Interactive Multiple Objective Decision Making based on Quadratic Programming

Frank Kriwaczek* **and Berç Rustem***

Departmental Technical Report: 2000/17
ISSN 1469-4174

December 2000

Abstract. In order to make a decision in the face of multiple objectives it is necessary to know the relative importance of the different objectives. Yet, it is often very difficult to specify a set of precise weights before possible alternatives solutions are known. In this paper, we present an interactive, iterative method for arriving at an acceptable solution. The decision maker gradually discerns what is achievable and adjusts his aspirations and (implicitly) the specification of weights and trade-offs between his objectives, in the light of what he learns. To aid the decision maker's cognition and to allow him to express his wishes in a natural way, we employ an intuitive interface, based on the parallel coordinates method of displaying and specifying points in multidimensional space.

Keywords: Goal programming, quadratic programming, compromise programming, multiple-objective decision making, interactive algorithms, parallel coordinates, visualisation.

1. Introduction

This paper describes an algorithm for helping a decision maker uncover his preference structure in a multiple-objective decision problem. As weightings and trade-offs between objectives become available explicitly, both to the decision maker and to the system, a solution can be found that is both feasible, satisfying any constraints that had been stipulated, and satisfactory, from the decision maker's point of view.

The setting for this solution method is constrained optimisation - simultaneously attaining multiple goals while maintaining feasibility. We assume that the decision maker is able to express what a perfect solution would be in the form of a vector of *bliss* values $\mathbf{x}^{\mathrm{d}} \in \Re^n$. The attainment of each component of the bliss point $\mathbf{x}^{\mathrm{d}}$ is a desired objective. We also assume that the set $\Im$ of feasible solutions is specified by:

$$\Im \equiv \{\mathbf{x} \in \Re^n \mid \mathbf{N}^{\mathrm{T}}\mathbf{x} = \mathbf{b}\}$$

where coefficient matrix $\mathbf{N} \in \Re^{n+m}$ has linearly independent columns and vector $\mathbf{b} \in \Re^m$ is the corresponding right-hand side of the model equations.

* Department of Computing
Imperial College of Science, Technology and Medicine
180 Queen's Gate
London SW7 2BZ
England

Email: br@doc.ic.ac.uk and frk@doc.ic.ac.uk

Clearly, if the bliss point is feasible, i.e., if $\mathbf{x}^{\mathrm{d}} \in \Im$, then the best decision would be to adopt $\mathbf{x}^{\mathrm{d}}$. The multiple-objective decision problem arises when $\mathbf{x}^{\mathrm{d}} \notin \Im$, and we need to consider decisions $\mathbf{x} \in \Im$ that are close to $\mathbf{x}^{\mathrm{d}}$.

Methods which employ measures of distance based on the weighted sum of absolute deviations along each dimension or the maximum weighted absolute deviation along the dimensions have linear programming formulations and have been treated elsewhere (Hillier and Lieberman, 1996; Pinto, 1995). We shall take the norm (or distance measure) to be the quadratic form:

$$\left\|\mathbf{x}-\mathbf{x}^{\mathrm{d}}\right\|_{\mathbf{Q}}^{2} = (\mathbf{x}-\mathbf{x}^{\mathrm{d}})^{\mathrm{T}}\mathbf{Q}(\mathbf{x}-\mathbf{x}^{\mathrm{d}})$$

where $\mathbf{Q} \in \Re^{n\times n}$ is a symmetric positive semi-definite weighting matrix. The diagonal elements of $\mathbf{Q}$ signify the relative importance of $\mathbf{x}^{i}$ attaining $(\mathbf{x}^{\mathrm{d}})^{i}$, and the off-diagonal elements measure the trade-offs between the achievement of one objective versus another.

If the decision maker was absolutely sure about his weights and trade-offs, encapsulated in matrix $\mathbf{Q}$, then the decision problem would amount to finding the solution to the quadratic programming problem:

$$\min_{\mathbf{x}}\{\left\|\mathbf{x}-\mathbf{x}^{\mathrm{d}}\right\|_{\mathbf{Q}}^{2} \mid \mathbf{x} \in \Im\} \tag{1}$$

In practice, the true value of $\mathbf{Q}$ will probably not be known and must be estimated, in the simplest case by the unit matrix $\mathbf{I}^{n}$. Because the *estimate* of $\mathbf{Q}$ is unlikely to reflect the real feelings of the decision maker accurately, a solution to (1) will very likely not be acceptable to him. However, given $\mathbf{x}^{\mathrm{d}}$ and $\Im$, the nearness of the solution of (1) to $\mathbf{x}^{\mathrm{d}}$ will depend only on $\mathbf{Q}$**.** So the only way of producing a solution (1) that is acceptable is to re-specify $\mathbf{Q}$.

Let the set $\Omega \subset \Re^{n}$ denote the set of solutions $\mathbf{x}$ acceptable or *admissible* to the decision maker. Even though Ω will not be known to the decision maker explicitly, but will exist only in an unexplored state in his mind, we still assume that $\Omega \cap \Im$ is a non-empty convex set.

In the algorithm described in this paper the decision maker does not re-specify $\mathbf{Q}$ directly, but rather proposes a solution point $\mathbf{x}_{\mathrm{p}}$ that is acceptable (but not necessarily feasible) and preferred to the latest optimal feasible solution $\mathbf{x}_{k}$ found by the system. The manner in which $\mathbf{x}_{\mathrm{p}}$ differs from $\mathbf{x}_{k}$ is given by the displacement or *correction vector:*

$$\boldsymbol{\delta}_{k} = \mathbf{x}_{\mathrm{p}} - \mathbf{x}_{k}$$

In a sense, vector $\boldsymbol{\delta}_{k}$ expresses the direction in which the decision maker wishes the system to search for the next solution, in the light of what has been achieved so far. The system does this by computing a new and refined weighting matrix based on the old value $\mathbf{Q}_{k}$ together with $\boldsymbol{\delta}_{k}$. It then solves the quadratic programming problem (1) with the revised version of $\mathbf{Q}$ and presents a new solution $\mathbf{x}_{k+1}$ to the user. If $\mathbf{x}_{k+1}$ is satisfactory then a feasible, acceptable solution has been found and the algorithm terminates. Otherwise, the process is repeated through another iteration.

2. The Algorithm in Detail

Step 0: Start with

- the bliss point $\mathbf{x}^{\mathrm{d}}$,
- an initial positive, semi-definite weighting matrix $\mathbf{Q}_0$, and
- the set of linear constraints.

Set k = 0.

Step 1: Compute the solution to the quadratic programming problem

$$\mathbf{x}_k = \arg\min\{(\mathbf{x}-\mathbf{x}^{\mathrm{d}})^{\mathrm{T}}\mathbf{Q}_k(\mathbf{x}-\mathbf{x}^{\mathrm{d}}) \mid \mathbf{N}^{\mathrm{T}}\mathbf{x} = \mathbf{b}\}$$

Step 2: Interact with the decision maker.
If $\mathbf{x}_k$ is acceptable to the decision maker then

Stop.

else

Elicit a preferred point $\mathbf{x}_{\mathrm{p}}$ from the decision maker.

Calculate displacement or correction vector $\boldsymbol{\delta}_k = \mathbf{x}_{\mathrm{p}} - \mathbf{x}_k$.

Step 3: Choose a positive scalar value η_k to reflect the degree to which the weighting matrix should be modified during this iteration. (If in doubt, set η_k equal to 1.)
Calculate a new weighting matrix

$$\mathbf{Q}_{k+1} = \mathbf{Q}_k + \eta_k \frac{\mathbf{Q}_k\boldsymbol{\delta}_k\boldsymbol{\delta}_k^{\mathrm{T}}\mathbf{Q}_k}{\boldsymbol{\delta}_k^{\mathrm{T}}\mathbf{Q}_k\boldsymbol{\delta}_k}$$

To ensure that the denominator $\boldsymbol{\delta}_k^{\mathrm{T}}\mathbf{Q}_k\boldsymbol{\delta}_k$ is non-zero, make a small change to $\boldsymbol{\delta}_k$, if necessary.
Set k = k+1
Go to **Step 1**.

As is shown in (Rustem, 1998, chapter 3), $\mathbf{Q}_{k+1}$ will be positive (semi-) definite, provided $\mathbf{Q}_k$ is.

It is also shown in (Rustem, 1998, chapter 3) that the iterates $\{\mathbf{x}_k\}$ generally obey the wishes of the decision maker, in that $\mathbf{x}_{k+1} - \mathbf{x}_k$ is the feasible alternative closest to the specified direction $\boldsymbol{\delta}_k$. The optimality of each iterate $\mathbf{x}_k$ in step 1 is also desirable, in order to maintain the multi-objective optimisation framework throughout, which underlies the evaluation of $\mathbf{x}_k$.

3. Parallel Coordinates

The multiple objective decision method described in this paper involves the decision maker specifying a bliss solution $\mathbf{x}^{\mathrm{d}}$, and iteratively a sequence of acceptable (but not necessarily feasible) solutions $\{\mathbf{x}_{\mathrm{p}}\}$ in response to feasible solutions $\{\mathbf{x}_k\}$ produced by the system. All of these solutions are points in the n-dimensional real space $\Re^n$. The decision maker gradually discerns what is feasible and adjusts the specification of what he finds acceptable, accordingly.

For the method to work successfully the decision maker must be able:

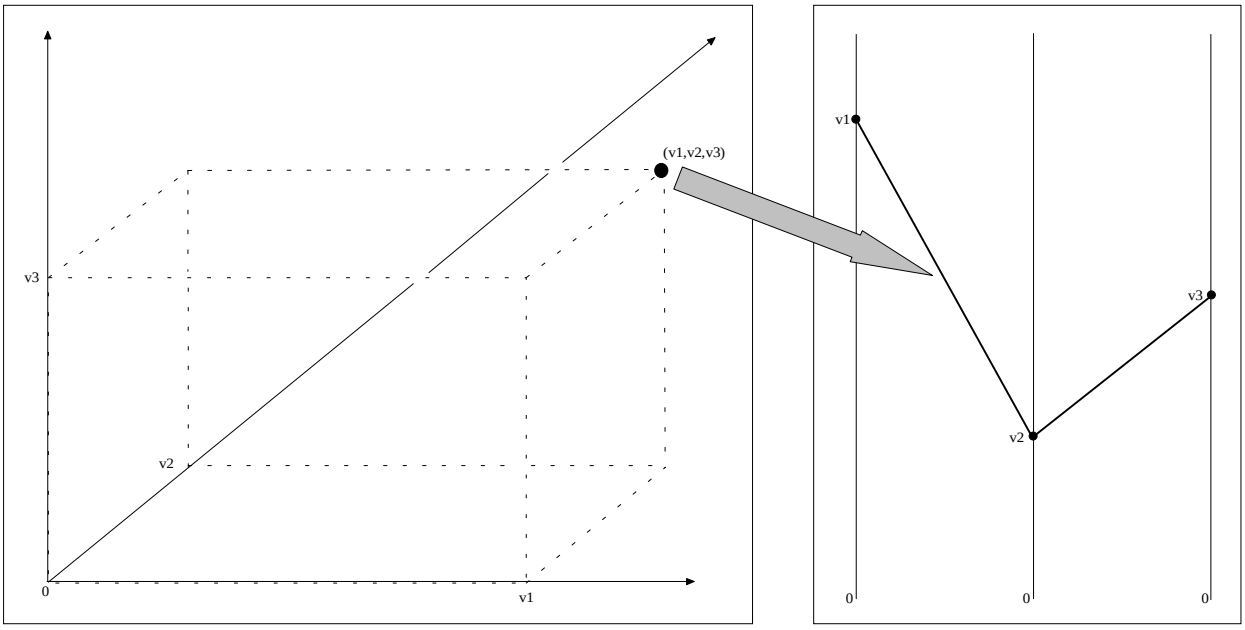
- to see clearly how close his desires are from what can be achieved,
- to specify accurately, yet in a natural way, what new compromise he is willing to make.

So, what is needed is an interface that:

- can display several points in n-dimensional space, showing clearly where and by how much they differ from each other,
- allows the user to adjust the position of such points as a way of expressing what is acceptable and the direction towards which the system should search for the next solution.

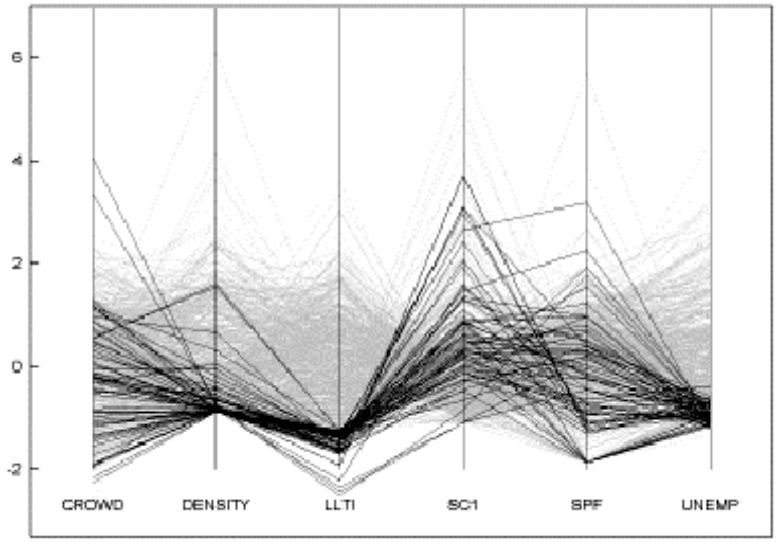
Although there are many ways of representing points in two, three or even four dimensions, it is not so easy to display points in n-dimensional space using a 2D display, where n may be 6, 7 or even higher. However, the parallel coordinates technique devised by Alfred Inselberg (Inselberg and Dimsdale, 1987). is able to do this, without any apparent limit on the number of dimensions.

In this approach, a point in n-dimensional space is represented as a polygonal line - a series of n-1 line segments in 2-dimensional space. So, the point with Cartesian coordinates $(v_1, v_2, \ldots, v_n)$ would, on a parallel coordinate display be represented by the poly-line comprising the n-1 line segments connecting the points $(1, v_1), (2, v_2), \ldots, (n, v_n)$.



Parallel coordinates can also be used when there is only a discrete set of values along a dimension, and, indeed, have met their most frequent application in the display of records in multi-dimensional databases, where dimensions might be product, sales area, customer location, and so on.

Parallel coordinate displays are a popular visualisation tool in data mining. Outliers are easy to observe and it is possible to spot trends, correlations and clustering of the data points (i.e., the database records) by detecting patterns between the poly-lines. When a large number of data points are represented simultaneously the results can be confusing, although some systems allow you to highlight subsets of points in an instructive fashion:



In our system, though, we normally display at most four data points at a time and so the danger of visual overload (i.e., clutter) is avoided.

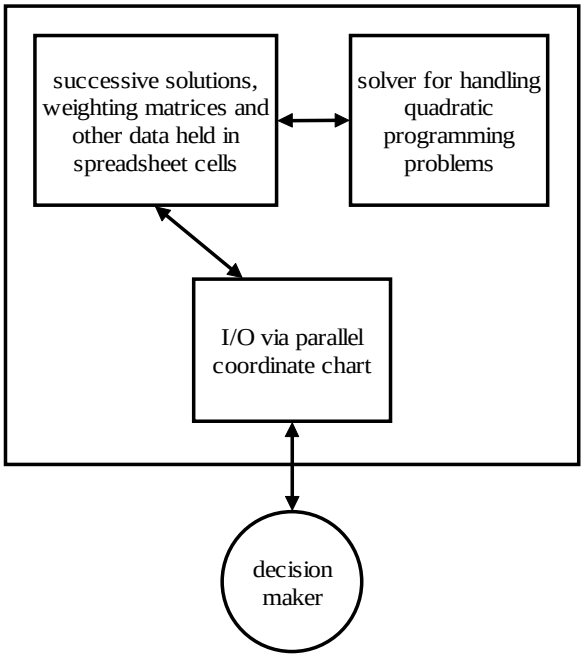
The conventional Cartesian representation of points in $\Re^n$ may well be helpful when considering the mathematics underlying our algorithm, but the decision maker is likely to gain greater insight into his problem from the parallel coordinate representation, in which the various components of a solution are laid out side-by-side.

4. The System in Action

Our interactive multiple objective decision making system has been implemented in Microsoft Excel. Excel has the virtues of:

- a built-in solver that can handle quadratic programming as well as linear programming problems,
- a very flexible graphing system that can produce pretty much any kind of chart and allows for items on the graph to be manipulated, thus modifying the corresponding underlying data items,
- the facility for positioning buttons, sliders and other input devices, and a simple, yet reasonably powerful programming language, Visual Basic for Applications, for harnessing and automating Excel's functionality - allowing useful custom tools to be prototyped with ease.

The simple architecture of the system is shown in the following diagram:

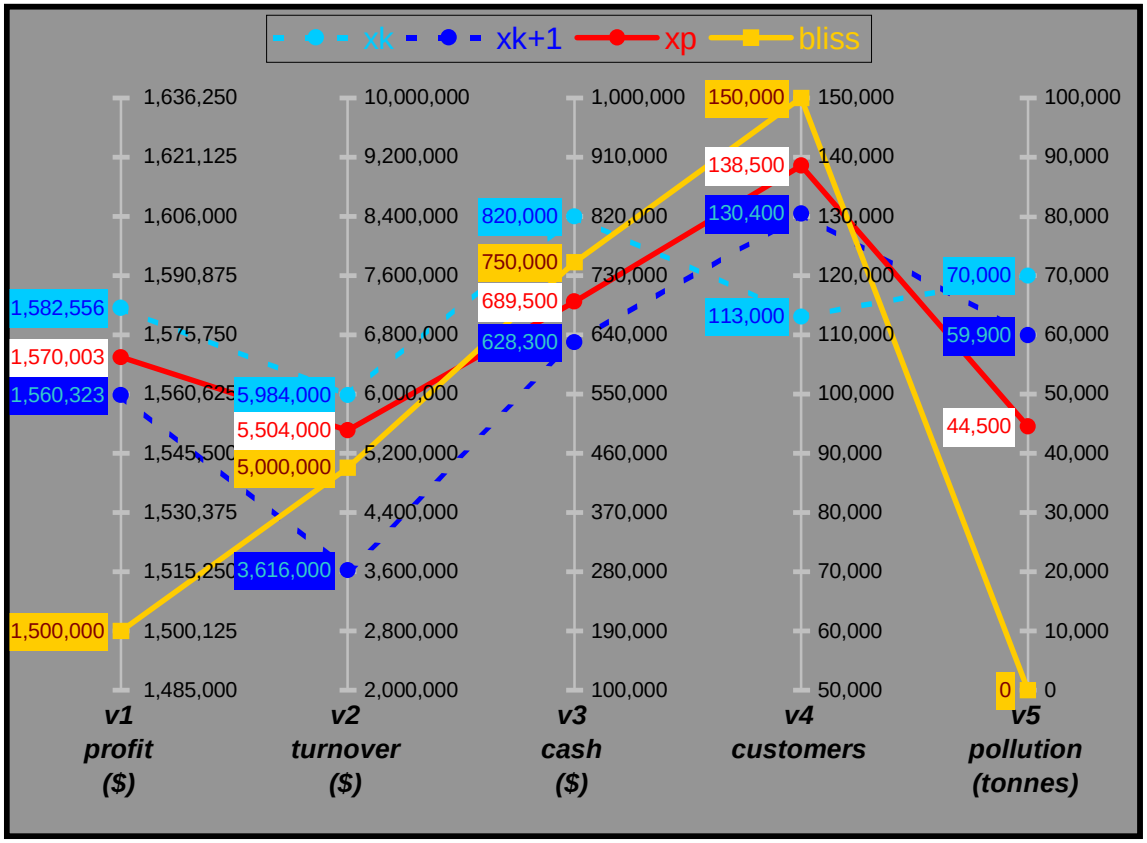


The parallel coordinate chart is implemented as a single X-Y scatter diagram with various series identified as the parallel axes and other series identified as the vertices of the polygonal lines which represent the different solution points.

The data underlying the parallel coordinate chart, including the solution points and the labels used for calibrating the parallel axes is held in worksheet cells. Algorithm step 3, which involves updating the weighting matrix, is carried out on worksheet, using Excel's built-in matrix functions, whilst step1 - finding an optimal solution to the quadratic programming problem - is carried out with the aid of Excel's solver, which is supplied by Frontline Systems and uses the Generalized Reduced Gradient (GRG2) nonlinear optimization algorithm of Leon Lasdon and Allan Waren (Fylstra, Lasdon, Watson and Waren, 1998).

The parallel coordinate chart acts as the user interface. It presents the user with the bliss point $\mathbf{x}^{\mathrm{d}}$ and current optimal feasible solution $\mathbf{x}_k$, in the form of poly-lines. The decision maker then drags vertices on the $\mathbf{x}_k$ poly-line to new positions along the axis lines, in accordance with the components of his new preferred solution $\mathbf{x}_{\mathrm{p}}$.

The decision maker can now see three poly-lines, for $\mathbf{x}^{\mathrm{d}}$, $\mathbf{x}_k$ and $\mathbf{x}_{\mathrm{p}}$. Digital readouts adjacent to the vertices show coordinate values. When the decision maker is satisfied with his choice of $\mathbf{x}_{\mathrm{p}}$, he hits the "NEXT" button and a new feasible solution $\mathbf{x}_{k+1}$ is computed, virtually instantaneously. The parallel coordinate display now shows all four points: $\mathbf{x}^{\mathrm{d}}$, $\mathbf{x}_k$, $\mathbf{x}_{\mathrm{p}}$ and $\mathbf{x}_{k+1}$ on the chart and allows the decision maker to form an impression of how successful the system has been in searching in the indicated direction for an acceptable solution.



If $\mathbf{x}_{k+1}$ is still not acceptable, the decision maker hits the "NEXT" button again. The $\mathbf{x}_k$ and $\mathbf{x}_p$ lines are removed, the $\mathbf{x}_{k+1}$ line is relabelled $\mathbf{x}_k$ and the system is ready for the next iteration.

Because of the speed of the system and the very natural way that the components of consecutive solution points can be displayed and modified, the system provides a congenial laboratory for exploring and eventually reaching an acceptable compromise solution.

References

Fylstra, D, L. Lasdon, J. Watson and A. Waren (1998). Design and Use of the Microsoft Excel Solver. Interfaces Volume 28, Number 5.

Hillier, F.S. and G.J. Lieberman. (1996). Introduction to Operations Research, McGraw-Hill, New York.

Inselberg, A. and B. Dimsdale. (1987). Parallel Coordinates for Visualizing Multi-Dimensional Geometry. Proc. Computer Graphics Intl. Conf.

Pinto, R.L.V. (1995). A Logic-Based Modelling Language and Integer Programming Framework for Multicriteria Optimization. PhD Dissertation. Department of Computing. Imperial College.

Rustem, B. (1998). Algorithms for Nonlinear Programming and Multiple Objective Optimisation. John Wiley, Chichester.