Short Software Descriptions

A. Lewandowski

December 1988
WP-88-109

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
A-2361 Laxenburg, Austria

Foreword

This paper presents briefly the software for interactive decision support and software tools for developing decision support systems, developed in years 1985 – 1988 within the contracted study agreement between the System and Decision Sciences Program at IIASA and several Polish scientific institutions, including the following:

- Institute of Automatic Control, Warsaw University of Technology,
- Institute of Systems Research, Polish Academy of Sciences,
- Institute of Informatics, Warsaw Univesrity,
- Institute of Automatics, Academy of Mining and Metalurgy, Krakow.

The theoretical part of the results developed within the project is presented in the IIASA Working Paper WP-88-071 entitled *Theory, Software and Testing Examples in Decision Support Systems.* This volume contains also the theoretical and methodological bacgrounds of the software systems developed within the project. These are presented shortly in this paper.

Detailed software descriptions and user manuals have been published as separate IIASA Working Papers – each Working Paper describes one software product and the title of such a paper corresponds to the title of the section in this paper.

All software products in executable form are available to educational and scientific institutions, assuming that these products will not be used for commercial applications. Inquires for software should be directed to the System and Decision Sciences Program at IIASA, Methodology of Decision Analysis Project.

Alexander B. Kurzhanski
Chairman
System and Decision Sciences Program

Software for Interactive Decision Support Short Descriptions

Andrzej Lewandowski
International Institute for Applied Systems Analysis, Laxenburg, Austria
Andrzej Wierzbicki
Institute of Automatic Control, Warsaw University of Technology

Contents

IAC–DIDAS–L
A Dynamic Interactive Decision Analysis and Support System for Multicriteria Analysis of Linear and Dynamic Linear Models on Professional Microcomputers

Tadeusz Rogowski, Jerzy Sobczyk, Andrzej P. Wierzbicki

Institute of Automatic Control, Warsaw University of Technology.

1 Purpose of the packages

IAC–DIDAS–L1 and L2 are two versions of software packages – prototype versions of decision analysis and support systems – for interactive, multiobjective analysis for *models describing substantive aspects of a decision situation* that are represented *in linear programming or dynamic linear programming form.* Such models are often used to represent situations of complex decisions involving economic, environmental and technological aspects. Models of this type must be formulated by teams of analysts specialized in a given field, before they can be helpful to decision makers. Once formulated, however, such models can be used in many ways to learn about decision options and predicted results.

2 Methodological background

IAC–DIDAS–L systems *can be used either by analysts* specialized in modelling *or by decision makers* experienced in a given field but not necessarily computer specialists. These systems help in organizing work with decision models in a process of interactive, dynamic decision support. First, they help in model edition and initial analysis; then in the formulation of a multiobjective decision analysis problem; then in the initial assessment of bounds of decision outcomes or objectives for a given problem.

A model of multiobjective linear programming type is characterized by its decision variables, its outcome variables defined by linear model equations, and its constraints or bounds on various variables. In multiobjective analysis, *the user can select objective variables* (mostly among outcome variables) *that might be minimized, maximized or stabilized* close to given values. This constitutes the definition of multiobjective analysis problem. For a given problem, the package can help in computing bounds on efficient (multiobjectively optimal) values of the objective variables and in suggesting a compromise efficient solution. All this is done in IAC–DIDAS systems by special multiobjective optimization methodology and special optimization solvers; but *the user* needs not to be bothered with these specialized details, because he *can influence the choice of an efficient solution* (decisions and outcomes) *by specifying his reference point or aspiration levels* for the objective outcomes he has determined, and asking the system to find the efficient solution that matches his aspirations most closely.

An interactive multiobjective analysis of the problem based on the principle of reference point optimization is performed, while the user (the analyst or the decision maker) indicates the type of solutions that he is interested in by specifying his aspiration levels for objective outcomes (or for their trajectories in the dynamic case) and the decision support system responds to his directions by solving a special optimization problem and answering, whether his aspirations are attainable. If not, the system proposes decisions with outcomes that come uniformly as close as possible to the stated aspiration levels. If the aspiration levels are attainable but cannot be exceeded, the system proposes decisions with outcomes that precisely match the aspiration levels. If the aspiration levels can be exceeded, the system proposes decisions with outcomes that uniformly exceed the aspirations. By changing the aspiration levels, ***the user can easily control and select such decision options that are best suited for his preferences.***

Finally, the system can also help in a post-optimal analysis of a selected decision option by examining various options that are close to it. All results of analysis can be illustrated graphically on the monitor screen. The user can also have print-outs of selected results of analysis.

The system helps also the user to keep track of consecutive results of analysis and stores results marked by the user as important in a special ***result directory***. Because the user can formulate various multiobjective decision problems for given model of substantive aspects of the decision situation (by maximizing or minimizing multiobjective various model outcomes, etc.), the system keeps also track of various problem formulations. Finally, the system allows also to work with various models of one or more decision situations; thus, the system has also ***model and problem directory***.

The system includes two demonstrative models with some problem formulations and illustrative results of their analysis; by watching these demonstrations, the user can easily learn how to work with the system. One of the demonstrative examples used in both versions illustrates ***a diet composition problem*** and the other example depends on the system version. In the version L1 that includes the possibility of working with dynamic models but uses a more specialized format of model edition, ***an example of controlling two reservoirs on a river*** illustrates dynamic problems of decision analysis. In the version L2 that is essentially limited to static models but has an easy and user-friendly spreadsheet format of model edition, ***an example from agricultural economics*** illustrates the possibilities of the system.

3 Short program description

IAC–DIDAS–L1 and L2 are software packages from the class of ***prototypes*** of interactive multiobjective decision analysis and support systems. When supplemented by a model of the substantive aspects of a decision situation (involving economic, environmental, technological etc. aspects), they can serve both as tools of model analysis used by specialists and as decision support systems used by decision makers that are experienced in a given substantive field (but not necessarily computer specialists) that want to investigate various aspects of many decision options. These packages are developed to the level of ***scientific transferable software***, that is, are tested and documented enough to be used widely in research. The IAC–DIDAS–L packages are designed for use with models of multiobjective linear programming and dynamic linear programming type.

IAC–DIDAS–L1 is written in FORTRAN, contains a special linear programming optimization ***solver*** for multiobjective problems, that results in relative fast execution of optimization runs during interactive analysis but requires the preparation of the model of substantive aspects of decision situation in the MPS format (standard for linear programming specialists but not necessarily easy for an average user); on the other hand, this version allows also for an easy edition and analysis of dynamic models.

IAC-DIDAS-L2 is written in PASCAL, contains another version of linear programming optimization solver adapted for multiobjective problems and supports an interactive definition and edition of the model of substantive aspects of decision situation in a user-friendly format of a spreadsheet.

Both versions have ***user interfaces*** with helps displaying various commands used depending on the stage of decision analysis process. Both versions have ***data bases with directories*** of various models, multiobjective analysis problem formulations and various results of analysis. Both versions have ***graphic interfaces*** helping to illustrate results of analysis.

4 Hardware requirements

Both IAC-DIDAS-L1 and L2 are implemented on professional microcomputers compatible with IBM-PC/XT (with a hard disk, Hercules or color graphics or EGA card and, preferably, a coprocessor) or a similar IBM-PC/AT configuration.

HYBRID — A Mathematical Programming Package for Multicriteria Dynamic Linear Problems

Short Program Description of version 3.1.

Marek Makowski [*]

International Institute for Applied Systems Analysis, Laxenburg, Austria

Janusz S. Sosnowski

Systems Research Institute of the Polish Academy of Sciences, Warsaw.

1 Purpose of the package.

HYBRID 3.1. is a mathematical programming package which includes all the functions necessary for the solution of linear programming problems, both static and dynamic (in fact also for problems with structure more general then the classical formulation of dynamic linear problems). HYBRID 3.1. may be used for both single- and multi-criteria problems. The package may be also used for solving single-criteria linear-quadratic problems. Since HYBRID is designed for real-life problems, it offers many options useful for diagnostics and verification of a model for a problem being solved.

HYBRID is a member of DIDAS family of decision analysis and support systems which is designed to support multicriteria analysis via reference point optimization . HYBRID can be used by an analyst or by a team composed of a decision maker and an analyst or—on last stage of application—by a decision maker alone. HYBRID is a tool which helps to choose a decision in a complex situation in which many options may and should be examined. Possible applications include problems of economic planning and analysis, management, technological or engineering design problems, problems of environmental control.

2 Methodological and theoretical backgrounds.

A multicriteria problem is transformed by HYBRID to an equivalent single criterion problem. Therefore a multicriteria problem is solved as a sequence of parametric optimization problems modified by a user in interactive way upon analysis of previous results.

HYBRID uses a non-simplex algorithm - a particular implementation of the Lagrange multiplier method - for solving linear programming problems. General linear constraints are included within an augmented Lagrangian function. The LP problem is solved by minimizing a sequence of quadratic functions subject to simple constraints (lower and upper bounds). This minimization is achieved by the use of a method which combines the conjugate gradient method and an active constraints strategy. The method exploits the sparseness of the matrix

[*]on leave from the Systems Research Institute of Polish Academy of Science, Warsaw.

structure. A dynamic problem is solved through the use of adjoint equations and by reduction of gradients to control subspaces. The simple constraints (lower and upper bounds for non-slack variables) for control variables are not violated during optimization and the resulting sequence of multipliers is feasible for the dual problem. Constraints other then those defined as simple constraints may be violated, however, and therefore the algorithm can be started from any point that satisfies the simple constraints.

3 Description of implementation.

HYBRID is coded partly in C language and partly in an extension of Fortran-77 (functions coded in Fortran, after processing by C preprocessor, conform to ANSI standard of Fortran-77). The PC version has also a user friendly driver (with context sensitive help) and functions that ease analysis of a solution and modification of parameters of multicriteria problem.

4 Hardware requirements.

HYBRID 3.1. may be used on VAX 780/11 and on IBM-PC XT/AT or compatibles (with any graphic card). The PC version for small problems requires 256kB RAM, for larger problems a configuration with more RAM is needed. The coprocessor is strongly recommended but a demo version of HYBRID is available also for a PC configuration without coprocessor.

IAC–DIDAS–N
A Dynamic Interactive Decision Analysis and Support System for Multicriteria Analysis of Nonlinear Models

Tomasz Kreglewski, Jerzy Paczynski, Janusz Granat, Andrzej P. Wierzbicki
Institute of Automatic Control, Warsaw University of Technology.

1 Purpose of the package

The decision analysis and support systems of DIDAS family—that is, Dynamic Interactive Decision Analysis and Support systems —are specially designed to support interactive work with a substantive model of a decision situation while using multicriteria optimization tools. They stress the learning aspects of such work, such as the right of a decision maker to change his priorities and preferences when learning new facts. DIDAS systems can be used either by analysts who want to analyse their substantive models, or by teams of analysts and decision makers, or even by decision makers working alone with a previously defined substantive model; in any case, we shall speak further about *the user* of the system.

There are several classes of substantive models that all require special technical means of support. The IAC–DIDAS–N version is designed to support models of multiobjective nonlinear programming type. Models of this type include two classes of variables: ***input variables*** that can be subdivided into ***decision variables*** (means of multiobjective optimization) and ***parametric variables*** (model parameters that are kept constant during multiobjective analysis but might be changed during parametric or sensitivity analysis)—and ***outcome variables*** that can be subdivided into several types, the most important of them being ***optimized outcomes*** or ***objectives*** (the ends of multiobjective optimization that can be either maximized or minimized or stabilized, that is, kept close to a desired level). The user might change the classification of outcome variables and select his objectives among various outcome variables when defining an multiobjective analysis problem.

For all input and outcome variables, a reasonably defined nonlinear model should include *lower* and *upper bounds*, that is, reasonable ranges of admissible changes of these variables. Moreover, an essential part of a nonlinear model definition are ***model equations***, that is, nonlinear functions that define the dependence of all outcome variables on input variables. To make the model definition easier for the user, it is assumed that outcome variables are defined consecutively and that they can depend not only on input variables, but also on previously defined outcome variables.

The IAC–DIDAS–N system helps in definition, edition, initial analysis and verification, optimization and multiobjective decision analysis of a rather broad class of nonlinear models. An important feature of IAC–DIDAS–N is that it supports also automatic calculations of all derivatives of nonlinear model functions.

2 Methodological background

A typical procedure of working with the IAC–DIDAS–N system consists of several phases.

In the first phase, a user—typically, an analyst—defines the substantive model and edits it on the computer. IAC–DIDAS–N supports the definition and edition of substantive models in an easy but flexible standard format of a spreadsheet, where the input variables correspond to spreadsheet columns and the outcome variables—to spreadsheet rows; special cells are reserved for various additional information. Another new feature of IAC–DIDAS–N is a symbolic differentiation facility that supports automatic calculations of all derivatives required by a nonlinear programming algorithm. The user does not need to laboriously calculate many derivatives and to check whether he did not make any mistakes; he only defines model equations or outcome functions in a form that is acceptable for the symbolic differentiation program—which admits functions from a rather wide class. The spreadsheet format allows also for display of computed values of automatically determined formulae for derivatives in appropriate cells. The user of IAC–DIDAS–N can also have several substantive models recorded in special model directories, use old models to speed up the definition of a new model, etc., while the system supports automatically the recording of all new or modified models in an appropriate directory.

In further phases of work with DIDAS-type systems, the user—here typically an analyst working together with the decision maker—specifies a multiobjective analysis problem related to his substantive model and participates in an initial analysis of this problem. There might be many multiobjective analysis problems related to the same substantive model: the specification of a multiobjective problem consists in designating types of model outcomes, especially objective outcomes that shall be optimized, and specifying bounds on outcomes. For a given definition of the multiobjective analysis problem, the decision and outcomes in the model are subdivided into two categories: those that are *efficient* with respect to the multiobjective problem (that is, such that no objective can be improved without deteriorating some other objective) and those that are inefficient. It is assumed that the user is interested only in efficient decisions and outcomes (this assumption is reasonable provided he has listed all objectives of his concern; if he has not, or if some objectives of his concern are not represented in the model, he can still modify the sense of efficiency by adding new objectives, or changing the types of objectives).

One of main functions of DIDAS-type systems is to compute efficient decisions and outcomes – following interactively various instructions of the user – and to present them to the user for analysis. This is done by using the principle of reference point optimization – that is, solving a special parametric nonlinear programming problem resulting from the specification of the multiobjective analysis problem; for this purpose, IAC–DIDAS–N contains a specialized nonlinear programming algorithm called solver. Following the experiences with previous versions of nonlinear DIDAS systems, a special robust nonlinear programming algorithm was further developed for IAC–DIDAS–N.

The main phase of work with the IAC–DIDAS–N system consists in interactive scanning of efficient outcomes and decisions, guided by the user through specifying two *reference points* called *reservation point* and *aspiration point* in the objective space, i.e. *reservation levels* and *aspiration levels* for each objective; the system admits also the more simple option of specifying either only an aspiration level or only a reservation level for some objectives. The user can get additional information about the range of possible outcomes during so called initial analysis of multiobjective problem and thus he can reasonably specify his reference levels: aspiration level that he would like to attain and reservation level that he would like to satisfy in any case. The system suggests some initial values for both reference points.

IAC–DIDAS–N utilizes the aspiration and the reservation levels as parameters in a special achievement function coded in the system, uses its solver to compute the solution of a nonlinear programming problem equivalent to maximizing this achievement function, and responds to the user with an attainable, efficient solution and outcomes that strictly correspond to the user-specified references (in the sense of being possibly close to the aspirations if they are unattainable, and uniformly better than aspirations if they are attainable).

3 Short program description

The IAC–DIDAS–N system (Institute of Automatic Control, Dynamic Interactive Decision Analysis and Support, Nonlinear version) is a decision support system designed to help in the analysis of decision situations where a mathematical model of substantive aspects of the situation can be formulated in the form of a multiobjective nonlinear programming problem.

The system supports the following general functions:

- definition and edition of a substantive model of the decision situation in a user-friendly format of a spreadsheet.
- specification of a multiobjective decision analysis problem related to the substantive model.
- initial multiobjective analysis of the problem, resulting in estimating bounds on efficient outcomes of decisions and in learning about some extreme and some neutral decisions.
- interactive analysis of the problem with the stress on learning possible efficient decisions and outcomes, organized through system's response to user-specified aspiration and reservation levels for objective outcomes. The system responds with efficient objective outcomes obtained by maximization of an achievement function that is parameterized by the user-specified aspirations and reservations.

4 Hardware requirement

The program can be run on an IBM–PC–XT, AT or a compatible computer with Hercules Graphics Card, Color Graphic Adapter or Enhanced Graphics Adapter and, preferably, with a numeric coprocessor and a hard disk. If a numeric coprocessor is available then the coprocessor version of the system can be used taking advantage of the coprocessor computational capacity, otherwise only the emulation version of the system can be used with less computational capabilities. The system programs are recorded on two diskettes. Each diskette contains compiled code of one version of the system together with some data files with demonstrative examples of nonlinear models.

DISCRET
An Interactive Decision Support System for Discrete Alternatives Multicriteria Problems

Janusz Majchrzak

Systems Research Institute, Polish Academy of Sciences, Warsaw.

1 Executive summary

DISCRET is a system created to solve basic multicriteria choice problems in which a finite set of feasible alternatives (and decisions) is explicitly given and for each alternative the value of all criteria describing its attributes interesting to the decision maker (DM) were evaluated and listed. The DM is assumed to be rational in the sense that he is looking for a Pareto-optimal solution as his final solution of the problem.

Such a decision problem is rather trivial for any human being as long as the number of criteria and alternatives is small (say, 3–5 criteria, 7–15 alternatives) However, if the number of alternatives and/or criteria grows, the limits of human information processing capabilities are reached and some decision support facilities have to be utilized to guarantee a proper and efficient decision making.

In many real-life problems the decision variables take their values from a discrete set rather than from a continuous interval. Usually there is a finite number of available facility location sites, the facility size or production capability may be chosen from a discrete set of values, during a design process the components are chosen from a set of typical elements available on the market, etc. Such problems form the "natural" field of applications for the DISCRET system.

Another field of possible applications of the DISCRET system consists of cases in which the original problem is actually continuous (rather than discrete) but the analysis restricted just to a finite number of alternatives appearing in this problem may be interesting and useful for the DM, since it may result in an enlightening and a more precise definition of his preferences, region of interest or aspiration levels.

Situations falling under the latter category may occur for at least two following reasons. Firstly, a sample of alternatives together with the corresponding criteria values may be readily available (from simulation model runs for example). Secondly, for the purpose of an initial analysis the DM may take into considerations just a few values for each decision variable or generate a random sample of alternatives.

The DISCRET system makes no restrictions on the forms of the criteria. Therefore, attributes as complicated as required may be considered.

To start the session with DISCRET the DM has to supply the file containing set of the criteria values for all feasible alternatives, the problem specification file and (optionally) the file containing the set of feasible decisions. These files, called the data, specification and the additional data file respectively, describe the problem under consideration.

After loading the problem the DM may obtain the information about the criteria values ranges and he may put the lower and/or upper bounds on the values of some/all criteria. The bounds setting may be utilized to eliminate irrelevant alternatives from further considerations or to specify the current region of interest in the objective space.

In the next step the DM may eliminate the dominated alternatives by an explicit enumeration technique. The tolerances for criteria values play an important role here. If they are all equal zero or have small positive values that correspond to indifference limits of the DM's for criteria values, the whole set of the nondominated solutions will be obtained. If the values of tolerances are equal to some significant fractions of the corresponding criteria ranges, then a representation of the set of nondominated solutions will be obtained. The representation is a subset of the set of nondominated solutions preserving its shape and containing the smaller number of elements, the larger were chosen tolerance coefficients.

The biggest advantage of the implemented enumeration method is its ability to select a representation of the nondominated set instead of the entire set. Unlike other known approaches which find the entire nondominated set first and then select a representation (differently defined for each of those methods), the presented method selects a representation at once. This fact profits in efficiency. Observe that because the representation contains less elements than the nondominated set, it will be obtained with a smaller computational effort.

The powerful tool of the reference point approach is also available for the DM. By determining a reference point, he exhibits his aspiration levels for criteria values, confronts them with the obtained solution and modifies them and the reference point. To learn more about the problem the graphic display of two–dimensional subproblems on the terminal screen can be utilized. The DM chooses two criteria for the vertical and horizontal axes, while the other criteria are restrictively bounded — a two–dimensional "slice" is cut out of the original m–dimensional problem. The graphical displays are very useful on this stage of the decision making process, since the DM can see clusters (groups) of alternatives.

DISCRET is an interactive system. The DM may execute its commands in any order. The variety of paths the DM may follow guarantees flexibility in meeting his demands. The implemented approach seems to be easy to understand and approve even for a user who is not very familiar with multicriteria optimization techniques.

2 Short program description

The interactive decision support system DISCRET has been designed to solve medium–size discrete multicriteria choice problems with the number of alternatives ranging from few hundreds to few thousands. The number of criteria is in the current version restricted to 20 (mainly due to the limitations of display facilities).

The program is recorded on a diskette(s) and should be installed on an IBM–PC–XT/AT or a compatible computer with a Color Graphic Adapter, Enhanced Graphic Adapter or Hercules Graphic Card and a hard disk. The compiled code is distributed together with a number of files it requires and with two test problems generators providing demonstrative examples.

The system supports the following menu–controlled general functions:

— Loading user's problems in an easy to prepare standard of an ASCII file form.

— Criteria values ranges (utopia and nadir points) display, and new criteria values bounds setting to define the user's current region of interest.

— Solving the discrete multicriteria optimization problem with explicit alternatives (implicit constraints), i.e. finding the set of nondominated or weakly nondominated elements or it's representation, keeping or rejecting duplicate elements, etc.

— The reference point approach, i.e. selection of nondominated alternatives that correspond to user-supplied aspiration levels for criteria.

— Graphic display of the two–dimensional "slices" of the problem showing the user alternatives clusters (groups).

DINAS
Dynamic Interactive Network Analysis System

Wlodzimierz Ogryczak, Krzysztof Studzinski, Krystian Zorychta
Institute of Informatics, Warsaw University.

1 Purpose of the system

DINAS is a scientific transferable software tool which enables the solution of various multiobjective transshipment problems with facility locations. For a given number of fixed facilities and customers and for a number of potential facilities to be optionally located, DINAS provides the user with a distribution pattern of a homogeneous product under a multicriteria optimality requirement. While working in an interactive mode, the user gets optimal locations of the potential facilities and a system of optimal flows of the product between nodes of the transportation network. With DINAS one can analyse and solve such problems as:

- the transportation problem with new supply and/or demand points location,
- the problem of warehouses location,
- the problem of stores location for agricultural production,
- the problem of service centers location and districts reorganization,

and many other real-life distribution-location problems.

2 The problem statement

DINAS works with problems formulated as multiobjective transshipment problems with facility location. A network model of such a problem consists of nodes connected by a set of direct flow arcs. The set of nodes is partitioned into two subsets: the set of fixed nodes and the set of potential nodes. The fixed nodes represent "fixed points" of the transportation network, i.e., points which cannot be changed, whereas the potential nodes are introduced to represent possible locations of new points in the network. Some groups of the potential nodes represent different versions of the same facility to be located (e.g., different sizes of warehouse etc.). For this reason, potential nodes are organized in the so-called selections, i.e., sets of nodes with the multiple choice requirements. A homogeneous good is distributed along the arcs among the nodes. Each fixed node is characterized by two quantities: supply and demand on the good, Each potential node is characterized by a capacity which bounds maximal good flow through the node. The capacities are also given for all the arcs but not for the fixed nodes. A few linear objective functions are considered in the problem. The objective functions are introduced into the model by given coefficients associated with several arcs and potential nodes (the so-called cost coefficients, independently of their real character). The cost coefficient connected to an arc is treated as the unit cost of the flow along the arc. The cost coefficient connected to a potential node is considered as the fixed cost associated with locating of the node (e.g., an investment cost). Summarizing, the following groups of input data define the transshipment problem under consideration:

- objectives,
- fixed nodes with their supply-demand balances,
- potential nodes with their capacities and (fixed) cost coefficients,
- selections with their lower and upper limits on number of active potential nodes,
- arcs with their capacities and cost coefficients.

The problem is to find the best (satisfying) location and flow scheme, i.e., to determine the number and locations of active potential nodes and to find the good flows (along arcs) so as to satisfy the balance and capacity restrictions.

3 Methodological and theoretical backgrounds

DINAS does not solve the multiobjective problem. It rather makes the user selecting the best solution during interactive work with the system. According to some user's requirements DINAS generates various efficient solutions which can be examined in details and compared to each other. The user works with the computer in an interactive way so that he can change his requirements during the sessions. The DINAS interactive procedure utilizes an extension of the reference point optimization. The basic concept of that approach is as follows:

- the user forms his requirements in terms of aspiration and reservation levels, i.e., he specifies acceptable and required values for given objectives;
- the user works with the computer in an interactive way so that he can change his aspiration and reservation levels during the sessions.

DINAS searches for the satisfying solution while using an achievement scalarizing function as a criterion in single-objective optimization.

4 Description of the implementation

DINAS is prepared as a menu-driven and easy in usage system. The system consists of three programs prepared in the C programming language:

- the network screen editor for friendly data input and results examination;
- the solver for single-objective optimization;
- the main interactive procedure for handling multiple objectives.

The basic version of the system is capable of solving problems with seven objective functions, about one hundred of fixed nodes, a few hundreds of arcs, and fifteen potential nodes organized in a few selections.

5 Hardware requirements

DINAS runs under Disk Operating System (DOS version 3.00 or higher) on an IBM-PC XT/AT or compatibles equipped with Hercules, EGA or CGA card and requires 640 K RAM. A hard disc is recommended but not necessary. One double-sided double-density floppy disk drive is sufficient to run DINAS. The system can be installed in two versions: with taking advantages of the Numeric Data Processor chip, or without using the NDP chip. However, for solving large real-life problems the version with the NDP chip is strongly recommended. A printer is useful but not necessary since all the reports can be routed directly to a printer or to a disk file for printing at a later time.

MCBARG
A System Supporting Multicriteria Bargaining

P. Bronisz, L. Krus, B. Lopuch

Systems Research Institute, Polish Academy of Sciences, Warsaw.

1 Executive summary

Many aspects of economic, environmental, or technological activity are influenced directly by bargaining between and among individuals, firms, and nations ("players"). In the pure bargaining problem, considered in the MCBARG system, the bargaining conditions are determined entirely by the bounds of discussion, within which the final outcome is determined by the interaction of the players. Even in the case of one individual, firm, and nation, there are many situations of complex decision, the decision maker needs help to learn about possible decision options, decision consequences. The MCBARG system enables learning process of the players, and supports reaching the final outcome in the multicriteria bargaining problem. It is based on the theoretical results presented in Bronisz, Krus, Wierzbicki (this volume).

The multicriteria bargaining problem is a generalization of classical bargaining problem, under assumption that the utility functions of participants are not given explicitly. This generalization follows from the fact that an aggregation of participant's or player's objectives is often impossible, because of various practical limitations of the utility theory. The problem is defined by an agreement set — the set of outcomes that can be reached under unanimous agreement of the players, and by a disagreement (status quo) point which is a result of the problem if there is no such an agreement.

The proposed interactive process consists in generation of sequence of outcomes leading to a nondominated solution. The process is based on limited confidence principle, taken from practical observation, which says that the players have limited confidence in their ability to predict consequences and possible outcomes, hence each player tries to prevent other players from receiving disproportionally large gains. The generated outcomes are consistent with preferences of the players. The process assures also some fairness rules and is resistent to various manipulations of the players .

The algorithm consists of a number of rounds. Each round starts at the current status quo point (the first round starts from the initial status quo point). At each round the player specifies his confidence coefficient (i.e. defines part of the maximal improvement of the outcomes the counter players can obtain in the round). Furthermore assuming some moves of the other players, he tests different improvement directions for his objectives. This phase of the work with MCBARG system consists in an interactive scanning of outcomes guided by each player through specifying reference points in the objective space. The reference points are composed of aspiration levels of each player for his objectives. The players get additional information about the range of possible outcomes for a given confidence coefficient and some assumed actions of the counter players. This information is useful for reasonable specification of the aspiration levels. The system generates also some initial values for

the aspiration levels and calculates corresponding outcomes (called neutral outcomes). The scanning of the player outcomes is performed in the system through directional optimization and lexicographic improvement of the week Pareto outcomes. The system responds to the player with an attainable, efficient (under the assumed confidence coefficient) outcomes that strictly correspond to the player-specified aspirations. The results obtained for a number of different reference points can be easily compared through scrolling option in both numerical and graphical form. To finish this phase, the player is required to select, according to his preferences, his reference point indicating his preferable improvement direction. These points selected independently by all the players are basis for calculation of the result of the round. The result,is calculated following the limited confidence principle (the minimal confidence coefficient is used for all players) and trying to improve outcomes for all players in the directions specified by their reference point. Thus, the system acts as a neutral mediator proposing a single-test provisional agreement improving the initial situation and forming a basis for next round of negotiations. The results are presented independently to the players in form of report, and the players can go to the next round assuming the obtained result as a new status quo point. The process terminates when an efficient, strictly Pareto-optimal solution in the agreement set is reached.

The system includes a generator and an editor of the model of the bargaining problem for which the interactive process is organized. The model describes the agreement set in form of a set of inequalities, and the status quo point. The generator and the editor enable introducing linear or nonlinear formulas defining the inequalities using standard operators and functions. An illustrative example has been prepared which relates to the problem of cooperation of two farms. The problem consists in division of products resulting from cooperation between two farms, according to the preference of farm owners.

2 Short program description

The MCBARG system is a decision support system designed to help in analysis of decision situation and mediation in multicriteria bargaining problem in which a mathematical model of the problem can be formulated by a status-quo point and a system of inequalities describing agreement set in objective space of the players.

The program is recorded on one diskette that should be installed on an IBM-PC-XT, AT or compatible computer with Hercules Graphics Card, Color Graphic Adapter (CGA) or Enhanced Graphics Adapter (EGA). A diskette contains compiled code of the program together with some data files for a demonstrative example of the bargaining problem.

The system supports the following general functions:

— The definition and edition of a model of the bargaining problem together with the specification of the multicriteria decision problem.

— Interactive mediation by generation of a sequence of outcomes (depending on player-specified aspirations), leading to a nondominated solution in agreement set.

— Report of the final, efficient solution of the problem.

The second function proceeds in a number of rounds and in each round the system supports:

— Initial multiobjective analysis of the bargaining problem for each player, that results in estimating bounds on efficient outcomes and learning about the extreme and neutral outcomes.

— Unilateral, interactive analysis of the problem with the stress on learning, organized through system response to user specified confidence coefficients and aspiration levels for objective outcomes. The systems responds with efficient (under the assumed confidence coefficient) objective outcomes.

— Calculation of the multilateral, cooperative solution of the round. Reporting the results of the already performed rounds.

NOA1: a Fortran Package of Nondifferentiable Optimization Algorithms. Methodological and User's Guide

Krzysztof C. Kiwiel and Andrzej Stachurski

Systems Research Institute, Polish Academy of Sciences,

Newelska 6, 01-447 Warsaw, Poland.

1 Purpose of the package

NOA1 is a package of Fortran subroutines designed to locate the minimum value of a locally Lipschitz continuous function subject to locally Lipschitzian inequality and equality constraints, general linear constraints and simple upper and lower bounds. The user must provide a Fortran subroutine for evaluating the (possibly nondifferentiable and nonconvex) problem functions and their single subgradients. The package implements several descent methods, and is intended for solving small-scale nondifferentiable minimization problems on a professional microcomputer. The following standard optimization problem form is assumed:

$$
\begin{aligned}
\textit{minimize} \quad & \max\{ F_I^U(X) : I = 1,\ldots, MOB \} && (1)\\
\textit{s.t.} \quad & F_I^U(X) \le 0 \ \text{ for } \ I = MOB+1,\ldots, MOB+MI && (2)\\
& F_I^U(X) = 0 \ \text{ for } \ I = MOB+MI+1,\ldots, MOB+MI+ME && (3)\\
& F_I^U(X) \le 0 \ \text{ for } I = MEPF{+}1,\ldots, MEPF{+}MFI && (4)\\
& \langle A_I, X \rangle \le B_I \ \text{ for } I = 1,\ldots, NLINEQ && (5)\\
& X_I^L \le X_I \le X_I^U \ \text{ for } I = 1,\ldots, N && (6)
\end{aligned}
$$

where $MOB \ge 1$, $MI \ge 0$, $ME \ge 0$, $MEPF = MOB + MI + ME$, $MFI \ge 0$, $NLINEQ \ge 0$, and X and A_I are N-vectors. (The two groups of nonlinear inequality constraints are distinguished because they are handled in NOA1 by the exact penalty and feasible point techniques, respectively.)

The user's problem subroutine evaluates the problem functions F_I^U, $I = 1,\ldots, MEPF+MFI$, and their subgradients. Its name must be declared **EXTERNAL** in the main program. The name is arbitrary (but it must differ from the names of NOA1 subroutines.

Some or all of the following items are supplied by the user:

- Main program MAINOA;
- Problem subroutine (called, e.g., USERS);
- Input data file;
- Data read by USERS on its first entry;

- Data read by USERS on its last entry.

The order of the files and data is important if all are stored in the same input stream. Some of the routines may require modification to suit a particular problem or a non-standard application.

The main program allocates the workspace for NOA1 and the user's problem subroutines, opens the primary input and output files (called FORT1 and FORT2), reads the algorithm's parameters and calls NOA1 to solve the problem.

For each problem the user may either provide his own subroutine USERS or insert a calling sequence to his subroutine in subroutine USERS. Then at run-time the problems will be distinguished by the value of the parameter IEXAMP. He must also append the names of his object files to the list of linked files contained in file NOA1.LNK.

The input data file defines various problem and run-time parameters (number of variables, iterations limit, etc.). Its name and unit number are defined at compile time (in the main program). It will normally be the first data set in the system card input stream.

At the end of a run, the solution is stored in the array X, whereas some additional information is stored at certain locations in the workspace arrays. The user may control the output level selecting the appropriate values of the control parameters.

NOA1 runs on IBM PC/XT or AT compatibles under the DOS operating system, version 3.1 or higher. The computer should have at least 512 kB of memory, a hard disk and an 8087 or 80287 mathematical coprocessor. The source code of NOA1 is written in Fortran 77; however, the object files were created by the Lahey Fortran 77 Compiler F77L, version 2.21. The user's subroutines should be compiled by the same compiler.

The NOA1 pacakage is still at an experimental stage, and we intend to increase both its efficiency and user friendliness. Any feedback from the users will be most welcome. The object code, some source code and data forNOA1 are distributed on a floppy disk containing 28 files.

2 Installation procedure

1. Create directories F77L and NOA1 in the root directory.

2. Copy the contents of the distribution diskette to the NOA1 directory.

3. Copy the Lahey F77L compiler and the linker (IBM linker, version 2.30 or higher) to the F77L directory.

4. Make sure the F77L directory is included in the path for DOS.

5. Connect to the NOA1 directory.

6. Create an executable file NOA1.EXE by executing the batch file LNOA1.BAT. This file contains one line

```
..\F77L\link @ NOA1.LNK
```

   It refers to the automatic response file NOA1.LNK (see the DOS manual for information about the stack and segment extensions).

7. Copy the file `QUADR3.DAT` to the file `FORT1`.

8. Run `NOA1` by executing the command `NOA1` (or `NOA1.EXE`). Check the output.

9. If necessary change the data in the `FORT1` file to run different versions of the QUADR problem (constrained, nonconvex, etc.) and to check the influence of certain parameters (EPSTOP, EPSFSB, etc.). If the summary output level MSGSUM is positive and the unit number NSUM=0, certain brief information will be output to the screen. (If NSUM is neither 0 nor NOUT, then a suitable file should be opened in the main program.)

3 A testing example

The files `QUADR.FOR` and `QUADR3.DAT` contain the source code and data for a simple minimax problem which may be used for testing `NOA1`.

POSTAN 3 and PLP
Extensions of MINOS for Postoptimal Analysis

Grzegorz Dobrowolski, Tomasz Rys
Joint System Research Department
of the Institute for Control and Systems Engineering,
Academy of Mining and Metallurgy, Cracow,
and of Industrial Chemistry Research Institute, Warsaw.
Adam Golebiowski, Krystyn Hajduk, Adam Korytowski
Institute for Control and Systems Engineering,
Academy of Mining and Metallurgy, Cracow.

1 General information

POSTAN3 (Dobrowolski et al.), an extended version of POSTAN (Dobrowolski et al. 1984) and POSTAN2 (Dobrowolski et al. 1987), is a postoptimal analysis package for linear and linear-fractional programming problems. It may also be used to solve linear-fractional problem by a direct, noninteractive method. The package is composed of a number of FORTRAN routines which are incorporated into MINOS, the well-known linear and non-linear programming code developed by Murtagh and Sanders (1977). The postoptimal analysis is performed after MINOS has found an optimal solution, and is initiated by adding new specifications to the original list of MINOS specifications.

The main objective of POSTAN3 is ranging, i.e., determining the ranges in which certain parameters (or groups of parameters) may be changed without affecting the optimal solution and/or the optimal basis. Sensitivity coefficients which are not included in the output of the unmodified version of MINOS are also determined.

Two new auxiliary modules have been implemented in POSTAN3 to improve its user interface:

- a module for programming a sequence of optimization problems,
- a module for decoding and selective printing of results.

PLP (Golebiowski) is a software package for parametric linear programming. It is also an extension of MINOS and is initiated by adding some specifications to the original list of MINOS specifications. PLP uses MINOS as the solver of optimization problems. It includes sections which create an interactive framework for parametric programming and perform ranging and housekeeping procedures.

Optionally, PLP gives a complete parametric programming analysis for one, or more, of the following vectors: cost, rhs, and bounds. In the same run, several problems of this kind can be solved and for each, the starting point may be the original optimal solution or the final solution obtained in the last problem.

2 Notes on implementations

The available implementations of the packages can be divided into two groups:

1. Main frame implementations that are destined for large scale optimization problems requiring a powerful computer system. An important parameter of the computer system at this point is operational memory available to a process.

2. Personal computer implementations that can run under restriction with respect to dimensionality of the problem. The minimum hardware configuration is 640 kB of operational memory and a mathematical co-processor.

Main frame

POSTAN	batch mode version. Software requirements: FORTRAN IV-E compiler.
PLP	batch mode version. Software requirements: FORTRAN IV-E compiler.
OPT	interactive mode, screen-oriented, menu-driven version. Software requirements: FORTRAN 77, C Language compilers, UNIX or XENIX operating system. There is a special implementation of POSTAN. In place of advanced postoptimal routines the reference point multiobjective optimization method is incorporated.

Personal computer

POSTAN	batch mode version. Software requirements: FORTRAN 77 compiler.
OPT	interactive mode, screen-oriented, menu-driven version. Software requirements: FORTRAN 77, C Language compilers, DOS or XENIX operating system. There is a special implementation of POSTAN. In place of advanced postoptimal routines the reference point multiobjective optimization method is incorporated.
PCPOST	interactive mode, window-oriented, menu-driven version. Software requirements: FORTRAN 77, C Language compilers, DOS operating system. There is a full implementation of POSTAN.

3 Formulation of linear programming problem

The formulation of the linear problem analyzed by POSTAN3 and PLP is the sane as for MINOS. Minimize (or maximize) a linear cost function

$$F(x) = a_0 x \tag{1}$$

subject to m row constraints:

$$d_i \leq a_i x \leq g_i \ , \quad i = 1, ..., m \tag{2}$$

and n constraints on separate variables:

$$d_{m+i} \leq x_i \leq g_{m+i} \ , \quad i = 1, ..., n \quad . \tag{3}$$

Here x is an n-dimensional column vector of decision variables, a_0 is an n-dimensional row vector of cost coefficients (also called the *objective row*), the $a_i, i = 1, ..., m$, are n-dimensional row vectors, the lower bounds d_i, $i = 1,...,m+n$, are real numbers or $-\infty$, and the upper bounds g_i, $i = 1,...,m+n$, are real numbers or $+\infty$. Of course, if the bounds take the values $+\infty$ or $-\infty$ the corresponding relation (2) or (3) must be replaced by a strict inequality. If $d_i = g_i$, then the variable x_i is said to be *fixed*. If $d_i = -\infty$ and $g_i = +\infty$ the variable x_i is said to be *free*. Analogous terms are used to describe the rows $a_i x$.

It should be recalled that in MINOS the two-sided inequality constraints (2) are not stated explicitly, but rather specified using *ranges*. More precisely, a one-sided inequality is introduced in the form $a_i x \leq g_i$ (type L) or $a_i x \geq d_i$ (type G), together with a real number r_i called the *range*. In the first case, the difference between the right-hand side g_i and this number yields the lower bound $(d_i = g_i - r_i)$; in the second case the sum of the right-hand side d_i and the real number r_i gives the upper bound $(g_i = d_i + r_i)$.

As option of POSTAN3 and PLP are expressed in terms of the internal formulation of the linear problem we shall recall this concept. The linear programming problem (1) - (3) is transformed by MINOS into the following internal form: Minimize (or maximize) the variable

$$-\tilde{x}_{n+1+\mathrm{obj}} \tag{4}$$

subject to equality constraints:

$$\tilde{A}\tilde{x} = 0 \tag{5}$$

and inequality constraints:

$$\tilde{l} \leq \tilde{x} \leq \tilde{u} \ . \tag{6}$$

Here $\tilde{A}$ is an $(m+1) \times (n+m+2)$-matrix:

$$\tilde{A} = \left[\begin{array}{ccc} \tilde{a}_1 & \tilde{b}_1 & \\ \cdot & \cdot & \\ \cdot & \cdot & I \\ \cdot & \cdot & \\ \tilde{a}_{m+1} & \tilde{b}_{m+1} & \end{array} \right] , \tag{7}$$

where I denotes the $(m+1) \times (m+1)$ identity matrix and

$$\begin{array}{l} \tilde{a}_i = a_i \quad i \leq \mathrm{obj} \ , \quad \tilde{a}_{obj} = a_0 \ , \quad \tilde{a}_i = a_{i-1} \quad i \leq \mathrm{obj} \ , \\ \tilde{b}_i = b_i \quad i \leq \mathrm{obj} \ , \quad \tilde{b}_{obj} = 0 \ , \quad \tilde{b}_i = b_{i-1} \quad i \leq \mathrm{obj} \ , \end{array} \tag{8}$$

where

$$b_i = \begin{cases} 0 & if \ d_i = -\infty \ and \ g_i = +\infty \\ d_i & if \ d_i \text{ is finite } and \ g_i = +\infty \\ g_i & if \ g_i \text{ is finite } . \end{cases}$$

The first n components of the extended vector of decision variables $\tilde{x} \in R^{n+m+2}$ form a subvector identical to x; these components are described as *structural*. Element $\tilde{x}_{n+1}$ is called the *right-hand-side component*; it is fixed at -1. The remaining components of $\tilde{x}$ are called *slack* or *logical components*. The objective variable $\tilde{x}_{n+1+\mathrm{obj}}$ is free. The vector of lower bounds $\tilde{l}$ and the vector of upper bounds $\tilde{u}$ are defined as follows:

$$\begin{array}{l} \tilde{l}_i = d_{m+i} \quad i = 1,...,n \ , \quad \tilde{l}_{n+1} = -1, \quad \tilde{l}_{n+1+\mathrm{obj}} = -\infty \ , \\ \tilde{u}_i = g_{m+i} \quad i = 1,...,n \ , \quad \tilde{u}_{n+1} = -1, \quad \tilde{u}_{n+1+\mathrm{obj}} = +\infty \ . \end{array} \tag{9}$$

Now let $i = n+1+j$, $j = 1,...,m$. Then

$$\tilde{l}_i = h_i \ , \quad \tilde{u}_i = k_i \ for \ j \ \mathrm{obj} \ and \ \tilde{l}_i = h_{i-1} \ , \quad \tilde{u}_i = k_{i-1} \ for \ j \ \mathrm{obj} \ , \tag{10}$$

where

$h_i = k_i = 0$ if the j-th row constraint is fixed (i.e., of type E)

$h_i = 0$, $k_i = +\infty$ if $d_j = -\infty$ and g_j is finite (one-sided constraint of type L)

$h_i = -\infty$, $k_i = 0$ if d_j is finite and $g_j = +\infty$ (one-sided constraint of type G)

$h_i = 0$, $k_i = g_j - d_j$ if d_j and g_j are finite

$h_i = -\infty$, $k_i = +\infty$ if the j-th row constraint is free .

4 Postoptimal analysis for linear problems in POSTAN3

Here we give a list of ranging options of POSTAN3 for linear problems.

Ranging on costs

Ordinary ranging

For each cost component a_0^i, $i = 1, \ldots, n$ the largest range is determined in which a_0^i may vary without affecting the optimal solution. While the range for a_0^i is being determined, all other components a_0^j, $j \neq i$, remain fixed at their original values. Some information on the change of state of variables at the boundaries is also given.

Directional ranging

For a given increment $\Delta a_0 \in R_n$ of the cost vector a_0 , the largest real $t_{\max} \geq 0$ is determined such that for every cost vector of the form $a_0 + t\Delta a_0$, $t \in [0, t_{\max}]$, the optimal solution is the same as at the point, a_0 (i.e., at $t = 0$). The boundary cost components $a_0^i + t_{\max}\Delta a_0^i$, $i = 1, \ldots, n$ and some information on the change of state of variables at the boundary are also given.

Ranging on right-hand sides

Ordinary ranging

For each component $\tilde{b}_i$, $i = 1, \ldots, m+1$ of the rhs vector (except for the objective row, $i \neq obj$), the maximum range is determined in which $\tilde{b}_i$ may vary without affecting the optional basis. While the range for $\tilde{b}_i$ is being determined, all other components $\tilde{b}_j$, $i \neq j$ are fixed at their original values. It should be noted that the rhs vector $\tilde{b}$ is not always the rhs of a constraint system in the original formulation (1) - (3); the user should refer to (5) - (11). Some information on the change of state of variables at the boundaries is given.

Directional ranging

For a given vector of increments $\Delta\tilde{b} \in R^{m+1}$ of the rhs vector $\tilde{b}$, the largest real $t_{\max} \geq 0$ is determined such that for every rhs of the form $b + t\Delta b$, $t \in [0, t_{\max}]$, the optimal basis is the same as at the point $\tilde{b}$ (i.e., at $t = 0$). At the boundary $t = t_{\max}$ either the optimal solution vanishes or one of the basic variables changes its state. Its name and the type of change are given. together with the boundary values of rhs elements.

Ranging on bounds

Ordinary ranging

For each lower bound $\tilde{l}_i$ and each upper bound $\tilde{u}_i$, $i = 1, \ldots, n+m+2$ two ranges are determined: range A which is the maximum range in which the bound may vary without affecting the optimal solution, and range B, which is the maximum range in which the bound may vary without affecting the optimal basis. While these ranges are being determined for $\tilde{l}_i$ (or $\tilde{u}_i$), all other bounds remain fixed at their original values. Some information is given on the change of state of variables at the boundaries. This analysis is not performed for fixed variables.

Directional ranging

For a given vector of increments $\mathrm{col}(\Delta\tilde{l}, \Delta\tilde{u}) \in R^{2(m+n+2)}$ of the vector of bounds $\mathrm{col}(\tilde{l}, \tilde{u})$, two real numbers are determined:

- $t_{\mathrm{maxa}} \geq 0$, the largest real number such that for every bound vector of the form $\mathrm{col}(\tilde{l}, \tilde{u}) + t\,\mathrm{col}(\Delta\tilde{l}, \Delta\tilde{u})$, $t \in [0, t_{\mathrm{maxa}}]$, the optimal solution is the same as for the bound vector $\mathrm{col}(\tilde{l}, \tilde{u})$, i.e., at $t = 0$.
- $t_{\mathrm{maxb}} \geq 0$, the largest real number such that for every bound vector of the form $\mathrm{col}(\tilde{l}, \tilde{u}) + t\,\mathrm{col}(\Delta\tilde{l}, \Delta\tilde{u})$, $t \in [0, t_{\mathrm{maxb}}]$, the optimal basis is the same as for the bound vector $\mathrm{col}(\tilde{l}, \tilde{u})$, i.e., at $t = 0$.

Boundary values of the elements of bound vector and some information on the change of state of variables at the boundaries are also given.

4..1 Ordinary ranging on elements of constraint matrix

For selected elements a_j^i, $i = 1, \ldots, n$, $j = 1, \ldots, m$ of the constraint matrix $\mathrm{col}(a_1, a_2, \ldots, a_m)$ (see (2)) the largest range is determined in which a_j^i may vary without affecting the optimal basis or the state of nonbasic variables. The list of the selected elements is given in the data. While the range for a_j^i is being determined, all other elements a_k^l, $k \neq l$ and/or $j \neq l$, are fixed at their original values. The sensitivity of the optimal cost with respect to the elements is also calculated. In addition, some information on the change of state of variables at the boundaries is given.

Directional ranging on constraint rows

For a given increment vector $\Delta a_i \in R_n$ of the constraint vector a_i, $i = 1, \ldots, m$, the largest range $(t_{\mathrm{min}}, t_{\mathrm{max}})$ is determined such that for every i-th constraint vector of the form $a_i + t\Delta a_i$, $t \in (t_{\mathrm{min}}, t_{\mathrm{max}})$, the optimal basis and the state of nonbasic variables are the same as at the point a_i (i.e., at $t = 0$). The sensitivity of the optimal cost with respect to parameter t at $t = 0$ is also calculated. In addition, the boundary values of the constraint row a_i and some information on the change of state of variables at the boundaries are given.

Directional ranging on structural columns of constraint matrix

For a given vector of increments Δa_i of the column a_i, $i = 1, \ldots, n$, the largest range $(t_{\mathrm{min}}, t_{\mathrm{max}})$ is determined such that for every i-th constraint column of the form $a_i + t\Delta a_i$,

$t \in (t_{\min}, t_{\max})$, the optimal basis and the state of nonbasic variables are the same as at the point a_i (i.e., at $t = 0$). The sensitivity of the optimal cost with respect to parameter t at $t = 0$ is calculated. addition, the boundary values of the column a_i and some information on the change of state of variables at the boundaries are given.

5 Parametric programming options of PLP

Parametric analysis of cost

The cost vector $a_0 = (a_0{}^1, a_0{}^2, ..., a_0{}^n)$ (see (1)) is changed along a direction given by the user, $\Delta a_0 = (\Delta a_0{}^1, \Delta a_0{}^2, ..., \Delta a_0{}^n)$ according to the formula:

$$a_0(t) = a_0(0) + t\Delta a_0 \ , \ t \geq 0 \tag{12}$$

where $a_o(0)$ is the starting value of cost. If the structural variable, say $\bar{x}_i$, is fixed then $\Delta a_0{}^i$ is automatically set to zero, regardless of the value given in the data.

PLP determines a sequence of values of the parameter denoted by $t_0, \ t_1, ..., \ t_k$, such that $0 = t_0 < t_1 < t_2 < ... < t_k$ and in each of the intervals $[t_i, \ t_{i+1})$, $i = 0, ..., k-1$ the optimal solution is constant and in each case the optimal basis is different. The integer k :

1. may be defined by the user as the maximum number of iterations,
2. may be determined by the condition that the optimal solution is constant for every $t \geq t_k$ and different from that in $[\ t_{k-1}, \ t_k)$,
3. may be determined by the condition that there are no optimal solutions for every $t < t_k$.

Parametric analysis of rhs

The right-hand side vector $\tilde{b} =$ col $(\tilde{b}_1, ..., \tilde{b}_{m+1})$ (see (7) and (8)) is changed along a direction given by the user, $\Delta\tilde{b} =$ col$(\Delta\tilde{b}_1, ..., \Delta\tilde{b}_{m+1})$, according to the formula:

$$\tilde{b}(t) = \tilde{b}(0) + t\Delta\tilde{b} \ \ , \ t \geq 0 \tag{13}$$

where $\tilde{b}(0)$ is the starting value of rhs. The component of $\Delta\tilde{b}$ which correspond to the objective row is automatically set to zero, $\Delta\tilde{b}_{obj} = 0$.

PLP determines a sequence of values of the parameter denoted by $t_0, \ t_1, ..., \ t_k$ such that $0 = t_0 < t_1 < t_2 < ... < t_k$ and in each of the intervals $[t_i, \ t_{i+1})$, $i = 0, ..., k-1$ the optimal basis is constant and in each

1. may be defined by the user as the maximum number of iterations,
2. may be determined by the condition that the optimal solution is constant for every $t \geq t_k$ and different from that in $[\ t_{k-1}, \ t_k)$,
3. may be determined by the condition that there are no optimal solutions for every $t < t_k$.

Parametric analysis of bounds

The vector of bounds $\mathrm{col}(\tilde{l},\ \tilde{u}) \in R^{2(n+m+2)}$ (see (9)) is changed along a direction given by the user, $\mathrm{col}(\Delta\tilde{l},\ \Delta\tilde{u})$, according to the formula:

$$\mathrm{col}(\tilde{l}(t),\ \tilde{u}(t)) = \mathrm{col}(\tilde{l}(0),\ \tilde{u}(0)) + t\ \mathrm{col}(\Delta\tilde{l},\ \Delta\tilde{u})\ ,\ t \geq 0 \qquad (14)$$

where $\mathrm{col}(\tilde{l}(0),\ \tilde{u}(0))$ is the starting value of bounds. The bound increments $\Delta\tilde{l}_i,\ \Delta\tilde{u}_i$ which corresponds to fixed variables are automatically set to zero regardless of the values given in the data.

If there is no lower and/or upper bound for the i-th variable $\tilde{x}_i$ (see (6)) the corresponding increment $\Delta\tilde{l}_i$ and/or $\Delta\tilde{u}_i$, respectively, is also automatically set to zero.

PLP determines a sequence of values of the parameter denoted by $t_o,\ t_1, ...,\ t_k$ such that $0 = t_0 < t_1 < t_2 < ... < t_k$ and in each of the intervals $[t_i,\ t_{i+1}),\ i = 0, ..., k-1$ the optimal basis is constant and in each case different. The integer k :

1. may be defined by the user as the maximum number of iterations,
2. may be determined by the condition that the optimal solution is constant for every $t \geq t_k$ and different from that in $[\ t_{k-1},\ t_k)$,
3. may be determined by the condition that there are no optimal solutions for every $t < t_k$.

Each interval $[t_i,\ t_{i+1}]$ is optionally divided into two subintervals $[t_i,\ t_i{}^a]$, $[t_i{}^a,\ t_{i+1}]$. The interval $[t_i, t_i{}^a]$ is the maximum interval where the optimal solution remains constant and not only the optimal basis. It often happens that $t_i = t_i{}^a$.

Dependent and independent work

All three kinds of analysis can be performed in one run. The starting point for the next kind of analysis may be either the original starting optimal solution (The Independent Work) or the last optimal solution obtained in the preceding analysis (The Dependent Work). The continuation is impossible if the optimal solution vanishes.

Controlling output

In each of the three kinds of analysis the following information is available. The user has to specify the frequency of printing the complete current optimal solution in MINOS format. This means that the complete printout is given for the values of parameters t equal to $t_{0+},\ t_{p+},\ t_{2p+}, ...,$ and $t_{(k-1)+}$ or t_{k+} depending on whether the optimal solution exists for $t < t_k$. The notation t_{i+} means that we take the right-hand limit of the optimal solution at t_i. The user specifies frequency of printing the so called PLP ITERATION LOG. This is a short message containing most important information about current change of optimal solution (value of the parameter t, change of optimal basis, current value of objective function).

Tolerances

The performance of PLP is strongly affected by the choice of tolerances. Especially important are two tolerances determined in MINOS : the tolerance of optimality (TOLD) and the tolerance of feasibility (TOLX). In the proper procedures of the PLP the following general rule is adopted. All quantities greater than or equal to 1.E+15 are taken as equal to infinity and all quantities whose absolute value is less than 1.E-9 are regarded as equal to zero.

6 Linear-fractional programming option in POSTAN3

The LFP part of POSTAN3 deals with linear-fractional programming problems of the form: Minimize

$$F(x) = \frac{cx + \alpha}{dx + \beta} \quad \text{where } x \in R^n \ , \quad c, d \in R_n \ , \quad \alpha, \beta \in R \tag{15}$$

subject to

$$A\,x \leq b \ , \quad x \geq 0 \text{ where } b \in R^m \ , \quad A - m \times m \,\text{matrix} \tag{16}$$

It is assumed that $dx + \beta \geq 0$ in the whole of the admissible region.

POSTAN3 may be used to solve the LFP problem using a modified simplex algorithm. In fact, the linear MINOS procedures are used to this end, after some modifications.

On user's request, POSTAN3 performs ordinary ranging on elements of vectors c and d (ranging on costs). The ranging on bounds and right-hand sides (both ordinary and directional) may be also performed in the LFP case.

References

Dobrowolski G., K. Hajduk, A. Korytowski and T. Rys (1984) POSTAN - A Package for Postoptimal Analysis (An Extension of MINOS). IIASA Collaborative Paper CP-84-32, Laxenburg, Austria.

Dobrowolski G., K. Hajduk, A. Korytowski and T. Rys (1987) POSTAN 2 - A Package for Postoptimal Analysis (An Extension of MINOS). IIASA Working Paper WP-87-26 Laxenburg, Austria.

Dobrowolski G., T. Rys, K. Hajduk, A. Korytowski (in preparation) POSTAN3 - Extended Postoptimal Analysis Package for MINOS. IIASA Collaborative Paper, Laxenburg, Austria.

Golebiowski A. (in preparation) PLP - A Package for Parametric Programming. IIASA Collaborative Paper, Laxenburg, Austria.

Murtagh B.A. and M.A. Saunders (1977) MINOS - A Large-Scale Linear Programming System. User's Guide. Technical Report SOL 77-9, System Optimization Laboratory, Stanford University, California.