CONCEPT

A System for Large Scale Dynamic
Nonlinear Optimization

USER'S MANUAL
by
Arne Drued
and
Alexander Meeraus

Development Research Center
World Bank
Washington, D.C., U.S.A.

Technical Note No. 16
Research Project 571-58
March 1980
Version 0.105/UPDATE 1
1. Introduction

The following manual describes the use of the optimization code CONOPT with the features that have been implemented by the beginning of November, 1979.

The code is still an experimental code, and all users are encouraged to report any problems or errors they meet. Also, users are encouraged to comment on this manual - if certain features are not explained very well or if apparently correct input decks causes error messages, and on the code itself - e.g., which additional features would be useful.

The manual is organized as follows: In section 2 we give a short definition of the mathematical problem we want to solve and we give the main assumptions on which the code is based. Section 3, 4 and 5 defines the optimization problem to the code. In section 3 we describe the MODEL-input that defines the structure of the problem, i.e. which variables enter in which relations, and defines the constant linear part of the model. The format of the MODEL-input is chosen as close to standard MPS-format for ordinary LP-problems as possible with a few additional features to define the time structure and the nonlinearities of the problem.

Section 4 describes the SERIES-input where the numerical values of the linear but time dependent parts of the problem can be defined.

The structure of the nonlinearities is defined in the MODEL-input. Section 5 describes how the actual nonlinear functions are entered. The user must supply a FORTRAN subroutine called FCOMP that can compute numerical
values of the nonlinear expressions, and the values are passed as black-box information to the optimizer. A plan for the future is to by-pass the FORTRAN subroutine and define the nonlinearities in a form that can be manipulated by the optimization code.

CONCEPT has a control language for defining and changing right hand sides, bounds, objective function, optimization direction, output levels, etc. Instead of inventing a new control language we have, being satisfied users of CDC's LP-system APEX, chosen to make a control language that very much follows the lines of APEX control language with the necessary additions of verbs to suit the dynamic and nonlinear problems. The control language is described in section 6.

The output from CONCEPT can be divided into different groups as reproduction of input, iteration log information, optimal or intermediate solutions, etc. The meaning of this output, especially the abbreviations in the page and table headings, is explained in section 7.

Finally, in section 8 we give an example of a small optimal control problem, and we show the four input files: MODEL, SERIES, FCOMP, and Control-Program.

At the moment we have not implemented any revise and restart facilities. Before we start implementing anything in this area we would like to have comments from users on the types of revisions and restarts they would find useful.

2. The Mathematical Formulation of the Problem

The type of nonlinear optimization problem we are concerned with can without loss of generality be defined as:
\[ \min \text{ or } \max \ z = \sum_{t=0}^{T} f (x_t, x_{t-1}, \ldots, x_{t-L}, t) \]
subject to
\[ g_i (x_t, x_{t-1}, \ldots, x_{t-L}) \cdot \text{rel.} \cdot b_i, \quad i=1, \ldots, n; \quad t=1, \ldots, T \]
and \[ \alpha_t \leq x_t \leq \beta_t, \quad t = 1, \ldots, T \]

where:
- \( x_t \) is a vector of continuous optimization variables in period \( t \).
  The dimension of \( x_t \) is \( m \), independent of \( t \).
- \( g_i \) is a linear or nonlinear function with continuous first derivatives.
- \( \text{rel.} \) is a relation i.e. \( <, =, > \) or non binding.
- \( \alpha_t \) and \( \beta_t \) are vectors of lower and upper bounds, some of which can be \( -\infty \) or \( +\infty \).
- \( f \) is an objective function with continuous first derivatives.
- \( L \) is the maximum number of lags.
- \( x_{t-s} \) indicates the vector of optimization variables lagged \( s \) periods, i.e. from \( s \) periods before period \( t \).

We assume that:
- the functions \( g \) and \( f \) can be expressed by the same formula in all time periods \( t \), especially each function depends always on the same set of variables. This assumption is certainly true for \( T=1 \), i.e. static problems.
- the components of \( x_0, x_1, \ldots, x_L \) that are used in the \( g \) and \( f \) functions are all known.
- the relation \( (\leq, =, >, \text{or non binding}) \) for a certain constraint is the same in all periods.
- all variables enter unlagged in at least one relation.
- the functions f and g are defined for all values of x that satisfies the bounds.

3. Structural Information - The MODEL-File

The structure of the problem is entered in a modified MPS-format with 4 to 3 sections. The sections must be entered in the order shown below:

<table>
<thead>
<tr>
<th>MODEL - card</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHS - section</td>
</tr>
<tr>
<td>LAGS - section (optional)</td>
</tr>
<tr>
<td>BOUNDS - section (optional)</td>
</tr>
<tr>
<td>RANGES - section (optional)</td>
</tr>
<tr>
<td>ENDATA - card</td>
</tr>
</tbody>
</table>

The format of the input cards are standard MPS-format with each card divided into 5 fields:

<table>
<thead>
<tr>
<th>Field</th>
<th>Columns</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2-3</td>
<td>Type of row or bound.</td>
</tr>
<tr>
<td>2</td>
<td>5-12</td>
<td>Name of column, right-hand side, boundset, or rangeset</td>
</tr>
<tr>
<td>3</td>
<td>15-22</td>
<td>Name of row or column.</td>
</tr>
<tr>
<td>4</td>
<td>25-36</td>
<td>Value indicator for derivative, right-hand side, bound, or range.</td>
</tr>
<tr>
<td>5-6</td>
<td>40-47</td>
<td>same as field 3.</td>
</tr>
<tr>
<td>5-6</td>
<td>50-61</td>
<td>same as field 4.</td>
</tr>
</tbody>
</table>

The name fields 2, 3 and 5 defined above contain 3 characters, and in the following we will always refer to names as 3 alphanumeric characters. However, in the CDC-version of the code all names can have 10 characters and the fields are extended 2 characters to the right, i.e. 5-14, 15-24, and 40-49.

In the names blanks are considered as ordinary characters, i.e. "X123456" and "X123456" are not identical. The values in field 4 and 6 are read with a F12.5 format and they should contain a decimal point. If a
decimal point does not appear or it is assumed to be located between column
30 and 31 or between column 55 and 56.

3.1 MODEL-card

The MODEL-card is the first card in the input deck. It is used to
identify the input deck. The format is:

Column 1-5:  MODEL

Field 3: A string of 3 alphanumeric characters identifying the
         input deck.

3.2 ROWS-section

The ROWS-card is the first card in the second data section. The
format is:

Column 1-4:  ROWS

After the ROWS-card each row or relation in one period of the model
must be described by one card with the following format:

Field 1: The type of the row or relation with L for less than or
         equal (\leq), G for greater than or equal (\geq), E for equal
         (=), and N for non binding. The type character can be
         punched in column 2 or 3.

Field 2: The name of the row, 5 alphanumeric characters.

The objective function expression must be entered as a row of type N.

3.3 COLUMNS-section

The COLUMNS-card is the first card in the third data section. The
format is:

Column 1-7:  COLUMNS
The COLUMNS-section describes which variables (columns) enters in which equations (rows) without a lag. The data is entered columnwise, and one column must be finished before the next is started. Within each column the order of the rows is unimportant.

The format of the data cards is:

Field 2: The name of the column, an 8 character variable name. Must be present on all data cards.

Field 3: Rowname, 8 characters.

Field 4: Derivative indicator.

Field 5: Rowname or blank.

Field 6: Derivative indicator if field 5 is not blank.

The derivative indicator describes how the variable (column) enters the equation (row). CONCEPT recognizes 4 different types of derivatives:

Derivative type 1: constant and time independent, i.e. the variable enters linearly with a constant coefficient.

2: constant but time dependent, i.e. the variable enters linearly with a time dependent coefficient.

3: variable derivative that only depends on lagged variables, i.e. the variable \( x_{it} \) enters in an expression like \( x_{it} = a(x_{t-1}, x_{t-2}, \ldots) \) where \( a(.) \) is any function of lagged variables.

4: variable derivative that depends on unlagged variables, i.e. the variable enters in a truly nonlinear fashion.

The derivative indicators (punched in field 4 or 6 of the data card) for the four derivative types are as follows:
Type 1: The value of the constant coefficient or derivative.
Type 2: 9999.
Type 3: 99999.
Type 4: 999999.

**Note:**

a. The special indicators 999., 99999., and 999999. are default value, and it is possible to change them in the control program, see section 6.

b. The CONOPT code does not distinguish between derivatives of type 3 and 4 at the moment.

c. It is always allowed to increase the type number of a derivative. However, the user should try to use the lowest possible type number because the solution algorithm takes advantage of this information.

d. In static problems, i.e. problems with T=1, derivatives of type 2 and 3 does not exist.

e. The values of a type 2 derivative can be supplied in the SERIES file, see section 4. Note however, that the definition of the non-linear expression that should be coded in a FORTRAN subroutine depends on which time series have been entered. Read section 5 carefully.

### 3.4 LAGS-sections

If the model relations contain lags (which all dynamic models should) the structure of the lagged part of the relations must be defined in one or more LAGS-sections.
The LAGS-card is the first card in each LAGS-section. The format is:
Column 1-4: LAGS
Field 4: lag number for this LAGS-section - a positive integer written
with a decimal point.
The LAGS-sections must be entered with increasing lag number.
Within each LAGS-section the data are entered with the same format
as in the COLUMNS-section with column-name, row-name and derivative indicator.
Each input element indicates that the variable named by the column name enters
the relation named by the row name lagged the number of periods indicated
by the lag number on the LAGS-card. The columns must be in the same order
as in the COLUMNS section although not all columns have to be present in the
LAGS-sections.

3.5 RHS-section

If a right-hand side is nonzero it must be specified in the RHS-
section. The first card in this section is the RHS-card with the following
format:
Column 1-3: RHS

More right-hand sides can be entered. The particular right-hand
side that should be used in the optimization is defined in the control
program. All right-hand sides are entered in a format similar to the format
of the COLUMNS-section, and as in the COLUMNS-section each right-hand side
must be finished before the next is started. The format of each card is:
Field 2: Name of the right-hand side, an 8 character name.
Field 3: Rowname, 8 characters.
Field 4: RHS-indicator.
Field 5: Rowname or blank.
Field 6: RHS-indicator if field 5 is not blank.
The RHS-indicator describes whether the right-hand side in the row depends on time, in which case 9999. is entered, or is time independent, in which case the right-hand side value is entered. Constant zero right-hand sides need not be specified. The 9999.-indicator is the same as the 9999.-indicator in the COLUMNS-section and it can be changed in the control program if necessary. The actual time series of the values of the right-hand sides are entered through the SERIES file, see section 4.

A special right-hand side named FUNCTION is reserved. This right-hand side is used to give the connection between the rownames and the row-indices used in the FORTRAN subroutine that computes the nonlinear expressions, see section 5. Field 4 and 6 of this right-hand side must contain a positive integer, written with a decimal point. The integer is interpreted as the row-index of the row, and it is the users responsibility that this row-index actually corresponds to the index used in the FORTRAN subroutine.

The only constraints on the row-indices are that they must be positive, less than or equal to the total number of rows, and distinct. If a FUNCTION right-hand side is present it must supply row-indices for all nonlinear functions. If no FUNCTION right-hand side is supplied, all the rows, both linear and nonlinear, will be given indices after their occurrence in the ROWS-section.

3.6 BOUNDS-section

If some of the variables have bounds different from \( \geq 0 \), which is the default bound, these bounds must be entered in a BOUNDS-section. The first card in this section is the BOUNDS-card with the following format:

Column 1-6: BOUNDS
More sets of bounds can be entered in one input deck. They are distinguished by a bound name. The particular bound set that should be used in an optimization is defined in the control program.

The format of the data cards is:

Field 1: The bound type. There are the following possibilities:
- UP - upper bound
- LO - lower bound
- FR - free variable
- FX - fixed variable
- MI - ≤ 0
- PL - ≥ 0 (≠ ult)

Field 2: Bound name, an 8 character name.

Field 3: Column name, 8 characters.

Field 4: BOUND-indicator or blank.

The following rules apply to the order of the data cards:

a. One bound set must be finished before the next bound set is started.

b. Within each bound set the columns must appear in the same order as in the COLUMNS-section, although not all columns need to be mentioned.

c. Only the following combination of two bounds for one column are allowed: (LO, UP), (LO, MI), (PL, UP), and (FR, UP). The last combination is used for bounds of the type $x \leq b$ where $b > 0$ since UP alone with a positive bound implies $0 \leq x \leq b$. Within a combination the order is unimportant, except with (FR, UP) where FR must be first.
The COLUMN-indicator in field 4 is only used with bound type JP, LO, and FX. If the bound is independent of time the value is simply entered. If the bound depends on time the time series indicator 9999. is used. If necessary, the 9999.-indicator can be changed as mentioned in the COLUMNS- and RHS-sections.

Some special bound set names, VARIABLE and all names starting with INIT, especially INITIAL, are reserved. The VARIABLE bound set is used to give the connection between the column names and the indices of the variables used in the FORTRAN subroutine that computes the nonlinear expressions, see section 5. Field 1 should be blank, and field 4 must contain a positive integer written with a decimal point. The integer is used as the index of the variable in the FORTRAN subroutine, and it is the users responsibility that the column names actually correspond to the indices used in the FORTRAN subroutine. The only constraints on the indices are that they must be positive, less than or equal to the total number of columns, and distinct. If a VARIABLE bound set is present it must supply indices for all nonlinear variables. If no VARIABLE bound set is supplied the variables will be given indices after their occurrence in the COLUMNS-section.

The INITIAL bound set is used to give the variables initial values that can be used as a starting point for the optimization. Field 1 can only contain the types FX, LO, and UP. If the bound type is FX the BOUND-indicator in field 4 contains the initial value. If the bound-type is LO or UP the variable is initialized at its lower or upper bound, respectively, and field 4 is not used. It is not necessary to supply any initial values and if an INITIAL bound set is present it need not contain all variables. Variables that are not initialized explicitly will be initialized as follows:
a. Fixed variables are of course initialized at their fixed value.
b. Variables without a bound are initialized at zero.
c. Variables with one bound like UP, LO, MI, or PL are initialized at this bound.
d. Variables with two bounds like (UP, LO), (MI, LO) or (PL, UP) are initialized at the numerically smallest bound and in case of a tie at the upper (positive) bound. Note, that a variable with a positive upper bound is assumed to have lower bound 0 unless another lower bound is defined, and it will be initialized at 0.
e. Lagged variables from periods before the planning horizon are initialized at 0.

If an initial value in the INITIAL bound set does not satisfy the bounds in the bound set that is selected for the particular optimization, the initial value is moved to the nearest bound.

It is possible to have different sets of initial values in the input deck. The initial values are entered in a bound set with a name starting with INIT using the same rules as for the INITIAL bound set. The actual INITXXX set to be used can be defined in the control program.

3.7 RANGES-section

A constraint of the type \( l_{it} \leq g_i (x_t, x_{t-1}, \ldots, t) \leq u_{it} \) can be specified in the RANGES-section. The value of either \( l_{it} \) or \( u_{it} \) as well as one of the inequalities must be defined in the ROWS-section, and the value of \( r_{it} = u_{it} - l_{it} \) is defined in the RANGES section. If \( b_{it} \) is the value defined for the row in the RHS-section the final constraint will be as follows:
row type G: \( b_{it} \leq g_i(x_t, x_{t-1}, \ldots, t) \leq d_{it} = r_{it} \)
row type L: \( b_{it} - r_{it} \leq g_i(x_t, x_{t-1}, \ldots, t) \leq b_{it} \)

The first card in the RANGES-section is the RANGES-card with the following format.

Column 1-6: RANGES

More range-vectors, i.e. \( r_{it} \)-vectors, can be entered. The particular range-vector to be used in the optimization is defined in the control program. All range-vectors are entered in a format similar to the format of the COLUMNS- and ROWS-sections, and as before each range-vector must be finished before the next is started. The exact format is:

Field 2: Name of the range vector, an 8 character name.
Field 3: Rowname, 8 characters.
Field 4: RANGE-indicator.
Field 5: Rowname or blank.
Field 6: RANGE-indicator if field 5 is nonblank.

The RANGE-indicator gives the value of \( r_{it} \) in the usual way: A time independent \( r_{it} \) is entered directly while a time-dependent \( r_{it} \) is indicated by the 9999.-indicator with the actual time series data entered in the SERIES section.

3.8 ENDATA-card

The file with the structural information, the MODEL-file, must always end with an ENDATA-card with the format:

Column 1-6: ENDATA
3.9 ERRORS

The code tests for inconsistencies in the input and prints an error message if something is wrong. In the cases where it is possible to recover simply by deleting the wrong data element the errors are counted as minor errors, and it will be possible to continue after a MODEL-file with minor errors. If there are more serious errors, counted as major errors, it is generally not possible to continue. However, an exception are major errors in RHS-, BOUND-, or RANGE-sets. The erroneous set is deleted, but the job can continue as long as the deleted set is not used to define the actual optimization problem.

4. Time Dependent Information - The SERIES-File

The structural description of the problem in the MODEL-file is independent of the number of time periods in the optimization problem. Before the actual optimization can be initialized we will have to define the optimization time periods and enter data for the time series in the model. The time period information is entered in the control program, and the time series are entered via the SERIES file.

The SERIES file consists like the MODEL-file of sections: SERIES-card, COLUMNS-section, LAGS-sections, RHS-section, BOUNDS-section, RANGES-section, and ENDDATA-card. All except the SERIES and ENDDATA cards are optional, but if they appear their relative order must be as in the MODEL-file. In each section data can be entered for the entries that were defined by the 9999.- indicator in the MODEL-file.

4.1 SERIES-card

The SERIES-card must always be the first card in the time series input deck. The format is as the MODEL-card in the MODEL-file.
4.2 COLUMNS-section

If a COLUMNS-section is present it must start with a COLUMNS-card as in the MODEL-file. For each 9999.-indicator in the columns-section a time series can be entered, but it is not required. See section 5 for a description of how the relations are between the FORTRAN subroutine describing the nonlinearities and the data supplied in the time series data section.

Each time series must be defined by a header card with the following format:

- Column 5-12: column name, 8 characters.
- Column 15-22: row name, 3 characters.
- Column 25-36: first time period in the time series, an integer written with a decimal point.
- Column 37-48: last time period in the time series, an integer written with a decimal point.

The time series must cover the whole optimization period but it is valid to enter a time series that covers a longer period both before and/or after the optimization period. Thus, it is possible to write one SERIES file and use it for optimizations over different time periods.

The time series itself is entered on subsequent cards using a (4F18.10) format.

The relative order of the time series must be the same as the order of the 9999.-entries in the MPS-file.

4.3 LAGS-sections

If one or more LAGS-sections are present they must all start with a LAGS-card as in the MODEL-file. The LAGS-sections must be entered with increasing lag number. The time series in these sections are also used to
enter data for the 9999.-indicators, and as in the COLUMNS-section it is not required to enter time series for all 9999.-indicators, but the time series that are actually entered must be in the same relative order as the 9999.s in the MPS-file. Each time series must be defined by a header card with the same format as in the COLUMNS-section, and the series are then written in (4F13.10) format on subsequent cards.

The definition of the time periods in connection with lagged derivatives is as follows: Let $s$ be the lag number. The derivative of a row in period $t$ with respect to a variable in period $t-s$ is assigned to period $t$, i.e. to the period of the row or constraint. If the optimization is performed from period $t_1$ to $t_2$ inclusive the time series must at least contain the derivatives from period $t_1$ to $t_2$ inclusive.

4.4 RHS-section

If the right hand side that is going to be used in the optimization contains any 9999.-indicators, the time series must be supplied in the RHS-section of the SERIES-file.

The RHS-section is started by a RHS-card with the same format as in the MODEL-file. The right hand sides must be mentioned in the same order as in the MODEL-file although not all right hand sides need to be mentioned, and within each right hand side the time series must be supplied in the same order as in the RHS-section of the MPS-file. If one 9999.-indicator in a right hand side is entered all time series in this right hand side must be entered.

Each time series must be defined by a header card with a format similar to the header card in the COLUMNS- and LAGS-sections:
Column 5-12: right hand side name, 3 characters
Column 15-22: row name, 8 characters
Column 25-36: first time period in the time series, an integer written with a decimal point
Column 37-48: last time period in the time series, an integer written with a decimal point

The time series itself is entered in (4F18.10) format immediately following the header card. The time series must at least cover the whole optimization period.

1.5 BOUNDS-section

If the bound set or INIT-bound set that is going to be used in the optimization contains any 9999.-indicators the time series must be supplied in the BOUNDS-section of the SERIES-file. The BOUNDS-section is started by a BOUNDS-card with the same format as in the MODEL-file.

The bound sets must be mentioned in the same order as in the MPS-file although it is not necessary to mention all bound sets. Within each bound set the time series must be entered in the same order as in the BOUNDS-section of the MPS-file. If a bound set is mentioned, all time series indicated by the 9999.-indicator in the bound set must be entered.

Each time series must be defined by a header card with the following format:

Column 5-12: Bound set name, 8 characters
Column 15-22: Column name, 8 characters
Column 25-36: First time period in the series, an integer written with a decimal point
Column 37-48: Last time period in the series, an integer written with a decimal point.

The time series itself is entered in (4F10.10) format immediately following the header card. For ordinary bound sets the time series must cover the whole planning period. For initial value bound sets the time series must at least cover the lagged values that are used in the model. Else the series can cover the whole, the first part of or none of the optimization period.

4.6 RANGES-section

The rules for the ranges is similar to the rules for the bounds: if the range set that is used in the optimization contains a 9999.-indicator the corresponding time series must be supplied in the RANGES-section of the SERIES-file. The RANGES-section is started by a RANGES-card with the format described for the MODEL-file.

Range sets must be mentioned in the same order as in the MODEL-file, but some of them can be left out. Within each range set the time series must be entered in the same order as in the RANGES-section of the MODEL-file. If a range set is mentioned all time series in the set must be supplied.

The header card for the time series has the format:

Column 5-12: Range set name, 8 characters
Column 15-22: Row name, 8 characters
Column 25-36: First time period in the time series, an integer written with a decimal point
Column 37-48: Last time period in the time series, an integer written with a decimal point

The series itself follows the header card and is entered in (4F10.10) format.
1.7 ENDATA-card

Like the MODEL-file the SERIES-file must always end with a card with ENDATA punched in columns 1 to 6.

4.3 ERRORS

The code tests for many types of errors in the input and if something wrong is found an error message is printed. In the cases where it is possible to recover simply by deleting the data element with the error the errors are counted as minor errors, and it will be possible to continue after a SERIES-file when minor errors have been read. After encountering more serious errors, counted as major errors, it is generally not possible to continue. However, as with the MODEL-file there is an exception with major errors in RHS-, BOUND-, or RANGE-sets. Again the erroneous sets are deleted and the job can continue as long as the deleted sets are not used later on.

After reaching a SERIES-file with non-recoverable major errors the MODEL-file will still be ready so it is possible to start reading a new SERIES-file.

5. The Nonlinear Functions

All nonlinearities must be supplied by the user through a subroutine called FCOMP, that can compute the numerical values of the nonlinear expressions and possibly also their nonzero derivatives.

The first cards in the FCOMP subroutine should be:

SUBROUTINE FCOMP (X,G)
DIMENSION X(*), G(**)

where * and ** should be replaced by the largest index of a variable or of an expression, respectively, or simply by 1. A common block with the name CRUSEI can be included as the third line in FCOMP:
The first five variables in CRUSE1 are input variables to FCOMP and they should never be changed. The content of the variables is as follows:

- **NROW** - the largest index of a nonlinear function and the dimension of $G$.

- **NCOL** - the largest index of a nonlinear variable and the dimension of $X$.

- **NCALL** - the number of times FCOMP has been called including this time. If the problem is changed, e.g., because a new objective is defined, NCALL starts from 1 again. This variable can be used by the user to control input to or output from FCOMP. In the CDC-version input can be read from TAPE7 and output can be written to TAPE6 = OUTPUT.

- **IPROB** - a cell that the user can set in the control program. It can be used to distinguish different model versions that are all included in one FCOMP subroutine.

- **ITIME** - the number of the time period for which the nonlinear expressions should be computed.

- **IERR** - a cell through which the user can report to the optimization code that something is wrong. It is initially set to zero, and if the user sets it to a nonzero value the optimization will stop immediately.

It should be mentioned, that FCOMP can call other user supplied subroutines, and it can store data in labelled commonblocks. In order to avoid duplicate common block names the user should avoid names starting with C.
A row or expression is defined to be nonlinear if it contains at least one 99999.- or 999999.-entry in either the COLUMNS- or one of the LAGS-sections of the MODEL-file, and a column or variable is defined to be nonlinear if the MODEL-file contains at least one 99999.- or 999999.-entry in the column in the COLUMNS- or one of the LAGS-sections. All 99999.-entries in the COLUMNS- or LAGS-sections that are not replaced by actual time series during input of the SERIES-file should be treated in the FCOMP-subroutine and in the following definitions as if the entries had been 99999.-entries, i.e. the variable is assumed to enter nonlinearly in the expression.

The subroutine FCOMP is only concerned with the nonlinear parts of the expressions. The nonlinear part of the expression in a certain row is the part of the total expression that depends on all the variables that enters nonlinearly in the row. I.e. if \( x_i \) enters as \( 3 \cdot x_i + x_i^2 \) both \( 3 \cdot x_i \) and \( x_i^2 \) are part of the nonlinear expression, and if a derivative is supplied it must be coded as \( 3 + 2 \cdot x_i \). If \( x_i \) also enters lagged but linearly, the lagged \( x_i \)-part is not a part of the nonlinear expression.

The values of the current nonlinear variables are supplied to FCOMP through the vector \( X(*) \) that is the first argument of FCOMP. \( X(ICOL) \) contains the value of the variable (column) that was defined to have no ICOL in the bound set named VARIABLE. \( X \) will only be defined for values of \( ICOL \) mentioned in the VARIABLE bound set. If no VARIABLE bound set was supplied \( X(ICOL) \) will contain the value of the ICOL'\text{th} column in the COLUMNS-section of the MODEL-file.

Lagged values of the nonlinear variables are supplied to FCOMP through the systems function \( XLAG(ICOL,ILAG) \). \( ICOL \) is the column number as defined above and \( ILAG \) is the number of lags. \( ILAG \) must be positive.
The values of the nonlinear expressions are returned through the vector \( g(*) \) that is the second argument of \( \text{FCOMP} \). The nonlinear expressions are referenced through row numbers. Expression no. \( \text{IROW} \) corresponds to the row that was defined to have no. \( \text{IROW} \) in the right hand side named \( \text{FUNCTION} \). If no \( \text{FUNCTION} \) right hand side was supplied, expression \( \text{IROW} \) corresponds to the \( \text{IROW}' \)th row in the \( \text{ROWS} \)-section of the \( \text{MPS} \)-file.

It is possible also to supply partial derivatives of the nonlinear expressions to the optimization code. However, the description must wait for a later version of the manual.
Table 1: Differences between standard MPS-format and CONOPT-format

<table>
<thead>
<tr>
<th>Item</th>
<th>MPS (e.g. APEX or MPSX)</th>
<th>CONOPT-format</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Names</td>
<td>Leading blanks are ignored, embedded blanks are sometimes allowed.</td>
<td>The whole 8-character string is used as the name. Blanks are not recognized.</td>
</tr>
<tr>
<td>2. Linear combination rows</td>
<td>Sometimes accepted.</td>
<td>Not allowed.</td>
</tr>
<tr>
<td>4. Time series indicator</td>
<td>Not applicable.</td>
<td>9999.- or an alternative indicator is recognized.</td>
</tr>
<tr>
<td>5. Nonconstant coefficients</td>
<td>Not applicable.</td>
<td>99999.- and 999999.- or alternative indicators are recognized.</td>
</tr>
<tr>
<td>6. F JN in rhs</td>
<td>Not applicable.</td>
<td>Optional.</td>
</tr>
<tr>
<td>7. VARIABLE bnd</td>
<td>Not applicable.</td>
<td>Optional.</td>
</tr>
<tr>
<td>8. INITIAL bnd</td>
<td>Not applicable.</td>
<td>Optional. Can be located anywhere in the BOUNDS-section.</td>
</tr>
<tr>
<td>9. INITXXX bnd</td>
<td>Not applicable.</td>
<td>Optional.</td>
</tr>
<tr>
<td>10. RANGES</td>
<td>Range values can be negative. Ranges can be applied to equalities.</td>
<td>Range values must be positive. Ranges can only be applied to inequalities.</td>
</tr>
<tr>
<td>11. Coefficient values</td>
<td>If there is no decimal point one is assumed in the first blank position after the number. E-format is sometimes accepted.</td>
<td>Standard FORTRAN F12.6 format is used. I.e. if there is no decimal point one is assumed after the first 6 positions in the field. E-format with right justified exponent is accepted.</td>
</tr>
</tbody>
</table>
6. The Control Language

The control language used for CONOPT is very similar to the control language used in APEX-III, especially in the way the flow of control is managed. Therefore, users that are familiar with APEX can look at Table 2 and proceed to section 6.5.

6.1 General Format of Control Statements

All control cards are divided into 7 fields that are used as follows:

Field 1 in column 1 to 10 is either blank or contains a label for the control card. All labels must be unique.

Field 2 in column 11 to 20 is used for the verb that describes the instruction to be performed.

Field 3 to 7 in column 20 to 30, 31 to 40 etc. up to 61 to 70 are used for parameter 1 through parameter 5. If more parameters are needed they are written on the following card with both label and verb fields blank.

The position of the label, verb, and parameters within the 10-character fields is unimportant and blank parameter fields are ignored.

6.2 Flow of Control

Most input/output and computational instructions return on exit a positive integer parameter, LBRANCH, that gives a summary of the result of the operations. The values returned in LBRANCH are described with the different verbs in Section 6.5. Based on the value of LBRANCH it is possible to jump to different parts of the control program by inserting a BRANCH-verb immediately after the verb that created the LBRANCH value.

The parameters of the BRANCH verb are labels, and if the value of LBRANCH is i, control is transferred to the control statement with the
label in parameter i. After the jump the value of LBRANCH is reset to 1.
If not all parameter fields are defined by the user the last label will be
copted to the rest of the parameter fields such that all large LBRANCH values
will give the same jump. To jump to the next card it is possible to give the
next card a label and use this label as a BRANCH parameter. However, an
asterisk (*) in the parameter field of the BRANCH card is equivalent to this
construction.

The second important control verb is EXIT. It is simply used to
finish the execution of the control program.

The next verbs are useful for writing sub-routines and for loop
construction etc. The first-time user will hardly need them and he can skip
to Section 6.3.

PERFORM is a verb very similar to BRANCH. It transfers control to
one of the labels used as parameters on the PERFORM card depending on the
value of LBRANCH. The difference is the association with the verb NEXT.
When a PERFORM verb is executed and the label is not * the address of the
instruction following the PERFORM statement is placed in a stack. When the
verb NEXT, that has no parameters, is executed, control is transferred to
the instruction in the top of the stack and this instruction is deleted from
the stack. PERFORM acts as a call of a subroutine and NEXT is the correspond-
ing return jump. The maximum nesting of PERFORM verbs is 10.

Conditional branches based on the values of certain CR-cells (see
Section 6.4) instead of on the outcome of computational procedures can be
set up using the TEST-verb. The first parameter must be the name of a cell,
either integer or real, and the second parameter is either the name of a
cell or a numerical value. TEST sets the value of LBRANCH to 1 if the value
of the cell in parameter 1 is less than parameter 2, LBRANCH is set to 2 if
the two values are equal, and it is set to 3 if parameter 1 is greater than
parameter 2. Based on the value of LBRANCH the verbs BRANCH or PERFORM can
transfer control to different instructions.

6.3 Default Branches and Default Procedures

If the user does not enter a BRANCH-card immediately after the
instruction that created the LBRANCH value, the system will in most cases
insert a default BRANCH-card. The labels on the default BRANCH-cards are
described together with the LBRANCH values in Section 6.5.

The default BRANCH-cards will in many cases transfer control
away from the next instruction if a major or minor error was found or if
the optimization for some reason did not give the optimal solution. If
the user wants to continue independent of the LBRANCH value or with
other jumps than the default he must enter an explicit BRANCH verb, e.g.
with an asterisk for continue in some of the parameter fields.

The default branches can have the following labels: MAJERR,
MINERR, PROBOUT, and *. Corresponding to each of these labels there is
a default procedure that is inserted in the control program unless the
user defines the label by writing it on one of his own control cards. The
default procedures are currently as follows:

<table>
<thead>
<tr>
<th>Label</th>
<th>Verb</th>
<th>Parameter fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAJERR</td>
<td>MESSAGE</td>
<td>MAJOR ERROR - TERMINATION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXIT</td>
</tr>
<tr>
<td>MINERR</td>
<td>MESSAGE</td>
<td>MINOR ERROR - TERMINATION</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXIT</td>
</tr>
<tr>
<td>PROBOUT</td>
<td>MESSAGE</td>
<td>NON OPTIMAL EXIT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>OUTPUT</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SAVE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EXIT</td>
</tr>
</tbody>
</table>
6.4 Communication Region Cells and Associated Verbs

The detailed operations of the input procedures and the computational procedures can be determined through a number of communication region cells, CR-cells. The names and description of the most important ones can be found in table 2. Almost all cells have default values that are also shown in the table, and except for some names, e.g. KNSOBJ (objective function name) the optimization can work with only default values.

Some of the cells can be changed through two verbs, SET and STEP, that have two parameters. The first parameter is the name of a CR-cell, and the second parameter can either be the name of a cell or a numerical constant. The SET instruction simply replaces the content of the cell in parameter 1 with the value or the content of the cell in parameter 2. The STEP instruction increases the content of the cell in parameter 1 with the value or the content of the cell in parameter 2.

6.5 Input/Output and Computational Verbs

6.5.1 MODEL

The MODEL-verb is an input verb that asks the system to read the MODEL-file. The verb has one parameter, an alphanumerical string in parameter field 1. The string can be blank.

After the MODEL-verb is called the corresponding file is rewound and read until the proper MODEL card is found. If the parameter of the MODEL-verb is blank the first MODEL-card is selected from the file, and if a non-blank parameter was supplied the first MODEL card with this name in field 2 (column 15-24) is selected. After the MODEL-card is selected the ROWs, COLUMNS etc. are read until an ENDATA card or until end of file.

The LBRANCH-values returned by MODEL are:

1 - Successful completion.
2 - Major error were found.
3 - Minor errors were found.
Table 2. Communication Region Cells

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Reset</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KINDEL</td>
<td>A</td>
<td>No</td>
<td>-</td>
<td>The name on the MODEL-card of the selected MODEL-file.</td>
</tr>
<tr>
<td>KIINC</td>
<td>A</td>
<td>Yes</td>
<td>Blank</td>
<td>The name of the selected bound set.</td>
</tr>
<tr>
<td>KNSGLK</td>
<td>A</td>
<td>Yes</td>
<td>MINIMIZE</td>
<td>Direction of optimization (MIN, MINIMIZE, MAX, or MAXIMIZE).</td>
</tr>
<tr>
<td>KSDEL</td>
<td>A</td>
<td>No</td>
<td>-</td>
<td>The name on the SERIES-card of the selected SERIES-file.</td>
</tr>
<tr>
<td>KNINIT</td>
<td>A</td>
<td>Yes</td>
<td>INITIAL</td>
<td>The name of the selected initial bound set.</td>
</tr>
<tr>
<td>KNSKAP</td>
<td>A</td>
<td>Yes</td>
<td>Blank</td>
<td>The name of the selected objective function.</td>
</tr>
<tr>
<td>KNSRHS</td>
<td>A</td>
<td>Yes</td>
<td>Blank</td>
<td>The name of the selected right hand side.</td>
</tr>
<tr>
<td>KNUSER</td>
<td>A</td>
<td>Yes</td>
<td>Blank</td>
<td>The name of the selected range set.</td>
</tr>
<tr>
<td>LRENL1</td>
<td>I</td>
<td>Yes</td>
<td>10</td>
<td>The upper bound on the number of minor error messages from each input section.</td>
</tr>
<tr>
<td>LRENL2</td>
<td>I</td>
<td>Yes</td>
<td>4</td>
<td>The upper bound on the number of major error messages from each input section.</td>
</tr>
<tr>
<td>LFITER</td>
<td>I</td>
<td>Yes</td>
<td>100,000</td>
<td>The upper bound on the number of iterations in OPTIMIZE.</td>
</tr>
<tr>
<td>LFHPS</td>
<td>I</td>
<td>Yes</td>
<td>0</td>
<td>The upper bound on the number of input records from each input section of the MODEL-file that are reproduced on the output.</td>
</tr>
<tr>
<td>LFITSU</td>
<td>I</td>
<td>Yes</td>
<td>0</td>
<td>As LFHPS for the SERIES-file.</td>
</tr>
<tr>
<td>LFSTIM</td>
<td>I</td>
<td>Yes</td>
<td>1</td>
<td>The function made for the FCOMP subroutine, see Section 5.</td>
</tr>
<tr>
<td>LFTRCH</td>
<td>I</td>
<td>Yes</td>
<td>1</td>
<td>The value of the first period in the optimization. Lagged values are prior to this period.</td>
</tr>
<tr>
<td>LGTO</td>
<td>I</td>
<td>Yes</td>
<td>1</td>
<td>The value of the last period in the optimization.</td>
</tr>
<tr>
<td>LTJAB</td>
<td>I</td>
<td>Yes</td>
<td>0</td>
<td>Base period used in encoding the period-code on the FCRTRAN-readable output file.</td>
</tr>
<tr>
<td>NWJAC1</td>
<td>R</td>
<td>Yes</td>
<td>9999</td>
<td>Value to indicate time series elements in the MODEL-file.</td>
</tr>
<tr>
<td>NWJAC2</td>
<td>R</td>
<td>Yes</td>
<td>999999</td>
<td>Value to indicate nonlinear Jacobi elements only depending on lagged variables.</td>
</tr>
<tr>
<td>NWJAC3</td>
<td>R</td>
<td>Yes</td>
<td>999999</td>
<td>Value to indicate general non-linear Jacobi elements.</td>
</tr>
</tbody>
</table>

**Explanation to table 2:**

- **Type:** A - Alphanumeric, I - Integer, R - Real.
- **Reset:** Yes - the cell can be changed in a SET or STEP instruction.
  - No - the cell cannot be changed directly by the user.
and the default BRANCH-card that is inserted immediately after
the MODEL verb if the user does not supply his own BRANCH card is:

```
BRANCH MAJERR MINERR
```

6.5.2 SERIES

The SERIES-verb is very similar to the MODEL-verb, the only differ-
ence being that it reads the SERIES file. The file is again rewound before
reading and the verb has one parameter that has the same function as the
parameter of the MODEL-verb.

Notice that the two CR-cells LJFROM and LJTO should be set before
calling SERIES if the default values are not wanted. Also, if a new opti-
mization should be performed with the same model but with a different
planning period, SERIES must be called again after the change in LJFROM
and/or LJTO.

The LBRANCH values and the default BRANCH-card are the same as for
MODEL.

6.5.3 OPTIMIZE

The OPTIMIZE-verb takes care of the optimization including the
search for a first feasible solution and the proper setting up of the
problem. The verb has no parameters.

For the setup some of the CR-cells are needed. First of all
the names KNSOBJ, KNSRHS, KNSBND, KNSRNG, KNSINT, and KNSDIR. If no
objective name, KNSOBJ, is supplied only a feasible solution will be searched.
The default blank names for KNSRHS, KNSBND, and KNSRNG correspond to an all
zero right hand side, a bound set of greater than or equal to zero bounds on all variables, and a range set without any ranges. The default initial value bound set, KNSINT, is INITIAL, and the default optimization direction, KNSDIR, is MINIMIZE.

The number of iterations performed including both iterations in the search for a feasible solution and in the optimization is limited by the CR-cell LFITER with a default value of 100,000. LFITER can be reset in a SET or a STEP Instruction.

The LBRANCH values returned by OPTIMIZE are:

1. Optimal solution.
2. Major error.
3. No feasible solution can be found.
4. Iteration count greater than or equal to LFITER.
5. Convergence too slow.
6. No change in objective even though the reduced gradient is greater than the tolerance.
7. Time limit reached.
8. A feasible solution is found but no objective was supplied.
9. There is not sufficient core to continue. And the default BRANCH-card that is inserted unless the user writes his own is

   BRANCH * MAJERR PROBOUT

If the LBRANCH-value was 4, i.e. if the solution process was stopped prematurely, calling the OPTIMIZE verb again will continue the optimization exactly as if there had been no interrupt. Notice that the iteration count
continues so LFILTER should be reset before OPTIMIZE is called again.

Other CR-cells that have to do with output can also be reset between the
two OPTIMIZE verbs, and the OUTPUT verb (see 6.5.4.) can be called without
destroying the status of the solution. Calling OPTIMIZE again after one
of the other L3BRANCH values causes a major error.

If some of the problem-defining cells such as the name cells
KNSOBJ, KNSRHS etc. are changed then the problem will be set up again at
the next call to OPTIMIZE, and the solution process is started from the
beginning again. If the time horizon is changed through a resetting of
LJFROM or LJTO the user must call SERIES again before OPTIMIZE.

6.5.4 OUTPUT

The OUTPUT verb produces an MPS-like listing of the current solu-
tion. A detailed description is found in section 7.5. In addition, a spe-
cial FORTRAN-readable solution file in APEX-III compatible format for further
processing in report-generators (TABGEN) can be requested. The following
options are available:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT: FULL</td>
<td>full MPS-like output</td>
</tr>
<tr>
<td>FULL/SPEC</td>
<td>full MPS-like output and special file</td>
</tr>
<tr>
<td>SPECIAL</td>
<td>special file only</td>
</tr>
<tr>
<td>blank</td>
<td>full MPS-like output</td>
</tr>
</tbody>
</table>

Further details on the special FORTRAN-readable file are given in Appendix.

If output is called before a feasible solution has been found only
the feasible periods and the current period are printed.
The MESSAGE verb is used to print a string on the solution output. The content of the 5 parameter fields is simply printed as a 50 character string at the moment the MESSAGE verb is executed.

The purpose of the SAVE verb is to save the current solution at a file with the necessary information to make a restart. The actual procedure and corresponding restart procedure are not implemented yet.

The WRITE verb writes a part of the current solution on the local file WAT. WRITE can have one parameter with the three possible values MODEL, SERIES, and TSP. If no parameter is specified, SERIES is assumed. WRITE can only be called after an OPTIMIZE verb has been executed.

WRITE MODEL writes a modified MODEL file. Only the currently selected right hand side, bound set, initial bound set, and range set will be included in the MODEL file, and in the initial bound set all variables will be present as type FX with a 9999.- indication.

WRITE SERIES writes a modified SERIES file. Again, only the currently selected right hand side, bound set, initial bound set, and range set will be included, and in the initial bound set all variables will be represented by full time series. The LFRGM and LUTG cells should not be changed between the execution of OPTIMIZE and WRITE SERIES. The length of the time series on file WAT will correspond to the optimization period including the necessary lagged variables.
WRITE TSP produces a file with the current solution values that can be entered directly into TSP. (WRITE MODEL and WRITE TSP have not been implemented yet.)

The output from the WRITE or WRITE SERIES verb can be used for restarts. Notice, however, that all variables are present as time series in the initial bound set. If they are not referenced by 9999.- indicators in the MODEL file a minor error will result and the initial values will not be used. Also, right hand sides, bound sets, initial bound sets, and range sets that were not selected in the previous optimization will not be part of the new SERIES file, so if they are referenced by 9999.s in the MODEL file a minor error will result. The coming WRITE MODEL verb will create a MODEL file that can be used directly.

7. Ordinary Output

Apart from the FORTRAN compilers listing of the FCCMP subroutine the ordinary output from CONOPT can be divided into 5 groups, each of which are described in the following:

7.1 Control Program Listing

The first part of the output is a listing of the control program that is used to control the order of the computations and the setting of different cells. The control program contains both the control cards defined by the user and default branch cards after the computational procedures if a branch card was not supplied by the user. Also default action for minor and major errors is added if it is not defined by the user. All cards are assigned a number in the control program listing, and this number will be used in later references.
The control cards used with the example in Section 8 are shown in Fig. 3, and the corresponding control program listing is shown in Fig. 4.

7.2 Execution Information

The execution of the control program can be followed closely on the output. Each time a line is executed the number of the line, the verb of the line and the parameters are printed. In case of a BRANCH-verb the value of the branch parameter LBRACh is also printed.

7.3 Output from iMUEL and SERIES

The default output from the input verbs MODEL and SERIES is a listing of the section cards (MODEL, ROWS, COLUMNS, etc.) with the card number, error messages for any errors found within a default limit of 10 minor errors and 10 major errors for each input section, and a summary of the number of errors in each section. The limit on the number of error messages is stored in LFERR1 and LFEKR2 for minor and major errors respectively, and they can be changed by the SET-verb.

It is possible also to get a listing of the ordinary input cards. The user cell LFIPS with default value 0 gives an upper bound on the number of records listed from each section of the MODEL-file, and the user cell LFISU has the same purpose for the SERIES-file. Notice, the listing of the input is not an exact reproduction of the input cards but a listing of the data as they are interpreted by the input routines.

7.4 Output from OPTIMIZE

The output from OPTIMIZE will be explained through an example. The iteration output from OPTIMIZE corresponding to the example in section 8 is
shown in Fig. 5 and 6. Fig. 5 shows the first phase of the optimization where we search for a feasible solution. The search is done period by period, and the numbers in the output listing have the following meaning (the explanation is a little technical and the user without interest in the methodology can jump to Section 7.5):

**PERIOD:** The number of the period in which we search for a feasible solution. All earlier periods are already feasible, or they are not part of the optimization.

**PHASE:** The search uses three different algorithms with decreasing speed but with increasing reliability:

1: - We use a Newton-type algorithm within the current time period, keeping lagged variables fixed. The basic variables are if possible chosen away from the bounds, and if there is more freedom left, a nearly triangular basis is chosen.

2: - If Newton is not successful, which mainly will be the case if there are not enough variables away from bounds to choose as basic variables, a gradient and conjugate gradient search is tried, again keeping lagged variables fixed.

3: - If both the Newton and the gradient search fails, the lagged variables will also have to be changed. This is done in phase 3. We iterate in an optimization problem with the sum of squared infeasibilities in the first infeasible period as the objective and with the equations in the earlier feasible periods as constraints, essentially using a reduced gradient algorithm.
OPTIMAL STEPLENGTH: This is the steplength, either in the Newton-direction (phase 1) or in the gradient or reduced gradient direction (phase 2 or 3), that was found to give the smallest sum of squared infeasibilities. In the Newton algorithm an initial steplength of 1 is tried. In the gradient search the initial rate of change of the vector of infeasibilities is computed and based on this an initial steplength is estimated.

MAXIMAL STEPLENGTH: The largest steplength that would keep all variables within the bounds. In Newton the maximal steplength is never assumed to be larger than 1.

SUM OF SQUARED INFEASIBILITIES: The sum of the squared residuals or infeasibilities in the first infeasible period after the iteration has been finished. The lines without an iteration number simply print the initial value of the sum of squared residuals before any search is performed.

CALLS OF FCUMP: This number counts the total number of calls of the user-subroutine FCUMP during the iteration including the calls used to compute the nonlinear partial derivatives of the constraints.

CHANGE COLUMNS: This number is only printed in phase 2 and 3. In phase 2 it counts the number of variables that are not restricted by a bound in the initial point of the search, or in other words the number of variables between bounds during the search. In phase 3 it is the number of superbasics in the earlier feasible periods plus the number of unconstrained variables in the first infeasible period.

CGIND, GRAD: A logical variable that is true if a conjugate search is used. This is the case if the changing variables are the same as in the last iteration.
In Fig. 6 is shown the ordinary iteration output during the optimization, also for the example in Section 8. The output is divided into 6 categories for the inversion, the superbasis, the one dimensional search, the reduced gradient, the tolerances, and the objective function. The explanation of the numbers is:

**INVERSION:**

SCn: The number of basis changes from last iteration counted as the number of new nonbasic variables.

SPC: The number of spike columns used to represent the inverse basis matrix for all time periods.

SPE: The number of nonzero elements in the spike columns.

PnP: The number of pivots between periods, i.e. the number of times it was not possible to cross a basic variable for a row from the time period of the row.

LFC: The number of calls of FCCMP used to compute the nonlinear derivatives of the constraints.

**SUPERBASIS:**

sb: The number of superbasic variables used during the iteration.

+sb: The number of the variables entering the superbasis in the iteration if any. ** indicates that more than one variable were added to the superbasis during the iteration.

−sb: The number of the variables leaving the superbasis during the iteration if any. The number of the variable here and in + sb corresponds to the number used in the solution output.
UNEQUIDMENSIONAL SEARCH:

STP: The number of different steplengths that were tried in the one-dimensional search.

LFC: The number of calls of FCCHP used during the onedimensional search in the Newton subroutine to check for feasibility and to return to the feasible surface.

L-OPT: The final steplength in the search. Values in the order of magnitude of 1 indicates a well-scaled and well-behaved problem.

LFAL: A logical variable that is true (T) if the onedimensional search was actively constrained by a bound on one of the basic or superbasic variables.

LFK: A logical variable that is false (F) if something goes wrong in the onedimensional search, especially if there are problems with the convergence of the Newton algorithm.

REDUCED GRADIENT:

L-k-GRA: Gives the length (Euclidian norm) of the part of the reduced gradient that corresponds to the superbasic variables including any superbasics added during the iteration.

L-kUX: The length of the change vector for the super-basics, also including any superbasics added during the iteration.

CF/DL: The initial rate of change of the objective function along the search direction.

H-CUNL: A lower bound on the condition number of the estimated hessian of the reduced objective function computed as the square of the ratio of the largest to smallest diagonal element in the Cholesky factorization of the hessian estimate. Large numbers can indicate that the problem is badly scaled.
TOLERANCES:

UPTIL: In the Newton iterations, where we restore feasibility after some variables have been changed, the error in the objective due to errors in the constraints is computed as the scalar product of the residuals and the Lagrange multipliers. During the one-dimensional search this estimate of the error is required to be less than the number shown. In the line printed when the optimal solution is reached the number in this column is the actual estimate of the error.

LUFTX: The bound used in the Newton iterations on the sum of squared infeasibilities in one time period.

OBJECTIVE FUNCTION: The value of the objective at the end of the iteration.

7.5 Output from OUTPUT

For a sample of the output of the solution produced by the verb OUTPUT, see Fig. 7 and 8 that again refer to the example in Section 8.

The output is divided into two parts, one describing the information that has to go with the constraints, and one that describes the variables.

The first information printed on the top of each page is the status of the solution. There are five different possibilities:

OPTIMAL: The length of the superbasic part of the reduced gradient is less than the tolerance, or the number of superbasic variables is zero with no candidate to enter the superbasis.

INTERMEDIATE NONOPTIMAL: A feasible solution has been found and we are in the process of improving this solution.
FLIVL NONOPTIMAL: A feasible solution has been found. The iterations have
been stopped, either because the convergence is too slow or because
it was not possible to find a better solution even though the
reduced gradient was greater than the tolerance.

INFEASIBLE: No feasible solution can be found because a local minimum in
the sum of squared residuals was found. The output will only show
the initial feasible periods and the first period with
infeasibilities.

INTERMEDIATE INFEASIBLE: No feasible solution has been found yet, but the
search can continue. Again only the initial feasible periods and
the first period with infeasibilities is printed.

The second piece of information is the value of the objective
function. If the status of the solution shows an infeasible solution the
value printed is the sum of squared infeasibilities in the infeasible period,
and if a feasible solution has been found the value printed is the value of
the objective defined by the user.

The next line gives the names from the MODEL and SERIES cards and
the names of the selected objective function, right hand side, bound set,
range set, and the direction of the optimization.

The rest of the page in the CONSTRAINT section (Fig. 7) is very
similar to ordinary LP-output with an extra column indicating the time
period. A few comments are probably needed:

1. The constraints are defined as $5L_{it} \leq y_{i}(x_{i1}, \ldots) \leq 8U_{it}$ where
   the lower and upper right hand sides can be equal (type EQ),
   one can be - or + infinity (type LE or GE), both can have
absolute value infinity (type FR) or they can both be finite but different (type RG). The value of \( g_{it} \) is printed in the ACtIVITY column and the values of \( SL_{it} \) and \( BU_{it} \) in the RHS LOWER and RHS UPPER columns respectively.

2. The value in the SLACK column is the value \( u_{it} \) defined by
   \[ g_{i}(x_t \ldots) + u_{it} = B_{it}, \]
   where \( B_{it} \) is the right hand side value.

3. The value in the MARGINAL column shows the partial derivative of the optimal objective function value with respect to \( B_{it} \), keeping in mind that \( SL_{it} \) and/or \( BU_{it} \) moves with \( B_{it} \) if they are not infinity.

4. Notice that the value of the objective function in the feasible case is computed as minus the sum of the slacks in the objective rows. Therefore, the marginals of the free nonbinding objective rows are always one.

5. When a feasible solution has not yet been found the MARGINAL column shows the partial derivatives of the sum of squared infeasibilities. When the search is in phase 1 or 2, see section 7.4, only unlagged variables and equations are considered and the marginals are not defined (UNDEF) for lagged constraints.

The first lines in the VARIABLES section (see Fig. 8) are the same as in the LUSTRANS section. The rest of the page is again very similar to LP-output with the additional PERIOD column. The differences from LP are:

1. The NUMBER printed in the first column is the number of the variable used internally, and it is also the number that is used in the iteration log to describe variables entering and
leaving the superbasis. The number sometimes jumps between time periods. The missing numbers correspond to slack variables that are reported in the CONSTRAINTS section.

2. The STATUS of the variables can assume more value than in LP. Apart from LOWER, UPPER, and BASIC we have LAGGED variables, i.e. variables from before the optimization period that are needed in one or more constraints, and SUPER(-basic) variables, i.e. nonbasic variables not at a bound. In the search for a feasible solution there are two more values of the status: CONSTANT, that is used for lagged variables not changed in the phase 1 and 2 procedures for finding a feasible solution, and ACTIVE, that is used for variables changed in the last gradient search. A last status, UNDEF, is used for all variables just after a feasible solution has been found and before the optimization is started.

3. The OBJ. GRADIENT is simply the sum of the derivatives of all the objective rows lagged properly relative to each other. Variables from outside the optimization period have an undefined objective gradient (UNDEF). Before a feasible solution has been found the number printed is the derivative of the sum of squared infeasibilities in the infeasible period with respect to the variable of the line. If we are in phase 1 or 2 of the search for a feasible solution where lagged variables are kept constant (status CONSTANT) the objective gradient is undefined.
4. The RED. GRADIENT column gives the reduced gradient. It corresponds to the reduced costs in LP and it can be interpreted in the same way. The sign is chosen such that the reduced gradient value of a variable at a bound corresponds to the partial derivative of the optimal objective function value with respect to the variable, taking into account that the constraints should be satisfied, except possibly the bounds on this variable. The sign is independent of the optimization direction.

5. Before a feasible solution is found the RED. GRADIENT values are a little different. In the infeasible period the constraints are not used and the reduced gradient is equal to the objective gradient. In the earlier feasible periods the reduced gradient is either not defined, as in phase 1 and 2, or it is the reduced gradient of the sum of squared infeasibilities in the infeasible period.

7.6 Testoutput

Most of the subroutines in the system contain write statements for testoutput and the testoutput can be controlled to cover single subroutines or small groups of subroutines at different levels of detail.

Testoutput is controlled by communication cells with names starting with LO. The names and the documentation of these cells can be found in the internal documentation of the system in subroutine INITLZ. All output lines should be more or less self-explanatory as long as the naming conventions mentioned in INITLZ are remembered. The listing of the internal documentation can be made by a liable program that follows the system. But be careful. The documentation uses over 120 pages.
Through the use of iteration interrupts, resetting of cells and re-start of the optimization it is fairly easy to limit the testoutput to certain iterations.

b. An Example

The following model is a small model describing the pricing problem of the OPEC countries if they want to maximize discounted profits from their oil reserves. The model is taken from R.S. Pindyck, "Gain to Producers from the Cartelization of Exhaustible Resources" in Review of Economics and Statistics, vol. 60, p. 238-251. Only the equations of the model and the necessary data will be given. For further explanation, see the original paper.

Variables:

$T_d_t$ - the total demand for oil on the world market in year $t$ in billion barrels (bb).

$s_t$ - the supply from non-OPEC countries in year $t$ (bb).

$S_t$ - the cumulative supply from non-OPEC countries to year $t$ inclusive (bb).

$O_t$ - the amount of oil supplied by OPEC countries in year $t$ (bb).

$H_t$ - the reserves of oil in the OPEC countries in the beginning of year $t$ (bo).

$P_t$ - the price charged by OPEC countries in year $t$. This price is supposed to set the world market price. In 1975-dollars/barrel.

The above names without subscript $t$ will be used as variable or column names in the MODEL-file.
Data:
In 1974 we had the following values for the variables:
$T_u = 18.0$, $S = 5.5$, $CS = 0.0$, $D = 11.5$, and $R = 500.0$.

Equations:
The names written before the explanation of the equations are the names used as row or equation names in the MODEL-file.

TDEQ: Demand equation of the world marked:

\[ T_u = 0.17T_{u, t-1} - 0.13 P_t + 1.0 + 2.3 (1.015)^{t-1975} \]
\[ T_u = T_u + 0.87T_{u, t-1} - 0.13 P_t = - 1.0 - 2.3 (1.015)^{t-1975} \]

\[ \triangleq \quad \text{Equation for non-OPEC countries:} \]
\[ \Delta_s = 0.75 S_{t-1} + (0.1 P_t + 1.1) (1.02)^{-t/7} \]
\[ \Delta_s + 0.75 S_{t-1} + (0.1 P_t + 1.1) (1.02)^{-t/7} = 0 \]

\[ \triangleq \quad \text{Accounting identity for cumulative supply from non-OPEC countries:} \]
\[ CS_t = CS_{t-1} + S_t \]
\[ - CS_t + CS_{t-1} + S_t = 0 \]

\[ \Delta \quad \text{Market balance equation:} \]
\[ u_t = T D_t - S_t \]
\[ - u_t + T U_t - S_t = 0 \]

\[ \triangleq \quad \text{Accounting identity for the OPEC reserves:} \]
\[ \kappa_t = R_{t-1} - u_t \]
\[ - \kappa_t + R_t + R_{t-1} - D_t = 0 \]

\[ PROFIT: \quad \text{Objective function term for year} \ t: \]
\[ w_t (P_t - 250/K_t) (1.05)^{1974-t} \]
The MUDEL-input file is shown in Fig. 1. Notice the following:

a. The nonlinearities, \( P \) and \( CS \) in SEQ and \( D, P, \) and \( R \) in PROFIT, are all indicated by the 999999.-indicator. All other expressions are linear with time independent coefficients, and the coefficients are entered as in ordinary LP.

b. There are 4 lagged term. They are all declared in the LAGS-section where also the lagnumber of 1 is declared.

c. There are nonlinearities in the SEQ and PROFIT expressions. Therefore, they are declared in the FUNCTION right hand side as having index 2 and 4 in FCCMP, respectively.

d. TUEQ is the only equation with a nonzero right hand side. Since the right hand side is time dependent it is declared with the 9999.-indicator.

e. The variables \( CS, D, R, \) and \( P \) enter nonlinearly somewhere and they are declared in the VARIABLE bound set as having indices 1, 6, 4, and 2 in FCCMP, respectively. Notice that the names appear in the same order as in the column section.

f. The \( CS, k, \) and \( P \) variables enter in nonlinear functions that have singularities in zero. They are therefore given positive lower bounds to avoid numerical difficulties. All other variables will by default have a lower bound of zero.

g. In order to test the influence of initial values two different bound sets with initial values are declared. In the control program we can choose the particular set of initial values.
In the initial bound set some of the variables are declared to be initialized at the same value in all time periods. In many cases the user has some apriori knowledge of the order of magnitude of the model variables and passing this information can increase the speed and numerical stability significantly.

The SEIKES input file is shown in Fig. 2. Notice the following:

a. A very long time series has been entered in the RHS-section. In the control program we will declare the actual optimization period and the relevant part of the series will be extracted. The advantage of the long series is that it can be used for many runs with different time horizon.

b. In the INITIAL bound set the time series are entered with only one element, the lagged initial value. This is the minimum amount of information.

c. In the INITIAL bound set the time series for CS is simply made up to fit the constant initial S-values. The series for the reserve R is chosen as a geometric series with the quotient known from 1974 to 75. These initial values make economic sense, and they can increase the speed significantly.

d. Notice that the time series are supplied in the same order as the 9999's in the MODEL-file.

The simplest set of control cards that can run the example is shown in Fig. 3, and the total control program with all default branches and default labels interested is shown in Fig. 4. The iteration output from the example is shown in Fig. 5 and 6, and the print of the optimal solution is shown in Fig. 7 and 8. Fig 3 to 8 have all been discussed in Section 7.
Table 3. Summary of CONOPT - Verbs

<table>
<thead>
<tr>
<th>Verb</th>
<th>Parameters */</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRANCH</td>
<td>$l_1$ $l_2$ ... $l_n$</td>
<td>jumps to $l_i$ _BRANCH and sets $l_{\text{BRANCH}} = 1$</td>
</tr>
<tr>
<td>CRPRINT</td>
<td>$c_1$ $c_2$ ... $c_5$</td>
<td>displays current value of one or more CR cells</td>
</tr>
<tr>
<td>EXIT</td>
<td>none</td>
<td>terminates CONOPT</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>&lt;text&gt;</td>
<td>prints &lt;text&gt;</td>
</tr>
<tr>
<td>NEXT</td>
<td>none</td>
<td>returns to verb following the latest PERFORM call</td>
</tr>
<tr>
<td>PERFORM</td>
<td>$l_1$ $l_2$ ... $l_n$</td>
<td>return-jump to $l_i$ _BRANCH and sets $l_{\text{BRANCH}} = 1$</td>
</tr>
<tr>
<td>SET</td>
<td>$c_1$ $p_2$</td>
<td>sets $c_1$ to $p_2$</td>
</tr>
<tr>
<td>STEP</td>
<td>$c_1$ $p_2$</td>
<td>increments: $c_1 = c_1 + p_2$</td>
</tr>
<tr>
<td>TEST</td>
<td>$p_1$ $p_2$</td>
<td>sets $l_{\text{BRANCH}} = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2$ $p_1 = p_2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$3$ $p_1 &gt; p_2$</td>
</tr>
<tr>
<td>TIME</td>
<td>none</td>
<td>prints current resource use</td>
</tr>
<tr>
<td>MODEL</td>
<td>blank/name</td>
<td>reads first/name model and sets $l_{\text{BRANCH}} = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2$ major error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$3$ minor error</td>
</tr>
<tr>
<td>OPTIMIZE</td>
<td>none</td>
<td>sets $l_{\text{BRANCH}} = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2$ optimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$3$ major error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$4$ infeasible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$5$ convergence too slow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$6$ no change in objective</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$7$ time limit</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$8$ feasible solution only</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$9$ insufficient core</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>blank/FULL</td>
<td>generates NPS-like output and/or FORTRAN-readable file</td>
</tr>
<tr>
<td></td>
<td>FULL/SPEC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPECIAL</td>
<td></td>
</tr>
<tr>
<td>SERIES</td>
<td>blank/name</td>
<td>reads first/name series and sets $l_{\text{BRANCH}} = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$2$ major error</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$3$ minor error</td>
</tr>
<tr>
<td>WRITE</td>
<td>blank/SERIES</td>
<td>generates new series or model file</td>
</tr>
<tr>
<td></td>
<td>MODEL/TSP</td>
<td></td>
</tr>
</tbody>
</table>

* $c_1$ = CR cell.

$p_1$ = CR cell or constant.

$l_i$ = label.
Table 4. CONOPT-Verbs and their Default BRANCH Labels

<table>
<thead>
<tr>
<th>Verb</th>
<th>Default Labels</th>
<th>LBRANCH Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>BRANCH</td>
<td>not applicable</td>
<td>1</td>
</tr>
<tr>
<td>CKPRINT</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>EXIT</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>MESSAGE</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>MODEL</td>
<td>* MAJERR MINERR</td>
<td>yes</td>
</tr>
<tr>
<td>NET</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>OPTIMIZE</td>
<td>* MAJERR PROBOUT</td>
<td>yes</td>
</tr>
<tr>
<td>PERFORM</td>
<td>not applicable</td>
<td>1</td>
</tr>
<tr>
<td>SERIES</td>
<td>* MAJERR MINERR</td>
<td>yes</td>
</tr>
<tr>
<td>SET</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>STEP</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>TEST</td>
<td>*</td>
<td>yes</td>
</tr>
<tr>
<td>TIME</td>
<td>not applicable</td>
<td>no</td>
</tr>
<tr>
<td>WRITE</td>
<td>* MAJERR MINERR</td>
<td>yes</td>
</tr>
</tbody>
</table>
Figures

Figure 1: Listing of Model File
Figure 1a: FCOMP Subroutine
Figure 2: Listing of Series File
Figure 3: Control Program
Figure 4: Expanded Control Program
Figure 5: Iteration Log - Search for a Feasible Solution
Figure 6: Iteration Log - Search for Optimal Solution
Figure 7: Output of Constraints Section
Figure 8: Output of Variables Section
Figure 1: Listing of Model File

MODEL OPEC
ROWS
E TDEG
N PROFIT
E SEQ
E CSEQ
E DEQ
E REQ
COLUMNS
TD TOEQ -1.
S SEQ -1.
S CSEQ +1.0000 SEQ -1.0
C9 SEQ 999999.0 CSEQ -1.
D DEQ -1.
D REQ -1.0
R PROFIT 999999.
R REG -1.0
P PROFIT 999999.
P SEQ 999999.
P TDEQ 0.13
LAGS
TD TDEQ 0.87
S SEQ 0.75
C9 CSEQ 1.
R REG 1.0
RHS
FUNCTION PROFIT 8.
FUNCTION SEQ 2.
RIGHT TDEQ 9999.
BOUNDS
LO BND C9 0.01
LO BND R 0.01
LO BND P 0.01
FX INITIAL TD 9999.
FX INITIAL S 9999.
FX INITIAL C9 9999.
FX INITIAL D 9999.
FX INITIAL R 9999.
VARIABLE C9 1.
VARIABLE D 6.
VARIABLE R 4.
VARIABLE P 2.
FX INITA TD 18.0
FX INITA S 8.5
FX INITA C9 9999.0
FX INITA D 11.5
FX INITA R 9999.5
FX INITA P 14.0
ENODATA
EXAMPLE NO 1 - THE MONOPOLY CASE OF THE OPEC PROBLEM.

SUBROUTINE FCOMP(X,G)
REAL G(4),X(6)
COMMON/CUSE1/NGO,NCOL,NCALL,IPROB,ITIME,IERN

THE NONLINEAR PART OF THE SUPPLY FUNCTION FOR THE NON-OPEC COUNTRIES.

G(2)=(1,1.0,0.1*X(2)*1.029*(-X(1)/7,)

THE OBJECTIVE FUNCTION.

G(4)=1.05*(1.974+ITIME)*X(2)-250./X(4)*X(6)
RETURN
END

Figure 1a: FCOMP Subroutine
Figure 2: Control Program

MODEL
SET L: FROM 1975
SET LJU 1979
SERIES
SET K: NHLHS RIGHT
SET KUSBAND END
SET KUSORJ PROFIT
SET KUSDJ MAXIMIZE

OPTIMIZE
OUTPUT
EXIT
**Figure 4: Expanded Control Program**

```
<table>
<thead>
<tr>
<th>NUMBR</th>
<th>LABEL</th>
<th>VERB</th>
<th>PARAM.</th>
<th>PARAM2</th>
<th>PARAM3</th>
<th>PARAM4</th>
<th>PARAM5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>MODEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>BRANCH</td>
<td></td>
<td>MAJERR</td>
<td>WNEAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>SET</td>
<td>LJFRO</td>
<td>1975</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>SET</td>
<td>LJTO</td>
<td>1979</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>SERIES</td>
<td></td>
<td>MAJERR</td>
<td>WNEAR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>BRANCH</td>
<td></td>
<td>KNSRHG</td>
<td>RIGHT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>SET</td>
<td>KNSRNR</td>
<td>AND</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>SET</td>
<td>KNSOBJ</td>
<td>PROFIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>SET</td>
<td>KNSDIR</td>
<td>MAXIMIZE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>OPTIMIZE</td>
<td></td>
<td>MAJERR</td>
<td>PROBOUT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>BRANCH</td>
<td></td>
<td>OUTPUT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>OUTPUT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>MAJERR</td>
<td></td>
<td>MESSAGE</td>
<td>MAJOR ERROR</td>
<td>TERMINATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>WNEAR</td>
<td></td>
<td>MESSAGE</td>
<td>WNEAR ERROR</td>
<td>TERMINATE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>PROBOUT</td>
<td></td>
<td>MESSAGE</td>
<td>NON OPTIMAL EXIT</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>SAVE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>EXIT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

0 ERROR(S) IN CONTROL PROGRAM (SIZE = 120),
```
**Figure 5: Iteration Log – Search for a Feasible Solution**

Search for a Feasible Solution

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Period</th>
<th>Phase</th>
<th>Optimal Steplength</th>
<th>Maximal Steplength</th>
<th>Sum of Squared Infeasibilities</th>
<th>Calls of Fcomp</th>
<th>Change in Columns</th>
<th>Grad.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1975</td>
<td>1</td>
<td>0.0000E+00</td>
<td>0.0000E+00</td>
<td>2.503851447E+05</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1975</td>
<td>2</td>
<td>7.9216E-01</td>
<td>1.1863E+00</td>
<td>4.80724733E+04</td>
<td>1</td>
<td>7</td>
<td>f</td>
</tr>
<tr>
<td>3</td>
<td>1975</td>
<td>1</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td>2.886415643E+08</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1976</td>
<td>1</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td>2.01701975E+02</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1977</td>
<td>1</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td>1.27614866E+00</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1978</td>
<td>1</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td>1.30157808E+03</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1979</td>
<td>1</td>
<td>1.0000E+00</td>
<td>1.0000E+00</td>
<td>8.527751557E+01</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Feasible Solution, Value of Objective = 1.384606444E+01**
Figure 6: Iteration Log - Search for Optimal Solution

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Inversion</th>
<th>Superbasis</th>
<th>NDP Dimensional Search</th>
<th>Reduced Gradient</th>
<th>Tolerances</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
<td>5</td>
<td>15</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>8</td>
<td>1.12E+00</td>
<td>T</td>
<td>2.85E+00</td>
<td>1.4E+00</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>20</td>
<td>5</td>
<td>1.09E+00</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>20</td>
<td>5</td>
<td>1.0E+00</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>20</td>
<td>5</td>
<td>1.0E+00</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>20</td>
<td>5</td>
<td>1.0E+00</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>20</td>
<td>5</td>
<td>1.0E+00</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>5</td>
<td>15</td>
<td>20</td>
<td>5</td>
<td>1.0E+00</td>
</tr>
</tbody>
</table>

** Optimal Solution: The length of the reduced gradient is less than the tolerance.

*** Line 12 Branch with Branch # 1

*** Line 13 Output
**Figure 7: Output of Constraints Section**

**SOLUTION STATUS:** OPTIMAL

<table>
<thead>
<tr>
<th>NUMBER</th>
<th>NAME</th>
<th>PRTIND</th>
<th>TYP</th>
<th>STATUS</th>
<th>ROW ACTIVITY</th>
<th>SLACK</th>
<th>RHS LOWER</th>
<th>RHS UPPLN</th>
<th>MARINAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INFO</td>
<td>1975</td>
<td>F</td>
<td>BNDNG</td>
<td>-3,30000</td>
<td></td>
<td>-3,30000</td>
<td>-3,30000</td>
<td>67,20544</td>
</tr>
<tr>
<td>2</td>
<td>PROFIT</td>
<td>1975</td>
<td>F</td>
<td>SLACK</td>
<td>120,57020</td>
<td>120,57020</td>
<td>INF</td>
<td>INF</td>
<td>-1,00000</td>
</tr>
<tr>
<td>3</td>
<td>SIA</td>
<td>1975</td>
<td>F</td>
<td>BNDNG</td>
<td>45,42026</td>
<td>-95,42026</td>
<td>INF</td>
<td>INF</td>
<td>34,03000</td>
</tr>
<tr>
<td>4</td>
<td>CSIA</td>
<td>1975</td>
<td>F</td>
<td>BNDNG</td>
<td>45,42026</td>
<td>-95,42026</td>
<td>INF</td>
<td>INF</td>
<td>-92575</td>
</tr>
<tr>
<td>5</td>
<td>OBS</td>
<td>1975</td>
<td>F</td>
<td>BNDNG</td>
<td>45,42026</td>
<td>-95,42026</td>
<td>INF</td>
<td>INF</td>
<td>12,01071</td>
</tr>
<tr>
<td>6</td>
<td>AN</td>
<td>1975</td>
<td>F</td>
<td>BNDNG</td>
<td>45,42026</td>
<td>-95,42026</td>
<td>INF</td>
<td>INF</td>
<td>0,03807</td>
</tr>
<tr>
<td>7</td>
<td>TIN</td>
<td>1976</td>
<td>F</td>
<td>BNDNG</td>
<td>-1,13450</td>
<td>1,13450</td>
<td>1,13450</td>
<td>1,13450</td>
<td>30,45371</td>
</tr>
<tr>
<td>8</td>
<td>PHFIT</td>
<td>1976</td>
<td>F</td>
<td>SLACK</td>
<td>45,92026</td>
<td>-95,42026</td>
<td>INF</td>
<td>INF</td>
<td>1,00000</td>
</tr>
<tr>
<td>9</td>
<td>SIA</td>
<td>1976</td>
<td>F</td>
<td>BNDNG</td>
<td>45,92026</td>
<td></td>
<td></td>
<td></td>
<td>31,40518</td>
</tr>
<tr>
<td>10</td>
<td>CSFA</td>
<td>1976</td>
<td>F</td>
<td>BNDNG</td>
<td>45,92026</td>
<td></td>
<td></td>
<td></td>
<td>-0,0179</td>
</tr>
<tr>
<td>11</td>
<td>ONS</td>
<td>1976</td>
<td>F</td>
<td>BNDNG</td>
<td>45,92026</td>
<td></td>
<td></td>
<td></td>
<td>10,00214</td>
</tr>
<tr>
<td>12</td>
<td>RRA</td>
<td>1976</td>
<td>F</td>
<td>BNDNG</td>
<td>45,92026</td>
<td></td>
<td></td>
<td></td>
<td>-0,2804</td>
</tr>
<tr>
<td>13</td>
<td>INFO</td>
<td>1977</td>
<td>F</td>
<td>BNDNG</td>
<td>-1,36952</td>
<td>1,36952</td>
<td>1,36952</td>
<td>1,36952</td>
<td>34,43853</td>
</tr>
<tr>
<td>14</td>
<td>PROFIT</td>
<td>1977</td>
<td>F</td>
<td>SLACK</td>
<td>68,58376</td>
<td>68,58376</td>
<td>INF</td>
<td>INF</td>
<td>-1,00000</td>
</tr>
<tr>
<td>15</td>
<td>SIA</td>
<td>1977</td>
<td>F</td>
<td>BNDNG</td>
<td>68,58376</td>
<td></td>
<td></td>
<td></td>
<td>23,00929</td>
</tr>
<tr>
<td>16</td>
<td>CSIA</td>
<td>1977</td>
<td>F</td>
<td>BNDNG</td>
<td>68,58376</td>
<td></td>
<td></td>
<td></td>
<td>-0,04626</td>
</tr>
<tr>
<td>17</td>
<td>OBS</td>
<td>1977</td>
<td>F</td>
<td>BNDNG</td>
<td>68,58376</td>
<td></td>
<td></td>
<td></td>
<td>-10,04223</td>
</tr>
<tr>
<td>18</td>
<td>AN</td>
<td>1977</td>
<td>F</td>
<td>BNDNG</td>
<td>68,58376</td>
<td></td>
<td></td>
<td></td>
<td>-0,01915</td>
</tr>
<tr>
<td>19</td>
<td>TFO</td>
<td>1978</td>
<td>F</td>
<td>BNDNG</td>
<td>-3,00506</td>
<td>3,00506</td>
<td>3,00506</td>
<td>3,00506</td>
<td>37,5027</td>
</tr>
<tr>
<td>20</td>
<td>PROFIT</td>
<td>1978</td>
<td>F</td>
<td>SLACK</td>
<td>68,05264</td>
<td>68,05264</td>
<td>INF</td>
<td>INF</td>
<td>1,00000</td>
</tr>
<tr>
<td>21</td>
<td>SIA</td>
<td>1978</td>
<td>F</td>
<td>BNDNG</td>
<td>68,05264</td>
<td></td>
<td></td>
<td></td>
<td>24,86524</td>
</tr>
<tr>
<td>22</td>
<td>CSIA</td>
<td>1978</td>
<td>F</td>
<td>BNDNG</td>
<td>68,05264</td>
<td></td>
<td></td>
<td></td>
<td>-3,31450</td>
</tr>
<tr>
<td>23</td>
<td>ONS</td>
<td>1978</td>
<td>F</td>
<td>BNDNG</td>
<td>68,05264</td>
<td></td>
<td></td>
<td></td>
<td>12,51901</td>
</tr>
<tr>
<td>24</td>
<td>RRA</td>
<td>1978</td>
<td>F</td>
<td>BNDNG</td>
<td>68,05264</td>
<td></td>
<td></td>
<td></td>
<td>-0,0112</td>
</tr>
<tr>
<td>25</td>
<td>INFO</td>
<td>1979</td>
<td>F</td>
<td>BNDNG</td>
<td>3,44114</td>
<td>-3,44114</td>
<td>3,44114</td>
<td>3,44114</td>
<td>17,04844</td>
</tr>
<tr>
<td>26</td>
<td>PROFIT</td>
<td>1979</td>
<td>F</td>
<td>SLACK</td>
<td>81,00024</td>
<td>81,00024</td>
<td>INF</td>
<td>INF</td>
<td>-1,00000</td>
</tr>
<tr>
<td>27</td>
<td>SIA</td>
<td>1979</td>
<td>F</td>
<td>BNDNG</td>
<td>81,00024</td>
<td></td>
<td></td>
<td></td>
<td>10,86042</td>
</tr>
<tr>
<td>28</td>
<td>CSIA</td>
<td>1979</td>
<td>F</td>
<td>BNDNG</td>
<td>81,00024</td>
<td></td>
<td></td>
<td></td>
<td>-1,14202</td>
</tr>
<tr>
<td>29</td>
<td>ONS</td>
<td>1979</td>
<td>F</td>
<td>BNDNG</td>
<td>81,00024</td>
<td></td>
<td></td>
<td></td>
<td>-17,04844</td>
</tr>
<tr>
<td>30</td>
<td>RRA</td>
<td>1979</td>
<td>F</td>
<td>BNDNG</td>
<td>81,00024</td>
<td></td>
<td></td>
<td></td>
<td>-0,00439</td>
</tr>
</tbody>
</table>
APPENDIX

Special Output File

The use of the FULLSPEC as SPECIAL option with the OUTPUT verb causes the current solution to be written onto a FORTRAN-readable file. The default file name TAPE12 can be changed via the SOL=fn parameter on the CONOPT call card. If more than one solution is written to the same file, each solution is separated by an end-of-file mark; no file positioning takes place either before or after writing.

The CONOPT variable and equation names have to be augmented by a period-code in order to be compatible with standard LP naming conventions. CONOPT uses character positions 9 and 10 (counted from the left) to encode a 2-character base 32 period-code. The characters and their order are A,B,...,Z,0,1,...,5. The following encoding scheme is used by default:

<table>
<thead>
<tr>
<th>period</th>
<th>code</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>AA</td>
</tr>
<tr>
<td>1</td>
<td>AB</td>
</tr>
<tr>
<td>31</td>
<td>A5</td>
</tr>
<tr>
<td>32</td>
<td>3A</td>
</tr>
<tr>
<td>-0</td>
<td>55</td>
</tr>
<tr>
<td>-1</td>
<td>54</td>
</tr>
</tbody>
</table>

The code for the base period can be changed with the LJTAB cell (default zero). For example, setting LJTAB to 1975 will associate the code AA with 1975, AB with 1976 and so on.

In any event, CONOPT prints a message to indicate the exact encoding scheme used in generating the special file.
The format of this file follows the APEX-III specifications, see Appendix B of the APEX-III reference manual. The code for the basis status had to be extended to account for super-basic status. Bits 11-10 of the special word (word 7) includes the following types:

- 00: nonbasic
- 01: nonbasic at upper bound
- 10: basic
- 11: super basic

Example:

The following example illustrates the use of the special file in connection with TABGEN. The control cards are:

```
.
.
.
.
OPT.
REWIND,TAPE12.
TABGEN.
7/8/9
<FCOMP subroutine>
7/8/9
<CONOPT control program>
7/8/9
<Matrix>
7/8/9
<Series>
7/8/9
<TABGEN input>
6/7/8/9
```
The expanded control program was:

```
NUMBER LABEL  VERB  PARAM1  PARAM2  PARAM3  PARAM4  PARAM5
1     MODEL
2     BRANCH  MAJERR  MINERR
3     SET      LJFROM   1975
4     SET      LJTN     1979
5     SERIES
6     BRANCH  MAJERR  MINERR
7     SET      KNSRHS  RIGHT
8     SET      KNSRHS  AND
9     SET      KNSORJ  PROFIT
10    SET      KNSORJ  MAXIMIZE
11    SET      LJTRM   1975
12    OPTIMIZE
13    BRANCH  MAJERR  PROBOUT
14    OUTPUT  FULL/SPEC
15    EXIT
16    MAJERR  MESSAGE  MAJOR ERROR = TERMINATE
17    EXIT
18    MINERR  MESSAGE  MINOR ERROR = TERMINATE
19    EXIT
20    PROBOUT MESSAGE  NON OPTIMAL EXIT
21    OUTPUT
22    SAVE
23    EXIT
```

0 ERROR(S) IN CONTROL PROGRAM (SIZE = 132),
The control file to TABGEN was:

```
1 TITLE EASY UPEC PROBLEM
2 SETS
3 92 Y YEAR
4 AA YEAR 1975
5 BB YEAR 1976
6 CC YEAR 1977
7 DD YEAR 1978
8 EE YEAR 1979
9 18 V VARIABLES
10 P PRICE
11 N D-OPEC
12 TD D-TOTAL
13 S S-NON-OPEC
14 CS SUM-S
15 Q RESERVES
16 TABLES
17 CA PRIMAL SOLUTION VALUES
18 3 Y V
19 END
```

The TABGEN output was:

```
TABLE 1., PRIMAL SOLUTION VALUES

<table>
<thead>
<tr>
<th>YEAR</th>
<th>PRICE</th>
<th>D-OPEC</th>
<th>D-TOTAL</th>
<th>S-NON-OPEC</th>
<th>SUM-S</th>
<th>RESERVES</th>
</tr>
</thead>
<tbody>
<tr>
<td>1975</td>
<td>13,151</td>
<td>10,007</td>
<td>17,249</td>
<td>7,242</td>
<td>7,242</td>
<td>489,993</td>
</tr>
<tr>
<td>1976</td>
<td>12,119</td>
<td>9,118</td>
<td>16,766</td>
<td>7,648</td>
<td>14,890</td>
<td>480,875</td>
</tr>
<tr>
<td>1977</td>
<td>12,871</td>
<td>8,309</td>
<td>16,282</td>
<td>7,974</td>
<td>22,964</td>
<td>472,567</td>
</tr>
<tr>
<td>1978</td>
<td>15,769</td>
<td>7,091</td>
<td>15,521</td>
<td>8,430</td>
<td>31,294</td>
<td>465,476</td>
</tr>
<tr>
<td>1979</td>
<td>22,347</td>
<td>4,752</td>
<td>14,044</td>
<td>9,292</td>
<td>40,586</td>
<td>460,724</td>
</tr>
</tbody>
</table>
```