J Glob Optim (2007) 38:237–247
DOI 10.1007/s10898-006-9073-5

ORIGINAL PAPER

An exact global optimization method for deriving weights from pairwise comparison matrices

Emilio Carrizosa · Frédéric Messine

Received: 12 January 2006 / Accepted: 27 July 2006 / Published online: 6 September 2006

Abstract Some multiple-criteria decision making methods rank actions by associating weights to the different criteria or actions, which are pairwise compared via a positive reciprocal matrix A. There is a vast literature on proposals of different mathematical-programming methods to infer weights from such matrix A. However, it is seldom observed that such optimization problems may be multimodal, thus the standard local-search resolution techniques suggested may be trapped in local optima, yielding a wrong ranking of alternatives. In this note we show that standard tools of global optimization based on interval analysis, lead to globally optimal weights in reasonable time.

Keywords Branch and bound · Interval analysis · Multiple-criteria decision making · Pairwise comparison matrices

1 Introduction

In order to rank a set of n given decisions, a possible strategy consists of associating with them weights $x_1, \ldots, x_n$ through a nonlinear regression model of the form

$$\frac{x_i}{x_j} = a_{ij}, \quad i, j = 1, 2, \ldots, n, \tag{1}$$

where the elements a_{ij}, representing the relative preference of the ith action over the jth action, are positive constant assumed to be given.

E. Carrizosa
Facultad de Matemáticas, Universidad de Sevilla, Tarfia s/n 41012, Sevilla, Spain
e-mail: ecarrizosa@us.es

F. Messine (✉)
ENSEEIHT-IRIT, 1 rue C Camichel 31071, Toulouse Cedex, France
e-mail: Frederic.Messine@n7.fr

Different numerical procedures, not absent of controversy, have been proposed in the literature to infer x from model (1) (see e.g. [1, 2, 8, 12, 14, 16–18] and the references therein).

A broad class of such procedures yields as x an optimal solution of a problem of the form

$$\min_{x\in\mathbb{R}^n_{++}} \gamma\left(\left(|x_i/x_j - a_{ij}|\right)_{i,j=1}^n\right), \tag{2}$$

where $\mathbb{R}_{++}$ denotes the set of strictly positive reals, and γ is a monotonic norm in the non-negative orthant $\mathbb{R}_+^{n\times n}$, i.e.,

$$\gamma(u) \le \gamma(v), \quad \forall u, v,\ 0 \le u \le v \tag{3}$$

such as any (weighted) ℓ_p norm, $1 \le p \le +\infty$. This approach has been proposed, amon others, in [2, 5, 9, 11]. In particular, in [5] it is claimed that "such methods are more appealing because it is undisputable that the matrix (x_i/x_j) associated to the solution vector x must be close to the judgment matrix A." Three particular instances of (2) are included in [5, 11], namely the unweighted ℓ_1, ℓ_2 and ℓ_∞, under the names of *Least Absolute Error*, *Least Square* and *Least Worst Absolute Error* (see e.g. [16–18]) for a different strategy to infer weights from A which is not based on optimization techniques and may lead, as shown in [2], to non-efficient weights.

In spite of the fact that (2) may be multimodal, local-search methods seem to be the only proposal so far, the exception being the recent works [3, 4, 7], where different global-optimization strategies for solving (2) for the Euclidean norm γ are analysed. The methods in these references exploit extensively the properties of the Euclidean norm, and thus cannot be easily extended to more general settings, as the one considered in this paper.

For choices of γ different of the Eculidean norm, the literature is rather scarce. In [5], for the particular instances of (2) addressed, (ℓ_1 and ℓ_∞ norm together with the ℓ_2), the problems are considered to be "difficult to solve", although no attempt to solving them is given. These three problems are later addressed in [11] via "a simple genetic algorithm", which may yield solutions far from optimality.

From a practical viewpoint, it has been observed that it is not so easy to provide precise values for the scalar a_{ij}, which are thus replaced by intervals $A_{ij} = [a_{ij}^L, a_{ij}^U]$ (see e.g. [6, 10, 19, 20] and the references therein). In this case, the regression model (1) is replaced by the more general model

$$\frac{x_i}{x_j} \in A_{ij}, \quad i, j = 1, 2, \ldots, n \tag{4}$$

and (2) is replaced by the problem of finding strictly positive weights x_j minimizing a norm γ of the distances between the ratios x_i/x_j and the corresponding intervals A_{ij},

$$\min_{x\in\mathbb{R}^n_{++}} f(x) := \gamma\left(\left(\varepsilon\left(x_i/x_j, A_{ij}\right)\right)_{i,j=1}^n\right) \tag{5}$$

with $\varepsilon(s,[a^L,a^U])$ defined as the distance between s and the closest point in $[a^L,a^U]$,

$$\varepsilon(s,[a^L,a^U]) = \begin{cases} 0, & \text{if } s \in [a^L,a^U], \\ a^L - s, & \text{if } s \le a^L, \\ s - a^U, & \text{if } s \ge a^U \end{cases} \tag{6}$$

$$= \max\left\{0, \left|s - \frac{a^U + a^L}{2}\right| - \frac{a^U - a^L}{2}\right\}. \tag{7}$$

Obviously (5) coincides with (2) in what we call the degenerate case in which all intervals A_{ij} are degenerate, $A_{ij} = \{a_{ij}\}$.

The aim of this paper is to show that well-known strategies of deterministic global optimization, namely, branch and bound algorithms with bounds based on interval analysis, provide in reasonable time globally optimal solutions to problem (5), and thus, as particular case, to model (2), for different choices of γ. Hence, if a decision-maker infers weights from a pairwise comparison matrix following model (5) for a given norm γ, we show that, although, according to [5], the problems are "difficult to solve", they are solvable *exactly* by standard Global Optimization methods.

At the same time, uncertainties in the comparisons, modelled as intervals A_{ij} for the weights ratios, are included naturally. This is an interesting advantage of model (5) against, for instance, the methods recommended in [5, 11, 16].

In what follows γ is an arbitrary norm monotonic in $\mathbb{R}^{n\times n}_+$, as defined in (3). We stress that γ is assumed to be given, thus we do not make any suggestion about the choice of γ. Instead, we refer to [5], where it is said that "no one method is expected to outperform other methods at all situations."

For $i,j = 1,\ldots,n$, $A_{ij} = [a^L_{ij}, a^U_{ij}]$ is an interval satisfying $0 < a^L_{ij} \le a^U_{ij}$. Again, this interval matrix is assumed to be given.

With these elements at hand, we now show how to infer the weights from such (interval) pairwise comparison matrix.

2 General results

According to its definition (5), $f(\lambda x) = f(x)$ for any $x \in \mathbb{R}^n_{++}$ and any choice of $\lambda > 0$. Hence, a normalization constraint can be added to (5) without loss of generality. In particular, we can fix x_1 to unity, and replace (5) by

$$\begin{array}{ll} \min & f(x) := \gamma\left(\left(\varepsilon\left(x_i/x_j, A_{ij}\right)\right)^n_{i,j=1}\right), \\ \text{s.t.} & x_1 = 1, \\ & x \in \mathbb{R}^n_{++}. \end{array} \tag{8}$$

Observe that this way, we lower in one the dimensionality of the problem, which may be of great importance when deterministic global optimization techniques are to be used.

Since norms are continuous, we have by (7) that the objective function (8) is continuous in the feasible region. However, such feasible region is not closed, which makes at first glance unclear whether the optimal value is attained. However, one can easily construct a compact interval containing all optimal solutions of (8), which are shown to exist. Indeed, one has

Proposition 1 *Let z be strictly greater than the optimal value of (8). Then, any optimal solution x to (8) satisfies*

$$x_j \in X_j := [x_j^L, x_j^U], \quad j = 1, 2, \ldots, n,$$

where

$$X_j = \begin{cases} \{1\}, & \text{if } j = 1, \\ \left[\max\limits_{i<j} \dfrac{x_i^L}{a_{ij}^U + \frac{z}{\gamma(e_{ij})}}, \min\limits_{i<j} x_i^U \left(a_{ji}^U + \dfrac{z}{\gamma(e_{ji})} \right) \right], & \text{if } j > 1 \end{cases} \tag{9}$$

and e_{ij} is the $n \times n$ matrix with all elements null except the element (i, j), with value 1.

In particular, (8), and thus (5), always has optimal solutions.

Proof We will show that for any x, feasible for (8), but not satisfying (9), $f(x)$ is at least as great as z, known to be strictly greater than the optimal value. Hence, x cannot be optimal. Having shown this, we have in particular that (8) is equivalent to the problem where the constraint $x \in \mathbb{R}^n_{++}$ is replaced by (9), and thus we are optimizing a continuous function over a compact domain, guaranteeing then the existence of optimal solutions.

Since, by assumption, γ is monotonic in $\mathbb{R}^{n\times n}_+$, one has

$$f(x) \geq \gamma(\varepsilon(x_i/x_j, A_{ij})e_{ij}), \quad \forall i, j, \forall x \in \mathbb{R}^n_{++}. \tag{10}$$

Take x not satisfying (9), and let j be the smallest index for which $x_j \notin X_j$. Then, there must exist $i < j$ such that either

$$x_j < \frac{x_i^L}{a_{ij}^U + \frac{z}{\gamma(e_{ij})}} \tag{11}$$

or

$$x_j > x_i^U \left(a_{ji}^U + \frac{z}{\gamma(e_{ji})} \right). \tag{12}$$

If (11) holds, then,

$$\begin{aligned} \varepsilon(x_i/x_j, A_{ij}) &= \frac{x_i}{x_j} - a_{ij}^U \\ &> \frac{z}{\gamma(e_{ij})}. \end{aligned}$$

By (10),

$$\begin{aligned} f(x) &\geq \gamma(\varepsilon(x_i/x_j, A_{ij})e_{ij}) \\ &\geq \gamma\left(\frac{z}{\gamma(e_{ij})} e_{ij} \right) = z \end{aligned}$$

thus x cannot be optimal.

Analogously, if (12) holds then

$$\begin{aligned} \varepsilon(x_j/x_i, A_{ji}) &= \frac{x_j}{x_i} - a_{ji}^U \\ &> \frac{z}{\gamma(e_{ji})}. \end{aligned}$$

Hence,

$$\begin{aligned} f(x) &\geq \gamma(\varepsilon(x_j/x_i, A_{ji})e_{ji}) \\ &\geq \gamma\left(\frac{z}{\gamma(e_{ji})}e_{ji}\right) = z, \end{aligned}$$

showing that such x cannot be optimal. □

Following Proposition 1, we can reformulate (8) as a nonlinear nonconvex box-constrained problem of the form

$$\begin{aligned} &\min f(x) := \gamma\left(\left(\varepsilon\left(x_i/x_j, A_{ij}\right)\right)_{i,j=1}^{n}\right), \\ &\text{s.t.} \quad x_j \in X_j, \quad j = 1, 2, \ldots, n. \end{aligned} \tag{13}$$

Let us now discuss how to come up with a globally optimal solution to (13), first in the particular case in which the norm γ is the ℓ_∞ norm, and then for arbitrary monotonic norms γ.

2.1 The ℓ_∞ norm

For γ equal the ℓ_∞ norm, we have by (7) that

$$\begin{aligned} f(x) &= \max_{1\leq i,j\leq n} \varepsilon(x_i/x_j, A_{ij}) \\ &= \max_{1\leq i,j\leq n} \left\{ \max\left\{0, \left|\frac{x_i}{x_j} - \frac{a_{ij}^U + a_{ij}^L}{2}\right| - \frac{a_{ij}^U - a_{ij}^L}{2}\right\}\right\} \\ &= \max\left\{0, \max_{1\leq i,j\leq n} \left|\frac{x_i}{x_j} - \frac{a_{ij}^U + a_{ij}^L}{2}\right| - \frac{a_{ij}^U - a_{ij}^L}{2}\right\}. \end{aligned}$$

It is then seen that f is quasiconvex and strictly quasiconvex on $\prod_j X_j$, thus local optima are global optima. Moreover, the search of (globally) optimal solutions can be reduced to solving a series of linear programs. Indeed, a simple calculation shows that, for $\beta > 0$ and $x \in \prod_j X_j$, the statement $f(x) < \beta$ is equivalent to the statement

$$0 < \min_{ij} \min\left\{x_j(a_{ij}^U + \beta) - x_i, x_i - x_j(a_{ij}^L - \beta)\right\}.$$

Hence, for $\delta > 0$ given, a δ-approximate optimal solution x will be obtained after solving (via a binary search), $O(\log(1/\delta))$ linear problems of the form

$$\begin{aligned} &\max\ t \\ &\text{s.t.} \quad t \leq x_j(a_{ij}^U + \beta) - x_i, \quad \forall i, j, \\ &\qquad t \leq x_i - x_j(a_{ij}^L - \beta), \quad \forall i, j, \\ &\qquad x_j \in X_j, \quad \forall j. \end{aligned}$$

Observe that this problem had been (erroneously) claimed to be "difficult" in [5, 11].

2.2 Arbitrary monotonic norms

It is not hard to obtain (convergent) bounds in f within intervals using basic tools of interval analysis [13]. Indeed, for intervals $S = [s^L, s^U], T = [t^L, t^U]$, it is easily

shown, by enumeration of all possible cases, that the range $\sharp\varepsilon(s,t) := \{\varepsilon(s,T) : s \in S\}$ is given by

$$\sharp\varepsilon(s,t) = [\alpha(s,t), \beta(s,t)]$$

with

$$\alpha(s,t) = \max\left\{t^L - s^U, s^L - t^U, 0\right\},$$
$$\beta(s,t) = \max\left\{s^U - t^U, t^L - s^L, 0\right\}.$$

Hence, by the monotonicity of γ, we have that

$$F((X_1,\ldots,X_n)) := \left[\gamma\left((\alpha(X_i/X_j, A_{ij}))_{i,j=1}^n\right), \gamma\left((\beta(X_i/X_j, A_{ij}))_{i,j=1}^n\right)\right] \tag{14}$$

is an inclusion function for f. Moreover, since all norms are equivalent, it turns out that F exhibits first-order convergence in the domain of (13). Hence, a branch and bound algorithm using $\gamma\left((\alpha(X_i/X_j, A_{ij}))_{i,j=1}^n\right)$ as lower bound of f on intervals is convergent.

3 Numerical experiments

A series of tests has been performed to check the feasibility of interval branch-and-bound methods to solve (8) and to explore how perturbations in model (1) affect accuracies and computing times. For a fixed dimension n, we generate a weight vector $x = (x_1,\ldots,x_n)$ randomly, with components independently and identically distributed as a uniform variable in the set $\{1,2,\ldots,9\}$; for each pair $i,j, i<j$, a perturbation factor ξ_{ij} is then randomly generated following a uniform distribution in an interval $[1-c, 1+c]$ with $c \in \mathbb{R}_+$, and the interval pairwise comparison matrix in model (4) is defined with

$$A_{ij} = \begin{cases} [(1-r)\frac{x_i}{x_j}\xi_{ij}, (1+r)\frac{x_i}{x_j}\xi_{ij}], & \text{if } i<j, \\ \{1\}, & \text{if } i = \text{j}, \\ 1/A_{ji}, & \text{if } i>j, \end{cases} \tag{15}$$

where r is a fixed positive real number which denotes the wished radius for the intervals A_{ij}, $i<j$: i.e. $2r$ is the width of the intervals A_{ij} for $i<j$.

For each size n and constant c delimiting the perturbations ξ_{ij} in the interval $[1-c, 1+c]$, 50 interval pairwise matrices with different widths (defined by their radius r) were generated according to (15). The analysis has been performed using as γ the Euclidean norm; exactly the same methodology can be used for other norms or other objective functions. Mean, best and worst-case results are shown in Tables 1–3.

The used algorithm derives from a standard interval branch-and-bound algorithm of Moore–Skelboe type, where the midpoint test is added (this generates the current solution named $\tilde{f}$ in Algorithm 1), [15]. These kinds of exact global optimization methods work by bisecting the initial domain into smaller and smaller subboxes. Thus, a list of boxes (named $\mathcal{L}$ in Algorithm 1) is generated; boxes are discarded as soon as it is shown that they cannot contain a global optimum (computing a lower bound of a function over a box using interval analysis [13] it can be proved that the best solution in the box cannot be superior to the current one $\tilde{f}$). The stopping criterion for a standard interval branch-and-bound algorithm is generally the accuracy reached between the current minimum $\tilde{f}$ and the lowest lower bound remaining in $\mathcal{L}$ ($\tilde{f} - \min_{(z,Z)\in\mathcal{L}} z < e$,

where e is the wished precision). The bounds are computed by using the inclusion function defined by Eq. (14) and using the definition of interval arithmetic, [13]. For all our numerical tests $(2, 250)$, it was not possible to fix the same accuracy for all of them, because the values of the objective functions vary from 0 up to 100 depending on the size and of the perturbation of the problem (parameters n, r, c). Thus, the algorithm was modified in order to compute the efficient accuracy which can be reached by the optimal solutions (within a fixed number of iterations $\mathrm{NB}_{\max}$). The parameter co represents the step by which the precision e is decreased during the iterations of the algorithm (it could be fixed to 0.0001). The maximal number of iterations $\mathrm{NB}_{\max}$ is fixed to $200,000$ in Algorithm 1 in order to solve quickly all the numerical tests presented in Tables 1–3. Therefore one has to discuss about the precision reached during this imposed time. This method is detailed as follows:

Algorithm 1

- Set $X :=$ the initial domain in which the global minimum is searched, $X \subseteq \mathbb{R}^n$.
- Set $\tilde{f} := +\infty$ (the current minimum).
- Set $\mathcal{L} := (+\infty, X)$ (the lower bound, the box).
- Set $e := 10$ (the wished accuracy).
- Set $co := 1$.
- Set $NB_{\max} := 200000$.
- Set $k := 1$.
- While $k \leq NB_{\max}$ and $e \geq 10^{-4}$ Do
  (1) Extract from $\mathcal{L}$ the box which has the lowest lower bound.
  (2) Bisect the considered box by the middle of its edge which has the maximal length, yielding V_1, V_2.
  (3) For m:=1 to 2 do
  * Compute $v_m := \gamma\left(\alpha\left(\frac{(V_m)_i}{(V_m)_j}, A_{ij}\right)_{ij}\right)$ the lower bound of f on V_m, by using (14).
  * if $v_m < \tilde{f}$ (only one global solution is sought) then
    . Insert (v_m, V_m) in $\mathcal{L}$.
    . Set $\tilde{f} := \min(\tilde{f}, f(mid(V_m)))$, where $mid(V_m)$ is the midpoint of V_m.
    . If $\tilde{f}$ has changed then remove from $\mathcal{L}$ all (z, Z) where $z \geq \tilde{f}$, and set $\tilde{y} := mid(V_m)$.
  (4) if $\tilde{f} - \min_{(z,Z)\in\mathcal{L}} z < e$ then
  * if $co \geq e$ then $co := co/10$.
  * $e := e - co$.
  * Compute the CPU-time in seconds until this instruction.
  (5) $k := k + 1$.
  End Do
- Results: $\tilde{f}, \tilde{y}, e, k, time$.

The results given at the end of the algorithm, are the global minimum value $\tilde{f}$, only one solution $\tilde{y}$ and the reached accuracy e, the CPU-time $time$, the number of iterations

Table 1 Mean time and mean precision for 50 problems

Pb size	$n=5$		$n=6$		$n=7$		$n=8$		$n=9$		$n=10$	
r,c	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec
0, 0.1	551.83	4.09×10^{-3}	598.08	1.82×10^{-2}	521.24	7.19×10^{-2}	409.94	1.48×10^{-1}	304.73	2.28×10^{-1}	211.56	4.55×10^{-1}
0, 0.2	573.74	1.67×10^{-2}	471.04	1.07×10^{-1}	415.63	3.32×10^{-1}	324.38	8.10×10^{-1}	313.54	1.18×10^{0}	327.89	1.67×10^{0}
0, 0.3	585.75	2.23×10^{-2}	486.93	2.28×10^{-1}	451.43	6.17×10^{-1}	473.96	1.41×10^{0}	439.37	1.89×10^{0}	188.35	2.91×10^{0}
0.05,0.1	21.62	1.08×10^{-4}	63.33	2.31×10^{-4}	151.98	5.39×10^{-4}	203.64	1.56×10^{-3}	304.38	2.83×10^{-3}	228.84	3.46×10^{-3}
0.05, 0.2	463.63	2.34×10^{-3}	571.14	1.22×10^{-2}	406.58	5.27×10^{-2}	414.92	1.07×10^{-1}	292.56	2.17×10^{-1}	294.40	2.88×10^{-1}
0.05, 0.3	547.99	8.17×10^{-3}	451.54	4.78×10^{-2}	475.70	2.17×10^{-1}	422.24	3.59×10^{-1}	243.99	8.16×10^{-1}	295.19	1.11×10^{0}
0.1, 0.1	0.03	1.00×10^{-4}	1.37	1.00×10^{-4}	0.44	1.00×10^{-4}	4.60	1.43×10^{-4}	21.42	1.84×10^{-4}	11.32	1.05×10^{-4}
0.1, 0.2	0.85	1.00×10^{-4}	51.34	2.15×10^{-4}	124.24	8.86×10^{-4}	51.34	2.15×10^{-4}	153.29	2.36×10^{-3}	158.38	6.18×10^{-3}
0.1, 0.3	315.51	1.63×10^{-3}	485.98	1.14×10^{-2}	286.86	2.63×10^{-2}	426.37	5.56×10^{-2}	298.92	1.33×10^{-1}	256.64	1.92×10^{-1}

Table 2 Best time and best precision for 50 problems

Pb size	$n=5$		$n=6$		$n=7$		$n=8$		$n=9$		$n=10$	
r,c	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec
0, 0.1	71.63	1.20×10^{-4}	38.00	4.00×10^{-3}	0.40	3.00×10^{-3}	3.36	3.00×10^{-2}	2.30	5.00×10^{-2}	0.19	1.00×10^{-1}
0, 0.2	66.74	5.00×10^{-4}	6.81	5.00×10^{-3}	1.58	4.00×10^{-2}	0.21	9.00×10^{-2}	0.45	2.00×10^{-1}	1.19	3.00×10^{-1}
0, 0.3	80.83	7.00×10^{-4}	9.97	8.00×10^{-3}	4.52	7.00×10^{-2}	2.93	3.00×10^{-1}	2.94	4.00×10^{-1}	0.80	5.00×10^{-1}
0.05, 0.1	0.01	1.00×10^{-4}	0.02	1.00×10^{-4}	0.05	1.00×10^{-4}	0.10	1.00×10^{-4}	0.27	1.00×10^{-4}	0.37	1.00×10^{-4}
0.05, 0.2	0.02	1.00×10^{-4}	50.51	1.00×10^{-4}	4.62	1.00×10^{-3}	13.30	7.00×10^{-3}	0.50	2.00×10^{-2}	2.41	5.00×10^{-2}
0.05, 0.3	168.80	2.00×10^{-3}	1.86	1.00×10^{-4}	60.97	2.00×10^{-3}	6.79	2.00×10^{-2}	7.61	2.00×10^{-2}	4.00	1.00×10^{-1}
0.1, 0.1	0.01	1.00×10^{-4}	0.02	1.00×10^{-4}	0.03	1.00×10^{-4}	0.05	1.00×10^{-4}	0.07	1.00×10^{-4}	0.09	1.00×10^{-4}
0.1, 0.2	0.02	1.00×10^{-4}	0.02	1.00×10^{-4}	0.04	1.00×10^{-4}	0.07	1.00×10^{-4}	0.51	1.00×10^{-4}	0.64	1.00×10^{-4}
0.1, 0.3	0.02	1.00×10^{-4}	0.09	1.00×10^{-4}	0.04	1.00×10^{-4}	5.19	2.00×10^{-3}	0.37	4.00×10^{-3}	0.74	2.00×10^{-2}

Table 3 Worst time and worst precision for 50 problems

Pb size	$n = 5$		$n = 6$		$n = 7$		$n = 8$		$n = 9$		$n = 10$	
r, c	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec	time (s)	Prec
0, 0.1	1504.04	2.00×10^{-2}	1102.12	1.00×10^{-1}	1167.66	4.00×10^{-1}	1477.52	5.00×10^{-1}	1233.16	6.00×10^{-1}	1015.84	2.00×10^{0}
0, 0.2	1110.23	2.00×10^{-1}	1135.60	4.00×10^{-1}	1115.34	2.00×10^{0}	1468.28	3.00×10^{0}	1102.48	4.00×10^{0}	1231.78	3.00×10^{0}
0, 0.3	1081.01	9.00×10^{-2}	1030.16	2.00×10^{0}	1162.78	2.00×10^{0}	1204.36	5.00×10^{0}	1219.34	5.00×10^{0}	1063.33	8.00×10^{0}
0.05, 0.1	1034.59	5.00×10^{-4}	991.14	3.00×10^{-3}	1073.25	6.00×10^{-3}	1113.44	2.00×10^{-2}	1007.06	2.00×10^{-2}	926.07	3.00×10^{-2}
0.05, 0.2	1048.06	3.00×10^{-2}	1095.47	9.00×10^{-2}	1114.07	3.00×10^{-1}	1284.31	7.00×10^{-1}	1112.54	1.00×10^{0}	944.01	1.00×10^{0}
0.05, 0.3	1077.03	6.00×10^{-2}	1140.27	4.00×10^{-1}	1135.59	2.00×10^{0}	1208.64	2.00×10^{0}	1099.78	3.00×10^{0}	1126.40	3.00×10^{0}
0.1, 0.1	0.18	1.00×10^{-4}	4.93	1.00×10^{-4}	8.01	1.00×10^{-4}	182.71	2.00×10^{-3}	617.16	3.00×10^{-3}	349.49	3.00×10^{-4}
0.1, 0.2	39.13	1.00×10^{-4}	613.89	3.00×10^{-3}	855.05	2.00×10^{-2}	1160.54	2.00×10^{-2}	875.15	6.00×10^{-2}	839.19	6.00×10^{-2}
0.1, 0.3	1485.63	3.00×10^{-2}	1131.15	2.00×10^{-1}	905.77	2.00×10^{-1}	1289.27	3.00×10^{-1}	1070.57	7.00×10^{-1}	1043.63	6.00×10^{-1}

k needed to find this solution. Hence, this algorithm allows us to determine a global solution, up to accuracy e, within an imposed limited number of iterations $\mathrm{NB}_{\max}$.

In order to compare the results in the following tables, one must first compare the accuracy e and then, for a same accuracy, the CPU-time needed.

All the results are performed on a classical modern PC-computer (AMD Athlon 1.1 GHz processor with 256 MB of RAM) using a Fortran 90 language.

In the following tables of numerical results Table 1–3, one can notice that the three parameters n, r and c induce different effects. For the parameter n, it is well known that when the dimension of a global optimization problem increases, the performance of exact algorithms often deteriorates extremely quickly; this phenomenon is observed in every line of Table 1.

The particular problems addressed in this paper have another technical drawback relatively common in interval arithmetic methods, [13]: each variable appears many times in the objective function, affecting the quality of the lower bounds (14).

For parameter c, when it grows, the perturbation of the weight matrix A increases and then, the corresponding global optimization problem becomes more difficult to be solved. Considering the last parameter r, its growth introduces a simplification of the resolution of the associated global optimization problem. This positive effect seems to come from the fact that very good solutions are quickly obtained and then a lot of sub-boxes are directly discarded.

Admitting that 10^{-2} is a sufficient accuracy for solving these problems (in fact this depends on the values of the global optima, which increase with the perturbation of the problems), one can remark in Table 1 that the most important part of the considered problems with $5, 6, \ldots, 10$ variables can be solved with efficiency even when the size of the problem is $n = 10$. This result is emphasized by considering the worst case (see Table 3, which is not so far from the mean case). Table 2 shows that for every parameters $n \leq 10, c, r$, there exists a possibility that the resolution of the considered problem can be performed in a very efficient way. Remarking that when one considers problems with an interval matrix ($r > 0$) and not a degenerated one ($r = 0$), the corresponding global optimization problem is more efficiently solved. This means that it will be more interesting to formulate problems with a real degenerated matrix A, by a corresponding one with a little radius-perturbation $r = 0.05$ for example, yielding an interval matrix.

Nevertheless, some problems are not solved with a efficient precision (10^{-2} for this function) (see Table 3 when $r = 0, c = 0.3$ and $n = 6$ until 10). That is due to the fact that the values of the corresponding functions are not close to 0. Therefore, a precision 10^{-4} is easy to reach if the solution is close to 0, but very hard to perform in other cases.

4 Conclusion

In this article, one shows that standard interval branch-and-bound algorithms can be directly used to solve successfully the optimization problems appearing when one faces the problem of inferring weights from a pairwise comparison matrix, in which coefficients may also be intervals. A general formulation of this problem is proposed, and a global optimization algorithm is proposed.

Some particular instances of this problem (not including interval elements in the pairwise comparison matrices) were simply considered to be “difficult to solve”, [5], or heuristically solved without measuring the possible gap.

The numerical experiments show that some problems can efficiently be solved even if the size is rather important (10 for example) by considering the classical stopping criterion and with a precision about 1 or 2%.

Hence, as soon as one decides to infer weights by (8), interval analysis methods provide an effective tool to find the weight vector globally optimal for such cirterion for problems of moderate size.

Acknowledgement E. Carrizosa is supported by the Ministerio de Educación y Ciencia, Spain, MTM2005-09362-C03-01.

References

1. Barzilai, J.: Deriving weights from pairwise comparison matrices. J. Oper. Res. Soci. **48**, 1226–1232 (1997)
2. Blanquero, R., Carrizosa, E., Conde, E.: Inferring efficient weights from pairwise comparison matrices. Math. Methods Oper. Res. (2006) To appear
3. Bozóki, S.: A method for solving LSM problems of small size in the AHP. Cent. Eur. J. Oper. Res. **11**, 17–33 (2003)
4. Bozóki, S., Lewis, R.H.: Solving the Least Squares Method Problem in the AHP for 3×3 and 4×4 matrices. Cent. Eur. J. Oper. Res. **13**, 255–270 (2005)
5. Choo, E.U., Wedley, W.C.: A common framework for deriving preference values from pairwise comparison matrices. Comput. Oper. Res. **31**, 893–908 (2004)
6. Conde, E.: Mean utility in the assurance region model. Eur. J. Oper. Res. **140**, 93–103 (2002)
7. Fülöp, J.: Global Optimization methods for approximation by consistent matrices. Presented at the ISMP 2003, Copenhaguen
8. Gass, S., Forman, E.: The Analytic Hierarchy Process: An exposition. Oper. Res. **49**, 469–486 (2001)
9. Golany, B., Kress, M.: A multicriteria evaluation of methods for obtaining weights from ratio-scale matrices. Eur. J. Oper. Res. **69**, 210–220 (1993)
10. Haines, L.M.: A statistical approach to the analytic hierarchy process with interval judgements. (I). Distributions on feasible regions. J. Oper. Res. Soc. **110**, 112–125 (1998)
11. Lin, C.-C.: A revised framework for deriving preference values from pairwise comparison matrices. J. Oper. Res. Soc. (2006). To appear
12. Lootsma, F.A.: A model for the relative importance of the criteria in the multiplicative AHP and SMART. Eur. J. Oper. Res. **94**, 467–476 (1996)
13. Moore, R.E.: Interval Analysis. Prentice-Hall, Englewood Cliffs, NJ, (1966)
14. Ramanathan, R.: A note on the use of Goal Programming for the multiplicative AHP. J. Multi-Criteria Decis. Mak. **6**, 296–307 (1997)
15. Ratschek, H., Rokne, J.: New Computer Methods for Global Optimization. Ellis Horwood, West Sussex, UK (1988)
16. Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980)
17. Saaty, T.L.: How to make a decision: The analytic Hierarchy process. Interfaces **24**, 19–43 (1994)
18. Saaty, T.L.: Decision-making with the AHP: Why is the principal eigenvector necessary. Eur. J. Oper. Res. **145**, 85–91 (2003)
19. Salo, A., Hamalainen, R.P.: Preference programming through approximate ratio comparisons. Eur. J. Oper. Res. **82**, 458–475 (1995)
20. Wang, Y.-M., Elhag, T.M.S.: A Goal Programming method for obtaining interval weights from an interval comparison matrix. Eur. J. Oper. Res. (2006) to appear