International Institute for Applied Systems Analysis
IIASA www.iiasa.ac.at

Design and Implementation of a Stochastic Programming Optimizer with Recourse and Tenders

Nazareth, J.L.

IIASA Working Paper

WP-85-063

October 1985

DESIGN AND IMPLEMENTATION OF A STOCHASTIC PROGRAMMING OPTIMIZER WITH RECOURSE AND TENDERS.

J.L. Nazareth

October, 1985
WP-85-063

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
2361 Laxenburg, Austria

ABSTRACT

This paper serves two purposes, to which we give equal emphasis. First, it describes an optimization system for solving large-scale stochastic linear programs with simple (i.e. decision-free in the second stage) recourse and stochastic right-hand-side elements. Second, it is a study of the means whereby large-scale Mathematical Programming Systems may be readily extended to handle certain forms of uncertainty, through post-optimal options akin to sensitivity on parametric analysis, which we term "recourse analysis". This latter theme (implicit throughout the paper) is explored in a proselytizing manner, in the concluding section.

ACKNOWLEDGEMENTS

It is a pleasure to thank Michael Saunders for making MINOS 5.0 (and an earlier version) available to us and for his very helpful advice on its usage in this context.

CONTENTS

DESIGN AND IMPLEMENTATION OF A STOCHASTIC PROGRAMMING OPTIMIZER WITH RECOURSE AND TENDERS

J.L. Nazareth

1. Introduction

This paper is a sequel to Nazareth and Wets [21] and serves two purposes, to which we give equal emphasis. First, it describes an optimization system for solving a restricted but important class of large-scale stochastic linear programs with recourse. Second, it is a study and detailed illustration of the means whereby any large-scale Mathematical Programming System (MPS) designed for solving deterministic linear programs, could be readily extended to handle some forms of uncertainty, in particular, via post-optimal analysis options. This latter theme (implicit throughout the paper) is explored, in a proselytizing manner, in the concluding section.

The class of practical stochastic linear programs with which we are concerned (termed C1 problems in [21]) arise as a natural extension of the linear programming model as follows: given a linear program with matrix $\mathbf{A}$, it is often the case that some of the components of the right-hand-side (exogenous) vector of resource availability or resource demand, are known only in probability and have been replaced (in the deterministic LP formulation) by some expected value. We seek to extend this linear program, using the recourse formulation. Rows of $\mathbf{A}$ corresponding to the stochastic right-hand-side are used to define the technology matrix T (we follow the notation and terminology of [21]) and the remaining rows of $\mathbf{A}$ are used to define the constraint matrix A, both A and T being typically large, sparse matrices. The recourse is assumed to be simple (i.e. *decision-free* in the second-stage problem) and specified in terms of costs (or penalties) on shortage and surplus. Furthermore, we restrict attention to the case where each component

This paper is a draft for Chapter 14 of *Numerical Techniques for Stochastic Optimization Problems*, Y. Ermoliev and R.J.-B. Wets, eds., Springer-Verlag, to appear.

of the stochastic right-hand-side has a given discrete probability distribution. There are many applications for such a model, see Ziemba [27], and more complex stochastic linear programs with recourse can sometimes be solved by an iterative discretization or sampling procedure involving definition and solution of a sequence of C1 problems.

The above considerations are very much in the background of our implementation design, our choice of algorithms and of the more general issues which we wish to discuss regarding the extension of conventional Mathematical Programming Systems, so as to be able to handle at least some forms of uncertainty. Our optimization system is based primarily upon a version of Wolfe's generalized programming algorithm (see Dantzig [4]) given in Nazareth and Wets [21] Section 3.2.1 and, in more detail, in Nazareth [18]. It also includes a version of an algorithm based upon bounded variables (see Wets [25]) given in [21] Section 2.1 and, again in more detail, in [20]. Two simpler options, namely to solve an initial linear program and to permit some of its constraints to be "elastic" are also included to help get a recourse problem "off the ground." In our implementation (see Nazareth [19] for an overview of our overall approach) we have utilized current mathematical programming technology for specifying the problem (using standard MPS conventions [14] for the LP portion and a suitable extension to provide the added stochastic information), to represent the data internally (in packed data structures, space for which is dynamically allocated within a work storage array) and to implement our solution strategies (using an efficient and numerically stable implementation of the simplex method, namely the MINOS System of Murtagh and Saunders [15], [16]).

Finally, we want our design to mesh as naturally as possible with current Mathematical Programming Systems. In particular, we argue in the concluding section of our paper, that "recourse analysis" (simple recourse to start off with, but also more general forms of recourse) could be provided as a post-optimal analysis option in any large-scale MPS, to augment the options for parametric and sensitivity analysis that are now usually available.

2. Overview of the SPORT System

2.1. Problem

SPORT (pronounced SupPORT) is an acronym for *S*tochastic *P*rogramming *O*ptimizer with *R*ecourse and *T*enders. The current version solves large-scale stochastic programs with simple (decision-free in the second stage) recourse and discrete distribution of right-hand-side elements (termed C1 problems). The formal statement of such problems may be found in [21] (see (1.1) through (1.3) where $W = [I, -I]$ and where the right-hand-side $h(\omega)$ is the only stochastic quantity, with a known discrete distribution) and we shall not repeat here. Instead, we shall state the problem from the perspective emphasized in this paper, namely that of a given linear program in which inherent uncertainty in some of the right-hand-side (exogenous) elements is to be more fully taken into account. Consider therefore the linear program

$$\begin{array}{ll} \text{minimize} & cx \\ \text{subject to} & \mathbb{A}x = d \\ & x \geq 0 \end{array} \tag{2.1}$$

where $\mathbb{A}$ is an $m \times n$ matrix (which is generally large and sparse), d is a given m-vector and c is a given n-vector. Some of the elements of d which correspond to demands (or available resources) may be, in reality, only known in probability and defined in (2.1) by taking some expected value. For simplicity, let us suppose that the corresponding "technology" constraints of (2.1) are the last m_2 constraints and let us denote them by $Tx = \bar{h}$, where T is an $m_2 \times n$ matrix. Let the remaining m_1 constraints be $Ax = b$ where A is an $m_1 \times n$ matrix and $d = \begin{pmatrix} b \\ \bar{h} \end{pmatrix}$.

A useful extension to the LP model (2.1) is to permit the constraints $Tx = \bar{h}$ to be "elastic" (Tomlin [24]) by imposing a penalty q_i^+ on shortage in the i^{th} technology constraint when demand (corresponding to the right-hand-side element $\bar{h}_i$) exceeds the supply $(Tx)_i$, so that $y_i^+ = \bar{h}_i - (Tx)_i \geq 0$. Similarly let q_i^- be the penalty imposed on surplus (when the reverse of the earlier conditions holds) so that $y_i^- = (Tx)_i - \bar{h}_i \geq 0$. (The choice of notation q_i^+ for shortage and q_i^- for surplus is a little unfortunate, but is now standard.) Thus associated with the decision x for the "first-stage" or decision variables, we have a penalty of

$$Q_i(x, \bar{h}_i) = \begin{cases} q_i^+(\bar{h}_i - (Tx)_i) & \text{when } (\bar{h}_i - (Tx)_i) \geq 0 \\ q_i^-((Tx)_i - \bar{h}_i) & \text{when } (\bar{h}_i - (Tx)_i) \leq 0 \,. \end{cases}$$

To minimize first stage costs and all penalty costs we can formulate the extension of (2.1) as a problem with "elastic" constraints as follows:

$$\begin{array}{lll} \text{minimize} & cx + q^+y^+ + q^-y^- & \\ \text{subject to} & Ax & = b \\ & Tx + y^+ - y^- & = \bar{h} \\ & x \geq 0,\ y^+ \geq 0,\ y^- \geq 0 & \end{array} \tag{2.2}$$

where q^+ and q^- are m-vectors with components q_i^+ and q_i^- respectively.

Unfortunately (2.2) does not address the uncertainty in the right-hand side vector, which so far has been replaced by $\bar{h}$. One way to address uncertainty is to compute the penalty cost associated with each *realization* of the random vector $h(\omega)$. Let us also define the "tender" or "bill of goods" associated with a decision x by $\chi = Tx$. Thus we have

$$Q_i(x, \bar{h}_i) \triangleq \psi_i(\chi_i, h_i(\omega)) = \begin{cases} q_i^+(h_i(\omega) - \chi_i) & \text{when } (h_i(\omega) - \chi_i) \geq 0 \\ q_i^-(\chi_i - h_i(\omega)) & \text{when } (h_i(\omega) - \chi_i) \leq 0 \end{cases}$$

Let $\Psi(\chi) \triangleq E_\omega(\sum_{i=1}^{m_2} \psi_i(x_i, h_i(\omega)) = \sum_{i=1}^{m_2} E_\omega(\psi_i(\chi_i, h_i(\omega)) \triangleq \sum_{i=1}^{m_2} \Psi_i(\chi_i)$. We seek to minimize the cost of the decision cx and the expected value of the penalty costs. Thus we can formulate this extension of (2.1) as

$$\begin{array}{lll} \text{maximize} & cx + \sum_{i=1}^{m_2} \Psi_i(\chi_i) & \\ \text{subject to} & Ax & = b \\ & Tx - \chi & = 0 \\ & x \geq 0 & \end{array} \tag{2.3}$$

For $C1$ problems it can be readily demonstrated (see, for example [25], [20]) that

$$\Psi_i(\chi_i) = \max_{l=0,\ldots,k_i} (s_{il}\chi_i + e_{il})$$

where s_{il} and e_{il} are defined from the probability distribution of $h_i(\cdot)$. Let this be given by values $h_{i1}, h_{i2}, \ldots, h_{ik_i}$ with $h_{il} \leq h_{i,l+1}$, with associated probabilities $p_{i1}, p_{i2}, \ldots, p_{ik_i}$. Then, for $l = 0, \ldots, k_i$

$$\begin{aligned} s_{il} &= \left[\sum_{t=1}^{l} h_{it}\right] q_i - q_i^+ \\ e_{il} &= q_i^+ \bar{h}_i - q_i \left[\sum_{t=1}^{l} h_{it} p_{it}\right] \end{aligned} \tag{2.4}$$

where, by convention, $\sum_{i=1}^{0} = 0$, $q_i = (q_i^+ + q_i^-) > 0$ and $\bar{h}_i$ is the expected value of $h_i(\omega)$. Finally, using a theorem in [18], it is possible to state (2.3) in an *equivalent* form and in so doing also unify with (2.2) as follows:

$$\begin{array}{lll} \text{minimize} & cx + q^+y^+ + q^-y^- + \sum_{i=1}^{m_2} \Psi_i(\chi_i) & \\ \text{subject to} & Ax & = b \\ & Tx + y^+ - y^- - \chi & = 0 \\ & x \geq 0,\ y^+ \geq 0,\ y^- \geq 0 & \end{array} \tag{2.5}$$

For $\chi \triangleq \bar{h}$ we obtain (2.2) since $\Psi(\bar{h})$ is then a constant term. (2.5) is a piecewise-linear separable convex programming problem with which we shall be concerned henceforth. It makes possible both convenient implementation of the algorithms which we employ and the various options that we provide, as discussed in the next section.

2.2 Algorithms

The system is based primarily upon the Wolfe generalized programming approach as discussed in [21], Section 3.2.1. The particular algorithm implemented here termed ILSRDD (Inner Linearization - Simple Recourse - Discrete Distribution) is described, in detail, in [18]. The generalized programming approach was chosen because it proved effective in earlier experimental versions (see [18]) and because of its potential applicability to a wide class of stochastic programs (including problems with complete recourse and problems with probabilistic constraints, see [18]). We also include an alternative to ILSRDD. This is algorithm based upon problem redefinition and the introduction of bounded variables given by Wets [25] and implemented in the simpler form given in Nazareth and Wets [20]. The algorithm is termed BVSRDD (Bounded Variables—Simple Recourse—Discrete Distribution). This approach is much more limited in its range of possible application as we have discussed in [21], but we include it for the following reasons: (a) it is very convenient to have a second algorithm that works on basically the same input as ILSRDD, for purposes of comparisons of answers and validation of implementation. Two identical answers on a particular problem from two different algorithms are rather comforting in this world of uncertainty and although this is no guarantee of correctness, it provides some indication that an error (if any) is in the input data or its conversion into internal representations. (b) A fair amount of experience has been accumulated with an early implementation of this method

for dense problems (see Kallberg & Kusy [11]) and a more advanced implementation (which handles sparsity) should be available. (When there are relatively few points in each distribution of $h_i(\cdot)$ then this may even be a quite efficient way to solve C1 problems. (c) The algorithm BVSRDD makes possible a simpler and more direct extension of a deterministic MPS when the aim is only to handle simple recourse.

Two further options are provided in order to be able to solve (2.5) with $\chi = \bar{h}$ (ELASTIC option) and in order to solve an initial linear program, equivalent to (2.5) with $\chi = \bar{h}$, $q_i^+ = q_i^- = \infty$ (MINOS option). Here, $\bar{h}$ denotes an arbitrary right-hand-side vector. Both of these options are useful as preliminaries to the recourse formulation.

2.3 Implementation

From a practical standpoint, the linear programs which we want to solve and extend are of the more general form:

$$\begin{array}{ll} \text{minimize} & cx \\ \text{subject to} & \mathbb{A}x \begin{bmatrix} \leq \\ = \\ \geq \end{bmatrix} b \\ & l \leq x \leq u \end{array} \tag{2.6}$$

where $\begin{bmatrix} \leq \\ = \\ \geq \end{bmatrix}$ indicates that constraints take one of three possible forms and u and l are vectors of upper and lower bounds. Furthermore, we cannot usually expect the partition $\mathbb{A} = \begin{bmatrix} A \\ T \end{bmatrix}$ with technology rows T coming last in the matrix $\mathbf{A}$. In general, rows of A and rows of T will be interleaved in $\mathbf{A}$. In addition, it is worthwhile to explicitly include a scale factor ρ to permit a weighting of the second-stage objective relative to the first (see [18]). Thus the practical problems which we seek to solve, are derived from (2.5) and (2.6) and take the form

$$\begin{array}{lll} \text{minimize} & cx + q^+y^+ + q^-y^- + \rho \sum_{i=1}^{m_2} \Psi_i(\chi_i) & \\ \text{subject to} & (A^{\alpha_i})x \ (\leq = \geq) b_i & , \alpha_i \in \Lambda \\ & T^{\tau_i}x + y_i^+ - y_i^- - \chi_i = 0 & , \tau_i \in \Gamma \\ & l \leq x \leq u \,,\ y^+, y^- \geq 0. & \end{array} \tag{2.7}$$

where A^{α_i}, $\alpha_i \in \Lambda$ defines the rows of A, T^{τ_i}, $\tau_i \in \Gamma$ defines the rows of T, and Λ and Γ are index sets with $|\Lambda| = m_1$, $|\Gamma| = m_2$ ($|\Lambda|$ denotes the number of indices in Λ).

Our system for solving recourse problems of the form (2.7) has three main phases:

Phase 1: Problem Setup and Generation
Phase 2: Specialized Setup and Solution
Phase 3: Output

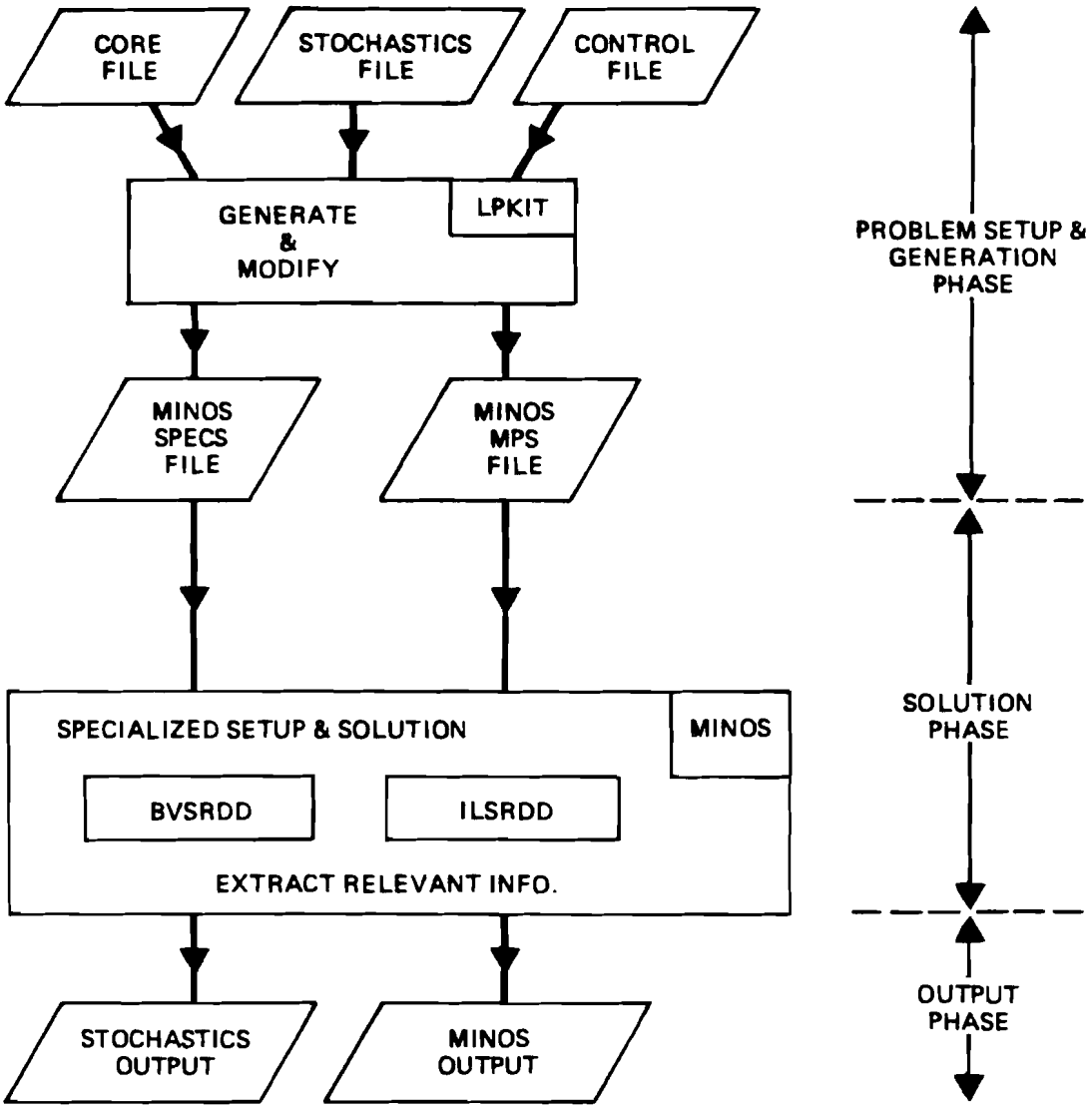
This is summarized in Figure 2.1. A design goal was that all algorithms work on essentially the same input and each ignore input data that is only required by the others, e.g. the limit on the number of cycles, which is only required by ILSRDD. The input is specified in the form of three files of information which are described in more detail in the next section. All that is *often* necessary to switch options is to change the algorithm card in the "control" file and check that enough work space has been provided for various items. The Problem Setup and Generation Phase results in the creation of two files required by MINOS — the SPECS file and the MPS file. The next main phase consists of reading in these files by MINOS, inserting additional columns into its packed data structures and finding the solution of the problem. Finally the Output Phase augments the solution output by MINOS with some additional information about the solution of the stochastic program with recourse.

The next three sections go into this in more detail.

3. Problem Setup and Generation

To be specific, we discuss this within the context of an example. Consider the following product-mix example (due to J. Ho [10]). The problem has two products and three ingredients. We seek to minimize cost of production while maintaining the levels of fat and protein at acceptable levels, and not exceeding availability of ingredients. The demand for each product is a random variable with discrete distribution but in an LP formulation this must be replaced by some expected value. The problem is summarized as follows, where x_i, y_i, z_i denote the amount of each ingredient in product i (i =1,2).

Figure 2.1 Overview



$$\text{minimize} \quad x_1 + 2y_1 + 3z_1 + x_2 + 2y_2 + 3z_2 \qquad \text{(OBJ)}$$

subject to

Fat/Protein Content of Product 1	$0.3x_1 + 0.4y_1 + 0.2z_1 \geq$	3.3	(A3)
Fat/Protein Content of Product 2	$0.5y_2 + 0.6z_2 \geq$	4.0	(A4)
Amount of Ingredient 1	$x_1 + x_2 \leq$	15.0	(A1)
Amount of Ingredient 2	$y_1 + y_2 \leq$	12.0	(A2)
Amount of Product 1	$x_1 + y_1 + z_1 =$	$\bar{h}_1$	(T1)
Amount of Product 2	$x_2 + y_2 + z_2 =$	$\bar{h}_2$	(T2)

$$x_i \, , y_i \, , z_i \geq 0 \, , \; i = 1,2$$

The penalties for under and over production are 2.0 and 1.0 units, respectively, for each product, and the probability distribution on demand $h(\cdot)$ is as follows:

	Product 1			Product 2		
Level	8.0	10.0	12.0	15.0	18.0	20.0
Probability	0.25	0.5	0.25	0.2	0.4	0.4

$h_1 = 10.0$ and $h_2 = 18.2$. The recourse function $\Psi(\chi)$ is defined in the usual way with $q^+ = (2.0,2.0)$ and $q^- = (1.0,1.0)$.

3.1 Corefile

The input data corresponding to the decision variables x of the problem forms the "corefile". This specifies

- the names and types of each row of the problem
- the objective c
- the coefficients of A and T
- the deterministic right-hand-side elements
- the bounds on variables and ranges on rows

The "corefile" is specified in standard MPS format, see [14] and will often originate in a prior LP formulation. A and T can have interleaved rows and rows corresponding to T should normally be equality rows. However if these correspond to $\geq$ or $\leq$ rows i.e if there is no penalty on surplus or shortage, respectively, then provision is made in the system to change these to equality rows and a warning message is printed to that effect. This means that q_i^+ or q_i^- must be chosen appropriately at value zero. Note also that if there were non-zero elements in the right-hand-side vector corresponding to rows in the technology matrix they will be ignored by ILSRDD or BVSRDD and a message printed to this effect.

For our example, the corefile is given in Figure 3.1. (Slack variables were introduced explicitly in this case, but this is not necessary and could have been avoided by appropriate definition of row types.)

3.2 Stochastics File

The "stochastics" file specifies the information pertaining to the recourse problem. It gives:

- the row names identifying the technology matrix
- the probability distribution for each stochastic right-hand side
- the penalties q^+ and q^- on shortage and surplus
- the set of initial tenders for ILSRDD or the base tender for BVSRDD

An MPS-like format was designed for each of these items of information and is explained in the rest of this subsection. (An extension of this format is given in Edwards et al. [7].)

NAME — This is a header card. The user may enter any characters in columns 15 to 72.

TECHNOLOGY — The data consists of a list of names, one for each row in the technology matrix. These must be a subset of the list of rownames in the "corefile". The submatrix corresponding to this set of rows in the COLUMNS section of the "corefile" defines the technology matrix. One name appears per line in columns 5 through 12.

DISTRIBUTION — The data consists of sets of entries of the form "rowname value probability". There is one such set for each of the rows named in the TECHNOLOGY section. "rowname" specifies the row associated with the entry (columns 5 through 12). "value" and "probability"

```
NAME      LP
ROWS
          N OBJ
          E A1
          E A2
          E A3
          E A4
          E T1
          E T2
COLUMNS
          CLM1      OBJ       1.0       A1        1.0
          CLM1      A3        0.3       T1        1.0
          CLM2      OBJ       2.0       A2        1.0
          CLM2      A3        0.4       T1        1.0
          CLM3      OBJ       3.0       A3        0.2
          CLM3      T1        1.0
          CLM4      A3        -1.0
          CLM5      OBJ       1.0       A1        1.0
          CLM5      T2        1.0
          CLM6      OBJ       2.0       A2        1.0
          CLM6      A4        0.5       T2        1.0
          CLM7      OBJ       3.0       A4        0.6
          CLM7      T2        1.0
          CLM8      A4        -1.0
          CLM9      A1        1.0
          CLM10     A2        1.0
RHS
          RTH       A1        15.0      A2        12.0
          RTH       A3        3.3       A4        4.0
          RTH       T1        10.0      T2        18.2
BOUNDS
ENDATA
```

Figure 3.1 The corefile

specify the point and its associated probability. They occupy the first and second numeric fields (columns 25 through 36 and 50 through 61) respectively and must be specified as real numbers. The "rowname" repeats itself for each possible value associated with the row and the probabilities for this "rowname" must sum to unity.

OBJECTIVE The data consists of entries of the form "name value value" where name is a rowname of T and the first value gives the value of q_i^+ and the second the value of q_i^- i.e the penalties on shortage and surplus respectively. The name occupies the first field (columns 5 through 12) and the values the first and second numeric fields

(columns 25 through 36 and 50 through 61) respectively. They must be specified as real numbers.

TENDERS — The data consists of entries of the form "name rowname value" where name is the name associated with tender, "rowname" specifies the row associated with the entry and "value" is the level of the tender for this row. "name" repeats itself over all entries associated with the tender and there is one such "name" for each tender specified. "name" and "rowname" occupy the first two name fields (columns 5 through 12) and (15 through 22) respectively and "value" the first numeric field (columns 25 through 36). (If a set of these are provided for ILSRDD then the first one is used by BVRDD as its base tender, see Sec. 2.1 of [21].)

ENDATA — This card must be specified and flags the end of the "stochastics" file.

For our example the "stochastics" file is given in Figure 3.2.

```
NAME TEST
TECHNOLOGY
        T1
        T2
DISTRIBUTION
        T1                      8.0                     0.25
        T1                     10.0                     0.5
        T1                     12.0                     0.25
        T2                     15.0                     0.2
        T2                     18.0                     0.4
        T2                     20.0                     0.4
OBJECTIVE
        T1                      2.0                     1.0
        T2                      2.0                     1.0
TENDERS
        TEND1       T1          8.0
        TEND1       T2         15.0
ENDATA
```

Figure 3.2. The stochastics file

3.3 Control File

The "control" file provides the information needed to guide the solution process. It gives:

- algorithm selected (generalized linear programming, bounded variable algorithm, elastic constraints or linear programming)
- input/output units for the files used by the system
- dimensioning information for various arrays within the system
- names of objective and right-hand-side vectors
- additional control parameters e.g. output level, cycle limit, etc.
- specification cards for MINOS

Our design here is similar to the MINOS SPECS file, but our format specification is more rigid and is based upon fields of four characters. Each main section is identified by a principal keyword which begins in column 1. Within each of these further options are identified by a second keyword which begins in column 5. Each of these options may have further suboptions and these are in turn identified by keywords beginning in column 9. The numerical strings or integers which provide the information that goes with a keyword are specified in a data field given by columns 23 through 30. Integers must of course be right justified. Only the first four characters (including blanks) of any keyword are significant.

The principal keywords, i.e. the keywords beginning in column 1, must be specified even when all defaults are selected.

The keywords are as follows:

BEGIN	This is a delimiter identifying the beginning of the control file	
ALGORITHM	This identifies the selected algorithm. Options are ILSRDD, BVSRDD, ELASTIC or MINOS.	
UNIT	The unit numbers are specified as follows:	
	CORE	unit number of "corefile". Default = 5
	STOCHASTICS	unit number of "stochastics" file. Default = 7
	SPECS	unit number of the MINOS SPECS file. Default = 8
	MPS	unit numbers of the MINOS file specifying the matrix. Default = 9

DEBUG — unit number for debugging information. Default = 0 (no output)

LOG — unit number of the log file. Default = 0 (no output)

DIMENSIONS — This specifies information for setting up the work array

ELEMENTS — an upper bound on the number of elements in the matrix (including space for input and generated tenders). Default = 1500

ROWS — an upper bound on the number of rows (including technology). Default = 100

TECHNOLOGY — an upper bound on the number of technology rows. Default = 20

COLUMNS — an upper bound on the number of columns in the matrix (including tenders). Default = 300

PROBABILITIES — an upper bound on the number of discrete levels associated with each stochastic right-hand side. Default = 30

TENDERS — This provides information on tenders as follows:

INPUT — an upper bound on the number specified in the "stochastics" file. Default = 1

GENERATED — an upper bound on the number of tenders saved. Used in the round robin strategy. Default = 20

ELEMENTS — an upper bound on the total number of tender elements. Default = 2000

Note: One must be careful about specifying these quantities.

SELECTORS

OBJECTIVE — name of the objective row – up to 8 characters (must be provided)

RHS — name of the right-hand-side vector – up to 8 characters (must be provided)

BOUNDS — name of the bounds vector – up to 8 characters

RANGES	name of the ranges vector – up to 8 characters
CONTROL OPTIONS	
OUTPUT	output level. Options are 1, 2 or 3, which provide increasingly verbose output. Default = 2
CYCLE	limit on number of tenders generated. Default = 1
SCALE	scale factor (see (2.1)), expressed as a percentage (ρ = SCALE/100). Default = 100.
MINOS SPECIFICATIONS	Here one specifies any MINOS options which are then echoed into the MINOS SPECS file.
END	Delimiter indicating the end of the control section

In our example the "control" file is given in Figure 3.3.

```
BEGIN
ALGORITHM                ILSRDD
UNIT NUMBERS
        CORE FILE                10
        STOCHASTICS FILE         11
        SPECS FILE               12
        MPS FILE                 13
        DEBUG FILE               14
        LOG FILE                 14
DIMENSIONS
        ELEMENTS                700
        ROWS                     10
        COLUMNS                  40
        PROBABILITIES            20
        TENDERS
                INPUT             1
                GENERATED        10
                ELEMENTS         99
SELECTORS
        OBJECTIVE                OBJ
        RHS                      RTH
CONTROL OPTIONS
        OUTPUT                    2
        CYCLE LIMIT               8
        SCALE FACTOR            100
END
```

Figure 3.3. The control file

3.4 Implementation of Problem Setup

This is done using some modules from LPKIT (see Nazareth [17]) suitably modified to suit our purposes. Additional routines have been written to set up information specified in the "stochastics" file into packed data structures and to generate the MINOS SPECS and MPS files.

4. Specialized Setup and Solution

This part of the implementation is built around MINOS Version 5.0 whose outermost routines MINOS1 and MINOS2 were modified for our purposes. In particular, the PHANTOM COLUMNS option of MINOS (simply a device to provide some "elbow-room" in the data structures holding the problem) is extensively used in order to complete the setup of the recourse problem in the packed data structures used by the MINOS system.

4.1 ILSRDD

The master program is defined by expression (3.7) in [21] with $W \triangleq [I, -I]$ and the obvious extension to match expression (2.7) in this paper. MINOS 5.0 sets up the A and T matrices in packed data structures from the MPS file which was generated in the previous phase. Then our modifications to subroutine MINOS2 pack in the additional columns corresponding to tenders. Other routines developed by us, which are called within the subroutine MINOS2, implement the generalized linear programming algorithm in coordination with the solution of each master program by MINOS 5.0. The detailed algorithm is given in [18].

4.2 BVSRDD

This is an implementation of the bounded variable method of Wets [25] in the form given in [21], Section 2.1. Further details of the algorithm may be found in [20]. There is a danger of performing a large number of pivot operations when the probability distribution of each right-hand-side element has many points (the so-called epsilon-to-death problem) but the associated computational effort is alleviated by the way in which MINOS updates its basis matrix representation. It is possible to improve the implementation (a) by using some of the acceleration techniques discussed in Wets [25] which, in effect, carry out several basis changes at the same time, (b) by specifying a good starting basis from the special structure in

(2.7).

In contrast to ILSRDD, implementation is much more straightforward because only an initial linear program must be set up.

4.3 ELASTIC

This option implements the linear program (2.2) (see Section 2.1 of this paper), thereby permitting the "technology rows" to be elastic. The row names defining the technology rows and the penalties q^+ and q^- are defined by the stochastics file. Other data in this file is ignored.

4.4 MINOS

This simply provides the preliminary option of solving an initial linear program. The data in the stochastics file is not required here.

5. Output Phase

The output consists of two parts:

(a) MINOS output in standard MPS format. For a description of this see Murtagh & Saunders [16].

(b) SPORT output. This gives the first-stage and second-stage costs the optimal tender, the dual multipliers (prices) associated with the technology rows in the optimal solution and the probability levels of the equivalent chance-constrained program.

For the earlier example the output is given in Figure 5.1.

Figure 5.1 The output for the earlier example

SPORT(ILSRDD): OPTIMAL SOLUTION OF RECOURSE PROBLEM FOUND

1. MINOS OUTPUT

1
PROBLEM NAME OBJECTIVE VALUE 4.3462500000d+01

STATUS OPTIMAL SOLN ITERATION 1 SUPERBASICS 0

OBJECTIVE OBJ (MIN)
RHS RTH
RANGES
BOUNDS

SECTION 1 - ROWS

NUMBER	...ROW..	STATE	...ACTIVITY...	SLACK ACTIVITY	..LOWER LIMIT.	..UPPER LIMIT.	.DUAL ACTIVITY	..I
25	OBJ	BS	43.46250	-43.46250	NONE	NONE	-1.00000	1
26	A1	EQ	15.00000	0.	15.00000	15.00000	-0.43750	2
27	A2 A	EQ	12.00000	0.	12.00000	12.00000	0.	3
28	A3	EQ	3.30000	0.	3.30000	3.30000	5.62500	4
29	A4	EQ	4.00000	0.	4.00000	4.00000	1.12500	5
30	T1	EQ	0.	0.	0.	0.	-0.25000	6
31	T2	EQ	0.	0.	0.	0.	1.43750	7
32	CONVXITY	EQ	1.00000	0.	1.00000	1.00000	26.96250	8

1
SECTION 2 - COLUMNS

NUMBER	.COLUMN.	STATE	...ACTIVITY...	.OBJ GRADIENT.	..LOWER LIMIT.	..UPPER LIMIT.	REDUCED GRADNT	M+J
1	CLM1	BS	8.00000	1.00000	0.	NONE	0.	9
2	CLM2	BS	2.25000	2.00000	0.	NONE	-0.00000	10
3	CLM3	LL	0.	3.00000	0.	NONE	2.12500	11
4	CLM4	LL	0.	0.	0.	NONE	5.62500	12
5	CLM5	BS	7.00000	1.00000	0.	NONE	0.	13
6	CLM6	BS	8.00000	2.00000	0.	NONE	0.	14
7	CLM7	LL	0.	3.00000	0.	NONE	0.88750	15
8	CLM8	LL	0.	0.	0.	NONE	1.12500	16
9	CLM9	LL	0.	0.	0.	NONE	0.43750	17
10	CLM10	BS	1.75000	0.	0.	NONE	0.	18
11	PHNT1001	LL	0.	2.00000	0.	NONE	2.25000	19
12	PHNT1002	LL	0.	2.00000	0.	NONE	0.56250	20
13	PHNT1003	LL	0.	1.00000	0.	NONE	0.75000	21
14	PHNT1004	LL	0.	1.00000	0.	NONE	2.43750	22
15	PHNT1005	LL	0.	3.66000	0.	NONE	0.36000	23
16	PHNT1006	LL	0.	10.40000	0.	NONE	3.00000	24
17	PHNT1007	BS	0.87500	7.90000	0.	NONE	0.00000	25
18	PHNT1008	LL	0.	4.20000	0.	NONE	0.11250	26
19	PHNT1009	BS	0.12500	8.40000	0.	NONE	-0.00000	27
20	PHNT1010 A	EQ	0.	0.	0.	0.	0.	28
21	PHNT1011 A	EQ	0.	0.	0.	0.	0.	29
22	PHNT1012 A	EQ	0.	0.	0.	0.	0.	30
23	PHNT1013 A	EQ	0.	0.	0.	0.	0.	31
24	PHNT1014 A	EQ	0.	0.	0.	0.	0.	32

ENDRUN

2. SPORT(ILSRDD) OUTPUT

COSTS ASSOCIATED WITH ABOVE SOLUTION

TOTAL = 43.46250 DIRECT = 35.50000 RECOURSE = 7.96250

QUANTITIES ASSOCIATED WITH ABOVE SOLUTION

ROW NAMES	TENDERS	PRICES	CC-EQUIVALENTS
T1	10.25000	-0.25000	0.75000
T2	15.00000	1.43750	0.18750

6. Testing

The program has been exercised on several test problems as follows:

(a) The product-mix example of Section 3 due to J. Ho. This is a "toy" problem with 5 rows of which 2 are technology rows and 6 first-stage decision variables.

(b) The test problem given by Kallberg & Kusy [11]. This too is a "toy" problem with 3 rows of which 2 are technology rows and 6 first-stage decision variables. (Documented in King [12].)

(c) The test problem given by Cleef [3]. This has 9 rows of which 6 are technology rows and 16 first-stage decision variables. (Documented in King [12].)

(d) The problem of allocating aircraft to routes given in Dantzig [4]. This has 9 rows of which 5 are technology rows and 29 first-stage decision variables. (Documented in King [12].)

(e) A discretized version of the stochastic transportation problem given by Qi [23] formulated as a standard stochastic linear program with simple recourse. This has 78 rows of which 44 are technology rows and 1496 first-stage decision variables.

The bank asset and liability model given by Kusy & Ziemba [13] and a full-scale version of problem (d) above both provide good illustrations of the practical applications for which our program is designed.

7. Sportsmanship

The current system can be applied to a wider range of problems than would appear at first sight. For example when the stochastic linear program has stochastic technology matrices with a few discrete probability levels (which are independent of the right-hand-side distribution) say, $T_1,\ldots,T_t$ with probabilities $p'_1,\ldots,p'_t$, then we can express this as an equivalent problem

$$\text{minimize} \quad cx + p'_1 q^+ y_1^+ + p'_1 q^- y_1^- + \cdots + p'_t q^+ y_t^+ + p_t q^- y_t^-$$

subject to

$$\begin{array}{lllll} Ax & & & & = b \\ T_1 x + & [I,-I]\begin{pmatrix} y_1^+ \\ y_1^- \end{pmatrix} & & & = h(\omega) \\ \cdot & \cdot & & & \cdot \\ \cdot & \cdot & & & \cdot \\ \cdot & \cdot & & & \cdot \\ T_t x & \cdots\cdots & \cdots + & [I,-I]\begin{pmatrix} y_t^+ \\ y_t^- \end{pmatrix} & = h(\omega) \end{array} \qquad (7.1)$$

$$x \,,\, y_j^+ \,,\, y_j^- \geq 0$$

Let us treat T defined by

$$T = \begin{bmatrix} T_1 \\ \vdots \\ T_t \end{bmatrix}$$

as a technology matrix in the usual way. Then we can set up the problem so that it can be solved by the system, as described earlier, with appropriate definition of penalties and distribution determined by (7.1).

In some situations the underlying probability distribution of $h(\cdot)$ is only known implicitly through a simulation model involving the random elements ω. Nazareth [18] discusses how the system can be extended to this case (see, in particular, Section 3.2 of [18] for some numerical experiments).

When the probability distribution of $h(\cdot)$ is not discrete, SPORT 2.0 can be used in conjunction with some iterative discretization procedure and computation of error bounds (see, for example, [26]).

When a more complex penalty structure is imposed on the second stage, program modifications would be required. This could, in many cases, be done fairly easily.

8. Availability

The Fortran implementation described here, SPORT 2.0 (pronounced SupPORT Version 2.0) was developed for use at IIASA on the VAX 11/780 (under the UNIX operating system). It uses MINOS 5.0 (the latest documented version), which is available in-house. Using the terminology in Nazareth [19], the current version of our system is a level-2 implementation, designed for algorithmic experimentation

and for problem solving by an experienced user (one expected to be familiar both with his problem and with the implemented algorithm).

To use SPORT 2.0 at another site, it would be necessary to obtain MINOS 5.0 independently from Stanford University and to *substitute* our set of Fortran routines for the two MINOS 5.0 files MI00MAIN and MI10MACH. (Note that SPORT 2.0 will not run with versions of MINOS below 5.0.)

An earlier version of our system, designed for MINOS 4.9, SPORT 1.1, is available on the SDS/ADO tape, which is a collection of a number of routines for stochastic programming. This version provides readable Fortran and a manual (see Edwards [6]) to document our implementation. Note that it is *not* executable, since MINOS 4.9 is not included with it.

In order to obtain a copy of SPORT 2.0, please contact the author of this article at either of the following addresses:

IIASA, System & Decision Sciences, A-2361, Laxenburg, Austria

or

CDSS, P.O. Box 4908, Berkeley, California 94704, USA

9. Stochastic Programming with Recourse as a Form of Post-Optimal Analysis in a Mathematical Programming System

Many large-scale Mathematical Programming Systems (for example, MPSX/370 [1]) provide options for performing *parametric* and *sensitivity analysis* in the optimal solution of a linear program and for repeated (and efficient) reoptimization through a dual simplex procedure, when the right-hand-side is changed. (For MINOS, post-optimal analysis routines have been developed by Dobrowski, et al [5].)

A common approach for handling uncertainty in the right-hand-side is to use *scenario analysis*, which is indeed greatly facilitated by the above post-optimal options. Ermoliev and Wets [8] characterize this approach to dealing with uncertainty as being "seriously flawed" and explain why as follows: "Although it (scenario analysis) can identify 'optimal' solutions for each scenario (that specifies some values for the unknown parameters), it does not provide any clue as to how these 'optimal' solutions should be combined to produce a merely reasonable decision." Another approach that has been utilized by mathematical programmers as discussed in Section 2.1 is to introduce *elastic constraints* by defining

penalties on shortage and surplus for a given right-hand-side. This, as we have noted, is in the spirit of the recourse model, but it does not yet address the stochastic aspect of the right-hand-side elements.

One aim of our paper has been to demonstrate (hopefully convincingly) that *recourse analysis* could be introduced in a very natural way as a post-optimal analysis option in an MPS and that its implementation is not substantially more difficult than that of other post-optimal analysis options currently provided within them. It could be argued, of course, since problem (2.7) can be directly expressed as a linear program, that it could be left up to the user to set up this linear program, create the appropriate MPS file and solve it in the conventional way. This is to impose upon him or her a laborious and error prone task. To do so would be as unreasonable as requiring that the user implement his own post-optimal parametric and sensitivity analysis. Another approach is to use an extended LP system based upon piecewise-linear (separable) programming (see Fourer [9]) to solve (2.5) or (2.7). Unfortunately such systems are not available as general purpose software. Thus it is necessary to fall back upon the more conventional mathematical programming systems.

The particular implementation described in earlier sections of this paper was developed for MINOS (specifically Version 5.0) in its linear programming mode, but an implementation for another large-scale linear programming system (MPS) could be patterned along rather similar lines (see, in particular, Figure 2.1). This would require the following:

(a) Firstly, augmentation of the standard MPS description of a linear program (which may be formulated and solved as a first step) by some standardized description of the stochastic information. A format along similar lines to Section 3.2 would be quite appropriate. Note that this does *not* conflict with the trend toward high-level modeling systems for defining mathematical programming problems (see, for example, the GAMS System of Brooke, et al. [2]). MPS format (and its extension to stochastic problems) primarily serves the purpose of formalizing the *interface* to optimization codes and indeed MPS format continues to play this role in systems like GAMS. (With regard to the third "control" file of Figure 2.1, note that this is specific to the MINOS implementation and would obviously vary with different MPS systems.)

(b) Secondly, set up of one or more linear programming problems corresponding to (2.7) by augmenting internal data structures. The more straightforward implementation (because it involves only one augmentation) is to use some version of the bounded variable method of Wets [25] as in BVSRDD (see Section 4.2.). Assuming that a deterministic version of the problem has already been solved, the additional columns could be inserted directly into the packed data representation used by the MPS from the stochastic information supplied as described in (a) above, and the problem reoptimized. (It would be wasteful to generate a fresh MPS file for (2.7).) In MPSX/370, the augmentation and reoptimization could be done through the Extended Control Language (see [1]). The difficulty with the bounded variable approach arises when the distribution has many points, for example, when it is obtained by discretizing a continuous distribution. See the discussion in Section 4.2. Also it does not generalize to non-simple recourse. The alternative is to implement the generalized linear programming approach, again directly inserting the added columns into internal data structures and solving a sequence of linear programs, each starting off where the previous one left off (as in ILSRDD, Section 4.1). As we have seen, implementation required modification only of the outermost level of MINOS and we believe this would be true for other MPS systems as well. The ILSRDD algorithm is very efficient in this context and as we may note, the approach applies to more general forms of recourse.

(c) Thirdly, the output of the solution in an appropriate way, again done most conveniently through access to the internal data structure.

To summarize, the mathematical programming field is ripe for incorporating some forms of stochastic programming with recourse into current large-scale MPS systems. We have provided a detailed illustration of how it can be done for one currently available MPS and how it could (possibly even should) be done for other systems.

References

[1] M. Benichou, J.M. Gauthier, G. Hentges and G. Ribiere, "The efficient solution of large-scale linear programming problems - some algorithmic techniques and computational results," *Mathematical Programming* **13**(1977) 280-322.

[2] A. Brooke, A. Drud and A. Meeraus, "High level modelling systems and nonlinear programming," Development Research Department, World Bank. (Presented at 12^{th} International Symposium on Mathematical Programming, Boston, Mass. 1985).

[3] H. Cleef, "A solution procedure for the two-stage stochastic program with simple recourse," *Z. Operations Research* **35**(1981) 1-13.

[4] G.B. Dantzig, *Linear Programming and Extensions*, Princeton University Press, 1963.

[5] G. Dobrowolski, T. Rys, J. Hadjuk and A. Korytowski, "POSTAN 2 - Extended postoptimal analysis package for MINOS," in A. Lewandowski and A. Wierzbicki eds., *Theory, Software and Test Examples for Decision Support Systems*, IIASA Collaborative Volume (1985), 142-176.

[6] J. Edwards, "Documentation for the ADO/SDS collection of stochastic programming codes," WP-85-02 (1985), IIASA, Laxenburg, Austria.

[7] J. Edwards, J. Birge, A. King and L. Nazareth, "A standard input format for computer codes which solve stochastic programs with recourse and a library of utilities to simplify its use," WP-85-03 (1985), IIASA, Laxenburg, Austria.

[8] Y. Ermoliev and R. J-B. Wets, "Numerical techniques for stochastic optimization problems," PP-84-04 (1984), IIASA, Laxenburg, Austria (in this volume).

[9] R. Fourer, "Piecewise-linear programming," Technical Report, Dept. of Ind. Eng. and Management Sciences, Northwestern University, Evanston, IL, USA (1985).

[10] J. Ho, private communication (1983).

[11] J. Kallberg and M. Kusy, "Code Instruction for S.L.P.R., a stochastic linear program with simple recourse,", Technical Report, University of British Columbia (1976).

[12] A.J. King, "Stochastic programming problems: examples from the literature," Department of Applied Mathematics, University of Washington, Seattle, USA (1985) (appears in this volume).

[13] M. Kusy and W. Ziemba, "A bank asset and liability management model," CP-83-59, IIASA, Laxenburg, Austria (1983).

[14] IBM Document No. H20-0476-2, "Mathematical Programming System 360 Version 2, Linear and Separable Programming - User's Manual," IBM Corporation, New York.

[15] B.A. Murtagh and M.A. Saunders, "Large-scale linearly constrained optimization," *Mathematical Programming* **14**(1978) 41-72.

[16] B.A. Murtagh and M.A. Saunders, "MINOS 5.0 User's Guide," Report No. SOL 83-20, Systems Optimization Laboratory, Stanford University (1983).

[17] J.L. Nazareth, "Implementation aids for optimization algorithms that solve sequences of linear programs by the revised simplex method, WP-82-107 IIASA, Laxenburg, Austria (1982).

[18] J.L. Nazareth, "Algorithms based upon generalized linear programming for stochastic programs with recourse," in: F. Archetti, ed., *Proceedings of IFIP International Workshop on Stochastic Programming: Algorithms and Applications*, Springer-Verlag (to appear).

[19] J.L. Nazareth, "Hierarchical implementation of optimizations methods," in: P. Boggs, R. Byrd and B. Schnabel, eds., *Numerical Optimization, 1984* SIAM, Philadelphia, 199-210 (1985)

[20] J.L. Nazareth and R.J-B. Wets, "Algorithms for stochastic programs: the case of non-stochastic tenders," A. Prekopa and R.J-B. Wets, eds., *Mathematical Programming Study*, (to appear).

[21] J.L. Nazareth and R.J-B. Wets, "Nonlinear programming techniques applied to stochastic programs with recourse," WP-85-62, IIASA, Laxenburg, Austria (1985) (appears in this volume).

[22] S.C. Parikh, Lecture notes on stochastic programming, unpublished, University of California, Berkeley (1968).

[23] L. Qi, Ph.D. Dissertation, Computer Science Department, University of Wisconsin, Madison (1983).

[24] J. Tomlin, private communication

[25] R. J-B. Wets, "Solving stochastic programs with simple recourse," *Stochastics* **10**(1983) 219-242.

[26] R. J-B. Wets, "Stochastic programming: solution techniques and approximation schemes," in A. Bachem, M. Groetschel and B. Korte eds., *Mathematical Programming: The State of the Art*, Springer-Verlag, Berlin (1983) 566-603.

[27] W.T. Ziemba, "Stochastic programs with simple recourse," in: P. Hammer and G. Zoutendijk, eds., *Mathematical Programming in Theory and Practice*, North-Holland, Amsterdam (1974) 213-274.