Acta Univ. Sapientiae, Informatica **7,** 1 (2015) 58–71

DOI: 10.1515/ausi-2015-0012

Minimization of the Perron eigenvalue of incomplete pairwise comparison matrices by Newton iteration

Kristóf ÁBELE-NAGY
Department of Operations Research and Actuarial Sciences, Corvinus University of Budapest, Hungary
email: kriszo_5@yahoo.de

Abstract. Pairwise comparison matrices are of key importance in multi-attribute decision analysis. A matrix is incomplete if some of the elements are missing. The eigenvector method, to derive the weights of criteria, can be generalized for the incomplete case by using the least inconsistent completion of the matrix. If inconsistency is indexed by CR, defined by Saaty, it leads to the minimization of the Perron eigenvalue. This problem can be transformed to a convex optimization problem. The paper presents a method based on the Newton iteration, univariate and multivariate. Numerical examples are also given.

1 Introduction

When faced with a multi-attribute decision problem, where all alternatives are already evaluated with respect to all relevant criteria, one has to determine the subjective weights of criteria to rank the alternatives. When a decision maker is asked to determine his own subjective weights of criteria, it is often impossible to determine them directly. However, it may be simpler to tell

Computing Classification System 1998: G.1.6
Mathematics Subject Classification 2010: 91B06, 49M15, 90B50
Key words and phrases: incomplete pairwise comparison matrix, Perron eigenvalue, Newton iteration

how many times more important a criterion is compared to another. Ratios are arranged in a positive $n \times n$ matrix $A = [a_{ij}]_{i,j=1,\dots,n}$, which is called a *pairwise comparison matrix* (PCM), where n is the number of criteria. For a PCM $a_{ij} = \frac{1}{a_{ji}}$ holds for all $i, j = 1, \dots, n$, thus every element is 1 in the diagonal. A PCM is *consistent*, if the cardinal transitivity property $a_{ik}a_{kj} = a_{ij}$ holds for all $i, j, k = 1, \dots, n$, otherwise it is called *inconsistent*. The aim is to derive weight vector $\underline{w} = (w_1, w_2, \dots, w_n)^T$ from a PCM that includes the decision maker's subjective judgments.

PCMs can also be used to rank alternatives with respect to a given criterion. Another application of PCMs is to determine the voting power of each decision maker in a group decision problem.

There are several methods for deriving the weight vector [4], we however use the so called eigenvector method proposed by Saaty [14, 15]. For a consistent PCM the following eigenvector equation holds: $A\underline{w} = n\underline{w}$, where $w_i/w_j = a_{ij}, w_i > 0, \sum w_i = 1$. However, PCMs given by real decision makers are rarely consistent. Following the previous equation, Saaty proposed the following method (called the eigenvector method), to gain a weight vector, and also to measure inconsistency. Applying the Perron-Frobenius theorem to a (consistent or inconsistent) PCM, it yields that there is a unique positive Perron eigenvalue λ_{max}, and the corresponding right eigenvector is also positive. It is also known that $\lambda_{max} \geqq n$. Weights can be approximated even in the inconsistent case by the right eigenvector corresponding to λ_{max} (normalizing such that the sum of the weights equal 1), also denoted $\underline{w}$. The eigenvector method provides the weights as the normalized right eigenvector corresponding to the Perron eigenvalue of the PCM.

There are several inconsistency indices in the literature [4], but in the paper we will only discuss one of them: Saaty defined [14] inconsistency index $\mathsf{CR} = \frac{\mathrm{CI}}{\mathrm{ACI}}$, where $\mathsf{CI} = \frac{\lambda_{max} - n}{n-1}$, and ACI denotes the mean value of CI calculated from randomly generated PC matrices of size $n \times n$. Saaty also proposed that PCMs below the threshold $\mathsf{CR} = 0.1$ are to be considered acceptably inconsistent. CR is a positive linear transformation of λ_{max}: the higher λ_{max} is, the more inconsistent the given PCM is.

In some cases, not all elements of a PCM can be or are desired to be filled in. It can take a lot of effort to obtain all $\frac{n(n-1)}{2}$ pairwise comparisons, especially for large PCMs. In this case, missing elements are allowed in the matrix. Such a matrix is called an incomplete pairwise comparison matrix [7, 9], and has the following general form:

$$A = \begin{pmatrix} 1 & a_{12} & * & \dots & a_{1n} \\ 1/a_{12} & 1 & a_{23} & \dots & * \\ * & 1/a_{23} & 1 & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1/a_{1n} & * & 1/a_{3n} & \dots & 1 \end{pmatrix},$$

where $*$ stands for missing elements. They can be in any position except the diagonal, and are symmetric in the sense that if a_{ij} is missing, then a_{ji} is missing, too.

Substitute a variable for each missing element while keeping the reciprocal symmetry rule in mind, and let M denote the number of missing elements above the main diagonal:

$$A(\underline{x}) = A(x_1, x_2, \dots, x_m, \dots, x_M) = \begin{pmatrix} 1 & a_{12} & x_1 & \dots & a_{1n} \\ 1/a_{12} & 1 & a_{23} & \dots & x_M \\ 1/x_1 & 1/a_{23} & 1 & \dots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1/a_{1n} & 1/x_M & 1/a_{3n} & \dots & 1 \end{pmatrix}.$$

The aim is still to obtain a weight vector from the matrix. To facilitate this, the eigenvector method can be generalized to the incomplete case, as proposed by Shiraishi, Obata and Daigo [16, 17]. The solution to the eigenvector method shall be the Perron eigenvector corresponding to the least inconsistent completion of the incomplete PCM. Let inconsistency index CR be applied, therefore, the aim is to minimize CR, or equivalently, the Perron eigenvalue λ_{max}:

$$\min_{\underline{x} \in \mathbb{R}^M_+} \lambda_{max}(A(\underline{x})), \tag{1}$$

where $\mathbb{R}^M_+$ denotes the positive orthant of the M-dimensional Eucledian space. This will be our basic problem from now on.

Key to the existence of the minimum of λ_{max} is that problem (1) can be transformed into a convex optimization problem [2], using the following method: Parametrize incomplete PCM $A(\underline{x}) = A(x_1, x_2, \dots, x_m, \dots, x_M)$ such that $x_m = e^{t_m}, (m = 1, 2, \dots, M)$. This way we gain matrix B:

$$A(\underline{x}) = B(\underline{t}) = B(t_1, t_2, \dots, t_m, \dots, t_M) = A(e^{t_1}, e^{t_2}, \dots, e^{t_m}, \dots, e^{t_M}).$$

$\lambda_{max}(B(\underline{t}))$ is now a convex function of $\underline{t}$ [2].

Bozóki et al. [2] characterized when a unique solution exists to problem (1). The graph corresponding to an incomplete pairwise comparison matrix is defined as follows: $G = (V, E)$, $V = 1, 2, \ldots, n$ (vertices correspond to criteria), $E = \{e(i, j) \mid a_{ij}$ is given in the matrix, $i < j\}$ (edges correspond to pairwise comparisons). In other words, two vertices are connected by an edge if the element corresponding to their pairwise comparison is not missing.

Theorem 1 *[2] There exists a unique solution to* $\min_{\underline{x} \in \mathbb{R}_+^M} \lambda_{\max}(A(\underline{x}))$ *if and only if the graph corresponding to matrix* A *is connected.*

We will also need the partial derivatives of $\lambda_{\max}$ with respect to the elements of the PCM. According to Harker's formulas [8] both the first and second derivatives can be calculated. These formulas can be found in the Appendix.

In the next section three methods are presented to solve problem (1). First one is the method used by Bozóki et al. [2], which is based on the method of cyclic coordinates, and uses Matlab's *fminbnd* function to solve univariate problems. Second and third ones are the main contributions of the paper. Both methods apply Newton iteration, univariate (Section 2.1.2) and multivariate (Section 2.2). The univariate method is similar to the method using *fminbnd*. It also uses cyclic coordinates, but the inner univariate problem is solved by Newton iteration. The multivariate method is based on the multivariate Newton iteration. A numerical example is presented in section 3. Some of the issues presented in the paper have already been considered, in Hungarian, in [1].

2 Algorithms for optimal completion

2.1 Cyclic coordinates

Let us consider an incomplete pairwise comparison matrix A with a connected graph. Let d denote the number of missing elements from the upper triangle of A, so $A = A(x_1, \ldots, x_M)$. Bozóki et al. [2] proposed a completion method based on cyclic coordinates, as follows. Every variable is given a starting value of $x_m^{(0)}, m = 1, 2, \ldots, M$. Every iteration is composed of M steps. In the first step of the first iteration, let x_1 be the only free variable, while the others are fixed at their starting values $x_m^{(0)}, m = 2, 3, \ldots, M$. Let the single optimum of this single variable optimization problem ($\min_{x_1} \lambda_{\max}$) be $x_1^{(1)}$. In the second step of the first iteration, let x_1 be fixed at the value of $x_1^{(1)}$, and let x_2 be the free variable, while all other variables are fixed at their value of $x_m^{(0)}, m = 3, 4, \ldots, M$. Again, from optimizing $\lambda_{\max}$ in x_2, we obtain the optimum $x_2^{(1)}$.

Continue these steps, until we obtain $x_M^{(1)}$. In the second iteration the starting values are $x_m^{(1)}, m = 1, 2, \ldots, M$. So the mth step of the kth iteration is as follows:

$$x_m^{(k)} = \arg\min_{x_m} \lambda_{max}\left(A(x_1^{(k)}, \ldots, x_{m-1}^{(k)}, x_m, x_{m+1}^{(k-1)}, \ldots, x_M^{(k-1)})\right), m = 1, 2, \ldots, M.$$

For the stopping criteria they propose the following: the algorithm stops at the end of the kth iteration if k is the smallest integer for which

$$\max_{m=1,2,\ldots,M} |x_m^{(k)} - x_m^{(k-1)}| < T, \tag{2}$$

where T is the tolerance threshold ($T = 10^{-4}$ is chosen for their and our tests as well).

Another important question is the choice of the starting values. Bozóki et al. [2] in their numerical example used $x_m^{(0)} = 1, m = 1, 2, \ldots, M$. In the paper we will use values based on the solution of the incomplete logarithmic least squares method (ILLSM) [11]. This method determines weight vector $\underline{w}$ by minimizing

$$\min \sum_{\substack{i,j=1 \\ a_{ij} \text{ is given}}}^{n} \left[\log a_{ij} - \log\left(\frac{w_i}{w_j}\right)\right]^2, \tag{3}$$

where $\sum_{i=1}^{n} w_i = 1$ and $w_i > 0 \quad i = 1, \ldots, n$. Solving this problem is based on solving a system of linear equations [2]. Although the ILLSM method can generate ordering different from that of the eigenvector method, it provides reasonable starting values for our iteration [10]. Therefore, the solution of the ILLSM problem will be used for the starting values for $\underline{x}$. Let $w_i^L, i = 1, \ldots, n$ denote the ith component of the weight vector derived from solving the ILLSM problem, and let x_m be in position (i, j). The starting values will be $x_m^{(0)} = w_i^L / w_j^L, m = 1, 2, \ldots, M$.

Again, in order to transform problem (1) to a convex optimization problem, rescaling $x_m = e^{t_m}, m = 1, \ldots, M$ is done [2]. Let $L(t_m) = \lambda_{max}(e^{t_m})$.

The global convergence of cyclic coordinates is stated and proved in [12, pages 253–254].

The cyclic coordinates method presented above will be the framework for the single variable method presented here as well, with the fundamental difference being in how we obtain the optimum.

2.1.1 Cyclic coordinates with Matlab's *fminbnd*

Bozóki et al. [2] used a general optimization function in Matlab (*fminbnd*) for obtaining $\min_{t_m} \lambda_{max}$. Function *fminbnd* uses an algorithm which combines golden section search and parabolic interpolation [3, 5, 13]. A method tailored for this problem and based on the Newton iteration is presented next.

2.1.2 Cyclic coordinates with univariate Newton iteration

Using the method of cyclic coordinates, we are optimizing in only one variable at a time. Let us denote this variable by x, while the other variables are fixed while the minimization occurs. Our goal is to write the Newton iteration of this optimization. Let $x = e^t$ (similarly $x^{(r)} = e^{t^{(r)}}$) and $L(t) = \lambda_{max}(e^t)$. With these notions, we are searching for t where $L'(t) = 0$. Because of this, the rth iteration of Newton's method can be written as

$$t^{(r+1)} = t^{(r)} - \frac{L'(t^{(r)})}{L''(t^{(r)})}.$$

According to Harker [8] the derivatives $\frac{\partial \lambda_{max}(x)}{\partial x}$ and $\frac{\partial^2 \lambda_{max}(x)}{(\partial x)^2}$ are known, and depend on the position (i, j) of x in of the matrix.

To write the Newton iteration we need $\frac{\partial L(t)}{\partial t}$ and $\frac{\partial^2 L(t)}{(\partial t)^2}$.

$$\frac{\partial L(t)}{\partial t} = \frac{\partial \lambda_{max}(e^t)}{\partial t} = \frac{\partial \lambda_{max}(x)}{\partial x} \cdot \frac{\partial e^t}{\partial t} = \frac{\partial \lambda_{max}(x)}{\partial x} \cdot e^t. \tag{4}$$

Similarly

$$\begin{aligned} \frac{\partial^2 L(t)}{(\partial t)^2} = \frac{\partial^2 \lambda_{max}(e^t)}{(\partial t)^2} = \frac{\frac{\partial \lambda_{max}(x)}{\partial x} \cdot e^t}{\partial t} = \\ = \frac{\frac{\partial \lambda_{max}(x)}{\partial x}}{\partial t} \cdot e^t + \frac{\partial \lambda_{max}(x)}{\partial x} \cdot \frac{\partial e^t}{\partial t} = \frac{\partial^2 \lambda_{max}(x)}{(\partial x)^2} \cdot e^{2t} + \frac{\partial \lambda_{max}(x)}{\partial x} \cdot e^t. \end{aligned} \tag{5}$$

Newton iteration can now be written as

$$\begin{aligned} t^{(r+1)} = t^{(r)} - \frac{L'(t^{(r)})}{L''(t^{(r)})} = t^{(r)} - \frac{\frac{\partial \lambda_{max}(x)}{\partial x}(x^{(r)}) \cdot e^{t^{(r)}}}{\frac{\partial^2 \lambda_{max}(x)}{(\partial x)^2}(x^{(r)}) \cdot e^{2t^{(r)}} + \frac{\partial \lambda_{max}(x)}{\partial x}(x^{(r)}) \cdot e^{t^{(r)}}} = \\ = t^{(r)} - \frac{\frac{\partial \lambda_{max}(x)}{\partial x}(x^{(r)})}{\frac{\partial^2 \lambda_{max}(x)}{(\partial x)^2}(x^{(r)}) \cdot e^{t^{(r)}} + \frac{\partial \lambda_{max}(x)}{\partial x}(x^{(r)})}. \end{aligned} \tag{6}$$

As mentioned, $\dfrac{\partial\lambda_{max}(x)}{\partial x}$ and $\dfrac{\partial^2\lambda_{max}(x)}{(\partial x)^2}$ depend on the position of the element x in the matrix (i,j). In a particular step of an iteration all the other variables are temporarily fixed, as described earlier.

A full step of the Newton iteration (which is only a subroutine of a step of the cyclic coordinate iteration) consists of the following steps, where x is in position (i,j):

1. $t^{(r)} = \ln x^{(r)}$,
2. Apply (6),
3. $x^{(r+1)} = e^{t^{(r+1)}}, 1/x^{(r+1)} = e^{-t^{(r+1)}}$.

With this algorithm we managed to apply the Newton iteration specifically for the problem of minimizing the Perron eigenvalue of incomplete PCMs.

2.2 Multivariate Newton iteration

Instead of using the method of cyclic coordinates and optimizing in one variable at a time, one can optimize in all of the variables at the same time, using the multivariate Newton iteration. Let $L(\underline{t}) = \lambda_{max}(e^{t_1},\dots,e^{t_M})$. We want to minimize L, so we need:

$$\underline{t}^{(r+1)} = \underline{t}^{(r)} - \gamma[HL(\underline{t}^{(r)})]^{-1}\nabla L(\underline{t}^{(r)}), \tag{7}$$

where $HL(\underline{t}^{(r)})$ is the Hessian matrix of $L(\underline{t})$, and $\nabla L(\underline{t}^{(r)})$ is the gradient vector of $L(\underline{x})$ (both in the rth Newton iteration), and γ is the step size as usual in multivariate Newton iteration. Again, because of the parametrization $\underline{x} = e^{\underline{t}}$, we have to adapt this formula for our case. All of the elements of the gradient vector $\nabla L(\underline{t}) = \left(\dfrac{\partial L(\underline{t})}{\partial t_1},\dots,\dfrac{\partial L(\underline{t})}{\partial t_M}\right)$ can be calculated with the method described earlier, namely (4).

Now let us write the Hessian matrix:

$$HL(\underline{t}) = \begin{pmatrix} \dfrac{\partial^2 L(\underline{t})}{\partial t_1^2} & \dfrac{\partial^2 L(\underline{t})}{\partial t_1 \partial t_2} & \cdots & \dfrac{\partial^2 L(\underline{t})}{\partial t_1 \partial t_M} \\ \dfrac{\partial^2 L(\underline{t})}{\partial t_2 \partial t_1} & \dfrac{\partial^2 L(\underline{t})}{\partial t_2^2} & \cdots & \dfrac{\partial^2 L(\underline{t})}{\partial t_2 \partial t_M} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 L(\underline{t})}{\partial t_M \partial t_1} & \dfrac{\partial^2 L(\underline{t})}{\partial t_M \partial t_2} & \cdots & \dfrac{\partial^2 L(\underline{t})}{\partial t_M^2} \end{pmatrix}.$$

Diagonal elements are calculated by (5). We still need to reformulate the off-diagonal elements of the Hessian matrix, where we differentiate with respect to different variables. From now on, $x_p = e^{t_p}$ is in position (i, j) and $x_q = e^{t_q}$ is in position (u, v).

$$\frac{\partial^2 L(\underline{t})}{\partial t_p \partial t_q} = \frac{\partial^2 \lambda_{max}(e^{t_1}, \ldots, e^{t_M})}{\partial t_p \partial t_q} = \frac{\partial \left(\frac{\partial \lambda_{max}(e^{t_1}, \ldots, e^{t_M})}{\partial t_q} \right)}{\partial t_p}.$$

We can apply (4), and get $\frac{\partial \lambda_{max}(e^{t_1}, \ldots, e^{t_M})}{\partial t_q} = \frac{\partial \lambda_{max}(\underline{x})}{\partial x_q} \cdot e^{t_q}$. Including the case $p = q$ as well,

$$\frac{\partial^2 L(\underline{t})}{\partial t_p \partial t_q} = \frac{\partial \left(\frac{\partial \lambda_{max}(\underline{x})}{\partial x_q} \cdot e^{t_q} \right)}{\partial t_p} = \frac{\partial \left(\frac{\partial \lambda_{max}(\underline{x})}{\partial x_q} \right)}{\partial t_p} \cdot e^{t_q} + \frac{\partial \lambda_{max}(\underline{x})}{\partial x_q} \cdot \frac{\partial e^{t_q}}{\partial t_p}. \tag{8}$$

Here

$$\frac{\partial e^{t_q}}{\partial t_p} = e^{t_q} \cdot \chi_{\{p=q\}}, \tag{9}$$

where

$$\chi_{\{p=q\}} = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}.$$

On the other hand,

$$\frac{\partial \left(\frac{\partial \lambda_{max}(\underline{x})}{\partial x_q} \right)}{\partial t_p} = \frac{\partial \left(\frac{\partial \lambda_{max}(\underline{x})}{\partial x_q} \right)}{\partial x_p} \cdot \frac{\partial x_p}{\partial t_p} = \frac{\partial \left(\frac{\partial \lambda_{max}(\underline{x})}{\partial x_q} \right)}{\partial t_p} \cdot \frac{\partial e^{t_p}}{\partial t_p} = \frac{\partial^2 \lambda_{max}(\underline{x})}{\partial x_p \partial x_q} \cdot e^{t_p}, \tag{10}$$

which also includes the case $p = q$. Writing (9) and (10) back into (8) we get the final form of

$$\frac{\partial^2 L(\underline{t})}{\partial t_p \partial t_q} = \frac{\partial^2 \lambda_{max}(\underline{x})}{\partial x_p \partial x_q} \cdot e^{t_p + t_q} + \frac{\partial \lambda_{max}(\underline{x})}{\partial x_p} \cdot e^{t_p} \cdot \chi_{\{p=q\}}. \tag{11}$$

Note that (5) is a special case of (11) with $p = q$. In (11) we can calculate $\frac{\partial^2 \lambda_{max}(\underline{x})}{\partial x_p \partial x_q}$ and $\frac{\partial \lambda_{max}(\underline{x})}{\partial x_p}$ according to Harker's formulas [8], and they depend on the positions of the variables in the matrix.

Using these formulas, and given a starting value of $\underline{t}_0$ for $\underline{t}$, we can calculate the gradient vector and Hessian matrix for every iteration of the multivariate Newton's method (7). Again, the iteration continues while (2) is satisfied. The stopping criteria is determined for $\underline{x}$ (not $\underline{t}$), because a small difference in $\underline{t}$ can lead to large differences in $\underline{x}$.

3 Numerical example

Let us consider the following incomplete pairwise comparison matrix:

$$\begin{pmatrix} 1 & 5 & 3 & 7 & 6 & 6 & 1/3 & 1/4 \\ 1/5 & 1 & x_1 & 5 & x_2 & 3 & x_3 & 1/7 \\ 1/3 & 1/x_1 & 1 & x_4 & 3 & x_5 & 6 & x_6 \\ 1/7 & 1/5 & 1/x_4 & 1 & x_7 & 1/4 & x_8 & 1/8 \\ 1/6 & 1/x_2 & 1/3 & 1/x_7 & 1 & x_9 & 1/5 & x_{10} \\ 1/6 & 1/3 & 1/x_5 & 4 & 1/x_9 & 1 & x_{11} & 1/6 \\ 3 & 1/x_3 & 1/6 & 1/x_8 & 5 & 1/x_{11} & 1 & x_{12} \\ 4 & 7 & 1/x_6 & 8 & 1/x_{10} & 6 & 1/x_{12} & 1 \end{pmatrix}.$$

This is an incomplete version of Saaty's "buying a house" example matrix [15]. Further, it is the same incomplete PCM which was used as an example by Bozóki et al. [2], where it is shown that the graph corresponding to this matrix is connected.

In Tables 1 and 2 the values of each variable in each iteration are shown using the univariate Newton iteration and the multivariate Newton iteration, respectively. However, iteration does not mean the same thing in these cases. For the univariate Newton method, an iteration k is the outer iteration which contains $m = 12$ complete univariate Newton iterations for each k. For the multivariate Newton iteration, an iteration r is one iteration of the multivariate Newton method itself.

As mentioned earlier, $T = 10^{-4}$ in (2), and the starting values $x_m^{(0)}$ are chosen to be equal to the optimal solution of the ILLSM problem (3). For the multivariate case, several γ values have been experimented with. $\gamma = 0.45$ yielded the lowest number of iterations, therefore this value was used for the results in Table 2. The number of iterations required was $k = 14$ in the univariate case, and $r = 14$ in the multivariate case.

Tests were also done for starting values $x_m^{(0)} = 1, m = 1, \dots, M$ (again with $\gamma = 0.45$ in the multivariate case). The univariate method required $k = 15$ iterations, while the multivariate method required $r = 26$ iterations.

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$x_4^{(k)}$	$x_5^{(k)}$	$x_6^{(k)}$	$x_7^{(k)}$	$x_8^{(k)}$	$x_9^{(k)}$	$x_{10}^{(k)}$	$x_{11}^{(k)}$	$x_{12}^{(k)}$
0	0.3823	1.8430	0.4758	8.9920	4.2690	0.5228	0.5361	0.1384	0.8855	0.1085	0.2916	0.4200
1	0.3460	1.7620	0.4699	9.6590	4.7370	0.5667	0.5320	0.1434	0.9206	0.1091	0.2932	0.4008
2	0.3338	1.7320	0.4690	9.8410	4.8230	0.5671	0.5283	0.1431	0.9270	0.1088	0.2924	0.4016
3	0.3315	1.7270	0.4678	9.8840	4.8370	0.5681	0.5271	0.1428	0.9283	0.1090	0.2919	0.4023
4	0.3308	1.7240	0.4671	9.9000	4.8440	0.5687	0.5264	0.1427	0.9295	0.1091	0.2916	0.4026
5	0.3305	1.7220	0.4668	9.9090	4.8470	0.5691	0.5259	0.1426	0.9302	0.1092	0.2914	0.4028
6	0.3303	1.7210	0.4666	9.9140	4.8500	0.5693	0.5256	0.1425	0.9306	0.1093	0.2913	0.4029
7	0.3302	1.7210	0.4665	9.9170	4.8510	0.5694	0.5255	0.1425	0.9309	0.1093	0.2913	0.4030
8	0.3301	1.7200	0.4664	9.9180	4.8520	0.5695	0.5254	0.1425	0.9310	0.1093	0.2913	0.4030
9	0.3301	1.7200	0.4664	9.9190	4.8520	0.5696	0.5253	0.1425	0.9311	0.1093	0.2912	0.4031
10	0.3300	1.7200	0.4664	9.9200	4.8520	0.5696	0.5253	0.1424	0.9311	0.1093	0.2912	0.4031
11	0.3300	1.7200	0.4664	9.9200	4.8520	0.5696	0.5253	0.1424	0.9312	0.1093	0.2912	0.4031
12	0.3300	1.7200	0.4664	9.9200	4.8520	0.5696	0.5253	0.1424	0.9312	0.1093	0.2912	0.4031
13	0.3300	1.7200	0.4664	9.9200	4.8520	0.5696	0.5253	0.1424	0.9312	0.1093	0.2912	0.4031
14	0.3300	1.7200	0.4664	9.9210	4.8520	0.5696	0.5253	0.1424	0.9312	0.1093	0.2912	0.4031

Table 1: Univariate Newton

r	$x_1^{(r)}$	$x_2^{(r)}$	$x_3^{(r)}$	$x_4^{(r)}$	$x_5^{(r)}$	$x_6^{(r)}$	$x_7^{(r)}$	$x_8^{(r)}$	$x_9^{(r)}$	$x_{10}^{(r)}$	$x_{11}^{(r)}$	$x_{12}^{(r)}$
0	0.3823	1.8430	0.4758	8.9920	4.2690	0.5228	0.5361	0.1384	0.8855	0.1085	0.2916	0.4200
1	0.3385	1.7160	0.4484	9.7830	4.7170	0.5786	0.5006	0.1276	0.9592	0.1221	0.2739	0.4427
2	0.3430	1.7540	0.4707	9.5490	4.6180	0.5500	0.5368	0.1446	0.9082	0.1087	0.2946	0.4040
3	0.3285	1.7070	0.4581	9.9020	4.8210	0.5755	0.5169	0.1376	0.9436	0.1149	0.2848	0.4172
4	0.3323	1.7260	0.4672	9.8070	4.7800	0.5640	0.5297	0.1435	0.9243	0.1096	0.2924	0.4035
5	0.3287	1.7140	0.4637	9.9140	4.8430	0.5715	0.5234	0.1412	0.9349	0.1112	0.2894	0.4074
6	0.3302	1.7200	0.4664	9.8880	4.8320	0.5682	0.5268	0.1428	0.9294	0.1096	0.2916	0.4034
7	0.3295	1.7180	0.4656	9.9170	4.8490	0.5702	0.5251	0.1422	0.9322	0.1099	0.2908	0.4043
8	0.3299	1.7200	0.4664	9.9110	4.8470	0.5693	0.5258	0.1426	0.9308	0.1094	0.2913	0.4033
9	0.3298	1.7190	0.4662	9.9190	4.8520	0.5698	0.5253	0.1424	0.9315	0.1095	0.2911	0.4035
10	0.3300	1.7200	0.4664	9.9180	4.8510	0.5696	0.5255	0.1425	0.9311	0.1094	0.2913	0.4032
11	0.3299	1.7200	0.4663	9.9200	4.8520	0.5697	0.5253	0.1424	0.9313	0.1094	0.2912	0.4032
12	0.3300	1.7200	0.4664	9.9200	4.8520	0.5696	0.5253	0.1425	0.9312	0.1093	0.2912	0.4031
13	0.3300	1.7200	0.4663	9.9200	4.8520	0.5696	0.5253	0.1424	0.9312	0.1093	0.2912	0.4031
14	0.3300	1.7200	0.4664	9.9200	4.8520	0.5696	0.5253	0.1424	0.9312	0.1093	0.2912	0.4031

Table 2: Multivariate Newton

4 Conclusions

When using the generalized eigenvector method for incomplete pairwise comparison matrices, the matrix is to be completed optimally with regard to its inconsistency, or equivalently minimizing its Perron eigenvalue λ_{max}. Although eigenvalue minimization problems are generally difficult due to nonconvexity, the special case of incomplete PCMs proves to be convex. In order to solve the problem, a method based on the Newton iteration (both univariate and multivariate) is presented in the paper.

Future research can be focused on the choice of γ in case of the multivariate method. Fülöp [6] has recently proposed an alternative method for minimizing λ_{max}, further research includes a comparative analysis of the algorithms.

Acknowledgements

The author thanks János Fülöp (Institute for Computer Science and Control, Hungarian Academy of Sciences – MTA SZTAKI; Óbuda University) for his remarks. The author thanks his PhD supervisor Sándor Bozóki (Institute for Computer Science and Control, Hungarian Academy of Sciences – MTA SZTAKI; Corvinus University of Budapest) for valuable discussions. The support of OTKA grant K 111797 is gratefully acknowledged.

References

[1] K. Ábele-Nagy, *Incomplete pairwise comparison matrices in multi-attribute decision making (In Hungarian, Nem teljesen kitöltött páros összehasonlítás mátrixok a többszempontú döntésekben)*, Master's Thesis, Eötvös Loránd University, Budapest, 2010. ⇒61

[2] S. Bozóki, J. Fülöp, L. Rónyai, On optimal completion of incomplete pairwise comparison matrices, *Math. Comput. Modelling* **52,** 1–2, (2010) 318–333. ⇒60, 61, 62, 63, 66

[3] R. P. Brent, *Algorithms for Minimization without Derivatives*, Prentice-Hall, Englewood Cliffs, NJ, 1973. ⇒63

[4] M. Brunelli, *Introduction to the Analytic Hierarchy Process*, SpringerBriefs in Operations Research, Springer, New York, 2015. ⇒59

[5] G. E. Forsythe, M. A. Malcolm, C. B. Moler, *Computer Methods for Mathematical Computations*, Prentice-Hall, Englewood Cliffs, NJ, 1976. ⇒63

[6] J. Fülöp, An optimization approach for the eigenvalue method, *IEEE 8th International Symposium on Applied Computational Intelligence and Informatics (SACI 2013)*, Timişoara, Romania, 23–25 May 2013. ⇒69

[7] P. T. Harker, Alternative modes of questioning in the Analytic Hierarchy Process. *Math. Model.* **9,** 3 (1987) 353–360. ⇒59

[8] P. T. Harker, Derivatives of the Perron root of a positive reciprocal matrix: with application to the Analytic Hierarchy Process. *Appl. Math. Comput.* **22,** 2–3 (1987) 217–232. ⇒61, 63, 65, 70

[9] P. T. Harker, Incomplete pairwise comparisons in the Analytic Hierarchy Process. *Math. Model.* **9,** 11 (1987) 837–848. ⇒59

[10] A. Ishizaka, M. Lusti, How to derive priorities in AHP: a comparative study. *Cent. Eur. J. Oper. Res.*, **14,** 4 (2006) 387–400. ⇒62

[11] M. Kwiesielewicz, The logarithmic least squares and the generalised pseudoinverse in estimating ratios. *European J. Oper. Res.* **93,** 3 (1996) 611–619. ⇒62

[12] D. G. Luenberger, Y. Ye, *Linear and Nonlinear Programming* (3rd Edition), Series: International Series in Operations Research & Management Science, **116**, Springer, New York, 2008. ⇒62

[13] Matlab Documentation: fminbnd `http://www.mathworks.com/help/matlab/ref/fminbnd.html`. ⇒63

[14] T. L. Saaty, A scaling method for priorities in hierarchical structures. *J. Math. Psych.*, **15,** 3 (1977) 234–281. ⇒59

[15] T. L. Saaty, *The Analytic Hierarchy Process*, McGraw-Hill, New York, 1980. ⇒59, 66

[16] S. Shiraishi, T. Obata, M. Daigo, Properties of a positive reciprocal matrix and their application to AHP. *J. Oper. Res. Soc. Jpn.* **41,** 3 (1998) 404–414. ⇒60

[17] S. Shiraishi, T. Obata, On a maximization problem arising from a positive reciprocal matrix in AHP. *Bull. Inform. Cybernet.* **34,** 2 (2002) 91–96. ⇒60

Received: January 24, 2015 • Revised: April 29, 2015

Appendix

Harker's [8] formulas for the derivatives of the Perron eigenvalue are presented in the appendix.

Let A denote a PCM, and let $\underline{x} = \underline{x}(A)$ and $\underline{y} = \underline{y}(A)$ denote its right and left Perron eigenvectors, and $\lambda_{max} = \lambda_{max}(A)$ its Perron eigenvalue, so $A\underline{x} = \lambda_{max}\underline{x}$ and $\underline{y}^T A = \lambda_{max}\underline{y}^T$. The normalization for the eigenvectors in this case is $\underline{y}^T\underline{x} = 1$. Let $Q = \lambda_{max} I - A$. Also let Q^+ denote the pseudoinverse of Q, which satisfies the following properties: $QQ^+Q = Q$, $Q^+QQ^+ = Q^+$, $Q^+Q = QQ^+$. Finally, ∂a_{ij} denotes differentiation with respect to the element in position (i, j) in A, and similarly ∂a_{kl} denotes differentiation with respect to the element in position (k, l). Using these notations, the formulas are as follows:

The first derivatives:

$$\frac{\partial \lambda_{max}}{\partial a_{ij}} = y_i x_j - \frac{y_j x_i}{a_{ij}^2}, \text{ if } i > j.$$

The second derivatives:

$$\frac{\partial^2 \lambda_{max}}{\partial a_{ij} \partial a_{kl}} = \begin{cases} (\underline{x}\underline{y}^T)_{li} Q_{jk}^+ + (\underline{x}\underline{y}^T)_{jk} Q_{li}^+ - \frac{(\underline{x}\underline{y}^T)_{ki} Q_{jl}^+ + (\underline{x}\underline{y}^T)_{jl} Q_{ki}^+}{a_{kl}^2} - \\ -\frac{(\underline{x}\underline{y}^T)_{lj} Q_{ik}^+ + (\underline{x}\underline{y}^T)_{ik} Q_{lj}^+}{a_{ij}^2} + \frac{(\underline{x}\underline{y}^T)_{kl} Q_{il}^+ + (\underline{x}\underline{y}^T)_{il} Q_{kj}^+}{a_{ij}^2 a_{kl}^2} \\ \qquad\qquad\qquad\qquad \text{if } i > j,\ k > l,\ (i,j) \neq (k,l) \\ 2\frac{(\underline{x}\underline{y}^T)_{ij}}{a_{ij}^3} + 2(\underline{x}\underline{y}^T)_{ji} Q_{ji}^+ - \\ -2\frac{(\underline{x}\underline{y}^T)_{ii} Q_{jj}^+ + (\underline{x}\underline{y}^T)_{jj} Q_{ii}^+}{a_{ij}^2} + 2\frac{(\underline{x}\underline{y}^T)_{ij} Q_{ij}^+}{a_{ij}^4} \\ \qquad\qquad\qquad\qquad \text{if } i > j,\ (i,j) = (k,l). \end{cases}$$