Advances in Science and Technology
Research Journal

Advances in Science and Technology Research Journal 2023, 17(1), 222–229
https://doi.org/10.12913/22998624/158019
ISSN 2299-8624, License CC-BY 4.0

Received: 2022.10.11
Accepted: 2023.01.17
Published: 2023.02.01

Using Inconsistency Reduction Algorithms in Comparison Matrices to Improve the Performance of Generating Random Comparison Matrices with a Given Inconsistency Coefficient Range

Paweł Kuraś[1*], Alicja Gerka[1]

[1] Department of Complex Systems, The Faculty of Electrical and Computer Engineering, Rzeszow University of Technology, ul. MC Skłodowskiej 8, 35-036 Rzeszów, Poland

* Corresponding author's e-mail: p.kuras@prz.edu.pl

ABSTRACT

The aim of this paper is to present a new method for generating random pairwise comparison matrices with a given inconsistency ratio (CR) interval using inconsistency reduction algorithms. Pairwise comparison (PC) is a popular technique for multi-criteria decision-making, its purpose is to assign weights to the compared entities, thus ranking them from best to worst. The presented method combines the traditional random generation of comparison matrices supported by inconsistency reduction algorithms: the "Xu and Wei" algorithm and the "Szybowski" algorithm. This paper presents research that shows an increase in performance when generating such matrices relative to the standard random comparison matrix generation procedure using the "Szybowski" algorithm. The other algorithms also improve the process, but to a lesser extent, making the "Szybowski" supporting algorithm the preferred solution for the new process. As a result of the research, a free online tool "PC MATRICES GENERATOR" has also been made available to efficiently generate a large number of comparison matrices with a given CR factor range, any matrix size, and any number of matrices, enabling much more efficient and less time-consuming research in many fields that use comparison matrices, as the analytic hierarchy/network process (AHP/ANP), ELECTREE, PAPRIKA, PROMETHE, VIKOR or the Best-Worst method (BWM).

Keywords: algorithm, pairwise comparison, inconsistency ratio, generate, new online tool

INTRODUCTION

Recent decades have seen the development of many new multi-criteria decision-making methods, mainly through in-depth research on them. The main ones are techniques such as PROMETHEE, PAPRIKA, VIKOR, ELECTREE, and especially BWM (Best-Worst Method) and AHP (Analytic Hierarchy Process, see e.g. Mazurek [1], Liang et al. [2], Saaty [3,4], Brans et al. [17,18], Hansen et al. [25], Opricovic et al. [26] and Alkihairi et al. [27].

All these methods use pairwise comparison matrices, which are used also in such many other fields, see e.g. Koczkodaj et al. [20,21,22,23], Cavallo et al [24]. Since the authors of this paper, have repeatedly participated in research on methods that use them, they have also confronted the problem of efficiently generating random comparison matrices – necessary, for example, to perform Monte Carlo simulations for such methods that require large amounts of random data, see e.g. Caflisch [5]. This problem was particularly evident in the case of matrices, the so-called "large" (from 6x6 to 10x10) and when one wanted to simultaneously obtain a low CR – consistency range (especially below the value of 0.1). Then the generation times for several thousand random matrices with the desired parameters could reach even several days. This was a big obstacle for performing efficient research, as the need to examine large amounts of data with a specific consistency interval may be needed on such fronts as the study of constructive consistent approximations in pairwise comparisons, see Smarzewski et. al [19]. Therefore, using inconsistency reduction algorithms, a method has been developed that will significantly streamline this process and help researchers save time and hardware resources.

The purpose of this paper is to present and compare the results between the normal and the improved method of generating random comparison matrices with the desired parameters, as well as to choose the best method of inconsistency reduction that supports the algorithm for generating random matrices and to present an on-line tool "PC MATRICES GENERATOR" that implements the presented algorithm and is made available for free to researchers from all over the world.

The paper has the following structure: in the next subsection of the introduction, the problem of pairwise comparison is described mathematically. In the second section, the normal method of generating the comparison matrix with a given CR will be described, the improved method of generating the comparison matrix will be described, and the inconsistency reduction algorithms used to improve it will be described. The third chapter presents and discusses the results of the algorithms in different variants, and the fourth chapter briefly presents the developed software. Chapter 5 summarize the paper.

Pairwise comparisons

We have a data set X – this is the set of n entities to compare, and let a_{ij} denote the preferences of the *i*-th entity over the j-th entity. In addition, we establish that $a_{ij} > 0; \forall i, j \in \{1, 2, ..., n\}$. We can call pairwise comparisons reciprocal only if this property (1) is satisfied.

$$a_{ij} = \frac{1}{a_{ji}}, \forall i, j \in \{1,2, ..., n\} \tag{1}$$

The property in equation (1) is required for multiplicative pairwise comparisons. All pairwise comparisons can be set into a square $n \times n$ matrix called a pairwise comparison matrix – PCM (2).

$$\begin{pmatrix} 1 & a_{12} & \dots & a_{1n} \\ a_{21} & 1 & \dots & a_{2n} \\ \dots & \dots & 1 & \dots \\ a_{n1} & a_{n2} & \dots & 1 \end{pmatrix} \tag{2}$$

Pairwise comparisons are called consistent, if the property (3) is satisfied.

$$a_{ij} \cdot a_{jk} = a_{ik}; \forall i, j, k \tag{3}$$

Then and only if the matrix A is considered consistent, the vector of weights (priority vector) $w = (w_1, ..., w_n)$ of all the components fits into the following equation (4).

$$a_{ij} = \frac{w_i}{w_j}, \forall i, j \tag{4}$$

The priority vector w can be calculated using Saaty's eigenvalue method – EVM (5), developed and described by Saaty [3], where λ_{max} is the largest eigenvalue of the matrix.

$$Aw = \lambda_{max} \tag{5}$$

THE PROPOSED ALGORITHM

Inconsistency reduction

The consistency index *CI* (6) and the consistency ratio *CR* (7), were presented by Saaty [3,4], where n in (6) is the size of a pairwise comparison matrix, and RI in (7) is the random consistency index, determined by Saaty (1980).

$$CI = \frac{\lambda_{max}}{n - 1} \tag{6}$$

$$CR = \frac{CI}{RI} \tag{7}$$

Usually in calculations, we aim to achieve a CR value below 0.10 – then the matrix is considered consistent according to Saaty's postulates [3]. Over the last years, many algorithms have been developed to reduce such inconsistency and have been presented in many research papers, see eg. Xu and Wei [6], Cao et al. [7], Ergu et al. [8], Benítez et al. [9, 10], Kulakowski et al. [11], Szybowski [12], Abel et al. [13], or Mazurek et al. [14].

In the developed matrix generation method, the algorithms of Szybowski [12] and the algorithm of Xu and Wei [6] will be used, since their effectiveness has been demonstrated in a recent compilation of different techniques of this type, see Mazurek et. al [15], which also describes the performance of these algorithms in detail.

Normal method of generating pairwise comparison matrices with a given CR

The block diagram of the normal method is shown in Figure 1a – we start by determining the range of the CR coefficient, then we generate a random pairwise comparison matrix of the given size (generating first the whole matrix full of ones, completing it with random values from 1 to 9 on one diagonal, and on the other – the inverses of these values). Then we calculate the value of CR and if it is within the given range – we add it to the list of generated matrices, if not – we generate another pairwise comparison matrices until the wanted result is obtained.

Improved method of generating pairwise comparison matrices with a given CR

The block diagram of the improved method is shown in Figure 1b – in relation to the normal method we added the point related to the reduction of inconsistency, but only in the case when the randomly generated matrix is larger than a given range – at the moment there are no algorithms increasing inconsistency of the comparison matrix, so we have to reject such cases and generate another random matrix. The matrix is reduced to the specified range by the algorithm and then it is already within the specified range, so it can go directly to the output of the algorithm.

RESULTS AND DISCUSSION

Methodology

The measurements were implemented using the Google Colab environment, where both methods were implemented using the Python 3.10.4 environment using – in addition to the standard libraries bundled with it – the NumPy 1.19.5, SciPy 1.6.0 and SymPy 1.7.1 libraries, while the measurements were implemented using the time library of the Python language. The times (in seconds) shown in the tables are the result of averaging 10 independent measurements performed on the same environment, with each cleaning of the cache memory and RAM contents of the environment.

Results for the normal method of pairwise comparison matrices random generation

The measurement results are presented in Table 1 and illustrated in the graph (Fig. 2). It can

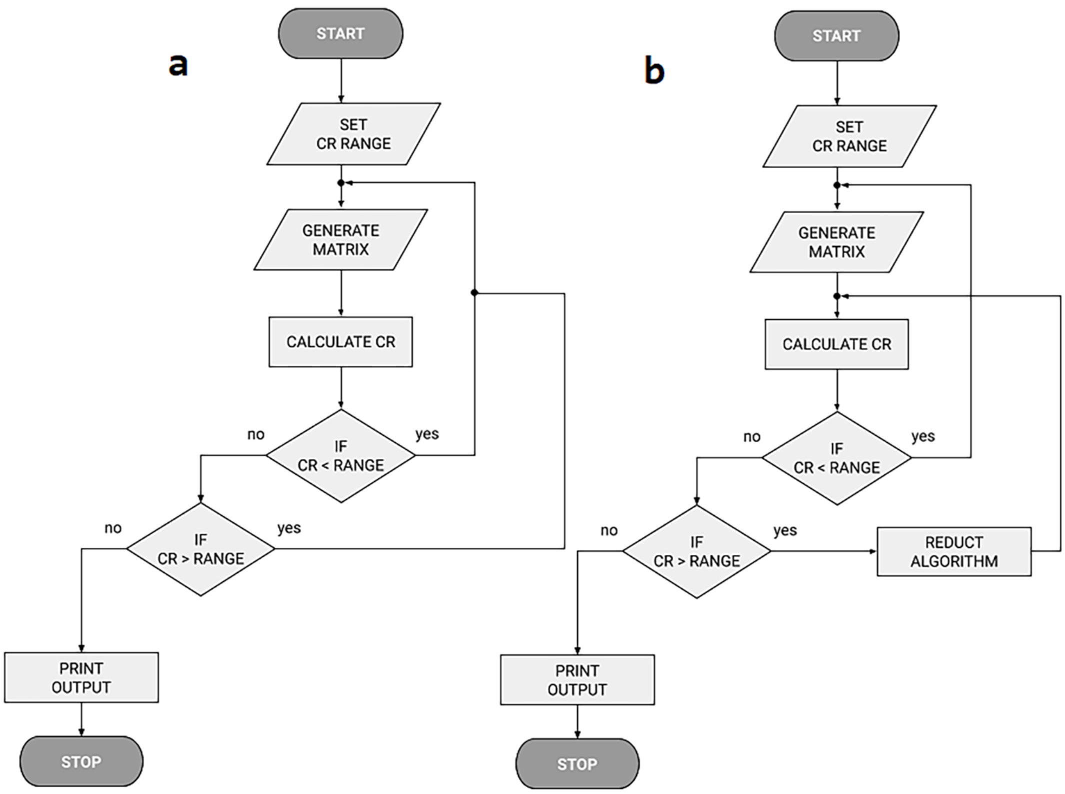

Figure 1. The block diagram of the regular (a) and improved (b) algorithm for generating random comparison matrices with given CR

Table 1. Time of generation 10,000 random matrices in different shapes using regular method in different CR ranges

[s]	3×3	4×4	5×5	6×6	7×7	8×8	9×9	10×10
<0.9 ; 1)	272	204	146	132	151	156	157	154
<0.8 ; 0.9)	200	195	154	148	174	205	227	274
<0.7 ; 0.8)	211	182	167	180	234	337	456	688
<0.6 ; 0.7)	113	162	180	224	366	649	1198	3809
<0.5 ; 0.6)	101	139	190	296	643	1479	4109	11641
<0.4 ; 0.5)	73	117	200	415	1263	4292	14739	28617
<0.3 ; 0.4)	68	97	212	617	2954	10941	29110	60009
<0.2 ; 0.3)	50	95	273	1279	10632	24092	49217	112277
<0.1 ; 0.2)	49	112	584	5870	25837	47059	73377	193067
<0.0 ; 0.1)	25	246	4632	18245	53673	83849	97602	311210

be clearly seen that while for small matrix sizes the times are acceptable, as the matrix size increases and CR decreases, the times reach more than 3 days for the highest result.

Results for the improved method of pairwise comparison matrices random generation

For the improved method, separate measurements were performed for both the Xu and Wei [6] algorithm and the Szybowski [12] algorithm. In both cases (Table 2, Table 3) a clear improvement and reduction in the order of magnitude of time for the generated matrices can be seen, but the clear winner was the Szybowski algorithm, as can be seen by considering both the comparisons of the "regular" algorithm with Xu and Wei (Fig. 3) and also with Szybowski (Fig. 4). However, it was not the "winner" of each compartment, but it achieved the best results in most cases – this is shown in Table 4.

PC MATRICES GENERATOR – IMPLEMENTATION OF PROPOSED NEW METHOD

An online service to generate random comparison matrices with a given CR coefficient has been launched at the URL found in the references of this article [16]. Its interface is shown in Figure 5, and it was developed using the Flask framework – serving Python scripts to a web application environment. It uses the same libraries as the environment used to test the matrix generation times, while the user side uses standard technologies – HTML, CSS and JavaScript.

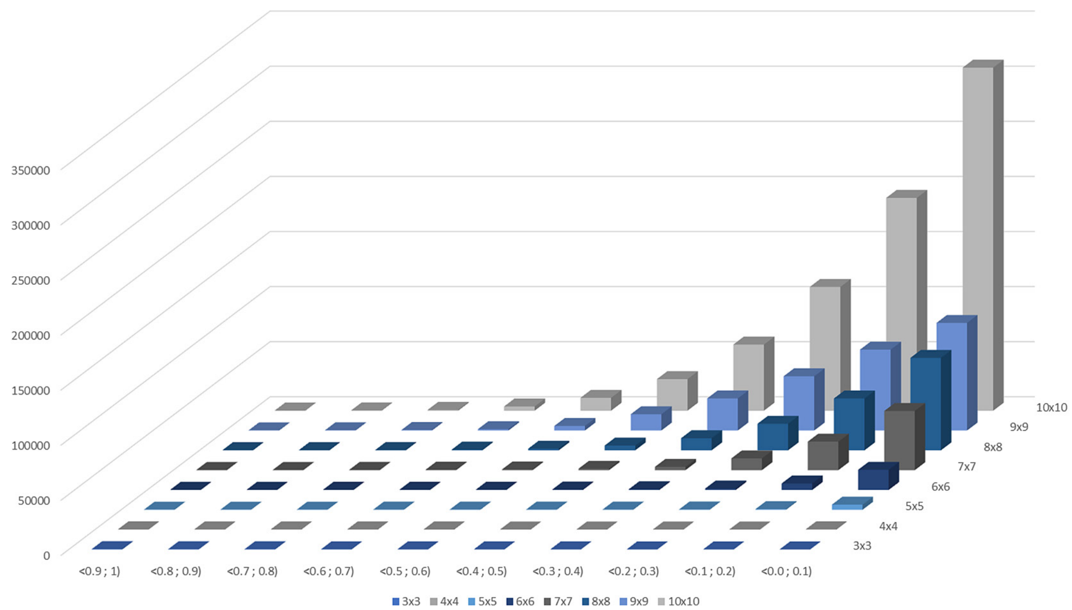

Figure 2. Time of generation 10,000 random matrices in different shapes using a regular method in different CR ranges

Table 2. Time of generation 10,000 random matrices in different shapes using an improved algorithm with Xu and Wei reducing method in different CR ranges

[s]	3×3	4×4	5×5	6×6	7×7	8×8	9×9	10×10
<0.9 ; 1)	148	132	129	133	104	105	117	133
<0.8 ; 0.9)	149	147	147	156	118	132	150	170
<0.7 ; 0.8)	158	161	170	191	150	179	201	257
<0.6 ; 0.7)	159	179	199	239	197	247	307	346
<0.5 ; 0.6)	162	197	237	299	263	338	423	533
<0.4 ; 0.5)	168	220	286	378	344	453	575	720
<0.3 ; 0.4)	172	248	347	489	461	607	774	951
<0.2 ; 0.3)	191	289	436	642	612	808	1049	1257
<0.1 ; 0.2)	219	365	593	872	834	1093	1409	1729
<0.0 ; 0.1)	259	533	876	1283	1207	1620	2071	2940

Table 3. Time of generation 10,000 random matrices in different shapes using an improved algorithm with Szybowski reducing method in different CR ranges

[s]	3×3	4×4	5×5	6×6	7×7	8×8	9×9	10×10
<0.9 ; 1)	45	45	61	76	99	126	150	175
<0.8 ; 0.9)	42	43	58	75	99	134	166	213
<0.7 ; 0.8)	43	42	57	76	103	148	216	273
<0.6 ; 0.7)	39	40	57	81	114	171	246	379
<0.5 ; 0.6)	37	40	57	84	127	204	298	434
<0.4 ; 0.5)	36	40	60	93	151	238	355	529
<0.3 ; 0.4)	33	40	66	105	172	289	444	655
<0.2 ; 0.3)	33	42	73	126	213	357	559	831
<0.1 ; 0.2)	34	46	90	159	279	469	733	1093
<0.0 ; 0.1)	34	58	121	222	394	673	1032	1560

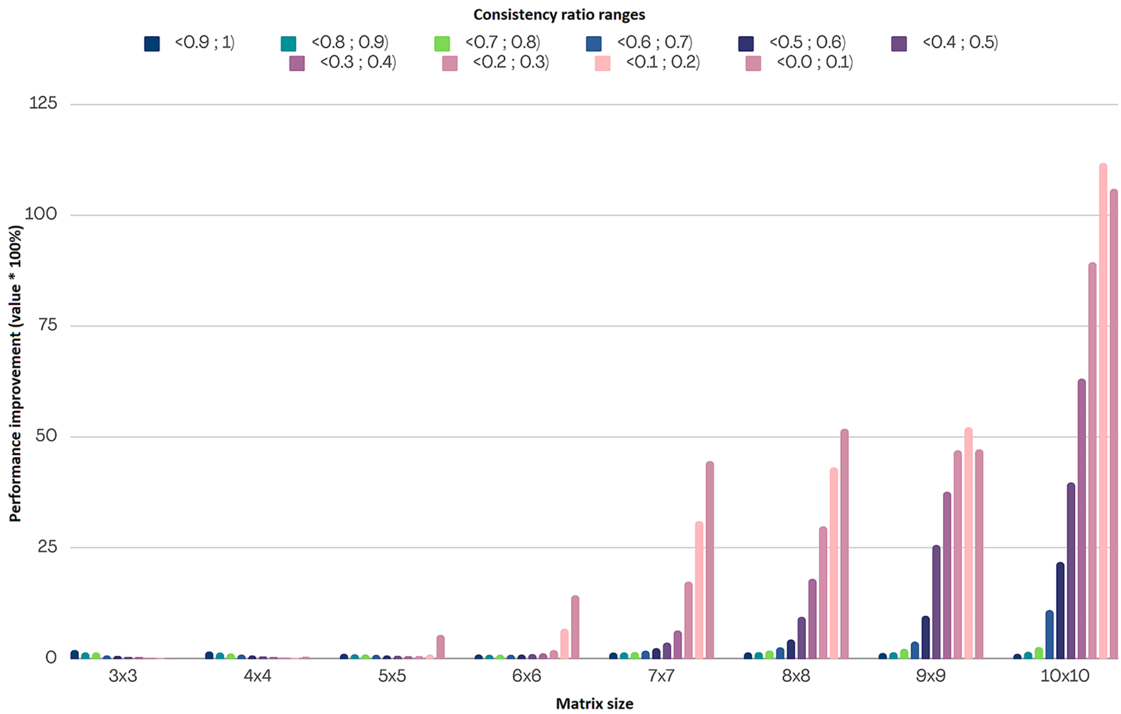

Figure 3. Comparing the performance of the improved "Xu and Wei" algorithm with the regular algorithm - a value in the graph indicates the value of percentage performance improvement (value * 100 %)

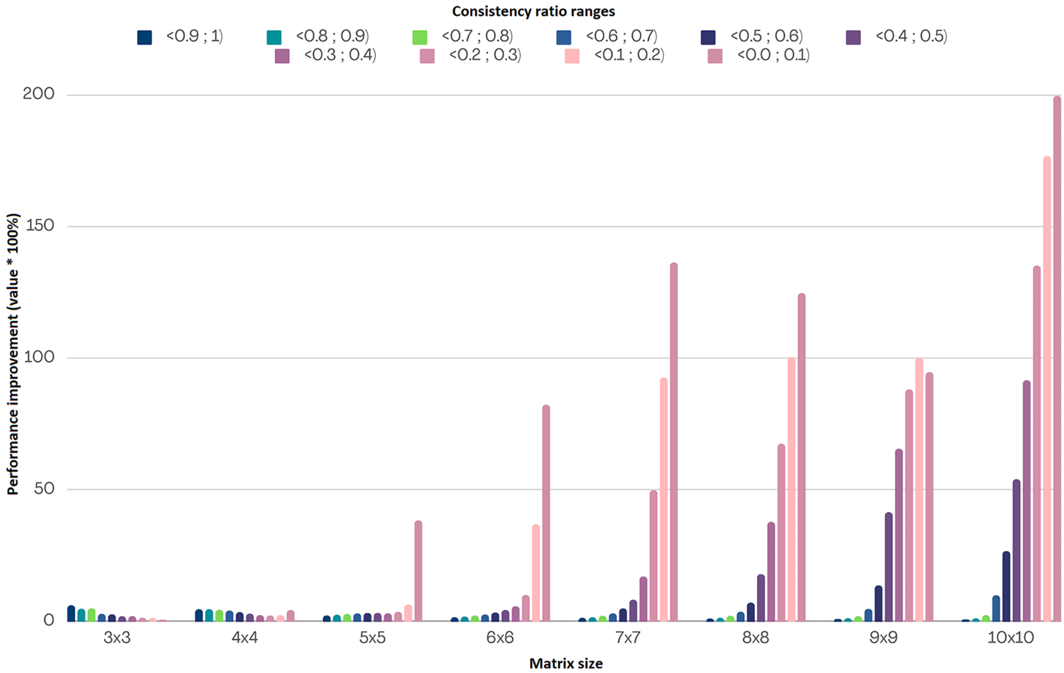

Figure 4. Comparing the performance of the improved "Szybowski" algorithm with the regular algorithm - a value in the graph indicates the value of percentage performance improvement (value * 100 %)

Table 4. Best method for each matrix size and CR range used in mixed (recommended) option in tool provided by authors of the paper

[s]	3×3	4×4	5×5	6×6	7×7	8×8	9×9	10×10
<0.9 ; 1)	Sz.	Sz.	Sz.	Sz.	Sz.	Xu.	Xu.	Xu.
<0.8 ; 0.9)	Sz.	Sz.	Sz.	Sz.	Sz.	Xu.	Xu.	Xu.
<0.7 ; 0.8)	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Xu.	Xu.
<0.6 ; 0.7)	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Xu.
<0.5 ; 0.6)	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.
<0.4 ; 0.5)	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.
<0.3 ; 0.4)	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.
<0.2 ; 0.3)	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.
<0.1 ; 0.2)	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.
<0.0 ; 0.1)	Normal	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.	Sz.

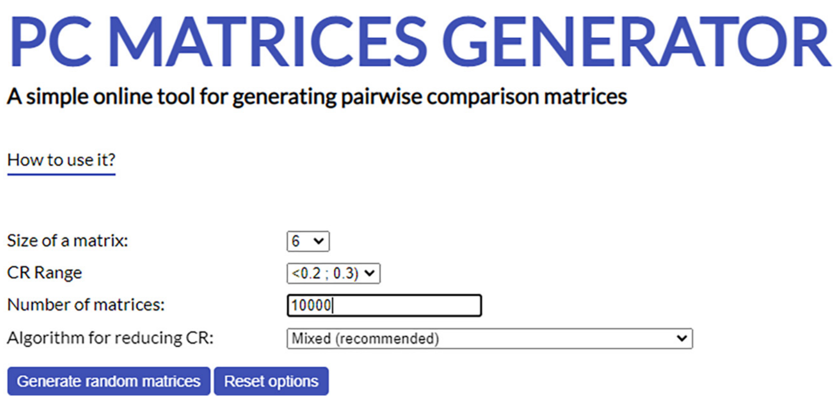

Figure 5. The interface of the "PC MATRICES GENERATOR" application

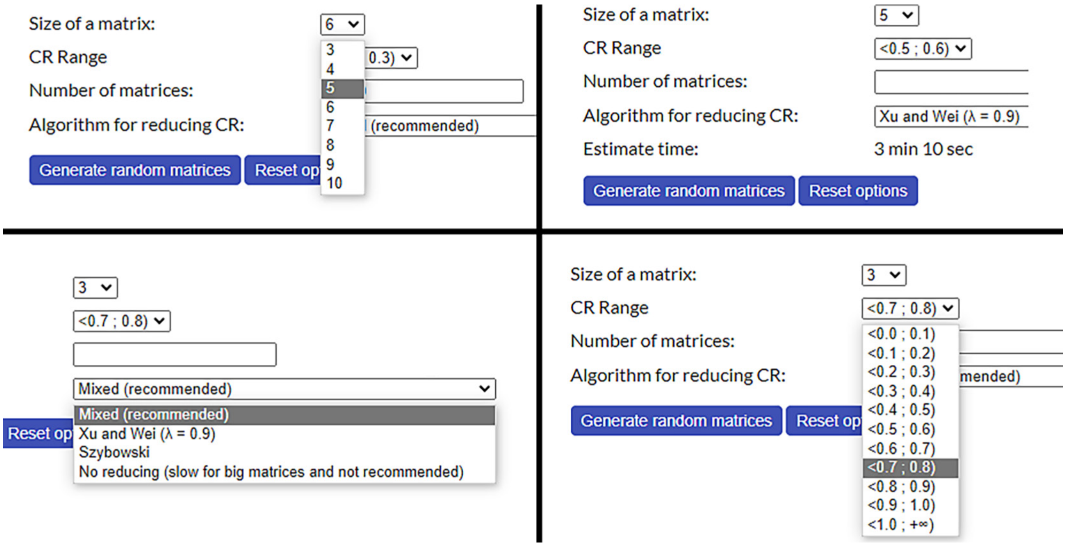

Figure 6. Available options in "PC MATRICES GENERATOR" application

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	**A11**	**A12**	**A13**	**A14**	**A21**	**A22**	**A23**	**A24**	**A31**	**A32**	**A33**	**A34**	**A41**	**A42**	**A43**	**A44**	**W1**	**W2**	**W3**	**W4**	**CR**
2	1.0	0.1667	2.0	1.0	6.0	1.0	9.0	6.0	0.5	0.1111	1.0	0.5	1.0	0.1667	2.0	1.0	0.1229	0.6879	0.0662	0.1229	0.0039
3	1.0	0.125	0.1667	0.5	8.0	1.0	1.0	3.0	6.0	1.0	1.0	3.0	2.0	0.3333	0.3333	1.0	0.0606	0.4199	0.3897	0.1299	0.0039
4	1.0	6.0	6.0	1.0	0.1667	1.0	1.0	0.125	0.1667	1.0	1.0	0.125	1.0	8.0	8.0	1.0	0.4059	0.0628	0.0628	0.4685	0.0039
5	1.0	1.0	9.0	7.0	1.0	1.0	7.0	6.0	0.1111	0.1429	1.0	1.0	0.1429	0.1667	1.0	1.0	0.4613	0.4163	0.0581	0.0642	0.0041
6	1.0	0.1429	1.0	0.1429	7.0	1.0	5.0	1.0	1.0	0.2	1.0	0.1429	7.0	1.0	7.0	1.0	0.0643	0.4152	0.0702	0.4502	0.0054
7	1.0	8.0	3.0	3.0	0.125	1.0	0.25	0.3333	0.3333	4.0	1.0	1.0	0.3333	3.0	1.0	1.0	0.5488	0.0603	0.203	0.1879	0.0062
8	1.0	0.125	1.0	0.25	8.0	1.0	8.0	3.0	1.0	0.125	1.0	0.3333	4.0	0.3333	3.0	1.0	0.0701	0.6191	0.0749	0.2359	0.0062
9	1.0	2.0	2.0	9.0	0.5	1.0	1.0	4.0	0.5	1.0	1.0	6.0	0.1111	0.25	0.1667	1.0	0.4687	0.2281	0.2531	0.0502	0.0062
10	1.0	1.0	1.0	0.5	1.0	1.0	1.0	0.5	1.0	1.0	1.0	0.3333	2.0	2.0	3.0	1.0	0.1948	0.1948	0.1768	0.4336	0.0078
11	1.0	0.5	0.3333	4.0	2.0	1.0	1.0	7.0	3.0	1.0	1.0	8.0	0.25	0.1429	0.125	1.0	0.1704	0.3633	0.4177	0.0485	0.0086
12	1.0	0.2	0.1429	1.0	5.0	1.0	1.0	8.0	7.0	1.0	1.0	9.0	1.0	0.125	0.1111	1.0	0.0678	0.414	0.4617	0.0565	0.0097
13	1.0	0.5	0.2	1.0	2.0	1.0	0.3333	3.0	5.0	3.0	1.0	8.0	1.0	0.3333	0.125	1.0	0.1024	0.2153	0.6	0.0823	0.0103
14	1.0	5.0	1.0	9.0	0.2	1.0	0.3333	2.0	1.0	3.0	1.0	6.0	0.1111	0.5	0.1667	1.0	0.4657	0.1082	0.3708	0.0554	0.0104
15	1.0	0.1429	1.0	0.3333	7.0	1.0	7.0	3.0	1.0	0.1429	1.0	0.25	3.0	0.3333	4.0	1.0	0.08	0.6006	0.075	0.2445	0.0104
16	1.0	0.3333	1.0	0.1429	3.0	1.0	3.0	0.25	1.0	0.3333	1.0	0.1111	7.0	4.0	9.0	1.0	0.0763	0.2001	0.0712	0.6524	0.0104
17	1.0	0.5	0.1429	0.125	2.0	1.0	0.2	0.3333	7.0	5.0	1.0	1.0	8.0	3.0	1.0	1.0	0.055	0.109	0.4382	0.3978	0.0116
18	1.0	1.0	4.0	6.0	1.0	1.0	3.0	4.0	0.25	0.3333	1.0	1.0	0.1667	0.25	1.0	1.0	0.4377	0.3671	0.1059	0.0893	0.0117
19	1.0	6.0	4.0	0.5	0.1667	1.0	1.0	0.125	0.25	1.0	1.0	0.1667	2.0	8.0	6.0	1.0	0.3202	0.0653	0.0775	0.537	0.0117
20	1.0	0.3333	0.1111	0.1111	3.0	1.0	0.2	0.3333	9.0	5.0	1.0	1.0	9.0	3.0	1.0	1.0	0.0435	0.1157	0.4491	0.3917	0.0124
21	1.0	0.5	0.1111	0.1111	2.0	1.0	0.25	0.1429	9.0	4.0	1.0	1.0	9.0	7.0	1.0	1.0	0.0459	0.0854	0.4026	0.4661	0.0124
22	1.0	7.0	9.0	1.0	0.1429	1.0	2.0	0.25	0.1111	0.5	1.0	0.125	1.0	4.0	8.0	1.0	0.471	0.0862	0.0478	0.395	0.0124
23	1.0	0.1111	0.1429	0.5	9.0	1.0	2.0	8.0	7.0	0.5	1.0	3.0	2.0	0.125	0.3333	1.0	0.0491	0.5696	0.2934	0.0879	0.015
24	1.0	0.1111	0.2	0.1667	9.0	1.0	2.0	1.0	5.0	0.5	1.0	1.0	6.0	1.0	1.0	1.0	0.0475	0.4018	0.2446	0.3061	0.0164
25	1.0	2.0	9.0	2.0	0.5	1.0	4.0	0.5	0.1111	0.25	1.0	0.1429	0.5	2.0	7.0	1.0	0.461	0.187	0.0465	0.3055	0.0168
26	1.0	0.125	0.3333	1.0	8.0	1.0	2.0	8.0	3.0	0.5	1.0	6.0	1.0	0.125	0.1667	1.0	0.0766	0.564	0.2949	0.0644	0.0173
27	1.0	8.0	2.0	4.0	0.125	1.0	0.25	1.0	0.5	4.0	1.0	3.0	0.25	1.0	0.3333	1.0	0.5301	0.0788	0.29	0.1011	0.0173
28	1.0	2.0	9.0	0.5	0.5	1.0	3.0	0.25	0.1111	0.3333	1.0	0.1111	2.0	4.0	9.0	1.0	0.3061	0.1365	0.0448	0.5126	0.0173
29	1.0	0.125	1.0	0.1667	8.0	1.0	6.0	2.0	1.0	0.1667	1.0	0.3333	6.0	0.5	3.0	1.0	0.0669	0.5433	0.085	0.3047	0.0173
30	1.0	0.1667	2.0	0.25	6.0	1.0	8.0	2.0	0.5	0.125	1.0	0.125	4.0	0.5	8.0	1.0	0.0891	0.5264	0.05	0.3346	0.0173
31	1.0	2.0	6.0	2.0	0.5	1.0	3.0	0.5	0.1667	0.3333	1.0	0.25	0.5	2.0	4.0	1.0	0.4513	0.1898	0.0692	0.2897	0.0173
32	1.0	4.0	0.3333	0.5	0.25	1.0	0.1667	0.1667	3.0	6.0	1.0	1.0	2.0	6.0	1.0	1.0	0.1777	0.0565	0.4048	0.3609	0.0173
33	1.0	0.5	6.0	2.0	2.0	1.0	8.0	4.0	0.1667	0.125	1.0	0.25	0.5	0.25	4.0	1.0	0.2821	0.5156	0.0492	0.1532	0.0173
34	1.0	6.0	1.0	0.5	0.1667	1.0	0.25	0.1667	1.0	4.0	1.0	0.5	2.0	6.0	2.0	1.0	0.2642	0.058	0.2355	0.4423	0.0173
35	1.0	0.5	0.5	0.3333	2.0	1.0	1.0	1.0	2.0	1.0	1.0	0.5	3.0	1.0	2.0	1.0	0.1238	0.2778	0.2326	0.3659	0.0173
36	1.0	0.5	6.0	2.0	2.0	1.0	6.0	3.0	0.1667	0.1667	1.0	0.5	0.5	0.3333	2.0	1.0	0.3073	0.4787	0.0671	0.1469	0.0173
37	1.0	6.0	6.0	4.0	0.1667	1.0	0.5	0.5	0.1667	2.0	1.0	1.0	0.25	2.0	1.0	1.0	0.63	0.0818	0.1376	0.1506	0.0173

Figure 7. The sample output of the application "PC Matrices Generator" – CSV file with random matrices in the given CR range

The user can specify the size of the matrix, set the desired CR interval and define the number of matrices, and select the desired algorithm (Fig. 6). The recommended option – mixed – was derived from Tab. 4 from this article and it uses different algorithms for maximum performance. After setting these values, the program will return the expected matrix generation time and we can use the "Generate random matrices" button to start the generation script.

When finished, our viewer will start downloading a CSV format file containing the ready-to-use matrices, along with their priority vectors and CR values (Fig. 7)

CONCLUSIONS

The purpose of this paper was to present a new method for generating random comparison matrices using inconsistency reduction algorithms and to demonstrate the significant performance gains associated with their use. An online tool implementing these methods in practice was also presented. We believe that our methods and tools will help to speed up the work of researchers working in the field of comparison matrices.

REFERENCES

1. Mazurek J. A New Approach to a Derivation of a Priority Vector from an Interval Comparison Matrix in a Group AHP Framework. In: International Conference on Intelligent Decision Technologies. Springer, Cham, 2016: 193-201.
2. Liang F., Brunelli M., Rezaei J. Consistency issues in the best worst method: Measurements and thresholds. Omega 2020; 96: 102175.
3. Saaty T.L. A scaling method for priorities in hierarchical structures. Journal of mathematical psychology 1977; 15(3): 234–281.
4. Saaty T.L. The analytic hierarchy process McGraw-Hill. New York 1980; 324.
5. Caflisch R.E. Monte carlo and quasi-monte carlo methods. Acta numerica 1998; 7: 1–49.
6. Zeschui X., Cuiping W. A consistency improving method in the analytic hierarchy process. European journal of operational research 1999; 116(2): 443–449.
7. Cao D., Leung L. C., Law J. S. Modifying inconsistent comparison matrix in analytic hierarchy process: A heuristic approach. Decision Support Systems 2008; 44(4): 944–953.
8. Ergu D., Kou G., Peng Y., Shi Y. A simple method to improve the consistency ratio of the pair-wise comparison matrix in ANP. European Journal of Operational Research 2021; 213(1): 246–259.
9. Benítez J., Delgado-Galván X., Gutiérrez J.A., Izquierdo J. Balancing consistency and expert judgment in AHP. Mathematical and Computer Modelling 2011; 54.7–8: 1785–1790.
10. Benítez J., Delgado-Galván X., Izquierdo J., Pérez-García R. Improving consistency in AHP decision-making processes. Applied Mathematics and Computation 2012; 219.5: 2432–2441.
11. Kułakowski K., Juszczyk R., Ernst S. A concurrent inconsistency reduction algorithm for the pairwise comparisons method. In: International Conference on Artificial Intelligence and Soft Computing. Springer, Cham 2015: 214–222.
12. Szybowski J. The improvement of data in pairwise comparison matrices. Procedia Computer Science 2018; 126: 1006–1013.
13. Abel E., Mikhailov L., Keane J. Inconsistency reduction in decision making via multi-objective optimisation. European Journal of Operational Research 2018; 267.1: 212–226.
14. Mazurek J., Perzina R., Strzałka D., Kowal B. A new step-by-step (SBS) algorithm for inconsistency reduction in pairwise comparisons. IEEE Access 2020; 8: 135821–135828.
15. Mazurek J., Perzina R., Strzałka D., Kowal B., Kuraś P. A numerical comparison of iterative algorithms for inconsistency reduction in pairwise comparisons. IEEE Access 2021; 9: 62553–62561.
16. https://reduce.prz.edu.pl/pc_matrices_generator/
17. Brans J., Vincke P., Mareschal B. How to select and how to rank projects: The PROMETHEE method. European Journal of Operational Research, 1986; 24.2: 228–238.
18. Brans, J.P., De Smet, Y. PROMETHEE methods. In Multiple criteria decision analysis. Springer, New York, NY 2016; 187–219.
19. Smarzewski R., Kozera R. Constructive Consistent Approximations in Pairwise Comparisons. Advances in Science and Technology Research Journal 2022; 16.4: 243–255.
20. Koczkodaj W.W., Orłowski M. An orthogonal basis for computing a consistent approximation to a pairwise comparisons matrix. Computers and Mathematics with Applications 1997; 34: 41–47.
21. Koczkodaj W.W., Szarek S.J. On distance-based consistency reduction algorithms for pairwise comparisons. Logic Journal of the IGPL 2010; 18: 859–869.
22. Koczkodaj W.W., Szwarc R. On axiomatization of inconsistency indicators for pairwise comparisons. Fundamenta Informaticae 2014; 132: 485–500.
23. Koczkodaj W.W., Smarzewski R., Szybowski J. On orthogonal projections on the space of consistent pairwise comparisons matrices. Fundamenta Informaticae 2020; 172: 379–397.
24. Cavallo B., Brunelli M. A general unified framework for interval pairwise comparisons matrices. International Journal of Approximate Reasoning 2018; 178–198: 93.
25. Hansen P., Ombler F. A new method for scoring additive multi-attribute value models using pairwise rankings of alternatives. Journal of Multi-Criteria Decision Analysis 2008; 15.3-4: 87–107.
26. Opricovic S., Tzeng G.H. Extended VIKOR method in comparison with outranking methods. European journal of operational research 2007; 178.2: 514–529.
27. Alkhairi P., Purba L.P., Eryzha A., Windarto A.P., Wanto A. The analysis of the ELECTREE II algorithm in determining the doubts of the community doing business online. Journal of Physics: Conference Series. IOP Publishing 2019; 1255(1).