

Chapter 3

Modeling Techniques for Complex Environmental Problems

*Marek Makowski*¹

Abstract

Mathematical models can be useful in decision-making processes whenever the amount of data and relations are too complex to be analyzed based solely on experience and/or intuition. Models, when properly developed and maintained, and equipped with proper tools for their analysis can integrate relevant knowledge available from various disciplines and sources. Most environmental decision problems are complex. However, some of them pose additional challenges owing to the large amount of data, the complex relations between variables, the characteristics of the resulting mathematical programming problems, and the requirements for comprehensive problem analyses. Such challenges call for applications of advanced techniques for model generation and analysis. Several of these techniques are outlined in this chapter and illustrated by the RAINS model, a large non-linear model, which has been used in international negotiations about the reduction of air pollution.

Keywords: Modeling paradigms, decision support systems, air quality, object-oriented programming, robustness, multicriteria model analysis, non-linear optimization, model management.

¹Reprinted from: *Natural Environment Management and Applied Systems Analysis*, M. Makowski and H. Nakayama (Eds), 2001, International Institute for Applied Systems Analysis, Laxenburg, Austria, ISBN 3-7045-0140-9.

3.1 Introduction

Most decision problems are no longer well-structured problems that are easy to solve by intuition or experience supported by relatively simple calculations. Even the same kind of problem that was once easy to define and solve, has now become much more complex because of the globalization of the economy, and a much greater awareness of its linkages with various environmental, social, and political issues. Modern decision makers (DMs) typically want to integrate knowledge quickly and reliably from these various areas of science and practice. Unfortunately, the culture, language, and tools developed to represent knowledge in the key areas (e.g., economy, engineering, finance, environment management, social and political sciences) are very different. Everyone who has ever worked on a team with researchers and practitioners having backgrounds in different areas knows this. Given the great heterogeneity of knowledge representations in various disciplines, and the fast-growing amount of knowledge in most areas, we need to find a way to integrate knowledge for decision support efficiently.

Rational decision making is becoming more and more difficult, despite the quick development of methodology for decision support and an even quicker development of computing hardware, networks, and software. Two commonly known observations support this statement:

- first, the complexity of problems for which decisions are made grows even faster;
- second, knowledge and experiences related to rational decision making develop rapidly but heterogeneously, therefore integration of various methodologies and tools is practically impossible.

A critical element of model-based decision support is a mathematical model, which represents data and relations that are too complex to be adequately analyzed based solely on experience and/or intuition of a DM or his/her advisors. Models, when properly developed and maintained, can represent not only a part of knowledge of a DM but also integrate relevant knowledge available from various disciplines and sources. Moreover, models, if properly analyzed, can help the DM to extend his/her knowledge and intuition. However, models can also mislead users by providing wrong or inadequate information. Such misinformation can result not only from flaws or mistakes in model specification and/or implementation, the data used, or unreliable elements of software, but also by misunderstandings between model users and developers about underlying assumptions, limitations of applied methods of model analysis, and differences in interpretation of results, to name a few. Therefore, the quality of the entire modeling cycle determines to a large extent the quality of the decision-making process for any complex decision problem.

A recent comprehensive overview of model-based decision support methodologies, tools, and environmental applications is provided in Wierzbicki *et al.* (2000). The monograph² also contains a detailed discussion on the modern decision making process, and on guidelines for model development and analysis, focusing mainly on multicriteria model analysis (MCMA).

This chapter concentrates on an overview of modeling paradigms and techniques applicable to complex models and illustrates them by the RAINS model. The structure of the chapter is as follows. The RAINS model is outlined in Section 3.2, which is followed by a discussion of modeling problems and applied techniques in Section 3.3. Section 3.4 presents an overview of MCMA methods.

3.2 Outline of the RAINS Model

In many parts of Europe the indicators of critical levels of air pollution are exceeded and measures to improve air quality in these areas are needed to protect the relevant ecosystems. Several international agreements have been reached over the last decade in Europe to reduce emissions. For several years, the Transboundary Air Pollution (TAP) Project³ at IIASA has been developing models that have been used for supporting international negotiations. The models help to identify cost-effective measures aimed at reducing various measures of ground-level ozone concentrations, acidification, and eutrophication at several hundred receptors over Europe. These measures correspond to policies for reducing emissions of NH₃ (ammonia), SO_x (sulphur oxides), NO_x (nitrogen oxides), and VOC (volatile organic compounds) by various economic sectors in European countries.

The structure of the RAINS model is outlined in *Figure 3.1*. The decision variables are composed of the levels of emissions NH₃, SO_x, NO_x, and VOC by various sectors in each country, which imply the corresponding emission control policies. For each country and type of emission, a cost function is defined. Such a function relates the emission level with the corresponding costs of reducing to this level a sum of various types of emissions caused by activities aggregated (for the purpose of this analysis) for each country in several sectors. Therefore, cost-effective measures can be calculated by minimizing the cost function that corresponds to the sum of costs related to reductions of all types of considered emissions in all sectors in each country. In order to determine the corresponding environmental impact, emission levels are used as inputs to the three dispersion submodels and to the ozone formation submodel. Studies of the impact of ozone, acidification, and eutrophication have resulted in the establishment of critical levels for various air-quality indicators in order to protect agricultural crops and forests. These are

²<http://www.iiasa.ac.at/~marek/pubs>

³<http://www.iiasa.ac.at/Research/TAP>

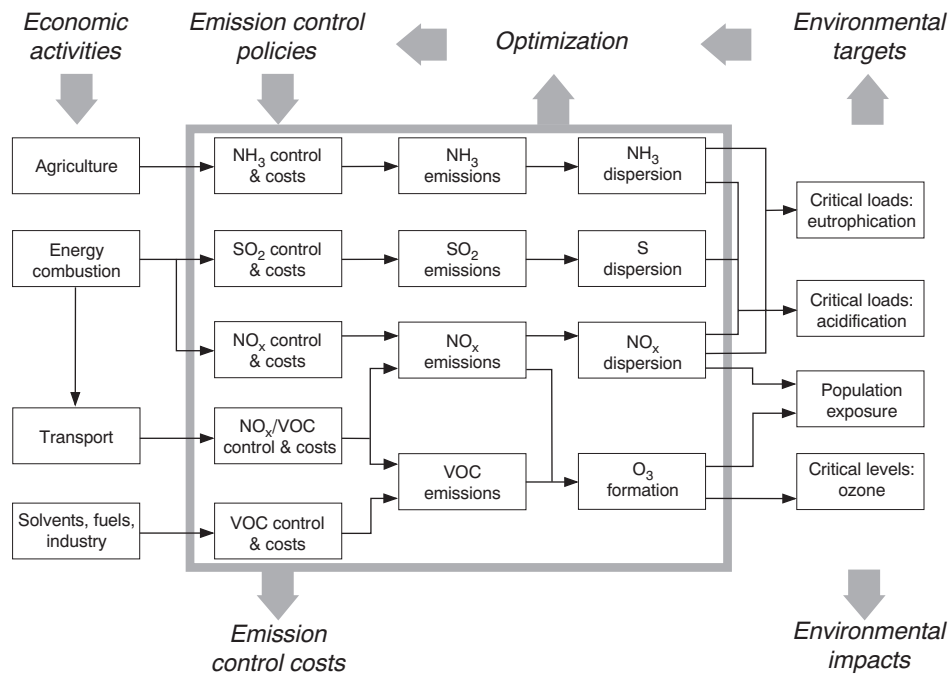


Figure 3.1. RAINS model structure.

determined using a long-term exposure measure, called the *accumulated excess*. Consequently, nine such exposure indices (six for ozone, two for acidification and one for eutrophication) have been defined for each of approximately 600 grids in Europe (also called *receptors*), and accumulated excess PWL (piece-wise linear functions) are defined for each grid and for each type of acidification and eutrophication excess. These indices are used to assess environmental effects of the applied emission control policies.

It is not only the structure of the RAINS model that is complex, but also the way in which it is used. In 1989, when the sulphur protocol was due for renegotiation, the United Nations Economic Commission for Europe (UN-ECE) accepted the RAINS model for use in the negotiations. Most probably, this was the first time when all parties to a major international negotiation accepted one computer model as a key tool in their negotiations. However, the acceptance of the model was just the beginning. The negotiators had to trust the model results and understand how the model works. The scientists had to understand the political realities and modify the model in order to respond better to the requests of the negotiators. In order to illustrate just one element of this process, let's consider an interpretation of the optimality of a solution. From the scientific perspective, a rational optimality criterion

is a minimization of the sum of costs of emission reductions subject to constraints on values of the air quality indices. However, this obviously results in solutions that would oblige some countries and/or industries to make larger emission reductions than others (which also implies substantial costs). Acceptance of such a solution would certainly distort competition; therefore, negotiators cannot accept such solutions. On the other hand, the RAINS model clearly demonstrates that uniform reductions (which are a sound idea from a political point of view) would not only be much more expensive but also would not result in achieving the desired air quality. Another example of this mutual learning process undertaken by the negotiators and scientists is illustrated by the evolution of the understanding of what the desired air quality should be. For example, the results of extensive research have shown that the critical acid loads should vary substantially between various ecosystems. Therefore, there is no justification to apply uniform environmental requirements for all grids in Europe.

In mathematical programming terms, RAINS is a large (about 30,000 variables and over 30,000 constraints), non-linear model. The original RAINS model described in Alcamo *et al.* (1990), which was a small linear programming (LP) model that dealt only with acidification, can be considered as a small pilot prototype of the current version of RAINS described in this chapter. The development of several versions of RAINS, made over ten years, was driven by the needs of the negotiators. The first version of RAINS was used for negotiating the sulphur protocol; therefore, it dealt only with a single pollutant. However, it has become clear that a multi-pollutant, multi-effect approach offers substantial environmental and financial advantages. Therefore, to respond to these needs, RAINS has been extended and gradually modified to its current form. A description of the current version of RAINS and of its use can be found in Schöpp *et al.* (1999), while a more formal model specification and a more detailed discussion of applied modeling paradigms is provided by Makowski (2000).

3.3 Modeling Problems and Techniques

Modeling of any complex problem is composed of the following mutually linked activities:

- model specification,
- data handling and model generation,
- model analysis.

These issues will be discussed in the following subsections.

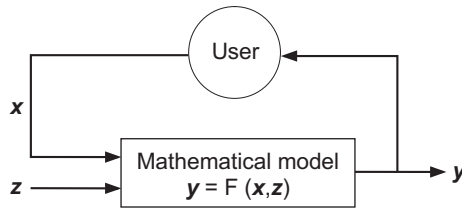


Figure 3.2. A mathematical model represents relations between decisions (inputs) x , external decisions (inputs not controlled by the user) z , and consequences (outcomes) y .

3.3.1 Model specification

Mathematical models are widely used in many areas of science and industry for predicting the behavior of a system under particular circumstances, when it is undesirable or impossible to experiment with the system itself. The understanding of the system gained through a comprehensive examination of its model can greatly help in finding decisions, the implementation of which will result in a desired behavior of the system. Therefore, a model used for decision support is focused on the basic function of a DSS (Decision Support System), namely, to provide an evaluation of consequences that will result from an implementation of given decisions.

All four of the basic concepts illustrated in *Figure 3.2*, namely, decision variables, external decisions, outcome variables, and a mathematical model are briefly discussed in the following subsections.

Decision Variables

In model-based decision support it is assumed that decisions have quantitative characters and therefore can be represented by a set of the model variables, hereinafter referred to as decisions⁴ $x \in E_x$, where E_x denotes a space of decisions. In a trivial case $x \in R$, which denotes that a decision is represented by a real number. However, in most cases x is a vector composed of various types of variables. For larger problems, the components of x are grouped in several subvectors. Let us illustrate this by specification of the decision variables of our illustrative model.

In the RAINS model the main decision variables are the annual emissions of the following four types of primary air pollutants from either sectors or countries:

- n_{is} , annual emission of NO_x from sector is ;
- v_{is} , annual emission of nonmethane VOC from sector is ;

⁴For the sake of brevity we call decision variables simply decisions.

- a_i , annual emission of NH_3 from country i ; and
- s_i , annual emission of SO_2 from country i .

where vectors n_{i_s} and v_{i_s} are combined for each country in subvectors n_i and v_i , respectively.

Additionally, optional decision variables are considered for scenarios that allow for limited violations of air quality targets. For such scenarios, variables corresponding to each type of considered air quality target are defined for each receptor. Each variable represents a violation of a given environmental standard. Optionally, violations of targets can be balanced with surpluses (understood as the difference between a target and its corresponding actual concentration/deposition).

External Decisions

Figure 3.2 illustrates two types of inputs to the core model: (1) decision variables \mathbf{x} controlled by a user, and (2) external decisions denoted by \mathbf{z} . In practice, inputs \mathbf{z} may include representations of various quantities that substantially influence the values of outcomes \mathbf{y} but are not controlled by the user, for example:

- regulations or commitments on environmental standards for air or water quality management models;
- meteorological conditions assumed for modeling physical relations in environmental models, e.g., *average*, *wet*, *dry*, *worst* year data for a water model; or
- forecasts of changes in demand for services, e.g., in telecommunication or transportation models.

In the RAINS model the external decisions \mathbf{z} are represented by:

- values representing the environmental standards that define constraints for various indices (such as maximum concentrations of various water and air quality indicators, respectively); and
- the set of meteorological data used for calibration of a respective model.

While the external decisions are beyond the control of the user of a DSS, s/he typically wants to examine a range of scenarios with various representations of external decisions in order to find out not only a solution which will best respond to a most likely representation of external inputs \mathbf{z} , but also a solution that will be *robust*, i.e., will also be good for various compositions of \mathbf{z} that should be considered.

Outcome Variables

The consequences of implementing various decisions x are evaluated by values of outcome variables $y \in E_y$. In various fields of applications, outcome variables are named differently, e.g., outcomes, metrics, goals, objectives, performance indices, attributes.

In the RAINS model, one outcome variable represents the sum of costs of reductions of emissions; four sets of other outcome variables correspond to various indices of air quality. While the definition of the cost is rather simple, an appropriate definition of air quality indices is rather complex. Environmental effects caused by acid deposition, by excess nitrogen deposition (the latter defined for two types of critical loads), and by eutrophication are evaluated at each receptor by a PWL function that represents an accumulated excess over the threshold of the environmental long-term target. If optional violations of environmental standards are allowed, then a maximum (over a set of receptors in each country) violation of each type of air quality indicator is also considered as an output variable.

Objectives

Out of the set of outcome variables $y \in E_y$, a user selects a subset of variables conventionally called objectives $q \in E_q$, where E_q is a space of objectives. Quite often objectives are referred to as criteria, and in this chapter these two terms will be used interchangeably. Usually E_q is a subspace of E_y , that is, the DM selects several criteria q_i from the set of outcomes y_j . Sometimes also some of the decision variables x are used as criteria, but for the sake of consistency we assume that such a variable is simply represented by one of the outcomes y .

The difference between objectives and outcome variables is not strict, and is mainly determined by the preferences of users. It has been commonly observed that a human being typically prefers to deal with seven plus/minus two criteria at a time. However, a complex model usually has many outcome variables. While users of models typically concentrate analysis by specifying preferences for several selected objectives, values of all outcome variables are reported, and sometimes a modification of the selection of objectives is desired. Therefore, depending on the stage and type of model analysis, the selection of the set of objectives is modified.

A partial preordering in E_q is usually implied by the decision problem and has obvious interpretations, such as the minimization of costs competing with the minimization of pollution. However, a complete preordering in E_q cannot usually be given within the context of a mathematical programming model. In other words, it is easy to determine for each objective separately, which solution (represented by vectors x and q) is the best one. However, for conflicting objectives there are two sets of solutions:

- Pareto-optimal (often called efficient), i.e., a solution, for which there is no other solution for which at least one criterion has a better value while values of remaining criteria are the same or better;
- dominated, i.e., solutions that are not Pareto-optimal.

Obviously, a Pareto-optimal solution is preferred over any solution it dominates (assuming that the selected criteria correspond well to the preferential structure of a DM). However, a set of Pareto-optimal solutions (often called Pareto-set, or Pareto frontier) is typically composed of an infinite number of solutions, many of which are very different. Pareto-optimal solutions are not comparable in a mathematical programming sense, i.e., one can not formally decide which is better than another one.

However, DMs are able to express their own preferences for various efficient solutions. One of the basic functions of multiobjective decision support is to provide various ways in which a DM may specify his/her preferences. There is no reliable formal way for separating a specification of preferences from a process of learning from the model analysis. It is a commonly known fact that decision making is not a point event, even in situations where it is realistic to assume that the problem perception does not change during the decision-making process. Therefore, the possibility of using a DSS in a learning and adaptive mode is a critical feature.

Mathematical Model

As already illustrated in *Figure 3.2*, a mathematical model (further on also called a core model) is used for predicting the consequences of decisions x , which can be either proposed by a DM or computed by a DSS. The consequences are measured by values of outcome variables y . Therefore, a model can be represented by mapping $y = F(x, z)$, where $x \in E_x$, $z \in E_z$, and $y \in E_y$ are vectors of values of decisions, external decisions, and outcomes, respectively. For the sake of brevity we will assume further on that the external decisions z are given and represented as parameters of the mapping F .

The core model (often called also substantive model) should include only logical and physical relations that are necessary to adequately represent relations between inputs x and outputs y . In addition to inputs and outputs, a model contains various intermediate and parametric variables (balance and/or state variables, resources, external decisions), conventionally called auxiliary variables. In a typical complex model, the decision and outcome variables are a small fraction of all variables. Auxiliary variables are introduced for easing the model specification and handling, and are typically not interesting for an end-user of the model. However,

the way in which auxiliary variables are selected and defined has a critical impact on the model performance and reliability.

In other words, the core model is composed of decision, outcome, and auxiliary variables, and of relations (inequalities, equations, etc., conventionally called *constraints*) between these variables that indirectly determine the sets of admissible (feasible) decisions and the corresponding solutions. Some of the constraints may reflect the logic of handling events represented by variables. For example, the model known as RWQM (Regional Water Quality Model) (Makowski and Somlyódy, 2000) has the constraint:

$$\sum_{k \in K(j)} x_{jk} = 1 \quad x_{jk} \in \{0, 1\}, \quad j \in E \quad (3.1)$$

where $K(j)$ is the set of technologies considered for emission node j , and E is the set of nodes where emissions occur. This condition assures that exactly one technology (represented by the corresponding binary variable x_{jk}) is selected in each waste water treatment plant.

Generally, the core model implicitly defines a set of feasible decisions $X_0 \subseteq E_x$. In other words, x is feasible, if and only if $x \in X_0$. The set X_0 is composed of all vectors x that fulfill all constraints representing all logical and physical relations among all the variables used in the model. Since every actual (and properly defined) decision problem has at least two solutions, X_0 is not empty.

A reader familiar with mathematical programming may be surprised, that such a model does not contain any goal function. This is done on purpose, and it is the recommended way of implementing any model-based DSS. We shall explain now, why the core model should not contain any representation of a preferential structure of a DM.

It is usually not possible to specify uniquely a model that can yield a unique solution reflecting the preferences of a DM. For example, very often it is practically impossible (even for a good analyst or an experienced DM) to specify values for a group of constraints that would determine a solution that corresponds well to preferences of a DM. In order to illustrate this point let us consider the RWQM model. A DM typically considers different wastewater treatment technologies and the related costs, as well as standards for water quality. However, s/he knows that specification of constraints for a group of (either ambient or effluent) water standards may lead to solutions that are too expensive. On the other hand, assuming constraints for costs (with water quality standards being goals) could result in an unacceptable water quality. Values of constraints are in such cases formally parameters in a corresponding optimization problem. But those values are, in fact, decisions that

reflect the preference structure of a user. Setting constraints' value too tight would result in restricting the analysis of the problem to a (possibly small) part of feasible solutions (often making the set X_0 empty). Textbooks on modeling typically provide the advice of using sensitivity analysis to deal with these limitations. However, as discussed in the section below on sensitivity analysis (see Section 3.3.3), applicability of sensitivity analysis to complex problems is very limited. A more practical approach in such situations is to specify two types of constraints, so called hard and soft constraints which correspond to *must* and *should* types of conditions, respectively. But, in fact, dealing with soft constraints can easily be done within multiobjective model analysis, which will be discussed later.

Therefore, the specification of a core model that defines X_0 should not include any relations that reflect conditions for acceptability of a solution by a user or a preferential structure of a DM. Hence, the *core model* accounts only for logical and physical relations between all the variables that define the set X_0 of feasible solutions. All other constraints and conditions that implicitly define acceptability of a solution by a user and those that represent a preferential structure of a DM will be included into an interactive procedure of the model analysis. This provides the flexibility of examining trade-offs between various solutions.

Such an approach to model specification and analysis allows us to design and implement a model-based DSS, which is conceptually composed of two parts:

- A constant and usually large core model. This part is built and verified before an actual analysis of a problem starts.
- A part that corresponds to a current specification of preferences defined by a user. This specification is interactively being changed, often drastically, by a DM.

Proper implementation of such an approach makes it possible for a DM to analyze feasible solutions that best correspond to his/her preference structure. Changing this structure is the essence of the model analysis and of the model-based decision support. There is an additional bonus in the fact that there always exists a feasible solution of the underlying mathematical programming problem, which is a prerequisite for an analysis of complex models.

Finally, we should point out that the value of a mathematical model as a decision aid comes from its ability to adequately represent reality. Therefore, there is always a trade-off between the requested accuracy (realism) of the model and the costs (also time) involved in developing it and providing the model with data. Hence, the requested accuracy should be consistent with the accuracy really needed for the model and with the quality of the available data.

Specification of the RAINS Mathematical Model

We shall now briefly comment on the specification of the RAINS model, which can be considered (as illustrated in *Figure 3.1*) as composed of three mutually linked parts:

- emission control costs and resultant emissions,
- atmospheric dispersion and tropospheric ozone formation models,
- environmental impacts.

Each of these components is backed up with a large amount of underlying research, which is presented in various specialized publications, see, e.g., Schöpp *et al.* (1999) and the RAINS Web site.⁵

Here we can provide only a general overview of these components.

The emission-cost module consists of three parts, estimating current and future levels of emissions of NH_3 , SO_x , NO_x , and VOC from each considered sector. These estimates are based on national statistics and projections of economic activities taking into account implemented and possible emission control measures and associated costs. These data are used to define parameters of PWL functions, which map for each sector considered in each country the emission levels of each type of pollutant to the corresponding cost.

The atmospheric dispersion processes over Europe for NH_3 , SO_x , NO_x , and VOC compounds are modeled using results of the European EMEP⁶ model, developed at the Norwegian Meteorological Institute and described, e.g., in Olenrzyński *et al.* (2000). However, the EMEP model is far too complex to be used for optimization, or even for many scenario analyses. Therefore, an essential requirement of an integrated assessment of the RAINS model is a simplified but reliable description of the dispersion processes in order to represent the source-receptor relationships involved. It is possible to envisage several ways of condensing the results of more complex models to achieve this. One approach is to use statistical techniques to build a simplified model based on the results obtained from a complex mathematical model for a large number of emission reduction scenarios. Such an approach has been implemented for, and is currently used by, the RAINS model. Of course, using simplified source-receptor relationships between the precursor emissions and the various thresholds of corresponding levels/loads results in a lesser accuracy than that assured by the EMEP photo-oxidants model. Therefore, selected results obtained from the simplified model are compared with results from

⁵<http://www.iiasa.ac.at/~rains>

⁶EMEP: European Monitoring and Evaluation Programme, cooperative program for monitoring and evaluation of the long-range transmission of air pollutants in Europe (see www.emep.int).

the EMEP model. This is done by running the EMEP model for the emissions obtained from the RAINS model, and comparing the levels/loads values provided by both models.

Another approach, which focuses on specification of a simplified ozone formation submodel, is based on using fuzzy-rules generation methodology and is presented by Ryoke *et al.* (2000). This method uses fuzzy rule generation methodology to represent numerous results of the EMEP model as a response surface describing the source-receptor relationships between ozone precursor emissions and daily tropospheric ozone concentrations. It has been shown that the fuzzy model provides better predictions of ozone concentrations than the traditional regression model based on all data at each grid. Furthermore, the membership functions (MFs) obtained appear to be sensible. When meteorological data are examined, the different fuzzy rules describe different meteorological conditions rather well. Unfortunately, the development of a fuzzy model requires manual tuning of parameters for each grid, therefore the model has been developed only for several grids in Europe. For these grids, the fuzzy model can be used to examine daily ozone concentrations caused by a selected emission scenario in a much faster and easier way than can be accomplished by the much more detailed EMEP model.

Space limitations prevent a full specification of the RAINS model. Such a specification is presented by Amann and Makowski (2000). Here we only outline two issues of more general interest, which are accounted for in this model specification: (1) the optimization problem and (2) soft constraints.

1. The resulting optimization problem has practically non-unique solutions. More exactly, it has many very different solutions with almost the same value of the original goal function. Let's consider two solutions x_1 and x_2 such that:

$$|c(x_1) - c(x_2)| < \epsilon \quad \|x_1 - x_2\| > \delta \quad (3.2)$$

where $c(\cdot)$ is a goal function, $\|\cdot\|$ is a norm used for determining the distance between vectors x_1 and x_2 , and ϵ, δ are two positive numbers, small and large, respectively. For most large optimization problems, this is a typical issue that, unfortunately, does not attract enough attention because analysts often look only at an optimal solution without analyzing other solutions, which have practically the same value of the goal function. Typically, a problem gets noticed when various instances of the mathematical programming problem that differ very little have very different optimal solutions (while the goal function remains practically the same).

There is a simple and practical technique called regularization, which provides a suboptimal solution that has additional properties specified by a user. The methodological background of regularization is presented, e.g., by

Makowski (1994a), and its implementation in the RAINS model is discussed in Section 3.3.3.

2. Representing environmental targets traditionally via hard constraints would result in the recommendation of expensive solutions. Only a few grids have active constraints for environmental targets, and for almost all grids the actual values of indices are substantially lower than the corresponding targets. In order to provide a more complete analysis, so called *soft constraints* (with compensation for the violation of original targets in some grids by a larger margin in other grids) can optionally be specified for environmental targets. They result in much cheaper solutions with more uniform differences between environmental targets and actual values of corresponding indices. The application of soft constraints in the RAINS model is presented in detail by Amann and Makowski (2000), and the mathematical background and also applications to other environmental problems can be found in several chapters of Wierzbicki *et al.* (2000).

3.3.2 Model generation and data handling

There are basically two approaches to the generation and analysis of a mathematical programming problem: either develop a problem-specific generator or use a modeling system (such as GAMS, AMPL, AIMMS). Several issues should be considered when selecting one of these approaches. These problems are discussed in detail by Makowski (2000). Here just seven of them are outlined:

- *Increasing the Efficiency of Model Generator Development* – A modeling system greatly simplifies the task of model specification, especially if compared with the amount of resources needed for the development of a model generator using traditional procedural programming languages like Fortran or C. However, the use of C++ substantially reduces this difference, especially with the Standard Template Library (recently included in the C++ standard), and with other class libraries supporting the implementation of mathematical programming problems.
- *Processing Input Data and Checking Data Consistency* – A model generator is more efficient in processing the input data needed for model specification. It is also preferred when a more sophisticated check of data consistency is desired.
- *Preprocessing* – A modeling system has limited possibilities for efficient preprocessing of optimization problems. This is not a serious problem for linear models because preprocessing is a standard feature of any good linear program-

ming (LP) solver. However, the preprocessing of non-linear models is much more difficult, as demonstrated, e.g., by Drud (1997).

- *Choosing a Starting Point* – For a large optimization problem, a good starting point might dramatically decrease the computation time. The computation of such a point is much easier for a problem-specific generator.
- *Comprehensive Model Analysis* – A modeling system greatly simplifies model analysis within the paradigm specific to a given system. However, using different modeling paradigms – such as soft and/or inverse simulation, regularization, soft constraints, or MCMA – which is necessary for comprehensive analysis of any complex problem, typically requires much additional effort if the particular paradigm is not included in a given modeling system.
- *Computing Nonlinear Constraints and Jacobian* – A modeling system releases a modeler from the complex task of providing code to compute the values of non-linear constraints and the non-linear elements of the Jacobian. However, a typical non-linear problem has only a few formulas for the non-linear part. Therefore, one can use, e.g., *Mathematica* (Wolfram, 1996) for generating C language code for formulas of the Jacobian and for the values of constraints, and then include this code in a class that provides values for particular elements of the Jacobian and for the constraints.
- *Decreasing Costs for Widely Distributed Models* – Finally, for models that are not only run on various platforms but are also widely distributed, a problem-specific generator substantially decreases costs for the users (typically, the cost of a solver plugged into the problem-specific software is a small fraction of the cost of the run-time license for a modeling system).

Taking into account the above-summarized points, the problem generator of the RAINS model has been implemented as a problem-specific C++ class that uses a template class library supporting the generation of mathematical programming problems. The generator includes an efficient preprocessor, which dramatically reduces the size of the non-linear optimization problem, and also performs an instance-specific scaling, which results in values of the Jacobian and Hessian that are unlikely to cause numerical problems for a non-linear solver.

The approach is conceptually very simple. Each of the above-mentioned solvers is available as a library of Fortran subroutines. The generator has C++ classes that are specific for each solver. These classes are inherited from the base classes that handle a common part of the generator. A problem-specific report writer processes the results into a form that eases their interpretations. Another class supports a portable interface between C++ and Fortran. Hence, three versions of executables can easily be produced, each composed of the generator, preprocessor, postprocessor, and one of the solvers.

A nonlinear solver requires routines that compute values as well as elements of the Jacobian of the non-linear constraints and the goal function. A large part of the total computation time is used for the execution of these functions, and therefore the efficiency of their implementation is important. The code for the Jacobian has been generated by *Mathematica* (Wolfram, 1996) with prior use of the *FullSimplify* operator, which simplifies the formulas substantially. This is an easy way to generate an efficient and bug-free code.

The RAINS model requires processing a large amount of data coming from various sources, including other complex models. Therefore, the data used by the TAP Project is maintained by several database management systems, which are coupled with other applications. To make the handling of data used in the RAINS model efficient and portable, the public domain library Hierarchical Data Format (HDF) (Koziol and Matzke, 1998), developed by the National Center for Supercomputing Applications, Illinois, USA,⁷ has been employed. The basic data structures are handled by a collection of well-tested template C++ classes that are also used for the LP-DIT.⁸ A C++ interface class has been implemented to make handling of the used data structures by the HDF library easy and efficient.

Costs of emission reductions discussed above are given as PWL (Piece-Wise Linear) functions of the corresponding emission levels. PWL functions are not smooth. Therefore, in order to be able to use efficient nonlinear solvers (which require smooth functions), the PWL cost functions are represented by corresponding smooth functions. Due to space limitations, these conversions are not presented here; however, they are described by Amann and Makowski (2000).

Finally, one should notice that the dimensions of the model are not fixed. For some scenarios, a part of the constraints and/or variables does not need to be generated. Moreover, the dimensions of the matrices and vectors used in the model definition vary substantially for various types of analysis. Fortunately, properly implemented constructors of C++ template classes handle such problems in a natural and efficient way.

3.3.3 Model analysis

There are many approaches to model analysis and typically a problem-specific combination of various approaches is needed for a comprehensive analysis of any complex problem. We summarize some general concepts of model analysis that are of more general interest, and then illustrate the need of combining some of the various methods by outlining the approach applied to the analysis of the RAINS model.

⁷<http://hdf.ncsa.uiuc.edu/HDF5>

⁸Linear programming data interchange tool.

General Concepts

One typically distinguishes two types of model-analysis methods, which are conventionally called simulation and optimization. They can be characterized as follows:

- In *simulation*, decision variables are inputs and goals are outcomes. Therefore this technique is good for exploring the intuition of a DM, not only for verification of the model, but also for providing a DM with information about the consequences – typically represented by values of outcome variables and constraints – of applying certain decisions. One can also consider simulation as an alternative-focused method of analysis that is oriented toward examining given alternatives.
- *Optimization* can be considered as a goal-oriented (value-focused) approach that is directed toward creating alternatives. Optimization is driven either by formulating a single objective in single-criterion optimization, or several objectives in multicriteria optimization, and looking for values of decision variables that optimize the value of the specified objective(s). Therefore, goals are the driving force and the values of decision variables are the outcomes.

Simulation– and optimization-based approaches are in fact complementary. For simulation, one needs to provide values for all decision variables. For this purpose, one may use random values for variables (as proposed by Goodwin and Wright, 1991, who present various techniques and examples), or assign values based either on the DM’s intuition or on a heuristic (possibly based on information from a knowledge base). One should, however, note that applicability of these appealing ideas is limited to rather small models; for models having hundreds or even more variables, a specification of values for all decision variables based on intuition is practically unrealistic.

However, even for a large model, simulation can be useful for a “*what if*” type of analysis, e.g., for comparing the results from optimizations with the outcomes from values of decision variables defined by the user, typically by modification of their values obtained from optimization. Of course, there is no way to assure that a given specification of the values of decision variables will result in a feasible solution. Therefore, instead of using a classical approach to simulation, one should use a *soft simulation*, where setting given values of decision variables \hat{x} is replaced by minimization of an outcome variable defined as:

$$\epsilon \|x - \hat{x}\| \tag{3.3}$$

where ϵ is a given positive number, x is a vector composed of decision variables, and \hat{x} is a vector composed of the corresponding desired values of these variables.

In a most simple soft simulation approach, one sets $\epsilon = 1$ and assigns to \hat{x} the given values of decision variables. However, a similar approach may be used also for more sophisticated types of analysis, where x is composed of not only decision variables, and the choice of \hat{x} depends on the desired properties of the solution. In particular, if values of some elements of \hat{x} are not known, then one can set them to be equal to zero, which implies a preference for the minimum norm solution.

Of course, term (3.3) can be used to define an outcome variable that can be used as a criterion in MCMA. It can be used also as a term in a composite goal function with larger values of the parameter ϵ for various simulation techniques. For example, by using a large value of ϵ (i.e., one that dominates the other terms of the goal function) and setting \hat{x} equal to desired values of decision variables, one can find a solution that is closest to such values. If these values are feasible, then a solution composed of these values will be found. If they are not feasible, then the closest feasible solution will be found. Note that in the latter case a traditional simulation will simply report “*infeasible problem.*”

For such an approach to soft simulation, the original goal function takes the role of the regularizing term, while for small values of ϵ , term (3.3) may be used as a regularizing term for the original goal function. An application of such an approach in the RAINS model is discussed in the next subsection.

Sensitivity Analysis

In mathematical programming, sensitivity analysis is typically understood as an analysis of changes of an optimal solution caused by an alteration of the data in the model. A traditional approach to such an analysis is based on properties of an optimal solution. It typically consists of calculations of ranges of changes of parameters for which an optimal solution does not change, and on using a dual solution for calculations of changes in value of a goal function for changes in some parameters that are small enough to allow such a simple evaluation procedure. These methodological topics, which all form the subject of post-optimal analysis, and the corresponding software tools have been extensively developed, especially for LP types of problems. However, their applicability is practically limited to rather small, linear models.

There are several problems concerned with applying the classical approaches to sensitivity analysis to problems represented by complex models. We summarize here only the three most important issues:

- The range of changes of parameters for which the classical sensitivity analysis is valid is typically too small to justify its application to models of mixed-integer and non-linear types.

- The concept and tools for sensitivity analysis have been developed and implemented for analysis of rather small models. Complex models are typically large, however; therefore use of these techniques is either cumbersome or practically impossible for complex models.
- In many models, the quality of dual solution is rather questionable, and for many other models the dual solution is practically non-unique. This is owing to the fact that most large models are numerically badly conditioned, and due to efficient presolve algorithms, which greatly decrease the resources (time and memory) needed for solving large problems. However, presolving always guarantees the quality of the primal solution but often results in an unreliable dual solution, which is the basis for classical sensitivity analysis. Therefore, a reliable sensitivity analysis requires a good understanding of various techniques and corresponding tools, which is rather limited to highly skilled specialists in mathematical programming.

Generally, one distinguishes two groups of problems which correspond to the two related but distinct issues that are typically used for a justification of applications of sensitivity analysis, namely:

- Model development, where some parameters of the model can hardly be precisely determined; here by parameters we understand only coefficients in logical and physical relations.
- Model analysis, where a classical single-objective optimization-based approach forces the analyst to treat all but one actual goal as constraints.

A discussion on how and when the selection of a type of model (such as fuzzy or stochastic) can adequately represent a problem for which a deterministic model with fixed parameters may be too simplified is far beyond the scope of this chapter. In many practical applications, a deterministic model is an adequate simplification provided that the developers of the model have enough data and experience to properly evaluate values of parameters. In some situations, a parametric analysis of a model is nevertheless needed, but this is typically done during the model validation. Another technique that is useful, and is more efficient than some elements of sensitivity analysis, is a specification of so-called *soft constraints*, and the use of such constraints for a definition of outcome variables.

The second issue (model analysis) can be easily addressed by using MCMA, which is based on a core model that does not include the preferential model of a user. In classical single-criterion optimization, several objectives were typically treated as constraints, for which one had to specify an acceptable value. This approach has not only the disadvantages discussed above, but it also requires analysis

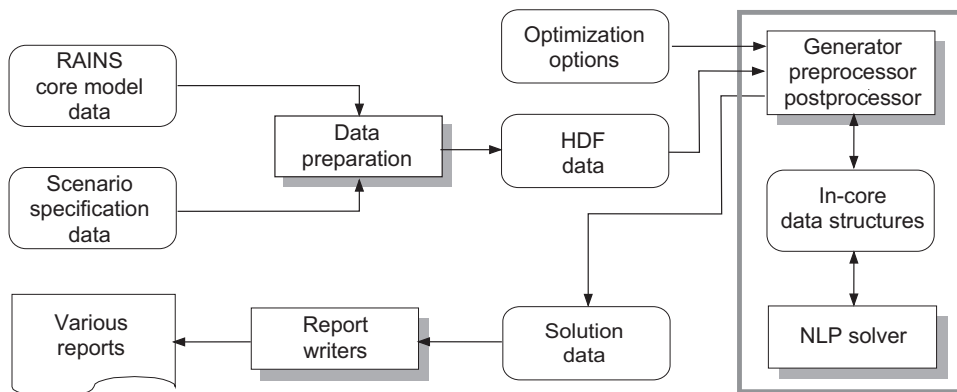


Figure 3.3. RAINS model analysis cycle.

of the impact of changes caused by specified constraining values for criteria that are treated as constraints. Such values cannot, in practice, be specified precisely, therefore their modifications are inevitable. Sensitivity analysis was developed in order to help in analysis of such modifications. However, the functionality of sensitivity analysis, which was applied to this part of classical analysis of optimization models, is replaced in multiobjective model analysis by more robust and natural approaches to problem analysis. Therefore, multiobjective model analysis offers better ways for providing some of the functionalities that are theoretically promised by sensitivity analysis.

Analysis of the RAINS Model

This section outlines how a combination of various methods of model analysis has been applied to the RAINS model, which is used extensively for various types of analysis that are needed for supporting international negotiations. Obviously, neither RAINS nor any other complex model provides any “best” solutions. This is simply because there are several problems and trade-offs that are both moral and social. No model can actually answer such questions, and this remains the domain of negotiations. However, models can help the negotiators concentrate on those parts of the negotiations that should not be represented by a mathematical model. This assistance is provided by various unbiased analyses, such as computing the consequences of given policies of emission reductions, or advising the values of emission levels that correspond best to given criteria and constraints.

Due to space constraints, I have limited this section to presenting the structure of one cycle of analysis followed by a summary of the implementation of a composite goal function for the RAINS model analysis.

The structure of one cycle of the RAINS model analysis is outlined in *Figure 3.3*. Prior to analysis, a data file is prepared that contains all parameters of the RAINS core model. Another data file with a definition of the parameters for a particular scenario is prepared by specialized software. These two data sets are converted by another specialized program into the HDF file. Additionally, a user has the possibility of selecting various options and specifying the corresponding parameters (e.g., of the composite goal function discussed below) and options (e.g., allowing for soft constraints, requesting the balancing of violations with surpluses) that overwrite the default selections and are used for a definition of a particular instance of the non-linear optimization problem.

The optimization problem is generated and solved by the problem-specific model generator linked with a selected non-linear solver library. The generator (the functions of which are described in Section 3.3.2) creates the necessary data structures, which are kept in-core and are used for functions that are required by each of the used solvers. Such an approach allows for the efficient generation and solution of the corresponding large non-linear optimization problem. After an optimal solution is found, a postprocessor converts the solution to a form that is convenient for analysis (by “undoing” the actions of the preprocessor and by computing values of variables, which were not generated).

A solution provided by the postprocessor is processed by a specialized report-writer program, which provides various types of information needed for the analysis of a solution. Afterwards, another scenario is designed based on this analysis and on requests from users. This scenario is used as an input to a new cycle of the analysis.

A particular scenario is defined by many parameters. A minimization of costs related to measures needed for improvement of air quality is a main goal; however, other objectives – such as robustness of a solution, trade-offs between costs and violations of environmental standards – are also important. Therefore, a MCMA has been applied to this case study. A composite goal function (3.4, below) is applied to support the analysis of trade-offs between the following three criteria:

- the minimization of total costs of emissions reduction,
- the minimization of violations of the environmental standards,
- the robustness of solutions.

The following composite criterion function is used:

$$goal_function = cost + \Theta + \Psi \quad (3.4)$$

where the *cost* term corresponds to the sum of the costs of emission reductions and Θ and Ψ are regularizing and the penalty term, respectively.

The regularizing term Θ is defined by:

$$\Theta = \epsilon \|z - \bar{z}\|^2 \quad (3.5)$$

where ϵ is a given positive (not necessarily small) number, z denotes a vector composed of decision variables that correspond to emissions, and \bar{z} is a given vector composed of desired (reference) values of emissions. The role of the term Θ is twofold. First, it helps to avoid large variations of solutions (with almost the same value as the original goal function) for problems that differ very little. Second, it substantially improves the numerical stability of the optimization problem. Additionally, the term Θ can be used for the technique called softly constrained inverse simulation. Thus, it is possible to analyze trade-offs between minimization of costs and solutions that correspond closely to various given patterns of emissions defined by \bar{z} .

The role of the term Ψ is also twofold. First, it serves as a penalty term for optional variables y , ya , and ye . Second, it provides regularization for these decision variables, which are not covered by the Θ term. The term Ψ is defined by:

$$\Psi = \sum_{l \in L} \sum_{j \in J} \psi(y_{lj}, \rho_o, \sigma_o) + \sum_{j \in J} \psi(ya_j, \rho_a, \sigma_a) + \sum_{j \in J} \psi(ye_j, \rho_e, \sigma_e) \quad (3.6)$$

where $\rho_o, \rho_a, \rho_e, \sigma_o, \sigma_a, \sigma_e$ are given positive parameters, and the function $\psi(\cdot)$ is defined by:

$$\psi(x, \rho, \sigma) = \begin{cases} -\rho\sigma x - \rho\sigma^2/2 & \text{for } x < -\sigma \\ \rho/2x^2 & \text{for } |x| \leq \sigma \\ \rho\sigma x - \rho\sigma^2/2 & \text{for } x > \sigma \end{cases} \quad (3.7)$$

Note that $\psi(x, \rho, \sigma)$ is a smooth function that, depending on the parameters ρ and σ , can be used for both purposes that correspond to the role of the term Ψ outlined above. First, it plays the role of a classical quadratic penalty function, if large values of the parameters ρ, σ are selected. Such a function can be used to examine the trade-offs between violations of air quality standards and minimization of costs. Second, it may not be desirable to apply any penalty function for some scenarios in which the balances between violations of environmental targets and surpluses are required. However, in such cases, it is still necessary to apply regularization in order to deal correctly with the soft constraints optionally defined by introduction of decision variables y_{lj}, ya_j, ye_j . A quadratic function is not suitable for this purpose because often violations and surpluses take small values in some grids and large

values in other grids; therefore, it is not possible to find a value of the parameter ρ such that it would allow for large values of violations/surpluses in some grids, while serving as a regularizing term for grids where violations/surpluses may be three orders of magnitude smaller, and a specification of different values of ρ for each of about 600 grids is not practicable. Therefore, when used for regularization purposes alone, the function ψ is defined using small values of both parameters ρ, σ , which implies using a flat PWL function with a small quadratic segment needed to make such a function smooth. Finally, the term Ψ provides a similar functionality as the approach commonly known as *soft constraints*.

To summarize the discussion on the form of the goal function (3.4), it is important to stress the fact that a properly defined goal function is the key element for achieving two objectives: namely, (1) providing a tool for a comprehensive problem analysis and (2) assuring possibly good numerical properties of the corresponding optimization problems. The specific form of this function – in particular, the penalty terms for soft constraint violations, the regularizing terms – makes the model analysis very similar to a multi-objective formulation, as applied, e.g., to softly constrained inverse scenario analysis. See Wierzbicki *et al.* (2000) for more details. In the near future the MCMA software, outlined in the next section, will be optionally used for multicriteria analysis as an alternative to the composite goal function.

3.4 Multicriteria Model Analysis

Many papers and books pointed out quite long ago the limitations of the traditional operations research (OR) approach. See, for example, Wierzbicki (1977), Ackoff (1979), Lewandowski and Wierzbicki (1989), Chapman (1992), and Wessels and Wierzbicki (1993). This chapter will briefly summarize only one of these techniques, which seems to be the most natural method that best corresponds to a real-life decision-making process. This is the Aspiration Reservation Based Decision Support (ARBDS) method which is an extension of the *aspiration level* (sometimes referred to as *reference point*) approach, originally proposed by Wierzbicki (1977), later developed and applied in many applications in various fields, and recently presented in detail in Wierzbicki *et al.* (2000). First, however, it is worth pointing out another successful approach, which is based on another constructive proposal for overcoming limitations of the traditional, optimization-centered OR approach. It was formulated by Sawaragi and coworkers in the introduction to their standard textbook on model-based decision support (Sawaragi *et al.*, 1985). These concepts have been elaborated on over the years, and one of the streams of the follow-up research is known as the *Shinayakana systems approach* (see, e.g., Sawaragi and

Nakamori, 1991). A more recent overview of this methodology for analyzing complex systems and environmental modeling is presented by Nakamori and Sawaragi (2000).

The following topics are discussed in ensuing subsections:

- basic concepts of MCMA,
- the Aspiration-Reservation method, and
- several basic problems with using weights to convert a multicriteria optimization problem into a parametric single-criterion mathematical programming problem.

3.4.1 Basic concepts of multicriteria model analysis

A key issue in MCMA is the identification and analysis of those parts of the Pareto-optimal solution set that are interesting for the user. Generation and analysis of the entire Pareto set is practically impossible. Therefore, any practical multi-objective method facilitates an analysis of a subset of Pareto solutions that best correspond to preferences of a user.

Multicriteria optimization methods typically assume that a multicriteria problem is converted into an auxiliary parametric single-objective problem whose solution provides a Pareto-optimal point having desired properties. Different methods apply different conversions, but the most commonly known methods can be interpreted (see Makowski, 1994b) in terms of the Achievement Scalarizing Function (ASF). The concept of ASF was introduced by Wierzbicki and it is very useful for comparing different approaches to multicriteria optimization. (See, e.g., Wierzbicki, 1977; Wierzbicki, 1986; and Wierzbicki *et al.*, 2000 for mathematical foundations, interpretations, and applications.)

A solution is called a Pareto-optimal (or an efficient) solution, if there is no other solution for which at least one criterion has a better value while values of remaining criteria are the same or better. In other words, one can not improve any criterion without seeing a value of at least one other criterion deteriorate. We refer to properly Pareto-optimal solutions with a prior bound on trade-off coefficients as Pareto solutions (unless otherwise mentioned). A Pareto-optimal point in objective space is composed of values of all criteria for a corresponding Pareto-optimal solution.

The basic distinction between various multicriteria methods is determined by the way in which a method supports selection of Pareto solutions that best correspond to the user's preferences. The aspiration-based methods use the aspiration values – a term used interchangeably with reference point – which are composed of values that a user wants to achieve for each criterion. If a specified aspiration

level \bar{q} is not attainable, then the Pareto-optimal point is that nearest to the aspiration level. If the aspiration level is attainable, then the Pareto-optimal point is uniformly better than \bar{q} . Properties of the Pareto-optimal point depend on the localization of the reference point (composed of aspiration values) associated with the criteria, and the applied definition of a distance.

To treat attainable aspiration points properly as well, instead of a distance one uses ASF. Therefore the selection of the Pareto-optimal solution depends on the definition of the ASF, which includes a selected aspiration point. Most of those methods use the maximization of an ASF in the form:

$$\mathcal{S}(q, \bar{q}, w) = \min_{1 \leq i \leq n} \{w_i(q_i - \bar{q}_i)\} + \epsilon \sum_{i=1}^n w_i(q_i - \bar{q}_i) \quad (3.8)$$

where $q(x) \in R^n$ is a vector of criteria, $x \in X_0$ are variables defined by the core model, X_0 is a set of feasible solutions implicitly defined by the core model, $\bar{q} \in R^n$ is an aspiration point, $w_i > 0$ are scaling coefficients and ϵ is a given small positive number. Maximization of (3.8) for $x \in X_0$ generates a properly efficient solution with the trade-off coefficients (as recomputed in terms of u_i defined below) smaller than $(1 + 1/\epsilon)$. For a non-attainable \bar{q} , the resulting Pareto-optimal solution is the nearest (in the sense of a Chebyshev weighted norm) to the specified aspiration level \bar{q} . If \bar{q} is attainable, then the Pareto-optimal solution is uniformly better. Setting a value of ϵ is itself a trade-off between getting a too restricted set of properly Pareto-optimal solutions or a too wide set practically equivalent to weakly Pareto-optimal solutions. Assuming the ϵ parameter to be of a technical nature, the selection of efficient solutions is controlled by the two vector parameters: \bar{q} and w .

It is commonly agreed that the aspiration point is a very good controlling parameter for examining a Pareto set (i.e., a set composed of Pareto-optimal solutions). Much less attention is given to the problem of defining the scaling⁹ coefficients w . A detailed discussion on scaling coefficients in a scalarizing function is beyond the scope of this chapter. The four commonly used approaches are summarized in Makowski (1994b).

The aspiration-based approaches correspond very well to the concept of satisficing behavior (also called bounded rationality), in which the DM attempts to attain aspiration levels, usually by first trying to improve the criterion that shows the worst performance. (See, e.g., March and Simon, 1958; Wierzbicki, 1982.) This method has several advantages over other multi-objective optimization methods, as discussed in detail in Wierzbicki *et al.* (2000) and is one of the most successful

⁹Note that the scaling coefficients w should not be used as weights for a conversion of a multi-criteria problem into a single criterion problem with a weighted sum of criteria (see Section 3.4.3 for a detailed discussion and examples). In the function (3.8) they play a different role than in a weighted sum of criteria.

classes of DSSs (see, e.g., Korhonen and Wallenius, 1989 for a justification of this statement). Moreover, as shown by Ogryczak and Lahoda (1992), the aspiration level approaches may be interpreted as extensions of goal programming (Charnes and Cooper, 1967), which was the precursor of most multicriteria optimization and model analysis methods.

The reference point approach can also be used for inverse simulation: instead of repeatedly adjusting the decision variables in order to determine acceptable states (expressed as constraints in the classical approach to optimization), the user chooses desired states (in terms of ranges of values of objectives) and the DSS determines for her/him the resulting values of the decision variables. The reference point approach also takes into account soft constraints often needed in the single-criterion optimization. Specifically, one can replace a soft constraint (or group of constraints) by an objective, and then set the aspiration level equal to the desired value of the constraint and the reservation level to the worst acceptable value. Thus, violations of soft constraints can be treated as criteria (to be minimized) in the multi-objective approach.

3.4.2 Aspiration reservation based decision support

The ARBDS method is an extension of the reference point method. In practical applications, the most promising approach is based on the calculation of scaling coefficients (used to define the weighted Chebyshev norm mentioned above), with the help of the aspiration level \bar{q} and a reservation level \underline{q} (the latter is composed of values of criteria that the user wants to avoid). This is the ARBDS approach that has been introduced by the DIDAS family of DSS described in Lewandowski and Wierzbicki (1989). Its extension is presented here, applied in the Interactive Specification and Analysis of Aspiration-based user Preferences (ISAAP) tool, which is described in detail in Granat and Makowski (2000).

Geometrical aspects of the reference point and the ARBDS approaches are shown in *Figure 3.4*, which illustrates a Pareto-solution set (between points **D** and **E**) for a two-criteria minimization problem. Utopia and Nadir¹⁰ points (denoted by **U** and **N**, respectively) are composed of the best and worst values of criteria in the Pareto set. For a given aspiration point **A**, any of the Pareto-optimal points between points **B** and **C** can be obtained for various definitions of the distance between an aspiration point and the Pareto set. In the classical reference point method the weights in the ASF defined by (3.8) had to be somehow specified. It

¹⁰One should note that a computation of a true Nadir point is often practically impossible, therefore, modern MCMA approaches don't use the Nadir point for model analysis. Usually only an approximation of the Nadir point is used for actual model analysis, which is not any actual limitation because users are typically interested in aspiration levels which are not attainable, and extremely bad solutions (which are close to the Nadir point) are not really interesting.

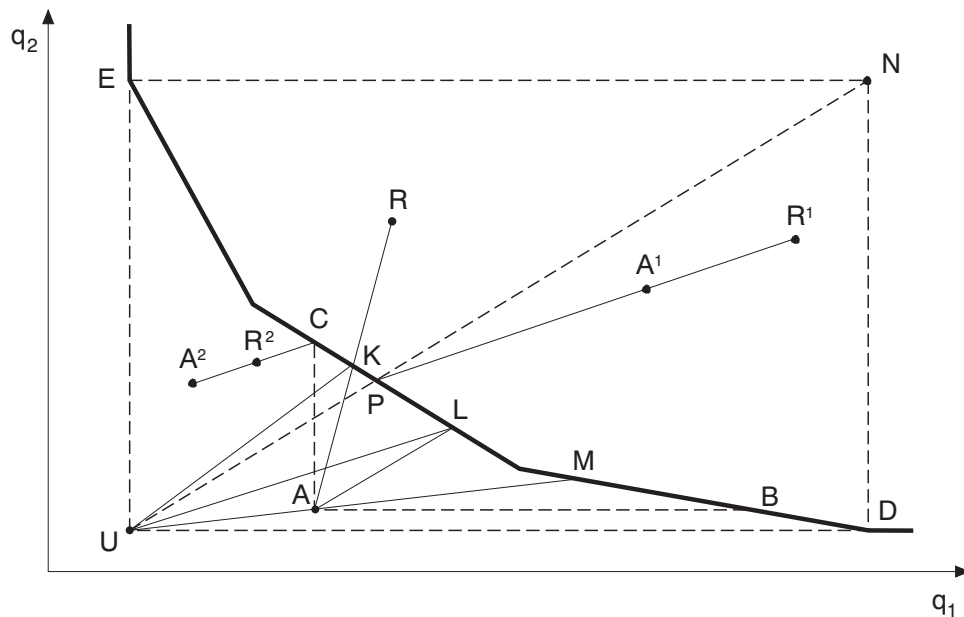


Figure 3.4. Illustration of a Pareto-optimal surface for a two-criteria case.

had been recognized that users have serious difficulties with defining weights that correspond well to their trade-offs between criteria (the problems were similar to those described in Section 3.4.3), and therefore the next stage in the development of the MCMA methods was to define weights indirectly using two points. Two approaches were most popular for a selection of such points: to use either a pair $\{U, A\}$, or $\{U, N\}$ for defining the weights. For the example in *Figure 3.4* this results in the Pareto-solutions **M**, and **L**, respectively. However, in practical applications both of these approaches created various difficulties, both for users and for the developers of the needed software.

These difficulties led to the development of the ARBDS method, in which a user specifies for each criterion a pair of values: aspiration (the desired value of a criterion) and reservation (the worst acceptable value), denoted further on by **A** and **R**, respectively. This is the most natural way of examination of interesting parts of the Pareto set because users typically have a good understanding of the range of criteria values that they want to achieve. Moreover, an appropriate implementation of the ARBDS method does not impose any restrictions on **A** nor on **R**. In *Figure 3.4* there are three pairs of aspiration and reservation points, denoted by $\{A, R\}$, $\{A^1, R^1\}$, and $\{A^2, R^2\}$, respectively. The corresponding Pareto-solutions are marked by **K**, **C**, and **P**, respectively. A selection of a pair like $\{A, R\}$ (i.e., a not attainable aspiration and a feasible reservation level) is typical for users who have learned

the properties of the problem and have a good feeling about the attainable ranges of criteria values. Selection of non-attainable reservation level (e.g., $\{\mathbf{A}^1, \mathbf{R}^1\}$) is typical for early stages of model analysis, when unrealistic reservation levels are specified. However, specifications of attainable aspiration levels (or at least some components of it) are not as rare as one can expect; especially, if some criteria are correlated. Therefore it is important that MCMA does not impose any restrictions on the feasibility of the aspiration nor of the reservation values.

In order to meet this requirement, and to support an option of a more exact specification of preferences for criteria values between aspiration and reservation values, the ASF for the ARBDS has a more general form than that shown in (3.8, above), and usually is defined as

$$\mathcal{S}(q, \bar{q}, \underline{q}) = \min_{1 \leq i \leq n} u_i(q_i, \bar{q}_i, \underline{q}_i) + \epsilon \sum_{i=1}^n u_i(q_i, \bar{q}_i, \underline{q}_i) \quad (3.9)$$

where \bar{q}, \underline{q} are vectors (composed of $\bar{q}_i, \underline{q}_i$, respectively) of aspiration and reservation levels, respectively, and $u_i(q_i, \bar{q}_i, \underline{q}_i)$ are the corresponding Component Achievement Functions (CAFs), which can be simply interpreted as nonlinear monotone transformations of the i -th criterion value q_i into ASF u_i (which reflects the degree of satisfaction of the user) based on the information represented by aspiration and reservation levels for this criterion, \bar{q}_i and \underline{q}_i , respectively. Maximization of the function (3.9) over the set of feasible solutions X_0 defined by the corresponding core model provides a properly Pareto-optimal solution with the properties discussed above for the function (3.8).

The ARBDS method is organized into the following steps typical for a MCMA:

1. The user or DM selects several criteria (objectives) from outcome variables. In typical applications there are 2–9 criteria.
2. The DM specifies an aspiration point $\bar{q} = \{\bar{q}_1, \dots, \bar{q}_k\}$, where \bar{q}_i are aspiration levels (the desired values for each criterion) and k is the number of criteria. Additionally, the DM specifies a reservation point \underline{q} , which is composed of the worst values of criteria that a DM would like to accept. Optionally, the user can specify his/her preferences for values of criteria between aspiration and reservation, by interactively selecting points that define PWL function, as illustrated in *Figure 3.5*.
3. The underlying formulation of the problem is the maximization of a ASF (3.9). This can be interpreted either as a value function of the DSS specified in response to the specific aspiration and reservation levels, or as an ad-hoc, non-stationary approximation of the value function of the DM, dependent on these

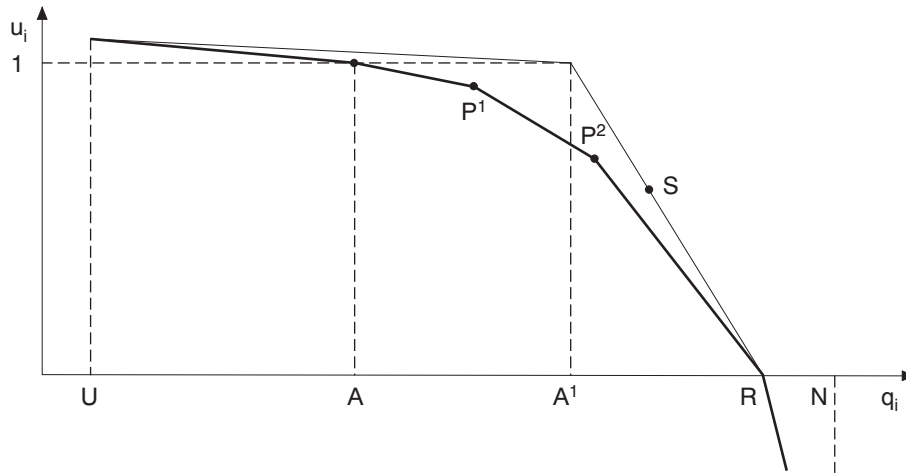


Figure 3.5. Illustration of the PWL CAF for a minimized criterion.

levels. The problem is then transformed by the DSS into an auxiliary parametric single-objective problem, the solution of which gives a Pareto-optimal point.

4. The DM explores various Pareto-optimal points by changing the aspiration point \bar{q} and reservation point \underline{q} for each criterion. Additionally, a DM may stabilize a criterion (i.e., specify a desired value instead of minimizing or maximizing the value of this criterion) or temporarily remove a criterion from the analysis.
5. The procedure described in points 2, 3, and 4 is repeated until a set of satisfactory solutions is found.

In order to allow for either specification of only aspiration and reservation levels or for additional specification of preferences (for the criteria values between aspiration and reservation levels), the ISAAP supports specification of the CAFs in a more general form than that originally proposed by Wierzbicki (1986). For this purpose, the PWL CAF u_i are defined by segments u_{ji} :

$$u_{ji} = \alpha_{ji}q_i + \beta_{ji}, \quad q_{ji} \leq q_i \leq q_{j+1,i} \quad j = 1, \dots, p_i \quad p_i \geq 3 \quad (3.10)$$

where p_i is a number of segments for the i -th criterion. Such a function for a minimized criterion is illustrated in *Figure 3.5*. The thin line corresponds to a function that is composed of three segments, which are defined by four points, namely **U**, **A**¹, **R**, and **N** (corresponding to the Utopia, aspiration, reservation, and Nadir points, respectively). The solid line represents a modified function for which the

previously defined aspiration level \mathbf{A}^1 was moved to the point \mathbf{A} and two more points – \mathbf{P}^1 and \mathbf{P}^2 – were interactively defined.

Values of CAF have a very easy and intuitive interpretation in terms of the degree of satisfaction from the corresponding value of the criterion. Values of 1 and 0 indicate that the value of the criterion exactly meets the aspiration and reservation values, respectively. Values of CAF between 0 and 1 can be interpreted as the degree of *goodness* of the criterion value, i.e., to what extent this value is close to the aspiration level and far away from the reservation level. These interpretations correspond to the interpretation of the membership function (MF) of the Fuzzy Sets, which is discussed in the section on the relations between fuzzy sets and CAF, below. However, the CAF values provide more functionality than the MF of the Fuzzy Sets, which does not distinguish the differences between elements that do not belong to a set. Namely, values of CAF greater than 1 correspond to the criterion values better than aspiration level while negative values of CAF show that the criterion value is worse than the reservation level, and the differences in values of CAF correspond to the differences of quality of solutions that are beyond aspiration and reservation levels.

By using an interactive tool for specification of the CAF defined by (3.10), such as ISAAP (Granat and Makowski, 2000), a DM can analyze various parts of a Pareto set that best correspond to various preferences of trade-offs between criteria. These preferences are typically different for various stages of analysis, and are often modified substantially during the learning process, when aspiration and reservation levels for criteria values are confronted with the attainable solutions, which correspond best to the aspiration and reservation levels. In such an interactive learning process, a user gradually comes to recognize attainable goals that correspond best to his/her trade-offs.

In some applications, the value of an outcome variable should neither be minimized nor maximized but should have a value close to a given *target* value. In such a situation, the goal-type criterion can be used. For this type of a criterion, the distance from a given target value (which can be changed during the interaction) is to be minimized. For such a criterion, a CAF is composed of two parts: the first part is defined for the criterion values smaller than the target value, and the second part for the criterion values larger than the given target. Such a function is illustrated in *Figure 3.6*. The conditions specified above for maximized and minimized criteria hold for the first and second function, respectively. There is obviously only one point i , for which $\alpha_{i-1,i} > 0$ and $\alpha_{i,i+1} < 0$ and the criterion value for such a point corresponds to a target value (denoted by \mathbf{T} in *Figure 3.6*) for the goal-type criterion. The function shown in *Figure 3.6* is symmetric, but for many applications an asymmetric function is appropriate and therefore both types of functions

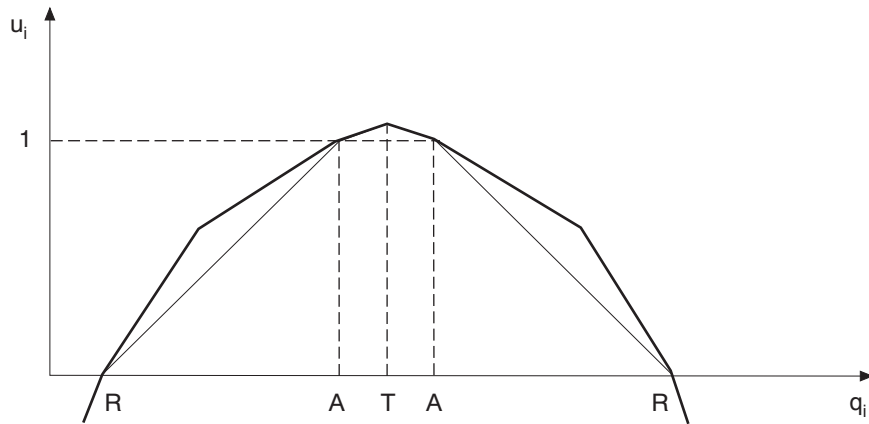


Figure 3.6. Illustration of the PWL goal-type CAF for a goal-type (or stabilized) criterion.

for the goal-type of criteria are supported by ISAAP. More details, including also a discussion on asymmetric CAF can be found in Granat and Makowski (1998).

The three types of criteria, i.e., minimized, maximized and goal-type, are used most often. However, sometimes it is desirable to consider more complex criteria. For example, for dynamic problems it is typical to deal with trajectories. In such cases, one can easily define outcome variables that correspond to a deviation from a given trajectory. Depending on the application, either a trajectory should be followed, or only the surplus (or deficit) should be minimized. The corresponding outcome variables can be defined as follows:

$$\max_{t \in T} |x_t - \bar{x}_t| \tag{3.11}$$

$$\max_{t \in T} (x_t - \bar{x}_t) \tag{3.12}$$

$$\min_{t \in T} (x_t - \bar{x}_t) \tag{3.13}$$

where T is a set of time indices and \bar{x}_t is a given reference trajectory. Such variables can be used as criteria: minimized for the first two cases and maximized for the last case.

The maximization of the ASF defined by (3.9) which uses the CAF in the form of (3.10) provides a natural way for selecting Pareto-efficient solutions that conform to the concept of satisficing behavior, that is, situations in which the DM attempts to attain aspiration levels by first trying to improve the criterion that shows the worst performance, i.e., which differs most from its aspiration level.

Relations Between Fuzzy Sets and Component Achievement Function

This section briefly comments upon an interpretation of the ASF in terms of Fuzzy Sets. Such functions can be interpreted in terms of the fuzzy MFs discussed in detail in Zimmermann (1985). MF are typically interpreted as functions that reflect the degree to which an element belongs to a set.

The ARBDS approach uses a so-called extended-valued MF proposed by Granat and Wierzbicki (1994), who suggested a method for constructing various forms of order-consistent component ASFs based on MFs that describe the satisfaction of the user with the attainment of separate objectives. Between aspiration and reservation levels, the values of this function coincide with the MF, as well as having an ordering properties. In other segments an ASF is used only for ordering alternatives (thus assuring that only Pareto-efficient solutions are found).

Thus, there are many similarities between the ARBDS and the *Fuzzy Multi-objective Programming* approaches. The main difference is due to the specification and use of CAF. The Fuzzy Multi-objective Programming method requires prior specification of aspiration and reservation levels that are used to define the MFs. It is implicitly assumed that the criteria values for all the interesting solutions are between the corresponding aspiration and reservation levels (because the applied MF does not differentiate between solutions with values better than aspiration level and between those with values worse than reservation level). In MCMA the user interactively specifies the reference membership levels for each CAF, which can be interpreted as a degree of achievements of the aspiration for each criterion. Moreover, CAF has order-preserving property, i.e., it has different values for all different criterion values.

The ARBDS method does not use the MF directly. It assumes that the user may change aspiration and reservation levels during the interaction upon the analysis of previously obtained solutions. The user specifies interactively the preferences in the space of the criteria values, which seems to be more natural than a specification of preferences in terms of degrees of achievements of CAF values. A selection in the criteria space can, however, be interpreted in terms of Fuzzy Sets by a definition of an MF for a linguistic variable (e.g., *good solution*) for each criterion, and an ex-post interpretation to which degree a solution belongs to a set of *good* solutions. There is no need for restrictions for the specification of aspiration and reservation levels in the criteria space. This is important for the analysis of large-scale complex problems for which the specification of attainable reservation levels might be difficult.

3.4.3 Problems with using weights for aggregation of criteria

One of the most popular approaches to multicriteria optimization is based on the idea of converting a multicriteria problem into a single-criterion one by summing up weighted criteria. This approach has a number of drawbacks as discussed in detail, e.g., by Makowski (1994b), by Nakayama (1994) and in Wierzbicki *et al.* (2000). However, this approach is still popular because it is believed to be simple, intuitive, and reliable. Thus it is necessary to summarize here the main limitations and misinterpretations of the properties of this approach.

First, there are fundamental problems with determining correct weights. One can easily observe via a simple example that weights (which always attempt to reflect a relative importance of criteria) must be in this approach defined as parameters that are constant for the whole Pareto set. However, the weights are actually very different in various areas of a Pareto set. To illustrate this point let us consider two minimized criteria:

- q_1 , costs of emission reduction, and
- q_2 , a measure of a concentration of pollution,

and the corresponding weights w_1 and w_2 . For two-criteria examples it is enough to consider the ratio:

$$\alpha = w_1/w_2. \quad (3.14)$$

Typically, when q_1 attains its best value (which corresponds to a minimum cost solution, which results in a high value of q_2) the value of α will be rather low, indicating much higher weight attached to the environmental criterion. An application of a very small α for the example presented in *Figure 3.7a* would result in the solution **A**. Conversely, for a best available purification technology the q_2 will attain minimum, which also corresponds to highest costs. In such a situation α will take a rather high value corresponding to a much higher weight attached to the economic criterion and the selected solution will be at point **C**.

Second, a weighted aggregation of criteria is a very unreliable way of scanning a set of Pareto solutions. Consider the simplest case with two minimized objectives illustrated in *Figure 3.7a*. For the linear case, a user can obtain only Pareto-optimal solutions corresponding to vertices **A**, **B**, and **C**. For any weighting coefficients vector α with a slope smaller than the slope of the vector α^1 , a solution will be in the vertex **A**. For a weighting coefficient vector that is parallel to α^1 , there is no unique solution,¹¹ and a very small increase of the slope of α will cause the solution to

¹¹Therefore the corresponding problem will be degenerated and any solution from the edge **AB** is optimal. Hence, the reported solution will differ, depending not only on the applied solver but also on the parameters used for a solver, including the possibly defined starting point for optimization.

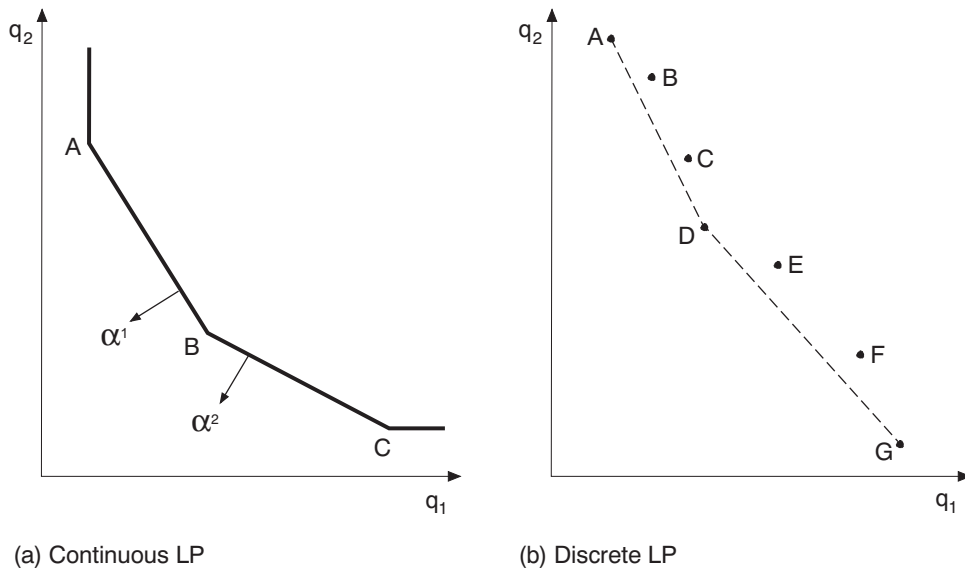


Figure 3.7. Limitations of selecting all Pareto solutions by aggregating criteria through their weighted sum: the cases of continuous (a) and discrete (b) linear models.

jump to the vertex **B**. A further increase in the slope of α will not cause any changes in the Pareto solution until the slope becomes greater than α^2 (which will cause another jump to the vertex **C**). This explains the experience known to everyone who has tried to use weights to analyze multiple-criteria LP models, namely, that often a relatively large change of weights does not cause any changes to the solution, but for some combinations of weights, a small modification creates in the same model a substantially (in practice the distances between vertices are often large) different solution.

Third, a weighted aggregation of criteria does not allow us to find all Pareto solutions. For a discrete model, a surface spanned over the Pareto set (that is composed of points) may be non-convex. Therefore, for the example depicted in *Figure 3.7b*, only some efficient solutions, namely, **A**, **D**, **G** will be found while possibly many other efficient solutions (e.g., **B**, **C**, **E**, **F**) will never be found, if weights are applied for an aggregation of criteria.

Fourth, contrary to the common belief, using weights can be counterintuitive, as one can find examples in which, for certain regions of the efficient frontier, there is no positive correlation between increasing the weight for a criterion and the corresponding improvement of the criterion value. Nakayama (1994) provides a more formal discussion of this issue illustrated by an example that shows that

there might be no positive correlation between increasing a weight for a criterion and the corresponding improvement of the criterion value.

Other problems and limitations of using weights for aggregating criteria are discussed by K. Hayashi and by A. Mohamed, in Chapters 4 and 13 of this volume, respectively.

Given such serious problems with using weights, there is no justification to use this approach any longer, especially because more reliable and natural approaches to MCMA are easily available.

3.5 Conclusions

Modeling of complex systems does, and will, require various elements of science, craftsmanship, and art (see, e.g., Wierzbicki *et al.*, 2000 for a collection of arguments that supports this statement). Moreover, development and comprehensive analysis of a complex model requires – and will continue to require – a substantial amount of time and resources. The main message of this chapter is to stress the often-forgotten fact that no single modeling paradigm can be successfully used to analyze a complex problem, especially if the results of such an analysis are used to support various elements of real decision-making processes. Several rules must be observed during the specification of a model in order to provide useful results. Also, various techniques of model analysis should be used, rather than just the classical approaches which are focused and driven either by simulation or optimization paradigms. Finally, the use of modular re-usable software tools should also be emphasized. They substantially ease the implementation of DSSs that provide more complete and comprehensive analysis of a problem than do closed systems focused on a specific model-analysis paradigm.

References

- Ackoff, R., 1979, The future of operational research is past, *Journal of OR Society*, **30**(2):93–104.
- Alcamo, J., Shaw, R., and Hordijk, L., eds, 1990, *The RAINS Model of Acidification*, Kluwer Academic Publishers, Dordrecht, Boston, London.
- Amann, A., and Makowski, M., 2000, Effect-focused air quality management, in A. Wierzbicki, M. Makowski, and J. Wessels, eds, *Model-Based Decision Support Methodology with Environmental Applications*, Kluwer Academic Publishers, Dordrecht, Netherlands, pp. 367–398. ISBN 0-7923-6327-2.
- Chapman, C., 1992, My two cents worth on how OR should develop, *Journal of Operational Research Society*, **43**(7):647–664.
- Charnes, A., and Cooper, W., 1967, *Management Models and Industrial Applications of Linear Programming*, J. Wiley & Sons, New York, London.

- Drud, A., 1997, Interactions between nonlinear programming and modeling systems, *Mathematical Programming*, **79**(1-3):99–123.
- Goodwin, P., and Wright, G., 1991, *Decision Analysis for Management Judgment*, J. Wiley & Sons, Chichester, New York.
- Granat, J., and Makowski, M., 1998, ISAAP – Interactive Specification and Analysis of Aspiration-Based Preferences, *Interim Report IR-98-052*, International Institute for Applied Systems Analysis, Laxenburg, Austria. Available on-line from <http://www.iiasa.ac.at/~marek/pubs>
- Granat, J., and Makowski, M., 2000, Interactive Specification and Analysis of Aspiration-Based Preferences, *European Journal of Operations Research*, **122**(2):469–485. Available also from International Institute for Applied Systems Analysis, Laxenburg, Austria, as RR-00-09.
- Granat, J., and Wierzbicki, A. P., 1994, Interactive specification of DSS user preferences in terms of fuzzy sets, *Working Paper WP-94-29*, International Institute for Applied Systems Analysis, Laxenburg, Austria.
- Granat, J., Makowski, M., and Wierzbicki, A., 2000, Optimization tools, in A. Wierzbicki, M. Makowski, and J. Wessels, eds, *Model-Based Decision Support Methodology with Environmental Applications*, Kluwer Academic Publishers, Dordrecht, Netherlands, pp. 167–214. ISBN 0-7923-6327-2.
- Korhonen, P., and Wallenius, J., 1989, Observations regarding choice behaviour in interactive multiple criteria decision-making environments: An experimental investigation, in A. Lewandowski and I. Stanchev, eds, *Methodology and Software for Interactive Decision Support*, Vol. 337 of *Lecture Notes in Economics and Mathematical Systems*, Springer Verlag, Berlin, New York.
- Koziol, Q., and Matzke, R., 1998, *HDF5 – A New Generation of HDF: Reference Manual and User Guide*, National Center for Supercomputing Applications, Champaign, Illinois, USA, <http://hdf.ncsa.uiuc.edu/nra/HDF5/>.
- Lewandowski, A., and Wierzbicki, A., eds, 1989, *Aspiration Based Decision Support Systems: Theory, Software and Applications*, Vol. 331 of *Lecture Notes in Economics and Mathematical Systems*, Springer Verlag, Berlin, New York.
- Makowski, M., 1994a, Design and implementation of model-based decision support systems, *Working Paper WP-94-86*, International Institute for Applied Systems Analysis, Laxenburg, Austria.
- Makowski, M., 1994b, Methodology and a modular tool for multiple criteria analysis of LP models, *Working Paper WP-94-102*, International Institute for Applied Systems Analysis, Laxenburg, Austria. Available on-line from <http://www.iiasa.ac.at/~marek/pubs>
- Makowski, M., 2000, Modeling paradigms applied to the analysis of European air quality, *European Journal of Operations Research*, **122**(2):219–241. Available also as from International Institute for Applied Systems Analysis, Laxenburg, as RR-00-06.
- Makowski, M., and Somlyódy, L., 2000, River basin water quality management, in A. Wierzbicki, M. Makowski, and J. Wessels, eds, *Model-Based Decision Support Methodology with Environmental Applications*, Kluwer Academic Publishers, Dordrecht, Netherlands, pp. 311–332. ISBN 0-7923-6327-2.

- March, J., and Simon, H., 1958, *Organizations*, J. Wiley & Sons, New York.
- Nakamori, Y., and Sawaragi, Y., 2000, Complex systems analysis and environmental modeling, *European Journal of Operations Research*, **122**(2):178–189.
- Nakayama, H., 1994, Aspiration level approach to interactive multi-objective programming and its applications, *Working Paper WP-94-112*, International Institute for Applied Systems Analysis, Laxenburg, Austria.
- Ogryczak, W., and Lahoda, S., 1992, Aspiration/reservation-based decision support — a step beyond goal programming, *Journal of Multi-Criteria Decision Analysis*, **1**(2):101–117.
- Olendrzyński, K., Berge, E., and Bartnicki, J., 2000, EMEP eulerian acid deposition model and its application, *European Journal of Operations Research*, **122**(2):426–439.
- Ryoke, M., Nakamori, Y., Heyes, C., Makowski, M., and Schöpp, W., 2000, A simplified ozone model based on fuzzy rules generation, *European Journal of Operations Research*, **122**(2):440–451. Available also from International Institute for Applied Systems Analysis, Laxenburg, Austria, RR-00-07.
- Sawaragi, Y., and Nakamori, Y., 1991, An interactive system for modeling and decision support – Shinayakana system approach, in M. Makowski and Y. Sawaragi, eds, *Advances in Methodology and Applications of Decision Support Systems, Collaborative Paper CP-91-17*, International Institute for Applied Systems Analysis, Laxenburg, Austria.
- Sawaragi, Y., Nakayama, H., and Tanino, T., 1985, *Theory of Multiobjective Optimization*, Academic Press, New York.
- Schöpp, W., Amann, M., Cofala, J., and Klimont, Z., 1999, Integrated assessment of European air pollution emission control strategies, *Environmental Modelling and Software*, **14**(1):1–9.
- Wessels, J., and Wierzbicki, A., eds, 1993, *User-Oriented Methodology and Techniques of Decision Analysis and Support*, Vol. 397 of *Lecture Notes in Economics and Mathematical Systems*, Springer Verlag, Berlin, New York.
- Wierzbicki, A., 1977, Basic properties of scalarizing functionals for multiobjective optimization, *Mathematische Operationsforschung und Statistik, s. Optimization*, **8**:55–60.
- Wierzbicki, A., 1982, A mathematical basis for satisficing decision making, *Mathematical Modelling*, **3**(5):391–405.
- Wierzbicki, A., 1986, On the completeness and constructiveness of parametric characterizations to vector optimization problems, *OR Spektrum*, **8**:73–87.
- Wierzbicki, A., Makowski, M., and Wessels, J., eds, 2000, *Model-Based Decision Support Methodology with Environmental Applications*, Series: Mathematical Modeling and Applications, Kluwer Academic Publishers, Dordrecht. ISBN 0-7923-6327-2.
- Wolfram, S., 1996, *The Mathematica Book, Third Edition, Mathematica Version 3*, Cambridge University Press, Cambridge.
- Zimmermann, H., 1985, *Fuzzy Set Theory – and Its Applications*, Kluwer Academic Publishers, Boston, Dordrecht, Lancaster.

