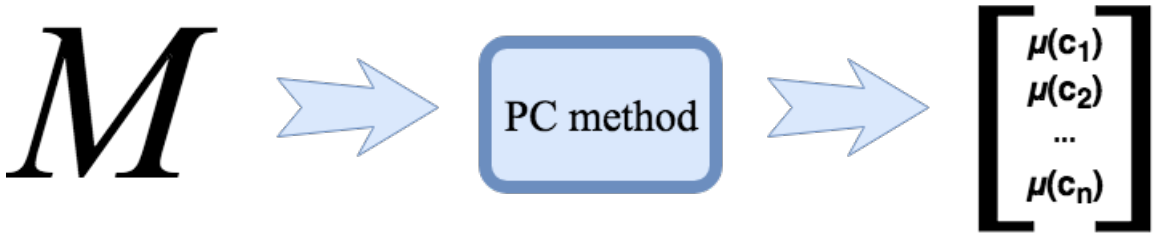


Fig. 2.1. Diagram of calculating *the weight vector*

2.3. Inconsistency

The second important parameter describing the *PC matrix* is *consistency*. The matrix is consistent if

$$\forall i, j, k \in \{1, \ldots, n\} : m_{ik} = m_{ij} m_{jk}. \tag{2.2}$$

Three numbers that should meet this assumption are called a *triad*.

If the matrix is consistent and the weight vector μ is computed, then for each variables i, j, where $1 \leq i, j \leq n$ meets the equation:

$$ij = \frac{\mu_i}{\mu_j}. \tag{2.3}$$

In practice, it is very rare for a matrix M to be completely consistent. In the long history of the PC method, many methods have been developed to calculate inconsistencies. Many of them are based directly on the definition of consistency ([eq:consistent]), some methods use the eigenvalues of the matrix, others are based on the assumption that each fully consistent matrix fulfills the condition ([eq:consistent2]).

3. Inconsistency indexes

3.1. Inconsistency indexes for complete matrixes

This subsection presents sixteen common inconsistency indexes. Their detailed description, including the formulas, is necessary to modify them in the next step so that they can also work for incomplete matrices. Many of them have been described and tested numerically in (Brunelli et al., 2013). In all methods, it is assumed that the *PC matrix* is reciprocal.

3.1.1. Saaty index

This is one of the most important and popular indexes and was introduced by *Saaty* (Saaty, 1977). In order to determine inconsistency, the matrix's eigenvalue should be computed. The author used the dependence that the largest eigenvalue of the matrix is equal to its dimension if and only if the given matrix is completely consistent. On this assumption, he based his thoughts and proposed the formula:

$$CI(A) = \frac{\lambda_{max} - n}{n-1}, \tag{3.1}$$

where λ_{max} is the principal eigenvalue of the PC matrix and n is its dimension.

3.1.2. Geometric consistency index

This index on the assumption ([eq:consistent2]) was proposed by *Craford* and *Williams* (Crawford et al., 1985) and then refined by *Aguaròn* and *Moreno-Jimènez* (Aguaron et al., 2003). In this case the priority vector should be calculated using the geometric mean method. Consider ([eq:consistent2]) one can create a matrix:

$$E = \left[e_{ij} \mid e_{ij} = a_{ij}\frac{w_j}{w_i} \right], \quad i,j = 1,...,n. \tag{3.2}$$

The inconsistency index is calculated as follows:

$$GCI = \frac{2}{(n-1)(n-2)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} ln^2 e_{ij}. \tag{3.3}$$

3.1.3. Koczkodaj index

One of the most popular inconsistency indexes was proposed by *Koczkodaj* (Koczkodaj, 1993). It is based directly on the definition of consistency ([eq:consistent]). The value of the inconsistency index for one triad ([triad]) was defined as:

$$K_{i,j,k} = min\{\frac{1}{a_{ij}} \mid a_{ij} - \frac{a_{ik}}{a_{jk}} \mid, \frac{1}{a_{ij}} \mid a_{ik} - a_{ij}a_{jk} \mid, \frac{1}{a_{jk}} \mid a_{jk} - \frac{a_{ik}}{a_{ij}} \mid\}. \tag{3.4}$$

This formula has been simplified by Duszak and Koczkodaj (Duszak et al., 1994) and is given as:

$$K(\alpha, \beta, \gamma) = min\{\mid 1 - \frac{\beta}{\alpha\gamma} \mid, \mid 1 - \frac{\alpha\gamma}{\beta} \mid\}, \quad where\, \alpha = a_{ij}, \beta = a_{ik}, \gamma = a_{jk} \tag{3.5}$$

Then it was generalized (Duszak et al., 1994) for $n > 2$. Finally, the inconsistency index has the following form:

$$K = max\{K(\alpha, \beta, \gamma) | 1 \leq i < j < k \leq n\}. \tag{3.6}$$

It is worth noting that not only does the coefficient find the greatest inconsistency but also indicates the place in which it occurs.

3.1.4. Kazibudzki indexes

Based on the Koczkodaj inconsistency index and observation that $ln(\frac{\alpha\gamma}{\beta}) = -ln(\frac{\beta}{\alpha\gamma})$, *Kazibudzki* proposed several additional inconsistency indexes (Kazibudzki, 2016). Instead of the formula for inconsistency of the triad [eq:k-abg], he introduced two new formulas:

$$LTI(\alpha, \beta\gamma) = \mid ln(\frac{\alpha\gamma}{\beta}) \mid, \tag{3.7}$$

$$LTI*(\alpha, \beta\gamma) = ln^2(\frac{\alpha\gamma}{\beta}). \tag{3.8}$$

Based on the above equations, *Kazibudzki* proposed new indexes. The simplest ones use the geometric mean of the triads. Thus, new indexes could be written in the form:

$$MLTI(LTI) = \frac{1}{n}\sum_{i=1}^{n}\left[LTI_i(\alpha, \beta\gamma)\right], \tag{3.9}$$

$$MLTI(LTI*) = \frac{1}{n}\sum_{i=1}^{n}\left[LTI*_i(\alpha, \beta\gamma)\right]. \tag{3.10}$$

After further research *Kazibudzki* introduces another inconsistency index (Kazibudzki, 2017), again based on ([eq:lti*]). It was defined as $CM(LTI*) = \frac{MEAN[LTI*(\alpha,\beta,\gamma)]}{1+MAX[LTI*(\alpha,\beta,\gamma)]}$. Hence,

$$CM(LTI*) = \frac{\frac{1}{n}\sum_{i=1}^{n}[LTI*_i(\alpha, \beta, \gamma)]}{1 + max\{LTI*_i(\alpha, \beta, \gamma)\}}. \tag{3.11}$$

3.1.5. Index of determinants

This index was proposed by *text*Pelaez and *Lamata* (PelÃ¡ez et al., 2003) and is also based on the concept of triad. The authors noticed that *PCM matrices* can be construct on the basis of triads. Their determinant is closely related to the consistency of the matrix.

For every triad (a_{ik}, a_{ij}, a_{jk}) one can build a matrix in the form:

$$T_{ijk} = \begin{pmatrix} 1 & a_{ij} & a_{ik} \\ \frac{1}{a_{ij}} & 1 & a_{jk} \\ \frac{1}{a_{ik}} & \frac{1}{a_{jk}} & 1 \end{pmatrix}, \quad where\, i < j < k. \tag{3.12}$$

The determinant of this matrix is:

$$det(A) = \frac{a_{ik}}{a_{ij}a_{jk}} + \frac{a_{ij}a_{jk}}{a_{ik}} - 2. \tag{3.13}$$

If the matrix is fully consistent, then $det(A) = 0$, else $det(A) > 0$. Based on the above considerations, the authors introduced the new inconsistency index that can be formulated as follows:

$$CI* = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{a_{ik}}{a_{ij}a_{jk}} + \frac{a_{ij}a_{jk}}{a_{ik}} - 2\right). \tag{3.14}$$

3.1.6. Kułakowski and Szybowski indexes

Kułakowski and *Szybowski* proposed two further inconsistency indexes (Kulakowski et al., 2014), which are also based on triads. They use the fact that the number of triads that can be found in a *PCM matrix* is

$$\binom{n}{3} = \frac{n!}{(n-3)!3!} = \frac{n(n-1)(n-2)}{6}. \tag{3.15}$$

The index is formulated as follows:

$$I_1 = \frac{6\sum_{t\in T} K(t)}{n(n-1)(n-2)}, \tag{3.16}$$

where $K(t)$ is the Koczkodaj index for triad $t = (\alpha, \beta, \gamma)$ of the set of all triads T.

The second inconsistency index is similar:

$$I_2 = \frac{6\sqrt{\sum_{t\in T} K^2(t)}}{n(n-1)(n-2)}. \tag{3.17}$$

Indexes can be combined with each other to create new coefficients. In this way *Kułakowski* and *Szybowski* proposed two new indexes. The first one is based on ([eq:K]) and ([eq:I1]). This index allows to choose what effect on the result should the greatest inconsistency found have and what the average

inconsistency of all triads. The new inconsistency index looks as follows:

$$I_\alpha = \alpha K + (1-\alpha) I_1, \tag{3.18}$$

where $0 \leq \alpha \leq 1$.

The second index expands the first one by ([eq:I2]):

$$I_{\alpha,\beta} = \alpha K + \beta I_1 + (1-\alpha-\beta) I_2. \tag{3.19}$$

3.1.7. Harmonic consistency index

Index introduced by *Stein* and *Mizzi* and it presents a completely new method of inconsistency counting (Stein et al., 2007). At the beginning it requires the creation of an auxiliary vector $s = (s_1, ..., s_n)^T$, where n is the dimension of the matrix A, for which the index will be calculated. Each element of the vector s is the sum of values in one column of the matrix A. Hence,

$$s_j = \sum_{i=1}^{n} a_{ji} \quad \forall j. \tag{3.20}$$

The authors proved that if the matrix A is consistent, then $\sum_{j=1}^{n} s_j^{-1} = 1$. The formula for the mean harmonic looks as follows (Brunelli, 2015):

$$HM = \frac{n}{\sum_{j=1}^{n} \frac{1}{s_j}}. \tag{3.21}$$

The final formula for inconsistency index was obtained by normalizing the above equation ([eq:hm']):

$$HCI = \frac{(HM(s) - n)(n+1)}{n(n-1)}. \tag{3.22}$$

3.1.8. Golden and Wang index

This index was introduced by *Golden* and *Wang* (Golden et al., 1989). It assumes that the priority vector was calculated using the geometric mean method, then normalized to add up to 1. In this way vector$g* = [g^*_{1,}, ..., g^*_n]$ was obtained, where n is the dimension of the matrix A. The next step is to normalize each column of the matrix A. After this, the sum of the elements of each column in matrix A is 1. The obtained matrix is marked with the symbol A^*. The inconsistency index is defined as follows:

$$GW = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} \mid a^*_{ij} - g^*_i \mid . \tag{3.23}$$

3.1.9. Salo and Hamalainen index

The index proposed by *Salo* and *Hamalainen* (Salo et al., 1995; Kazibudzki, 2016) uses the definition of inconsistency ([eq:consistent]), however it requires the creation of an auxiliary matrix, in which each element is the smallest and largest discrepancy from consistency based on formula ([eq:consistent]). The index takes all triads into account:

$$R = (r_{ij})_{nxn} = \begin{pmatrix} [\underline{r}_{11}, \overline{r}_{11}] & \dots & [\underline{r}_{1n}, \overline{r}_{1n}] \\ \vdots & \ddots & \vdots \\ [\underline{r}_{n1}, \overline{r}_{n1}] & \dots & [\underline{r}_{nn}, \overline{r}_{nn}] \end{pmatrix}, \tag{3.24}$$

where $\underline{r_{ij}} = min\left\{a_{ik}a_{kj} \mid k = 1, \dots, n\right\}$, $\overline{r_{ij}} = max\left\{a_{ik}a_{kj} \mid k = 1, \dots, n\right\}$ and n is the dimension of the tested matrix A. A numerical example was presented in (Brunelli, 2015). Based on the resulting matrix R, the authors proposed the following inconsistency index:

$$CM = \frac{2}{n(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{\overline{r}_{ij} - \underline{r}_{ij}}{\left(1 + \overline{r}_{ij}\right)\left(1 + \underline{r}_{ij}\right)}. \tag{3.25}$$

3.1.10. Cavallo and D'Apuzzo index

The authors *Cavallo* and *D'Apuzzo* based their index on triads but they conducted studies on a new path, generalizing them for linear, ordered abelian groups (Cavallo et al., 2009; Cavallo et al., 2010). Thanks to this, the index can be used also with other relations (Brunelli et al., 2013). Index for relation max can be presented in the form of a formula:

$$I_{CD} = \prod_{i=1}^{n-2} \prod_{j=i+1}^{n-2} \prod_{k=j+1}^{n} \left(max\left\{ \frac{a_{ik}}{a_{ij}a_{jk}}, \frac{a_{ij}a_{jk}}{a_{ik}} \right\} \right)^{\frac{1}{\binom{n}{3}}}. \tag{3.26}$$

3.1.11. Relative error

This index, proposed by *Barzaili* (Jonathan, ??), requires calculation of the weight vector using the arithmetic mean method for each row and creation of two additional matrices. Thus, the weight vector is

$$w_i = \frac{1}{n} \sum_{j=1}^{n} a_{ij},$$

where n is the dimension of the matrix. The two auxiliary matrices are calculated according to the formulas:

$$C = (c_{ij}) = (w_i - w_j)$$

$$E = (e_{ij}) = (a_{ij} - c_{ij})$$

Ultimately, the formula for the *Relative error* is following:

$$RE(A) = \frac{\sum_{ij} e_{ij}^2}{\sum_{ij} a_{ij}^2}. \tag{3.27}$$

3.2. Inconsistency indexes for incomplete matrixes

There are no inconsistency indexes for incomplete matrices. However, Those presented in chapter (3) could be use in such cases. It requires usually a slight modification of the index definition or calculation only for selected data. Below the ways in which the examined indexes have been adjusted to be able to deal with incomplete matrices are presented.

Saaty index

The input matrix is modified using the method proposed by *Harker* (Harker, 1987). It means that values $c+1$,where c is the number of non-empty elements in a given row, are places on the diagonal.

Geometric consistency index

During calculating the weight vector by the geometric mean, empty values are omitted. Additionally, in the formula ([eq:GCI]) only non-empty elements e_{ij} are used. The reason for this exclusion is that the domain of the logarithmic function is R^+.

Koczkodaj index, Kazibudzki indexes, Index of determinants:

Only those triads which do not contain empty values are taken into account.

Kułakowski and Szybowski indexes

Only those triads which do not contain empty values are taken into account. In addition, the number of triads is no longer calculated according to the formula ([eq:KulSzynPo3]) but determined directly by counting the number of triads.

Harmonic consistency index:

No modification.

Golden and Wang index:

During calculating the weight vector by the geometric mean empty values are omitted.

Salo and Hamalainen:

No modification.

Cavallo and D'Appuzo:

During calculating the product ([eq:CavDAp]) empty values are omitted.

Relative index:

No modification.

Proposed methods of adjustments of indexes allow to apply their in incomplete matrixes. However, their do not guarantee the best results. It requires further experiments. The existing indexes can be modified in many different ways. In this paper one put emphasis on the fact that the adjustments were slight. The aim is to examine existing indexes, not to create a new index.s

4. Studies of inconsistency indexes for incomplete matrices

The presented inconsistency indexes have been tested utilizing the Monte Carlo method. Their aim was to select those indexes which will give reliable results for incomplete matrices. Therefore, it was decided that the measure of the indexes' quality would be a *relative error* (expressed as a percentage), which took into account the value of the index for a full, inconsistent matrix and the value of the index for the same matrix after partial decomposition. To be sure that the results were fair, all indexes were tested on the same set of matrices. The different sizes of the matrices, the levels of incompleteness and the levels of inconsistency were taken into account. Then, in order to compare the indexes easily and to select the best ones, the results were averaged using the arithmetic mean. While building the algorithm to solve the problem (Kazibudzki, 2017) was used.

4.1. Algorithm

4.1.1. Steps of the algorithm

Procedure steps:

1. Randomly generate a vector $w = [w_1, ..., w_n]$ and a consistent *PCM matrix* associated with it $PCM = (m_{ij})$, where $m_{ij} = \frac{w_i}{w_j}$.
2. Disrupt the matrix by multiplying its elements (excluding the diagonal) by the value of d, randomly selected from the range $\left(\frac{1}{x}, x\right)$.
3. Replace values m_{ij}, where $i < j$ by values m_{ji}.
4. Calculate values of index with all methods for the created matrix.
5. Remove some values from the matrix by removing some of values. The level of incompleteness should be $g\%$.
6. Calculate the values of inconsistencies by all methods for the decomposed matrix.
7. Calculate the relative error for each index.
8. Repeat steps 1 to 10 X_1 times.
9. Calculate the average relative error for each inconsistency index for the *PCM matrix*.

10. Repeat steps 1 to 10 X_2 times.

11. Calculate the average relative error for each index by averaging the values obtained in step 9.

4.1.2. Details of algorithm

The above algorithm was carried out for values $X_1 = 100$, $X_2 = 100$. Tests were started for values d in the range $(1.1, 1.2, ..., 4)$ and then the results were averaged. It means that the average relative error of one index was calculated on the basis of 4000 matrices, each of which decomposed randomly 100 times. It gave together 400000 tests how good the index was.

In addition, tests were carried out for various sizes of matrices.
The results are divided into two parts:

1. A constant degree of incompleteness, different size of the matrix.

2. Different degrees of incompleteness, constant size of the matrix.

The aim of such a division is to pay attention to how the inconsistency indexes behave when the size of the matrix and the degree of incompleteness are changing. The results of the research are presented below.

4.2. Implementation

4.2.1. Development environment

Tests of indexes have been developed in R language which is appropriate for nuimerical calculations. It contains dozens of functions which support operations on matrixes and vectors. Integrated development environment (IDE) called RStudio has been used during implementation. This tool allows to create own packages which contains not only code but also documentation and information about licence and author. Package named *indexesForIncomplete* has been created. The most important part of this package is file *indexes.R* which performs calculations necessery to test indexes. RStudio supports programmer's work by syntax highlighting, built-in console, easy documentation searching and many others. The program is available on common operating systems. Before using RStudio one have to install R programming language.

4.2.2. Implementation of tests of inconsistency indexes

Implementation of tests inconsistency indexes consists of two steps:

1. Implementation of functions which calculates inconsistency indexes for given matrix (full or incomplete).

2. Implementation of tests which studies indexes for different matrixes and collects all results of these tests.

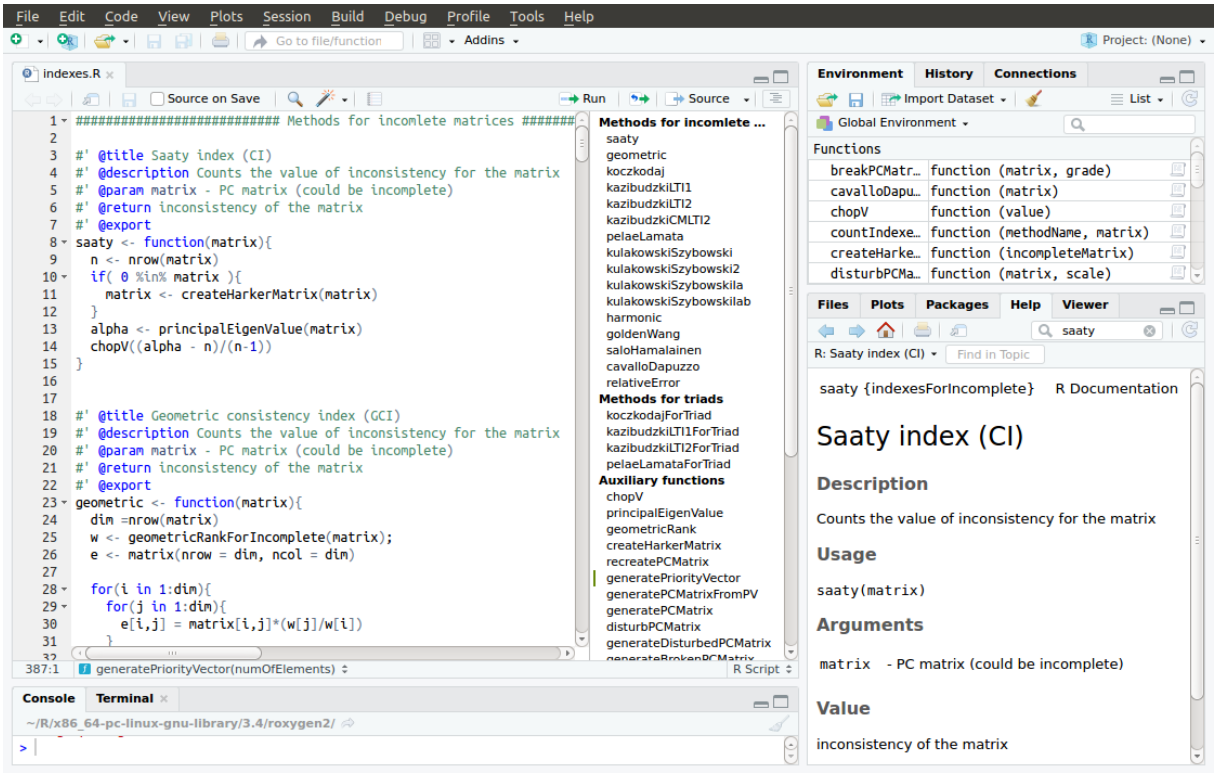


Fig. 4.1. Program *RStudio*

Implementation of inconsistency indexes

Sixteen functions which calculate inconsistency indexes using methods described in chapter 3. The functions have been wroten in such a way that allows handle both full and incomplete matrixes. One have not taken account of wrong matrixes, it means nonreciprocal or inconsistent PC matrixes. Each of these function has only one parameter - PC matrix. Exceptions are two methods implementing Kulakowski and Szybowski index which additionally take parameters α, β. The result of each function is value of inconsistency index. The functions have been extended by comments which informs about name of index related to given function, parameters and returned value. It allows to easily read and modify the code. Several examples of functions are presented below.

It is worth drawing attention to function which is called within the functions intended for indexes based on triads. This function generates triads from a matrix and next returns inconsistency for each o them. However, the way to calculate inconsistency for one triad depends on first function parameter. It informs about function name that calculates inconsistency of a triad specified method.

Implementation of tests

In the second step functions, which calculate the quality of the indexes for incomplete matrices, have been created. Functions, which generate specified matrixes, plays an important role. PC matrixes are created depending on size, the level of inconsistency and the degree of incompleteness.

The last part of functions relates testing how big relative error occurs for inconsistency indexes after deleting some values. To begin with functions, which test one index. They consider matrix size, the level

```
#' @title Saaty index (CI)
#' @description Counts the value of inconsistency for the matrix
#' @param matrix - PC matrix (could be incomplete)
#' @return inconsistency of the matrix
#' @export
saaty <- function(matrix){
  n <- nrow(matrix)
  if( 0 %in% matrix ){
    matrix <- createHarkerMatrix(matrix)
  }
  alpha <- principalEigenValue(matrix)
  chopV((alpha - n)/(n-1))
}
```

Fig. 4.2. The implementation of *Saaty* index

```
#' @title Koczkodaj matrix inconsistency
#' @description Counts the value of inconsistency for the matrix
#' @param matrix - PC matrix (could be incomplete)
#' @return inconsistency of the matrix
#' @export
koczkodaj <- function(matrix){
  triadsAndIdxs <- countIndexesForTriads("koczkodajForTriad", matrix)
  chopV(max(triadsAndIdxs))
}
```

Fig. 4.3. The implementation of *Koczkodaj* index

of inconsistency, the degree of incompleteness and the number of attempts which are performed for given matrix.

Then functions, which perform tests for each indexes, has been developed basing on the same matrixes. In that the set of matrixes , on which indexes go, is common. Thus, results are reliable and each index is considered the same way.

4.2.3. Documentation

Comments in code have been used to generating documentation. the package *roxygen2* has been made for this purpose. It has allowed to easy review code and know the functions./ Exemplary portions of the documentation are presented below.

```
#' @title Kulakowski and Szybowski consistency index (Ia)
#' @description Counts the value of inconsistency for the matrix
#' @param matrix - PC matrix (could be incomplete)
#' @return inconsistency of the matrix
#' @export
kulakowskiSzybowskiIa <- function(matrix, alfa, beta=0){
  triadsAndIdxs <- countIndexesForTriads("koczkodajForTriad", matrix)
  chopV(alfa*max(triadsAndIdxs) + (1-alfa)*mean(triadsAndIdxs))
}
```

Fig. 4.4. The implementation of *Kulakowski and Szybowski* index

```
#' @title Salo and Hamalainen consistency index (SH)
#' @description Counts the value of inconsistency for the matrix
#' @param matrix - PC matrix (could be incomplete)
#' @return inconsistency of the matrix
#' @export
saloHamalainen <- function(matrix){
  n <- nrow(matrix)
  sum <- 0

  rMin <- matrix(nrow = n, ncol = n, data = 0)
  rMax <- matrix(nrow = n, ncol = n, data = 0)

  for(i in 1:n){
    for(j in 1:n){
      c <- rep(0,n)
      for(k in 1:n){
        c[k] = matrix[i,k]*matrix[k,j]
      }
      rMin[i,j] = min(c)
      rMax[i,j] = max(c)
    }
  }

  for(i in 1:(n-1)){
    for(j in (i+1):n){
      sum <- sum + (rMax[i,j]-rMin[i,j])/((1+rMax[i,j])*(1+rMin[i,j]))
    }
  }

  cm <- 2/(n*(n-1))*sum
  chopV(cm)
}
```

Fig. 4.5. The implementation of *Salo and Hamalainen* index

```
#' @title Generates inconsistency indexes for each triad from the matrix
#' @param methodName - name of the method which computes inconsistency index for triad
#' @param matrix - PC matrix
#' @return vector of inconsistency indexes for triads
countIndexesForTriads <- function(methodName, matrix){
  triads <- generateTriads(matrix)
  if(is.null(dim(triads)) && length(triads) == 3){
    triads <- matrix(triads, 3, 1)
  }
  triadIdxs <- apply(triads, 2, methodName)
  triadIdxs
}
```

Fig. 4.6. The implementation of method *countIndexesForTriads*, which calculates inconsistency for each triad of a specified matrix

```
#' @title Generates PC Matrix on the basis of the size
#' @param numOfElements - number of elements of the matrix
#' @return reciprocal PC matrix
generatePCMatrix<- function(numOfElements){
  priorityVector <- generatePriorityVector(numOfElements)
  generatePCMatrixFromPV(priorityVector)
}
```

Fig. 4.7. The implementation of function *generatePCMatrix* which generates the PC matrix depending on matrix size

```
#' @title Disturbs the PC matrix
#' @description Disturbs the PC matrix in order to obtain inconsistency
#' @param matrix - PC matrix
#' @param scale - extend of disorders. This parametr is the upper limit
#' of the interval that is used to scale the elements. The lower limit
#' is defined as 1 / scale
#' @return disturbed PC matrix
disturbPCMatrix<- function(matrix, scale){
  dim = nrow(matrix)

  for(r in 1:dim-1)
    for(c in (r+1):dim)
      matrix[r,c] <- runif(1, min = 1/scale, max = scale) * matrix[r,c]

  diag(matrix) <- 1
  recreatePCMatrix(matrix)
}
```

Fig. 4.8. The implementation of function *disturbPCMatrix* which disturbs the PC matrix regarding inconsistency depending on given level of inconsistency

```r
#' @title Breaks PC Matrix
#' @description Breaks PC Matrix to obtain reciprocal, incomplete matrix
#' @param  matrix - PC matrix
#' @param grade - percentage of the value to be removed (not applicable to the diagonal)
#' @return incomplete PC matrix
breakPCMatrix<- function(matrix, grade){

  dim <- nrow(matrix)
  numOfEmptyElements <- grade*0.01*(dim*(dim-1))

  while(numOfEmptyElements > 1){
    rowToClear <- sample(1:dim, 1)
    colToClear <- sample(c(1:dim)[-rowToClear], 1)

    if(matrix[rowToClear, colToClear] != 0) {
      matrix[rowToClear, colToClear] <- matrix[colToClear, rowToClear] <- 0
      numOfEmptyElements <- numOfEmptyElements - 2
    }
  }

  matrix

}
```

Fig. 4.9. The implementation of function *breakPCMatrix* which breaks up the PC matrix regarding incompleteness depending on given degree of incompleteness

```r
#' @title Explore the incomplete PC matrix
#' @description Examines what is the relative error between the full matrix and indomplete matrix
#' @param methodName - a name of the method which is tested
#' @param scale - extend of disorders. This parametr is the upper limit of the interval that is used
#' to scale the elements. The lower limit is defined as 1 / scale
#' @param numOfElements - dimension of tested matrix
#' @param gradeOfIncomplete -  percentage of the value to be removed (not applicable to the diagonal)
#' @param numOfAttempts - number of test cases
#' @param alfa - a parameter for kulakowskiSzybowskiIa and kulakowskiSzybowskiIab method
#' @param beta - a parameter for kulakowskiSzybowskiIab method
#' @return average value of the relative error between the full matrix and indomplete matrix
exploreMatrix <- function(methodName, scale, numOfElements, gradeOfIncomplete, numOfAttempts, alfa=0, beta=0) {

  matrix <- generateDisturbedPCMatrix(numOfElements, scale)
  n <- ncol(matrix)
  if(alfa==0 && beta==0){
    realIdx <- methodName(matrix)
  } else {
    realIdx <- methodName(matrix, alfa, beta)
  }

  vectorOfIdsx <- integer(numOfAttempts)

  for( i in 1:numOfAttempts ) {
    brokenMatrix <- matrix(nrow = n, ncol = n, data = 0)
    numOfTriads <- length(generateTriads(brokenMatrix))/3

    while( (0 %in% (matrix %^% (n-1))) || numOfTriads==0){
      brokenMatrix <- breakPCMatrix(matrix, gradeOfIncomplete)
      numOfTriads <- length(generateTriads(brokenMatrix))/3
    }

    if(alfa==0 && beta==0){
      idx <- methodName(brokenMatrix)
    } else {
      idx <- methodName(brokenMatrix, alfa, beta)
    }

    vectorOfIdsx[i] <- abs(realIdx - idx)
  }

  incompleteIdx <- mean(vectorOfIdsx)

  abs(incompleteIdx/realIdx*100)
}
```

Fig. 4.10. The implementation of function *exploreMatrix* which tests given inconsistency index

```
#' @title Explore the incomplete PC matrixes for every method and scale <1.1, 1.2, ... , 4.0>
#' @description Examines what is the relative error between the full matrixes and indomplete matrixes.
#' @param numOfElements - dimension of tested matrix
#' @param gradeOfIncomplete -  percentage of the value to be removed (not applicable to the diagonal)
#' @param numOfAttempts - number of tested matrixes
#' @param numOfAttemptsForOneMatrix - number of test cases for each matrix
#' @param alfa - a parameter for kulakowskiSzybowskiIa method
#' @param beta - a parameter for kulakowskiSzybowskiIab method
#' @return average value of the relative error between the full matrix and indomplete matrix
test <- function(numOfElements, gradeOfIncomplete, numOfAttempts, numOfAttemptsForOneMatrix, alfa=0, beta=0){

  results <- matrix(nrow=31, ncol=16, data=0)
  counter <- 1

  for(i in seq(1.1, 4, 0.1)){
    print(counter)
    results[counter,] <- monteCarloOnTheSameMatrix(numOfElements, i, gradeOfIncomplete, numOfAttempts,
                                                   numOfAttemptsForOneMatrix, alfa, beta)
    counter <- counter+1
  }

  for(i in 1:16){
    results[31, i] = sum(results[,i])/30
  }

  results
}
```

Fig. 4.11. The implementation of function *test* which tests all indexes regarding given matrix size and the degree of incompleteness

Studies inconsistency indexes for incomplete matrixes

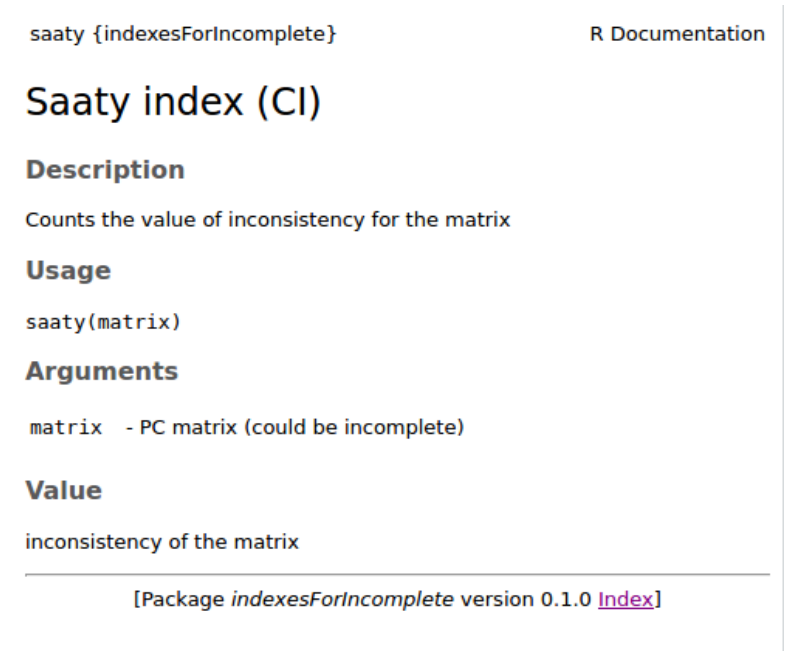
Documentation for package ‘indexesForIncomplete’ version 0.1.0

- DESCRIPTION file.

Help Pages

breakPCMatrix	Breaks PC Matrix
cavalloDapuzzo	Cavallo D'Apuzzo consistency index (CD)
chopV	Checks if the number is much greater than 0
countIndexesForTriads	Generates inconsistency indexes for each triad from the matrix
createHarkerMatrix	Create Harker matrix
disturbPCMatrix	Disturbs the PC matrix
exploreMatrix	Explore the incomplete PC matrix
exploreMatrixOnTheSameMatrix	Explore the incomplete PC matrixes
generateBrokenPCMatrix	Generate broken PC Matrix
generateDisturbedPCMatrix	Generates disturbed PC Matrix
generatePCMatrix	Generates PC Matrix on the basis of the size
generatePCMatrixFromPV	Generates PC Matrix on the basis of the priority vector
generatePriorityVector	Generates random priority vector
generateTriads	Generates triades from the matrix
geometric	Geometric consistency index (GCI)
geometricRank	Rank list given as geometric means
geometricRankForIncomplete	Rank list for the incomplete matrix given as geometric means
goldenWang	Golden and Wand consistency index (GW)
harmonic	Harmonic consistency index (HCI)
kazibudzkiCMLTI2	Kazibudzki matrix inconsistency (CMLTI2)
kazibudzkiLTI1	Kazibudzki matrix inconsistency (LTI1)
kazibudzkiLTI1ForTriad	kazibudzki (LTI1) innconsistency for one triad
kazibudzkiLTI2	Kazibudzki matrix inconsistency (LTI2)
kazibudzkiLTI2ForTriad	kazibudzki (LTI2) innconsistency for one triad
koczkodaj	Koczkodaj matrix inconsistency
koczkodajForTriad	Koczkodaj innconsistency for one triad

Fig. 4.12. The portion of the documentation: general view



saaty {indexesForIncomplete} R Documentation

Saaty index (CI)

Description

Counts the value of inconsistency for the matrix

Usage

```
saaty(matrix)
```

Arguments

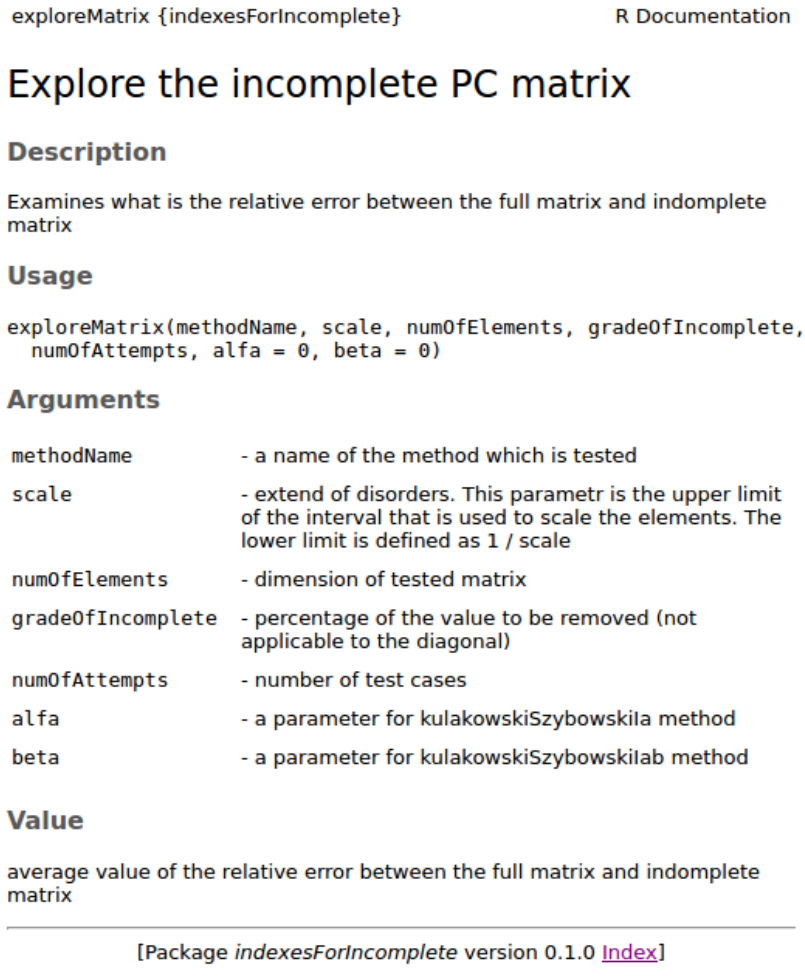
matrix	- PC matrix (could be incomplete)

Value

inconsistency of the matrix

[Package *indexesForIncomplete* version 0.1.0 Index]

Fig. 4.13. The portion of the documentation: function *saaty*



exploreMatrix {indexesForIncomplete} R Documentation

Explore the incomplete PC matrix

Description

Examines what is the relative error between the full matrix and indomplete matrix

Usage

```
exploreMatrix(methodName, scale, numOfElements, gradeOfIncomplete,
  numOfAttempts, alfa = 0, beta = 0)
```

Arguments

methodName	- a name of the method which is tested
scale	- extend of disorders. This parametr is the upper limit of the interval that is used to scale the elements. The lower limit is defined as 1 / scale
numOfElements	- dimension of tested matrix
gradeOfIncomplete	- percentage of the value to be removed (not applicable to the diagonal)
numOfAttempts	- number of test cases
alfa	- a parameter for kulakowskiSzybowskiIa method
beta	- a parameter for kulakowskiSzybowskiIab method

Value

average value of the relative error between the full matrix and indomplete matrix

[Package *indexesForIncomplete* version 0.1.0 Index]

Fig. 4.14. The portion of the documentation: function *exploreMatrix*

5. Results and discussion

5.1. Results

The tests have been performed using Operating System *Ubuntu 16.04* and IDE *RStudio*. Results are presented below.

5.1.1. Tests taking into account different matrix sizes

Table 5.1. Relative error of inconsistency indexes for incomplete matrices with constant degrees of incompleteness $g = 15\%$ and variable matrix size.

Index	n = 4	n = 7	n = 8	n = 10	n = 15	mean	rank
saaty	33,41	19,82	18,78	19,16	17,37	21,71	10
geometric	616,68	124,73	77,94	68,62	39,13	185,42	13
koczkodaj	**13,86**	**3,69**	**2,14**	**1,62**	**0,80**	**4,42**	**1**
kazibudzkiLTI1	24,80	10,21	6,62	4,97	2,73	9,87	6
kazibudzkiLTI2	42,31	17,93	11,88	9,03	5,03	17,24	8
kazibudzkiCMLTI2	35,40	17,07	13,26	11,20	6,81	16,75	7
pelaeLamata	44,65	19,90	13,46	10,36	5,84	18,84	9
kulakSzyb	20,34	7,68	4,88	3,63	1,96	7,70	5
kulakSzyb2	44,61	26,05	27,12	29,64	28,46	31,18	11
kulakSzybIa	16,47	5,18	3,09	2,27	1,16	5,63	3
kulakSzybIab	17,40	4,89	2,81	2,04	1,01	5,63	2
harmonic	9 573,02	1 577,49	1 127,33	1 066,35	866,00	2 842,04	15
goldenWang	115,92	54,37	43,90	43,16	36,26	58,72	12
saloHamalainen	381,57	205,06	176,11	160,06	136,55	211,87	14
cavalloD'Apuzzo	16,94	6,85	4,46	3,36	1,87	6,70	4
relativeError	1 792,64	226 313,60	746,21	100,87	20,42	45 794,75	16

5.1.2. Tests taking into account different degrees of incompleteness

5.1.3. Tests taking into account different levels of inconsistency

Among tested inconsistency indexes ones, which have given the best results, are presented below. The exhaustive results are placed in Appendix.

Table 5.2. Relative error of inconsistency indexes for incomplete matrices with varying degrees of incompleteness and constant matrix size $n = 8$.

Index	$g = 4\%$	$g = 7\%$	$g = 14\%$	$g = 25\%$	$g = 50\%$	mean	rank
saaty	4,71	9,40	18,78	32,89	65,56	26,27	10
geometric	23,60	48,44	86,61	135,68	207,99	100,46	13
koczkodaj	**0,48**	**0,99**	**2,17**	**4,52**	16,41	4,92	2
kazibudzkiLTI1	2,90	4,31	6,64	10,05	23,09	9,40	6
kazibudzkiLTI2	5,12	7,71	11,91	18,08	40,77	16,72	7
kazibudzkiCMLTI2	5,16	8,05	13,34	22,03	61,16	21,95	9
pelaeLamata	5,73	8,72	13,52	20,54	45,64	18,83	8
kulakSzyb	2,17	3,18	4,91	7,43	17,30	7,00	5
kulakSzyb2	5,90	12,22	27,12	56,71	202,27	60,84	12
kulakSzybIa	1,20	1,88	3,12	5,13	13,97	5,06	3
kulakSzybIab	1,00	1,65	2,84	4,74	**12,97**	**4,64**	**1**
harmonic	291,74	544,60	1 152,26	1 962,25	3 995,58	1 589,29	16
goldenWang	14,23	25,23	46,18	68,83	98,24	50,54	11
saloHamalainen	88,40	137,70	180,17	182,54	148,74	147,51	14
cavalloD'Apuzzo	1,95	2,91	4,46	6,81	16,11	6,45	4
relativeError	18,99	20,81	206,74	68,98	1 056,96	274,50	15

5.2. Discussion

Analyzing the above results, one can draw several conclusions.

5.2.1. Tests taking into account different matrix sizes

Along with a growth of the matrix size, examined relative error falls. Considered the best indexes, relative error amount to a dozen percent for small matrixes (size 4) so it is relatively big. However, it decreases quickly, when the size of PC matrix increases. For matrix size 7 exist indexes which give the results about $3 - 5\%$, what is a gratifying score. For PC matrixes with a considerable size (over 10 concepts), the relative error is just about $1 - 2\%$, what is an acceptable score.

Definitely, the best index came out *Koczkodaj index*. It wins in each test regardless of the matrix size. It gives really satisfy results for not very small matrices. The good results were obtained also through indexes *kulakowskiSzybowskiIab*, *kulakowskiSzybowskiIa* and *cavalloD'Apuzzo*. It seems that these indexes are reliable. Absolutely one should reject indexes *relativeError*, *harmonic*, *saloHamalainen*, *geometric*, *goldenWang* for incomplete matrices. Results obtained through these methods shows that score examined inconsistency for incomplete matrixes can depart significantly from a right values.

It is worth noticing that the presented results involve matrixes with the degree of incompleteness equals 15%. It means that 7 comparisons are missing for matrixes with size $n = 10$. The results will be different from presented in table 5.1. for others degrees of incompleteness. Impact the degree of incompleteness on tested relative error shows consecutive experiments.

5.2.2. Tests taking into account different degrees of incompleteness

Along with a growth of the degree of incompleteness, examined relative error grows. Considered the best indexes, relative error amount to just about $1 - 2\%$ percent for degree of incompleteness (size 4) so it is really satisfy. However, it increases slowly, when the degree of incompleteness falls. For the degree of incompleteness $g = 14\%$ exist indexes which give the results about $2 - 3\%$, what is a gratifying score too. The relative error is equal about a dozen percent for significant matrices ($g = 50\%$).

Definitely, the best indexes came out *Koczkodaj index* and *kulakowskiSzybowskiIab*. First one wins in most cases and gives really satisfy results. Second one improves along with a growth of the degree of incompleteness. It obtained a significant advantage over other methods in the last test, what caused that the average value turned out to be the lowest. The good results were obtained also through indexes *kulakowskiSzybowskiIa* and *cavalloD'Apuzzo*. It seems that these indexes are reliable. Absolutely one should reject indexes *harmonic*, *relativeError*, *saloHamalainen*, *geometric*. Results obtained through these methods shows that score examined inconsistency for incomplete matrixes can depart significantly from a right values.

It is worth noticing that the presented results involve matrixes with the size 8%. The results will be different from presented in table 5.2. for others matrix sizes. Impact the degree of incompleteness on tested relative error shows previous experiments.

5.2.3. Tests taking into account different levels of inconsistency

All tests for different matrix sizes and degrees of incompleteness has been performed taking into account varying levels of inconsistency. The scale of this incompleteness is proportional to the value d included in table 5.3. This table shows detailed results for which averages values was presented in table 5.1. It is worth taking a closer look at these scores. They point out that the quality of inconsistency indexes is determined also by the level of inconsistency.

The lowest values of the relative error has been obtained by *Koczkodaj index*, what was discussed before. One should notice and accent that it gives the best results only from particular moment which the level of inconsistency in matrix begins to grow. Before, so for small inconsistencies, the most satisfy index is *Cavallo and D'Apuzzo index*. This in interesting regularity which indicates that the choice of inconsistency index for incomplete matrixes should depend on the level of inconsistency.

It is worth noticing that this regularity repeats in each performed tests. For small level of inconsistency, regardless of the matrix size and the degree od incompleteness, *Cavallo and D'Apuzzo index* comes out to be the best. The results presented in Appendix confirms it.

5.2.4. General discussion

Certainly, the tests managed to show that the error increases with a growth of the level of incompleteness. At the same time, it decreases when the size of the matrix increases. However, the most important question was about which indexes cope well with incomplete matrices. The Koczkodaj index ([sub:Koczkodaj-index]) won in 9 out of 10 tests and its average error in both cases turn out

to be the lowest (below 5%). The next places are occupied by two indexes introduced by Kułakowski and Szybowski ([eq:Ia], [eq:Iab]) and Cavallo and D'Apuzzo index ([sub:Cavallo-and-D'Apuzzo]). It is worth noting that all of these indexes are based on triads.

A question about what makes the Koczkodaj index giving such good results and whether is it worth using, may arise. One should return to the definition of this index ([eq:K]) and notice that it is equal to the value of the most inconsistency triad. Therefore, if the level of incompleteness is low, there is a good chance that after deletion some values from the matrix and recalculating the index the value of it will not change at all. It will only change if the element included in the most inconsistent triad is removed. However, in many cases, the examination of the full matrix and th matrix after removing some values give exactly the same results (the error is 0%). This is the only index of this kind among those presented in the paper.

If one uses the Koczkodaj index, one may be worried that the removed comparison belonged to the most inconsistent triad. In such a case, it is difficult to predict what error will be contained in the index of the incomplete matrix. It seems that one should pay particular attention to the indexes proposed by Kułakowski and Szybowski. First of them ([eq:I1]) averages the inconsistencies of the triads. Therefore, it is safe and gives good results (in both tests it took the sixth place and achieved an error below 8%). From this perspective, another index suggested by the same authors ([eq:Ia]) turns out to be very interesting. In the tests it took the third place. It has the parameter α allowing to determine the effect of the greatest inconsistency of the triad (α), and the average $(1-\alpha)$. In the tests carried out, the parameter α was 0.4.

Table 5.3. Relative error of inconsistency indexes for incomplete matrices with varying degrees of inconsistency, constant matrix size $n = 8$ and constant level of incompleteness $g = 15\%$.

d	koczkodaj	kazibudzkiLTI1	kulakSzyb	kulakSzybIa	kulakSzybIab	cavalloD'Apuzzo
1.1	4.09	6.38	6.13	4.52	4.35	**0.49**
1.2	3.68	6.45	5.98	4.26	4.06	**0.96**
1.3	3.58	6.55	5.87	4.16	3.96	**1.37**
1.4	3.55	6.62	5.77	4.12	3.91	**1.80**
1.5	3.19	6.35	5.42	3.80	3.59	**2.02**
1.6	3.26	6.57	5.46	3.84	3.64	**2.39**
1.7	3.03	6.59	5.35	3.71	3.48	**2.81**
1.8	**2.47**	6.45	5.13	3.31	3.04	3.07
1.9	**2.47**	6.65	5.26	3.35	3.06	3.34
2.0	**2.20**	6.66	5.13	3.16	2.85	3.64
2.1	**2.38**	6.51	4.93	3.21	2.93	3.83
2.2	**2.16**	6.52	4.81	3.04	2.76	4.09
2.3	**2.14**	6.66	4.89	3.07	2.80	4.25
2.4	**2.00**	6.67	4.85	2.98	2.67	4.49
2.5	**1.96**	6.59	4.73	2.91	2.63	4.71
2.6	**1.73**	6.62	4.72	2.81	2.48	4.87
2.7	**1.94**	6.79	4.73	2.96	2.67	5.14
2.8	**1.75**	6.67	4.64	2.78	2.47	5.24
2.9	**1.56**	6.67	4.54	2.68	2.38	5.60
3.0	**1.82**	6.71	4.68	2.84	2.51	5.56
3.1	**1.47**	6.69	4.48	2.62	2.30	5.74
3.2	**1.46**	6.52	4.40	2.59	2.28	5.75
3.3	**1.48**	6.85	4.57	2.65	2.35	6.14
3.4	**1.38**	6.60	4.30	2.53	2.23	6.15
3.5	**1.17**	6.60	4.25	2.40	2.10	6.50
3.6	**1.25**	6.73	4.39	2.50	2.18	6.65
3.7	**1.18**	6.83	4.41	2.48	2.19	6.77
3.8	**1.24**	6.59	4.17	2.40	2.09	6.68
3.9	**1.28**	6.73	4.25	2.47	2.19	6.79
4.0	**1.11**	6.52	4.17	2.35	2.06	6.79
mean	**2.13**	6.61	4.88	3.08	2.81	4.45

6. Summary

The aim of the work - testing common inconsistency indexes for incomplete PC matrixes has been achieved. One has examined sixteen different methods for calculating inconsistency. The performed tests have consulted many factors which could have impact on results. It considered matrices with different size and the level of inconsistency. They was checked for varying degree of incompleteness (from 4% to 50%). All results has been gathered and presented in tables.

All tests has been developed in *R* language which is appropriate for numerical calculations and it has fulfilled the task. Functions, which are responsible for calculations, has been described and documented in details. They can compute inconsistency indexes for both full and incomplete matrices. If applicable, one can implement another indexes easily.

The tests has obtained expressly that some of existing indexes are up to calculating inconsistency for incomplete matrices after slight modifications. The ways of this modifications are presented in this work. Obviously there are many different possibilities to make changes in these indexes. Perhaps they can give even more favorable results. However, the purpose og this work was testing only existing indexes, therefore, one has decided to do not make many modifications. The most reliable results has been achieved by *Koczkodaj index*, *KulakowskiSzybowskiIa*, *KulakowskiSzybowskiIab* and *Cavallo DAapuzzo*. Selection one from these indexes should be done based on matrix parameters which has been described in this work and which has been presented in results.

Very interesting came out indexes proposed by *Kułakowski* and *Szybowski* containing parameters α and β. Only one assignment these arguments was checked in the performed tests. It seems that further experiments of these indexes should be dedicated to choice values of parameters α and β in a way that will give even better results. How one has been mentioned in this work, these parameters allow to choice between a more reliable result and sure that the relative error will be known.

The interesting line of enquiry can be using PC method for incomplete matrixes when somehow a part of comparisons is missing. Such a works already appear. However, this paper shows that also counting inconsistency for such matrices is possible. The PC method expands for many years, still discover new potential and tests new places where can be applied.