stichting
mathematisch
centrum

MC

DEPARTMENT OF MATHEMATICAL STATISTICS SW 35/76 JUNE

M. VAN GELDEREN

A USER'S PROGRAM FOR MULTIPLE LINEAR REGRESSION ANALYSIS

First printing April 1975
Second edition June 1976

2e boerhaavestraat 49 amsterdam

CONTENTS

1. INTRODUCTION

Many computer programs exist which in one way or another calculate estimates for the parameters of a linear regression model, but most of those programs can hardly be used by the layman. Sometimes even knowledge of the specific machine is required, although higher programming languages solve many of the machine dependent problems. As a layman, however, one is not always acquainted with these languages. Further, the programs usually require the data to be presented in a standard form, which in practice means that the data must be transformed into that one very special form or that the data should be punched in exactly that form. More difficult still, and often more confusing, is the way in which the program is told what model one wants to use. For instance, for the program for multiple polynomial regression of the Mathematical Centre a code matrix of zeros and ones must be punched to indicate the form of the regression polynomial. Transformations of one or more variables are hardly ever automatically possible and require a separate program to be run in advance to prepare the new data matrix.

Fortunately, it has been recognized recently that standard programs should not bother the user too much with awkward input specifications. In our opinion a statistician is more interested in the results of a program and in how he can obtain them without interference of software specialists of the computing centre, than in a few seconds gain in actual computing time. This is one of the main reasons for the development of a new program, in the course of which we had in mind from the beginning that the convenience should be as great as possible, with an input specification that is as appealing and as simple as possible. The system thus accepts as modeldescription a formula which resembles the models given in common statistical literature quite closely, while an accompanying inputdescription indicates which (series of) numbers of the data file go with which variable. This gives the user the opportunity to work with existing data and to process possible transformations without having to read in the adjusted data each time.

Still one should always realize well what one is doing: the modeldescription must be given explicitly and completely and the datastructure must be known exactly before an inputdescription can be given. The identification of observations and specific variables of the model is more "verbal" now and does not depend anymore on codenumbers or on a particular place of these observations between other (code)numbers. In the output the results are being identified with names given by the user in the model. The program enables the user to prepare the data and to specify the operations completely even if he knows no programming language at all. In order to get the computer to execute the program the (machine dependent) indications of section 2-8 have to be followed.

The original ideas for this inputarrangement came from A.P.B.M. VEHMEYER, while the article "ALGOL 60 translation for everybody" by F.E.J. KRUSEMAN ARETZ [6] supplied the basis for the translator and the execution section of the program. Also the basic ideas for the structure of the mass storage section are from the latter. A similar input system for linear programming problems by JAC. M. ANTHONISSE also supplied many starting points.

CHAPTER 1

MULTIPLE LINEAR REGRESSION ANALYSIS

1.1. THE MODEL

In a regression problem the experimentor searches for a relationship between a random variable $\underline{y}$ (the realizations of which are subject to some form of disturbance) on the one side and a number of variables $x_1,\ldots,x_p$ (which are without or at least almost without disturbances) on the other side. This relationship is expressed by a mathematical formula, which is called the model, for instance:

$$(1) \qquad \underline{y} = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \underline{e} \ .$$

The variables $x_1,\ldots,x_p$ and the variable $\underline{y}$ can also represent (other) transformed variables. The experimentor might have reasons to believe (from background information concerning the experiment) that transformations are necessary, for instance:

1) to obtain normally distributed disturbances,
2) to get a greater homogeneity of the variance of the disturbances,
3) to linearize non-linear models (if possible).

The transformed model can be written as:

$$(2) \qquad \underline{g} = g(\underline{y}) = \beta_0 + \sum_{i=1}^{p} \beta_i f_i(x_1,\ldots,x_n) + \underline{e}$$

in which g, $f_1,\ldots,f_p$ represent the transformations,
$\beta_0,\ldots,\beta_p$ represent the parameters to be estimated,
$\underline{y}$ represents the dependent variable,
$x_1,\ldots,x_n$ represent the independent variables,
$\underline{e}$ represents the disturbance.

The choice of a transformation by means of "trial and error" is rather time consuming and costly. The importance of the location parameter makes for the difficulty. It is not unusual that $\log(x)$ yields no improvement, but that $\log(c+x)$ gives better results for a particular choice of $c \neq 0$. Because this holds for almost any transformation of some importance, we must actually solve in each case a nonlinear adjustment problem. Often though, a simple form of the transformation is suggested by the experimentor who is better acquainted with the peculiarities of the experiment.

1.2. LEAST SQUARES

Regression analysis consists in fact of the adjustment of a hyperplane of the required dimension to the data. The fitting is done with the method of least squares, which means that the sum of the squares of the differences between the observed values for $\underline{y}$ and the estimated values for the expectation of $\underline{y}$, are being minimized. This sum of squares is also called the residual sum of squares.

In matrix notation the model can be written as

$$\underline{Y} = X\beta + \underline{e},$$

in which $\underline{Y}$ is a $(n\times1)$ random vector of observations,
X is a $(n\times p)$ matrix of known (fixed) values,
β is a $(p\times1)$ vector of (unknown) parameters,
and $\underline{e}$ is a $(n\times1)$ random vector of disturbances.

It is supposed that $E(\underline{e}) = 0$ and $\text{var}(\underline{e}) = I\sigma^2$, in which I is the unit matrix; thus:

$$E(\underline{Y}) = X\beta.$$

Let Y be a realization of $\underline{Y}$. The sum of squares of deviations then equals

$$(1) \qquad (Y-X\beta)'(Y-X\beta) = Y'Y - \beta'X'Y - Y'X\beta + \beta'X'X\beta$$
$$= Y'Y - 2\beta'X'Y + \beta'X'X\beta$$

(for $\beta'X'Y$ is a scalar and therefore equal to $Y'X\beta$).

The least squares estimation of β is the value b which, when substituted in (1), will minimize the sum of squares and is therefore given by the solution of the following system:

$$\left[\frac{\partial(Y-X\beta)'(Y-X\beta)}{\partial\beta}\right]_{\beta=b} = 0.$$

From the definition of the derivate of a matrix and an elementary theorem follows:

$$\frac{\partial}{\partial\beta}(-2Y'X\beta) = -2X'Y$$

and

$$\frac{\partial}{\partial\beta}(\beta'X'X\beta) = \beta'X'X + X'X\beta = 2X'X\beta.$$

The minimizing value b for β can therefore be extracted from:

$$-2X'Y + 2X'Xb = 0,$$

and so from:

$$X'Y = X'Xb.$$

This system is called the *normal equations*.

X'X is nonsingular if the rank of X equals p. In that case the inverse of X'X exists and the solution of the normal equations can be written as

$$b = (X'X)^{-1}X'Y.$$

Observe that $n \geq p$ must hold, in order that the rank of X can be p at all. Therefore at least as many observations must be made as there are parameters in the model.

$$\underline{b} = (X'X)^{-1}X'\underline{Y}$$

is called the *least squares estimator* of β and has the following properties:

1. It is an estimator which minimizes the sum of squares of deviations irrespective of any distribution properties of the disturbances. The assumption that the disturbances are normally distributed is, however, necessary

for tests which depend on this assumption, such as t- or F-tests, or for obtaining confidence intervals based on the t- or F-distributions.

2. According to the Gauss-Markov theorem the elements of $\underline{b}$ are unbiased estimators of the elements of β which have minimum variance (of any linear function of the Y's which provide unbiased estimators), again irrespective of the distribution properties of the disturbances.
3. If the disturbances are mutually independent and normally distributed (with $E(\underline{e}) = 0$ and $\mathrm{var}(\underline{e}) = I\sigma^2$), then $\underline{b}$ is the maximum likelihood estimator of β, for the likelihood function is in that case:

$$\frac{1}{(\sigma\sqrt{2\pi})^n}\exp\left(\frac{(Y-X\beta)'(Y-X\beta)}{2\sigma^2}\right).$$

For any fixed value of σ maximizing the likelihood function is equivalent to minimizing $(Y-X\beta)'(Y-X\beta)$.

The variance-covariance-matrix of $\underline{b}$ is:

$$\begin{aligned}\mathrm{var}(\underline{b}) &= \mathrm{var}((X'X)^{-1}X'\underline{Y}) = \mathrm{var}((X'X)^{-1}X'(X\beta+\underline{e}))\\ &= \mathrm{var}(\beta+(X'X)^{-1}X'\underline{e}) = (X'X)^{-1}X'I\sigma^2((X'X)^{-1}X')'\\ &= (X'X)^{-1}X'X(X'X)^{-1}\sigma^2 = (X'X)^{-1}\sigma^2.\end{aligned}$$

The variances are the diagonal elements and the covariances are the off-diagonal elements.

An estimator for σ^2 is given by

$$\underline{s}^2 = \frac{(\underline{Y}-X'\underline{b})'(\underline{Y}-X'\underline{b})}{n-p} = \frac{\underline{Y}'\underline{Y} - \underline{b}'X'\underline{Y}}{n-p}$$

Let v_{ij} be the element in the i-th row and j-th column of $(X'X)^{-1}$, then $\underline{sd}_i = \sqrt{v_{ii}}\,\underline{s}$ estimates the standard deviation of $\underline{b}_i$, and $c_{ij} = \frac{v_{ij}}{\sqrt{v_{ii} * v_{jj}}}$ gives the correlation coefficient between $\underline{b}_i$ and $\underline{b}_j$ (for $i = 1,\ldots,p$ and $j = 1,\ldots,p$). So:

$$v_{ii} = \left(\frac{\underline{sd}_i}{\underline{s}}\right)^2$$

and

$$v_{ij} = c_{ij} * \frac{\underline{sd}_i * \underline{sd}_j}{\underline{s}^2} = c_{ij} * \sqrt{v_{ii}*v_{jj}} \ .$$

A frequently used statistical measure for evaluating regression models is the multiple correlation coefficient, which is defined as the square root of the proportion of the total sum of squares accounted for by the model, that is:

$$\underline{R} = \left(\frac{b'X'Y-n\bar{y}^2}{\underline{Y}'\underline{Y}-n\bar{\underline{y}}^2}\right)^{\frac{1}{2}} .$$

$\underline{R}^2 * 100$ is often called the percentage of variance explained.

In table 1 the different contributions to the total sum of squares Y'Y are given. (Underlinings omitted because this table contains the realized values of the random variables; cf. the output described in 2.6).

Table 1. *Analysis of variance table*

source	df	sum of squares	mean squares (MS)	F-ratio (FR)	right tail probability
total (uncorrected)	n	$Y'Y$			
mean	1	$n\bar{y}^2$			
total (corrected)	n - 1	$Y'Y - n\bar{y}^2$			
regression	p - 1	$b'X'Y - n\bar{y}^2$	MS_R	$FR_R = \frac{MS_R}{MS_E}$	$P(\underline{F}_R \geq FR_R)$
residual	n - p	$Y'Y - b'X'Y$	$MS_E = s^2$		
lack of fit	k - p	$\tilde{Y}'\tilde{Y} - b'X'Y$	MS_L	$FR_L = \frac{MS_L}{MS_P}$	$P(\underline{F}_L \geq FR_L)$
pure error	n - k	$Y'Y - \tilde{Y}'\tilde{Y}$	MS_P		

In this table the presence of an unknown constant term in the model is assumed; if this term is absent, the "mean"-line and the "total (corrected)"-line vanish and p - 1 changes into p in the "regression"-line.

The lower part of the table is only valid when repeated observations for the dependent variable are available, in which case

k is the number of groups of replications,

m_i is the number of replications per group, and

$\tilde{y} = (\tilde{y}_1,\ldots,\tilde{y}_k)'$, with $\tilde{y}_i = \frac{1}{\sqrt{m_i}} \sum_{j=1}^{m_i} y_{ij}$.

The column "mean square" is obtained by division of the sum of squares by the corresponding degrees of freedom. So

$$MS_R = \frac{b'X'Y - n\bar{y}^2}{p - 1}, \quad MS_L = \frac{\tilde{Y}'\tilde{Y} - b'X'Y}{k - p} \text{ and } MS_P = \frac{Y'Y - \tilde{Y}'\tilde{Y}}{n - k} .$$

If the disturbances are mutually independent and normally distributed (with $E(\underline{e}) = 0$ and $\mathrm{var}(\underline{e}) = I\sigma^2$) we can test:

1. The overall regression equation, or more specifically: the regression null hypothesis is:

$$H_0: \beta_1 = \ldots = \beta_p = 0, \text{ except for the } \beta_i \text{ that denotes the constant term (if present)}$$

which is tested against the alternative hypothesis:

$$H_1: \text{at least one of } \beta_1,\ldots,\beta_p \text{ is unequal to zero,}$$

by treating the F ratio $FR_R = \frac{MS_R}{MS_E}$ as a realization of a $\underline{F}_R = F(p-1,n-p)$ variate.

2. The adequacy (linearity) of the model, by treating the ratio $FR_L = \frac{MS_L}{MS_P}$ as a realization of a $\underline{F}_L = \underline{F}(k-p,n-k)$ variate. Note that a significant lack of fit indicates that the model is wrong and that $\underline{s}^2$ overestimates σ^2. However, $\underline{MS}_p$ still is an unbiased estimator of σ^2.

The vector of residuals is defined as the difference between the vector of observations Y and the vector of fitted values obtained by using the regression equation $\hat{Y} = Xb$. So $D = Y - \hat{Y}$ or $d_i = y_i - \hat{y}_i$, $i = 1,\ldots,n$. If the model is correct, the residual mean square $s^2 = MS_E$ estimates σ^2, and $\sigma^2(\underline{d}_i) = \frac{n-p}{n}\sigma^2$.

iable, followed by an equal sign, followed by the sum of a number of terms, each of which must be the product of an identifier to denote a parameter (to be estimated) and an identifier to denote a dependent variable. An exception is made for the optional constant term in the model, which is given as a single identifier denoting that constant term. Correct model formulae are for instance: that from (1) and

"model" y variable = constant term + parameter * x variable

and

"model" depvar = const + beta1 * xvar1 + beta2 * xvar2.

TRANSFORMATIONS

Almost all transformations a user would like to perform on his data fit quite naturally in the model formula: each transformation is expressed as a formula itself. If, for instance, one wants to include in the model as an independent variable the natural logarithm of the sum of two other variables, one writes: (if those two other variables are called: xvar1 and xvar2)

ln(xvar1 + xvar2).

As operators +, -, *, and / are allowed, all with their conventional meaning (addition, subtraction, multiplication and division, respectively). Also the following twelve *standard functions* are allowed:

abs(E), sign(E), sqrt(E), sin(E), cos(E), arctan(E),
ln(E), exp(E), entier(E), arcsin(E), min(E1,E2) and max(E1,E2)

in which E, E1 and E2 are expressions in terms of variables, operators and standard functions.

Special operators are: // the integer division,
and: ** the exponentiation.

The operator // is defined only for two operands both of type integer and will yield a result of type integer, defined as follows:

a // b = sign(a/b) * entier(abs(a/b)).

The operation <factor> ** <primary> denotes exponentiation, where the factor is the base and the primary is the exponent. Thus for example

2 ** n ** k means $(2^n)^k$

while

2 ** (n**m) means $2^{(n^m)}$

Also the dependent variable may be transformed in this way. As a consequence the model description in its most general form looks like (2) on page 3. Only the disturbance term $\underline{e}$ is omitted and therefore, the dependent variable is not underlined.

Some examples of transformations are:

```
"model" y = a0 + a1 * sqrt(x1+x2) + a2 * sqrt(x3)
"model" arcsin(sqrt(y)) = a0 + a1 * x + a2 * x**2
                          + a3 * x**3 + a4 * x**4.
```

A user can specify model formulas in which terms with *known* regression coefficients appear by subtracting those terms from the left hand part, for instance:

```
"model" y - 5.4321 * x3 = a0 + a1 * x1 + a2 * x2
```

This applies especially to a0; if this term is known it must be shifted to the left hand side.

2.2. THE INPUT DESCRIPTION

To indicate which numbers or series of numbers from the data belong to which variable from the model and which numbers can be skipped, the system expects an input description. It consists of the code word "input" followed by a description (<input statement>) of the arrangement of the observations in the data. The basic idea in that description is that numbers from the data are being identified with the names from the <input statement> in such a way that (in order of entry) numbers belonging to the same name are put in a queue appended to that name, for instance:

"input" 100 * (codenr, 10 * [yvar], [xvar1, xvar2], -1)

means that one hundred series of numbers (each, as a check, terminated in this example by -1) are present in the data. Each series consists of fourteen numbers: first one value which is read to the name codenr, then ten values for the name yvar, then one value for the name xvar1, followed by one value for the name xvar2 and finally, the value -1.

The basic constituent of an <input statement> is a <variable> enclosed in square brackets, in the example: [yvar]. The corresponding number from the data is appended to the (already existing) queue for that name. Several variables can be put together in a <variable list> by separating them by commas and enclosing them in square brackets, in the example: [xvar1, xvar2]. This only serves to save the writing of several opening and closing brackets.

Separate numbers, series or blocks of numbers can be treated by putting a repetition factor (<control>) followed by an asterisk in front of a <variable list> (or in front of an <input statement> which must then be enclosed in round brackets), in the example: 100 * and 10 *.

If a repetition factor is 1, it may be omitted together with the asterisk and a round bracket pair, but square bracket pairs must remain. When a name is used as a repetition factor, a value must already have been assigned to it, which is done by giving that name, without square brackets and followed by a comma, earlier in the <input statement> than the use of that name as a repetition factor. The corresponding number from the data is then assigned as a value to that name. If names are used repeatedly in the <input statement>, the corresponding numbers from the data are being compared. In case of inequality an errormessage is supplied. This may be used as a check against shifted data reading. A similar check can be obtained by using a number separately (that is: followed by a comma or a round closing bracket), in the example the -1. The corresponding number from the data is then compared with that (check)number and again, in case of inequality, an error-message is produced.

Also a simple arithmetic expression is allowed as a repetition factor, or for that matter as a check value, provided that it is enclosed in <- and >-brackets. As in the case of single names used as a repetition factor, each (nonstandard function) name used in such a (special) simple arithmetic expression must have been given, followed by a comma, earlier in the <input statement> than the use of that name in the simple arithmetic expression.

The linkage between the <model statement> and the <input statement> is established by using the same names in the <model statement> and in the input <variable list>. Numbers from the data belonging to such input names will be treated as observations for the model variables, while numbers belonging to input names between square brackets which do not appear in the <model statement> are being skipped.

Often, repeated observations for the dependent variable are available. In order to be able to process this automatically, it is necessary that a <variable list> consisting entirely of dependent variables is preceded by a repetition factor (followed by an asterisk) indicating the number of replications. If a <variable list> contains independent as well as dependent variables, the number of replications is assumed to be 1. A series of (say 100) observations for a dependent variable with *no* replications is denoted as

```
100 * ([dep var])
```

The repetition factor in front of the opening square bracket is omitted (because it is 1), although the round *and* square bracket pair are not. Without the round brackets it would mean 100 replications of [dep var].

EXAMPLE

```
k, n, <k+n> * (check, m, m * [y], [x1,x2,x3,x4], check), -999
```

means that:

first one value is read and assigned to k,
then one value is read and assigned to n,
then k+n times the following happens:
a value is read and assigned to check,
the next value is read and assigned to m,
m values of y are read,
the values of x1, x2, x3 and x4 are read,
a value is read and compared with the check value
finally a value is read and compared with -999.

When the comparison fails an errormessage is supplied.

A combination with the following two models is automatically made by the program:

"model" y = a0 + a1 * x1 + a2 * x2

and

"model" y = a0 + a3 * x3 + a4 * x4.

2.3. THE DATA

The <data> consist of an unstructured series of numbers preceded by the code word "data" (the structure is being imposed onto it by the <input statement>). The series is terminated at the next code word (cf. 2.0).

A sequence of symbols is considered a number when it satisfies the BNF definition of <number> in appendix 1. Moreover, blanks are allowed as lay-out symbols as follows:

a) any number of blanks may follow the sign of a number, the exponent symbol "#", or the sign of the exponent,

b) a single blank may follow a digit or a decimal point.

If two or more blanks follow a digit, the second blank acts as a delimiter. Also the following symbols can act as a delimiter if they follow a digit directly or are being separated from that digit by at most one blank:

a) all symbols not being a digit, "." or "#".
b) the symbol "#" if it follows a digit of the exponent
c) the symbol "." if it follows a digit of the exponent or a digit of the decimal fraction.

It is recommended always to use commas or two (or more) blanks as delimiters.

EXAMPLES

real number	value
1.234	1.234
-0.5673#2	-56.73
0.02#-1	0.002
+#3	1000.0
-463.89#2	-46389.0

2.4. THE OPTIONS

It is possible to have the system perform some tasks optionally by providing an <options statement> in a <job>. It consists of the code "options" followed by a number of option identifiers or by the corresponding option numbers, separated by commas. The following options are available:

option number	option identifier
1	transformed data matrix
2	correlation matrix
3	residual analysis
4	no regression analysis
5	input data from file
6	output data to file
7	process submodels

Options 1, 2 and 3 cause the corresponding piece of information to be printed. Option 4 suppresses the regression analysis; it is meant to be used in combination with option 1 or 2. Option 5 and 6 interfere with the operating system of the computer and are therefore explained in section 2.8.

Option 7 causes the system to process submodels, which are formed by some form of backward elimination: each time the last <term> from the <right hand side> from the <model statement> is omitted (by deleting the last column from the transformed data matrix) and a regression analysis is performed with the reduced design matrix. Messages are generated about which terms are omitted, while further processing of the <job> ceases when the number of terms left is one. Moreover, a test is made (under the usual assumptions, see section 1.2) whether the omitted terms did contribute significantly to the regression sum of squares, or more specifically:
Suppose that the last p - k terms are omitted from the original model, then the reduction null hypothesis is:

$$H_0 : \beta_{k+1} = \dots = \beta_p = 0$$

which is tested against the alternative hypothesis:

H_1: at least one of $\beta_{k+1},\dots,\beta_p$ is unequal to zero,

by treating the F ratio: $F_{RED} = \frac{MS_{RED}}{MS_{RES}}$ as a realization of a $\underline{F} = F(p-k,n-p)$ variate. Herein is MS_{RED} the difference of the regression sum of squares of the original and the reduced model, divided by its degrees of freedom (that is: p - k).

2.5. THE USERS PROGRAM

In a <users program> several <job>s can be submitted to the system. Each <job> is separated from its preceding one by the code word "run", while the entire <users program> is terminated by the code word "exit". In the first <job>, the <model>, <input>, <options> and <data> must be given in some order. In each following <job> a <statement> (cf. section 2.0) which is unchanged may be omitted. The program then retains the last given <statement>. If options have been specified in a previous job and one wants to delete them, this is to be done by a new options-statement which may be empty if no options are to be executed (i.e. by only given: "options").

In front of each <job> or in front of the code word "exit" a text may be given for further identification of the output of a <job> or of the output of the entire <users program>. The use of quotes in that text should be avoided in view of confusion with the code words. The system starts reading a (possibly empty) text just after the code word "run" of the previous <job> (with the first <job> the system starts with the first column of the first card).

2.6. OUTPUT

After having read the code word "run", the processing of the <job> is initiated. First the <model>, <input> and <options> texts are printed in this order, preceded by the (possibly empty) accompanying identification text. Next an attempt is made to translate the <statement>s. Errors against syntax or semantics cause errormessages to be printed below each <statement>, while further processing of that <job> ceases. The processing of a next <job> then will have little or no use when the <statement> which developed the error(s) is not changed. Next the (transformed) data matrix is formed and passed to the regression routine, which supplies the following output in the order indicated:

1) the transformed data matrix (option 1).
2) per (transformed) variable the: mean, standard deviation, minimum and maximum.
3) the correlation matrix of the variables (option 2).
4) the multiple correlation coefficient.
5) the percentages of variance explained and not explained.
6) the estimates for the regression coefficients with estimated standard deviations.
7) the correlation matrix of the estimates (option 2).
8) the analysis of variance table.
9) the residual analysis (option 3).

Ad 1) The transformed data matrix gives the data after transformations according to the model specifications. If the model contains no transformations, the original data are given. Each (transformed) independent variable is indicated by its corresponding parameter. This stems from the fact that it is not obvious how to denote a transformed variable like: arcsin(sqrt(y+25)), with "arcsin", with "sqrt" or perhaps with "y" itself. The dependent variable is indicated by "dep.var."; in the case of replications for the dependent variable, the mean value of them is given.

Ad 3) and 7) The matrix of the estimated correlation coefficients of the variables and of the estimated regression coefficients are both supplied depending on whether option 2 is chosen or not.

Ad 9) Option 3 provides a table of the residuals d_i and of the "standard residuals": $\frac{d_i}{s}\sqrt{\frac{n}{n-p}}$. As a check on the computations, the sum of the residuals is also given. If an unknown constant term is present in the model this sum should be zero.

Without options the output from the program consists of 2), 4), 5), 6) and 7). Option 4 suppresses 4) to 8). If option 7 is used, the output for the model itself is given as specified by the other options, but for the submodels the superfluous parts of the output (that is: the transformed data matrix and the correlation matrix of the variables) are suppressed.

Most numbers in the output are given in 6 decimals, except:

numbers in 6) which are given in 12 decimals

numbers in 1) which are given in 3 decimals

and numbers in 5) which are given in 2 decimals.

2.7. ERRORMESSAGES

Errormessages have the following layout: error number: <error number>. The meaning of the <error number>s is:

600	statement does not terminate properly.
601	in a number . is not followed by a digit.
602	in a number # is not followed by +, - or a digit.
611	left hand part is not followed by an equal sign.
612	) is missing in a simple arithexp.
613	primary starts with an inadmissible symbol.

614	option name used in a simple arithexp.
615	inadmissible identifier in a simple arithexp (used as a control).
616	standard function call with incorrect number of parameters.
617	parameter list is not terminated with).
620	> is missing in a control.
621	inadmissible identifier as a control.
622	) is missing in a description.
623	] is missing in a description.
624	description starts with an inadmissible symbol.
625	standard function name used in a variable list.
626	option name used in a variable list.
627	inadmissible identifier in a variable list.
628	variable starts with an inadmissible symbol.
631	option starts with an inadmissible symbol.
632	incorrect option number used (or generated).
640	no defined identifier to the right of the equal sign.
641	regression parameter used incorrectly.
642	undefined identifier to the left of the equal sign.
643	term does not have the form: par × factor or factor × par.
644	undefined identifier in a term.
645	no regression parameter in a term.
701	constantlist exhausted.
702	namelist exhausted.
703	orderlist exhausted.
704	stack overflow.
800	division by zero.
801	integer division by zero.
802	exponentiation with zero base and non-positive exponent.
803	exponentiation with negative base and real exponent.
804	argument of "sqrt" is negative.
805	argument of "ln" is not positive.
806	given, read or computed replication factor is not an integer.

811 control reads an incorrect number in the data.

812 the replications on a line do not correspond in the left data.

813 the total number in the columns does not correspond in the left data.

814 the total number in the columns does not correspond in the right data.

If the <error number> starts with:

60, 61, 62 or 63 it is followed by the most recently processed symbol (or by the first two characters of the most recently processed code word) which is followed by the first eight characters of the most recently processed identifier.

64, it is followed by the (sequence) number of the right hand part term which causes the error, or a zero if the left hand part is at fault.

70, it is followed by the exceeded upperbound, after which the job starts again with larger lists. The user has no bother.

80, it is followed by the wrong value and the (sequence) number of the line of the transformed data matrix (counting backwards) which causes the error. Instead of the wrong value, the (sequence) number of the right hand part term which causes the error is displayed if the error number is 800 or 801.

81, it is followed by the check value and the wrong value.

2.8. TECHNICAL REMARKS

A program has been written in ALGOL 60 to run under the CDC - SCOPE 3.4 operating system.

The program reads from and writes to SCOPE files, which are linked to the program channels by means of the so-called "channel cards". These channel cards appear as first or only cards on the standard input device. The program expects a <users program> on channel 63 and writes its output to channel 64. By means of the "channel equate cards"

CHANNEL, 63 = 60

and

CHANNEL, 64 = 61

it is possible to read the <users program> from the standard SCOPE file INPUT (associated with channel 60) and write the results to the standard SCOPE file OUTPUT (associated with channel 61). By means of the "channel define cards":

```
CHANNEL, 63 = USERSPR, P126, R
CHANNEL, 64 = USEROUT, P136, PP60, R
```

the <users program> is read from the SCOPE file USERSPR and the output from the program is written to the SCOPE file USEROUT. (Of course all other legal SCOPE file names are allowed!)

The channel cards are ended by

```
CHANNEL, END
```

When option 5 (input data from file) is requested, the program tries to read a series of numbers in ALGOL free field format up to the first EOR-mark from channel 65 and treat that series as <data>. When a <data statement> is present in the <users program> it is ignored. Especially for large amounts of data, considerable gain in computing time is achieved in this way, as reading is done by ALGOL defined I/O procedures rather than program defined ones. Note that the CDC representation for the exponent symbol is: " and not: #.

When option 6 (output data to file) is requested, the program writes the following pieces of information to channel 66:

_a_ when option 1 is requested: the transformed data matrix, preceded by the number of rows and columns respectively.

_b_ when option 3 is requested: the number of (sub)models processed and the number of respondents, followed by for each (sub)model and for each respondent the: observation, fitted value, residual and standardized residual.

and finishes by writing an EOR-mark. Note that when none of the options 1 and 3 is requested, option 6 yields no effect.

An "input" statement to describe one record of data produced this way, could read:

```
"input" n, m, n * (m * [data element]),
        k, n, k * (n * [observation, fitted value, residual,
                        standardized residual])
```

EXAMPLES

On the following pages four users programs and the computer output resulting from them are reproduced.

```
THE FOLLOWING INPUT FOR THE PROGRAM:


*************************************
*                                   *
*  EXAMPLE  1  ORIGINATES FROM:     *
*                                   *
*  REFERENCE [5],  PAGE 472, 479    *
*                                   *
*************************************


*****************************************
*                                       *
*  NOTICE:  2,30258 50930  =  LN(10)    *
*                                       *
*****************************************

"MODEL"  Y = A + B * X + C * (LN(X) / 2,30258 50930)

"INPUT"  5 * ([X], 10 * [Y])

"OPTIONS"  TRANSFORMED DATA MATRIX,
           CORRELATION MATRIX,
           RESIDUAL ANALYSIS,
           PROCESS SUB MODELS

"DATA"

 25  0.67  0.70  0.75  0.76  0.78  0.80  0.83  0.84  0.88  0.89
 50  0.88  0.92  0.93  0.96  0.98  1.00  1.01  1.03  1.06  1.07
 80  0.96  0.98  0.99  1.03  1.05  1.06  1.08  1.11  1.15  1.17
130  1.07  1.09  1.11  1.13  1.14  1.14  1.19  1.22  1.25  1.29
180  1.10  1.13  1.17  1.19  1.20  1.21  1.23  1.25  1.28  1.33

"RUN"

"EXIT"
```

```
RESULTS IN THE FOLLOWING OUTPUT FROM THE PROGRAM;


***********************************
*                                 *
*  EXAMPLE  1  ORIGINATES FROM;   *
*                                 *
*  REFERENCE [5],  PAGE 472, 479  *
*                                 *
***********************************


***************************************
*                                     *
*  NOTICE:  2.30258 50930  =  LN(10)  *
*                                     *
***************************************



"MODEL"  Y = A + B * X + C * (LN(X) / 2.30258 50930)


"INPUT"  5 * ([X], 10 * [Y])


"OPTIONS"  TRANSFORMED DATA MATRIX,
           CORRELATION MATRIX,
           RESIDUAL ANALYSIS,
           PROCESS SUB MODELS
```

TRANSFORMED DATA MATRIX
======================

OBS.NO.	A	B	C	DEP.VAR.
1	1.000	25.000	1.398	.790
2	1.000	50.000	1.699	.984
3	1.000	80.000	1.903	1.058
4	1.000	130.000	2.114	1.163
5	1.000	180.000	2.255	1.209

CONTROL INFORMATION

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION	MINIMUM	MAXIMUM
A	1.000000	.000000	1.000000	1.000000
B	93.000000	56.387870	25.000000	180.000000
C	1.873843	.306746	1.397940	2.255273
DEP.VAR.	1.040800	.163655	.670000	1.330000

CORRELATION MATRIX OF THE VARIABLES

	A	B	C	DEP.VAR.
A	1.000000			
B	*	1.000000		
C	*	.962417	1.000000	
DEP.VAR.	*	.849838	.907742	1.000000

MULTIPLE CORRELATION COEFFICIENT .911959

PERCENTAGE OF VARIANCE

EXPLAINED 83.17

NOT EXPLAINED 16.83

REGRESSION PARAMETERS

PARAMETER	ESTIMATE	STANDARD DEVIATION
A	-.089981931872	.164146913453
B	-.000936132563	.000639569543
C	.649916854704	.117569464868

CORRELATION MATRIX OF THE ESTIMATES

	A	B	C
A	1.000000		
B	.929333	1.000000	
C	-.993392	-.962417	1.000000

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO	RIGHT TAIL PROBABILITY
TOTAL (UNCORRECTED)	50	55.475600			
MEAN	1	54.163232			
TOTAL (CORRECTED)	49	1.312368			
REGRESSION	2	1.091456	.545728	116.105965	.000000
RESIDUAL	47	.220912	.004700		
LACK OF FIT	2	.005012	.002506	.522336	.596685
PURE ERROR	45	.215900	.004798		

REGRESSION NULL HYPOTHESIS : B = C = 0

RESIDUAL ANALYSIS
=================

OBS.NO.	OBSERVATION	FITTED VALUE	RESIDUAL	STANDARDIZED RESIDUAL
1	.790000	.795160	-.005160	-.118992
2	.984000	.967401	.016599	.382824
3	1.058000	1.071978	-.013978	-.322363
4	1.163000	1.162208	.000792	.018260
5	1.209000	1.207254	.001746	.040272

SUM OF RESIDUALS -.000000

CONTROL INFORMATION - SUB MODEL 1
==================

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION	MINIMUM	MAXIMUM
C OMITTED				
A	1.000000	.000000	1.000000	1.000000
B	93.000000	56.387870	25.000000	180.000000
DEP.VAR.	1.040800	.163655	.670000	1.330000

MULTIPLE CORRELATION COEFFICIENT .849838
===============================

PERCENTAGE OF VARIANCE
=====================

EXPLAINED 72.22

NOT EXPLAINED 27.78

REGRESSION PARAMETERS

PARAMETER	ESTIMATE	STANDARD DEVIATION
A	.811415917843	.023947824429
B	.002466495507	.000220785272

CORRELATION MATRIX OF THE ESTIMATES

	A	B
A	1.000000	
B	-.857407	1.000000

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO	RIGHT TAIL PROBABILITY
TOTAL (UNCORRECTED)	50	55.475600			
MEAN	1	54.163232			
TOTAL (CORRECTED)	49	1.312368			
REGRESSION	1	.947825	.947825	124.801688	.000000
RESIDUAL	48	.364543	.007595		
LACK OF FIT	3	.148643	.049548	10.327219	.000027
PURE ERROR	45	.215900	.004798		
REDUCTION	1	.143631	.143631	30.558112	.000001

REGRESSION NULL HYPOTHESIS : B = 0 (IN THE REDUCED MODEL)

REDUCTION NULL HYPOTHESIS : C = 0 (IN THE ORIGINAL MODEL)

RESIDUAL ANALYSIS
================

OBS.NO.	OBSERVATION	FITTED VALUE	RESIDUAL	STANDARDIZED RESIDUAL
1	.790000	.873078	-.083078	-1.230717
2	.984000	.934741	.049259	.729724
3	1.058000	1.008736	.049264	.729800
4	1.163000	1.132060	.030940	.458338
5	1.209000	1.255385	-.046385	-.687146
		SUM OF RESIDUALS	-.000000	

END OF JOB

THE FOLLOWING INPUT FOR THE PROGRAM:

```
************************************
*                                  *
*  EXAMPLE  2  ORIGINATES FROM:    *
*                                  *
*  REFERENCE [3],   PAGE 218, FF.  *
*                                  *
************************************

"MODEL"  CHAMBER PRESSURE = BETA0 + BETA2 * VIBRATION + BETA4 * STATIC FIRE + BETA34 * DROP SHOCK * STATIC FIRE

"INPUT"  24 * [UNIT NO, CYCLE TEMP, VIBRATION, DROP SHOCK, STATIC FIRE, CHAMBER PRESSURE]

"OPTIONS"  TRANSFORMED DATA MATRIX, CORRELATION MATRIX, NO REGRESSION ANALYSIS

"DATA"

 1   -75     0     0   -65    1.4
 2   175     0     0   150   26.3
 3     0   -75     0   150   26.5
 4     0   175     0   -65    5.8
 5     0   -75     0   150   23.4
 6     0   175     0   -65    7.4
 7     0     0   -65   150   29.4
 8     0     0   165   -65    9.7
 9     0     0     0   150   32.9
10   -75   -75     0   150   26.4
11   175   175     0   -65    8.4
12     0   -75   -65   150   28.8
13     0   175   165   -65   11.8
14   -75   -75   -65   150   28.4
15   175   175   165   -65   11.5
16     0   -75     0   150   26.5
17     0   175     0   -65    5.8
18     0     0   -65   -65    1.3
19     0     0   165   150   21.4
20     0   -75   -65   -65    0.4
21     0   175   165   150   22.9
22     0   -75   -65   150   26.4
23     0   175   165   -65   11.4
24     0     0     0   -65    3.7

"RUN"

"EXIT"
```

RESULTS IN THE FOLLOWING OUTPUT FROM THE PROGRAM:

```
***********************************
*                                 *
*  EXAMPLE  2  ORIGINATES FROM:   *
*                                 *
*  REFERENCE [3],  PAGE 218, FF.  *
*                                 *
***********************************
```

"MODEL" CHAMBER PRESSURE = BETA0 + BETA2 * VIBRATION + BETA4 * STATIC FIRE + BETA34 * DROP SHOCK * STATIC FIRE

"INPUT" 24 * [UNIT NO, CYCLE TEMP, VIBRATION, DROP SHOCK, STATIC FIRE, CHAMBER PRESSURE]

"OPTIONS" TRANSFORMED DATA MATRIX, CORRELATION MATRIX, NO REGRESSION ANALYSIS

TRANSFORMED DATA MATRIX

OBS.NO.	BETA0	BETA2	BETA4	BETA34	DEP.VAR.
1	1.000	.000	-65.000	.000	1.400
2	1.000	.000	150.000	.000	26.300
3	1.000	-75.000	150.000	.000	26.500
4	1.000	175.000	-65.000	.000	5.800
5	1.000	-75.000	150.000	.000	23.400
6	1.000	175.000	-65.000	.000	7.400
7	1.000	.000	150.000	-9750.000	29.400
8	1.000	.000	-65.000	-10725.000	9.700
9	1.000	.000	150.000	.000	32.900
10	1.000	-75.000	150.000	.000	26.400
11	1.000	175.000	-65.000	.000	8.400
12	1.000	-75.000	150.000	-9750.000	28.800
13	1.000	175.000	-65.000	-10725.000	11.800
14	1.000	-75.000	150.000	-9750.000	28.400
15	1.000	175.000	-65.000	-10725.000	11.500
16	1.000	-75.000	150.000	.000	26.500
17	1.000	175.000	-65.000	.000	5.800
18	1.000	.000	-65.000	4225.000	1.300
19	1.000	.000	150.000	24750.000	21.400
20	1.000	-75.000	-65.000	4225.000	.400
21	1.000	175.000	150.000	24750.000	22.900
22	1.000	-75.000	150.000	-9750.000	26.400
23	1.000	175.000	-65.000	-10725.000	11.400
24	1.000	.000	-65.000	.000	3.700

CONTROL INFORMATION
===================

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION	MINIMUM	MAXIMUM
BETA0	1.000000	.000000	1.000000	1.000000
BETA2	33.333333	107.001287	-75.000000	175.000000
BETA4	42.500000	109.812092	-65.000000	150.000000
BETA34	-997.916667	9503.497402	-10725.000000	24750.000000
DEP.VAR.	16.579167	10.857976	.400000	32.900000

CORRELATION MATRIX OF THE VARIABLES
===================================

	BETA0	BETA2	BETA4	BETA34	DEP.VAR.
BETA0	1.000000				
BETA2	*	1.000000			
BETA4	*	-.596668	1.000000		
BETA34	*	.058825	.201315	1.000000	
DEP.VAR.	*	-.463978	.943534	-.032554	1.000000

NO REGRESSION ANALYSIS BY OPTION

END OF JOB

THE FOLLOWING INPUT FOR THE PROGRAM;

```
**********************************
*                                *
*  EXAMPLE  3  ORIGINATES FROM;  *
*                                *
*  REFERENCE [3],  PAGE 228, 339 *
*                                *
**********************************
```

"MODEL" LN(MEAN SURFACE VOLUME) = LNALPHA + BETA * LN(FEED RATE) + GAMMA * LN(WHEEL VELOCITY) + DELTA * LN(FEED VISCOSITY)

"INPUT" 35 * [RUN NO, FEED RATE, WHEEL VELOCITY, FEED VISCOSITY, MEAN SURFACE VOLUME]

OPTIONS" TRANSFORMED DATA MATRIX, CORRELATION MATRIX, RESIDUAL ANALYSIS

```
"DATA"  1    0.0174    5300   0.108   25.4
        2    0.0630    5400   0.107   31.6
        3    0.0622    8300   0.107   25.7
        4    0.0118   10800   0.106   17.4
        5    0.1040    4600   0.102   38.2
        6    0.0118   11300   0.105   18.2
        7    0.0122    5800   0.105   26.5
        8    0.0122    8000   0.100   19.3
        9    0.0408   10000   0.106   22.3
       10    0.0408    6600   0.105   26.4
       11    0.0630    8700   0.104   25.8
       12    0.0408    4400   0.104   32.2
       13    0.0415    7600   0.106   25.1
       14    0.1010    4800   0.106   39.7
       15    0.0170    3100   0.106   35.6
       16    0.0412    9300   0.105   23.5
       17    0.0170    7700   0.098   22.1
       18    0.0170    5300   0.099   26.5
       19    0.1010    5700   0.098   39.7
       20    0.0622    6200   0.102   31.5
       21    0.0622    7700   0.102   26.9
       22    0.0170   10200   0.100   18.1
       23    0.0118    4800   0.102   28.4
       24    0.0408    6600   0.102   27.3
       25    0.0622    8300   0.102   25.8
       26    0.0170    7700   0.102   23.1
       27    0.0408    9000   0.613   23.4
       28    0.0170   10100   0.619   18.1
       29    0.0408    5300   0.671   30.9
       30    0.0622    8000   0.624   25.7
       31    0.1010    7300   0.613   29.0
       32    0.0118    6400   0.328   22.0
       33    0.0170    8000   0.341   18.8
       34    0.0118    9700   1.845   17.9
       35    0.0408    6300   1.940   28.4     "RUN"     "EXIT"
```

```
RESULTS IN THE FOLLOWING OUTPUT FROM THE PROGRAM;


**********************************
*                                *
*  EXAMPLE  3  ORIGINATES FROM;  *
*                                *
*  REFERENCE [3],  PAGE 228, 339 *
*                                *
**********************************


"MODEL"  LN(MEAN SURFACE VOLUME)  =  LNALPHA + BETA * LN(FEED RATE) + GAMMA * LN(WHEEL VELOCITY) + DELTA * LN(FEED VISCOSITY)


"INPUT"  35 * [RUN NO, FEED RATE, WHEEL VELOCITY, FEED VISCOSITY, MEAN SURFACE VOLUME]


"OPTIONS"  TRANSFORMED DATA MATRIX, CORRELATION MATRIX, RESIDUAL ANALYSIS
```

TRANSFORMED DATA MATRIX
======================

OBS.NO.	LNALPHA	BETA	GAMMA	DELTA	DEP.VAR.
1	1.000	-4.051	8.575	-2.226	3.235
2	1.000	-2.765	8.594	-2.235	3.453
3	1.000	-2.777	9.024	-2.235	3.246
4	1.000	-4.440	9.287	-2.244	2.856
5	1.000	-2.263	8.434	-2.283	3.643
6	1.000	-4.440	9.333	-2.254	2.901
7	1.000	-4.406	8.666	-2.254	3.277
8	1.000	-4.406	8.987	-2.303	2.960
9	1.000	-3.199	9.210	-2.244	3.105
10	1.000	-3.199	8.795	-2.254	3.273
11	1.000	-2.765	9.071	-2.263	3.250
12	1.000	-3.199	8.389	-2.263	3.472
13	1.000	-3.182	8.936	-2.244	3.223
14	1.000	-2.293	8.476	-2.244	3.681
15	1.000	-4.075	8.039	-2.244	3.572
16	1.000	-3.189	9.138	-2.254	3.157
17	1.000	-4.075	8.949	-2.323	3.096
18	1.000	-4.075	8.575	-2.313	3.277
19	1.000	-2.293	8.648	-2.323	3.681
20	1.000	-2.777	8.732	-2.283	3.450
21	1.000	-2.777	8.949	-2.283	3.292
22	1.000	-4.075	9.230	-2.303	2.896
23	1.000	-4.440	8.476	-2.283	3.346
24	1.000	-3.199	8.795	-2.283	3.307
25	1.000	-2.777	9.024	-2.283	3.250
26	1.000	-4.075	8.949	-2.283	3.140
27	1.000	-3.199	9.105	-.489	3.153
28	1.000	-4.075	9.220	-.480	2.896
29	1.000	-3.199	8.575	-.399	3.431
30	1.000	-2.777	8.987	-.472	3.246
31	1.000	-2.293	8.896	-.489	3.367
32	1.000	-4.440	8.764	-1.115	3.091
33	1.000	-4.075	8.987	-1.076	2.934
34	1.000	-4.440	9.180	.612	2.885
35	1.000	-3.199	8.748	.663	3.346

CONTROL INFORMATION

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION	MINIMUM	MAXIMUM
LNALPHA	1.000000	.000000	1.000000	1.000000
BETA	-3.454469	.748055	-4.439656	-2.263364
GAMMA	8.849891	.298180	8.039157	9.332558
DELTA	-1.778466	.899585	-2.322788	.662688
DEP.VAR.	3.239746	.228501	2.856470	3.681351

CORRELATION MATRIX OF THE VARIABLES

	LNALPHA	BETA	GAMMA	DELTA	DEP.VAR.
LNALPHA	1.000000				
BETA	*	1.000000			
GAMMA	*	-.181207	1.000000		
DELTA	*	-.036208	.180096	1.000000	
DEP.VAR.	*	.680447	-.811063	-.202936	1.000000

MULTIPLE CORRELATION COEFFICIENT .977342

PERCENTAGE OF VARIANCE

EXPLAINED 95.52

NOT EXPLAINED 4.48

REGRESSION PARAMETERS

PARAMETER	ESTIMATE	STANDARD DEVIATION
LNALPHA	8.549532333086	.266023898546
BETA	.168424405160	.011808119577
GAMMA	-.537137014095	.030096077272
DELTA	-.014413466996	.009817045795

CORRELATION MATRIX OF THE ESTIMATES

	LNALPHA	BETA	GAMMA	DELTA
LNALPHA	1.000000			
BETA	-.024345	1.000000		
GAMMA	-.985554	.177707	1.000000	
DELTA	.242973	.003696	-.176561	1.000000

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO	RIGHT TAIL PROBABILITY
TOTAL (UNCORRECTED)	35	369.133526			
MEAN	1	367.358289			
TOTAL (CORRECTED)	34	1.775238			
REGRESSION	3	1.695702	.565234	220.306257	.000000
RESIDUAL	31	.079536	.002566		

REGRESSION NULL HYPOTHESIS : BETA = GAMMA = DELTA = 0

RESIDUAL ANALYSIS
================

OBS.NO.	OBSERVATION	FITTED VALUE	RESIDUAL	STANDARDIZED RESIDUAL
1	3.234749	3.293078	-.058329	-1.223587
2	3.453157	3.499877	-.046720	-.980073
3	3.246491	3.266833	-.020342	-.426727
4	2.856470	2.845581	.010889	.228428
5	3.642836	3.671117	-.028281	-.593273
6	2.901422	2.821409	.080013	1.678467
7	3.277145	3.185264	.091881	1.927422
8	2.960105	3.013233	-.053128	-1.114483
9	3.104587	3.095864	.008723	.182980
10	3.273364	3.319189	-.045825	-.961298
11	3.250374	3.244114	.006261	.131334
12	3.471966	3.537118	-.065151	-1.366704
13	3.222868	3.246139	-.023271	-.488175
14	3.681351	3.642772	.038579	.809285
15	3.572346	3.577499	-.005154	-.108113
16	3.157000	3.136624	.020376	.427441
17	3.095578	3.089933	.005644	.118401
18	3.277145	3.290415	-.013270	-.278377
19	3.681351	3.551596	.129755	2.721926
20	3.449988	3.424209	.025778	.540765
21	3.292126	3.307827	-.015701	-.329363
22	2.895912	2.938617	-.042705	-.895839
23	3.346389	3.281716	.064673	1.356676
24	3.306887	3.319607	-.012720	-.266842
25	3.250374	3.267523	-.017148	-.359731
26	3.139833	3.089357	.050476	1.058853
27	3.152736	3.127162	.025574	.536468
28	2.895912	2.917634	-.021722	-.455674
29	3.430756	3.410283	.020473	.429475
30	3.246491	3.261192	-.014701	-.308384
31	3.367296	3.392279	-.024983	-.524074
32	3.091042	3.110356	-.019313	-.405145
33	2.933857	3.051431	-.117574	-2.466405
34	2.884801	2.862104	.022697	.476118
35	3.346389	3.302140	.044249	.928227
		SUM OF RESIDUALS	.000000	

END OF JOB

```
THE FOLLOWING INPUT FOR THE PROGRAM;


****************************************
*                                      *
*  EXAMPLE  4  ORIGINATES FROM;        *
*                                      *
*  REFERENCE [1],  PAGE 88, 93, FF.    *
*                                      *
****************************************

"MODEL"  Y  =  ALFA0  +  ALFA1 * X

"INPUT"  5 * ([X], N, N * [Y])

"OPTIONS"

"DATA"

 1   4    1.1    0.7    1.8    0.4
 3   5    3.0    1.4    4.9    4.4    4.5
 5   3    7.3    8.2    6.2
10   4   12.0   13.1   12.6   13.2
15   4   18.7   19.7   17.4   17.1

"RUN"

      **************************
      *                        *
      *  MARTEN VAN GELDEREN   *
      *                        *
      *  MATHEMATISCH CENTRUM  *
      *                        *
      **************************

"EXIT"
```

RESULTS IN THE FOLLOWING OUTPUT FROM THE PROGRAM;

```
**************************************
*                                    *
*  EXAMPLE  4  ORIGINATES FROM;      *
*                                    *
*  REFERENCE (1),  PAGE 88, 93, FF.  *
*                                    *
**************************************



"MODEL"  Y  =  ALFA0  +  ALFA1 * X


"INPUT"  5 * ((X), N, N * (Y))


"OPTIONS"
```

CONTROL INFORMATION
===================

TRANSFORMED VARIABLE DENOTED BY PARAMETER	MEAN	STANDARD DEVIATION	MINIMUM	MAXIMUM
ALFA0	1.000000	.000000	1.000000	1.000000
ALFA1	6.700000	5.262579	1.000000	15.000000
DEP.VAR.	8.385000	6.545571	.400000	19.700000

MULTIPLE CORRELATION COEFFICIENT .987051
================================

PERCENTAGE OF VARIANCE
======================

EXPLAINED 97.43

NOT EXPLAINED 2.57

REGRESSION PARAMETERS

PARAMETER	ESTIMATE	STANDARD DEVIATION
ALFA0	.159483086279	.396807248659
ALFA1	1.227689091600	.047026168992

ANALYSIS OF VARIANCE TABLE

SOURCE	DF	SUM OF SQUARES	MEAN SQUARE	F - RATIO	RIGHT TAIL PROBABILITY
TOTAL (UNCORRECTED)	20	2220.210000			
MEAN	1	1406.164500			
TOTAL (CORRECTED)	19	814.045500			
REGRESSION	1	793.099430	793.099430	681.549798	.000000
RESIDUAL	18	20.946070	1.163671		
LACK OF FIT	3	4.252403	1.417468	1.273658	.319196
PURE ERROR	15	16.693667	1.112911		

REGRESSION NULL HYPOTHESIS : ALFA1 = 0

END OF JOB

```
***************************
*                         *
*  MARTEN VAN GELDEREN    *
*                         *
*  MATHEMATISCH CENTRUM   *
*                         *
***************************
```

END OF USERS PROGRAM

NUMBER OF RUNS: 4

APPENDIX 1

FORMAL DEFINITION

The syntax of the system is described in a notation known as the Backus Normal Form (BNF). It was introduced specifically for the description of ALGOL 60 and is similar to techniques for describing formal languages that have been used by logicians and linguists.

The BNF may be regarded as a metalanguage for the description of the system. In addition to the symbols that are admissible in the system, the metalanguage requires a number of additional symbols called metasymbols. The four metasymbols used in BNF are ::=, |, <, >. The , and . are part of the metalanguage English in which we are describing BNF.

We write:

```
<identifier>::=<letter>|<identifier><letter>|<identifier><digit>.
```

The metasymbols < and > are used as delimiters to enclose the name of a class. The metasymbol ::= may be read as "is defined as" or "consists of". The | is read as "or". The above phrase defines an <identifier> as a <letter> or as an <identifier> followed by a <letter> or as an <identifier> followed by a <digit>. Such an identifier is at least one letter, optionally followed by any number of letters and/or digits.

These (recursive) definitions may seem a bit peculiar since a class is defined in terms of itself. They make sense only because of the presence in the definition of at least one alternative that does not contain the class definition.

Syntax:

```
<letter>::= A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z

<digit>::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<adding operator>::= + | -
<multiplying operator>::= * | / | //

<model symbol>::= "MODEL" | "MO"
<input symbol>::= "INPUT" | "IN"
<options symbol>::= "OPTIONS" | "OP"
<data symbol>::= "DATA" | "DA"
<run symbol>::= "RUN" | "RU"
<exit symbol>::= "EXIT" | "EX"

<number>::= <unsigned number> | <adding operator> <unsigned number>
<unsigned number>::= <decimal number> | <exponent part> |
                     <decimal number> <exponent part>
<decimal number>::= <unsigned integer> | <decimal fraction> |
                    <unsigned integer> <decimal fraction>
<exponent part>::= # <integer>
<decimal fraction>::= . <unsigned integer>
<integer>::= <unsigned integer> |
             <adding operator> <unsigned integer>
<unsigned integer>::= <digit> | <unsigned integer> <digit>

<identifier>::= <letter> | <identifier> <letter>
                         | <identifier> <digit>

<data>::= <data symbol> <data list>
<data list>::= <number> | <data list> <number>

<options>::= <options symbol> <option list>
<option list>::= <option> | <option list> , <option>
<option>::= <identifier> | <number>

<input>::= <input symbol> <input statement>
<input statement>::= <part> | <input statement> , <part>
<part>::= <control> | <description> | <control> * <description>
<control>::= <number> | <identifier> | < <simple arithexp> >
<description>::= ( <input statement> ) | [ <variable list> ]
<variable list>::= <variable> | <variable list> , <variable>
<variable>::= <identifier>
```

```
<model>::= <model symbol> <model statement>
<model statement>::= <left hand part> = <right hand part>
<left hand part>::= <simple arithexp>
<right hand part>::= <term> | + <term> | <right hand part> + <term>
<simple arithexp>::= <term> | <adding operator> <term> |
                     <simple arithexp> <adding operator> <term>
<term>::= <factor> | <term> <multiplying operator> <factor>
<factor>::= <primary> | <factor> ** <primary>
<primary>::= <unsigned number> | <identifier> |
             <function designator> | ( <simple arithexp> )
<function designator>::= <identifier> |
                         <identifier> ( <parameter list> )
<parameter list>::= <simple arithexp> |
                    <parameter list> , <simple arithexp>

<users program>::= <job> <exit symbol> | <job> <users program>
<job>::= <statement> <run symbol> | <statement> <job>
<statement>::= <model> | <input> | <options> | <data>
```

APPENDIX 2

TECHNICAL DESCRIPTION OF THE PROGRAM

In this section a more or less technical description of the program in terms of identifiers, procedures and overall job-flow is given. Each name referring to an identifier in the program is quoted.

Basically the program logic is as follows

```
     INIT SYSTEM & BUFFERS
JOB: READ NEXT JOB
     INIT COMPILER TABLES
     COMPILE MODEL & INPUT
     INIT DATA BUFFERS
     EXECUTE (TO PRODUCE DESIGN MATRIX)
     REGRESSION ANALYSIS
     PRINT RESULTS
     GOTO JOB
```

In case of an errorsituation in one of the sections no further action is undertaken and control is transfered to JOB.

1. The system is initialized by declaring various system and compiler variables and constants, while setting the constants to their appropriate values. The internal representation of the symbols is equal to the CDC display code.

The two basic buffers are "symbollist" for handling symbols and "datalist" for handling numbers; "column" is used in transporting two-dimensional buffers to and from mass storage.

2. A job is read statement by statement (after printing and skipping preceding text) by means of successive calls of the procedure "readlist", which in its turn calls "readsymbol" or "readnumber" (depending on which statement from the job is read).

3. The compiler uses three tables: a "namelist", a "constantlist" and an "orderlist". The first two tables are used for handling alphanumerical information, while the (macro) orders generated by the compiler are delivered in the "orderlist" via a call of "store".

The function procedure "nextlistsymbol" assigns to itself the next symbol from the model, input or option statement, while "nextnumber" assigns to itself the next number from the datalist. The function procedure "nextsymbol" is used by the compiler in scanning the various statements. It calls "nextlistsymbol" and isolates a to-the-power-symbol from two consecutive times-symbols and an integer-division-symbol from two consecutive division-symbols, moreover blanks are skipped.

The function procedure "unsigned number" reads a number, looks it up in the "constantlist" and assigns to itself an address in that list where the value of that number can be found.

In a quite similar way, the function procedure "identifier" has to read an identifier, look it up in the namelist and assign to itself an address where in the "namelist" a reference to the first eight characters of that identifier and in the "stack" the value of that identifier can be found.

The compiler assumes that the running system has at its disposal a (programmed pseudo) register "F" which is capable of handling as well floating point numbers as integers. Furthermore, a memory organization known as a "stack" must be available to the running system. A so-called "stackpointer" refers to the first free position in the "stack". All binary operations will take place with the top of the "stack" as first operand and "F" as the second, the result being delivered in "F". As a side effect the "stackpointer" is decreased by 1. When the contents of "F" are saved in the "stack", the "stackpointer" is increased by 1.

The fundamental idea behind almost all procedures for translating the model and input statements is that the first basic symbol of the syntactical unit to be processed by that procedure has been read already (its value being assigned to "lastsymbol"). The procedure considers itself to have finished its task after reading the first basic symbol that no longer can belong to that unit syntactically. Meanwhile the translation of that unit has been produced. A more elaborate description of the procedure system "simple arithexp" can be found in [6] and [7]. Reference [8] provides the description of a complete ALGOL 60 compiler. We only mention here that every "simple arithexp" is being transformed into a macro program corresponding to the so-called reversed polish form, thus:

$$(a+b) * (c-d) \uparrow e \quad \text{becomes:} \quad (ab+)\ (cd-)\ e \uparrow *.$$

The procedure for translating the input statement must, among others, generate orders to perform the linkage between identifiers from the model and numbers from the data.

While translating a model statement, identifiers to the right of the equal sign are assigned type 1, those to the left of the equal sign type 2. If these identifiers appear in a "variable list" the types are changed into 3 and 5 respectively. Identifiers in a "variable list" not appearing in the model statement are assigned type 4, those in the input statement not appearing in a "variable list" are assigned type 6. Meanwhile orders are generated to put the next number from the data in the appropriate column of the (yet untransformed) design matrix, or to skip that number. For "variable lists" that consist entirely of identifiers not appearing in the model statement, special orders to skip the corresponding numbers all in one, are generated.

4. In "check model" a check is made if the model statement after the linkage to the data (by means of the input statement) still satisfies some elementary statistical conditions, like:

a) each term must be the product of a parameter and a factor
b) in that factor no identifier may appear that is not present in a variable list in the input statement. (An attempt to perform regression analysis with variables for which no data is present may not succeed.)

In "check input" a check is made (by means of the data-pointer) if all numbers in the data list have actually been processed. Moreover, a check is made if for each variable in the model statement an equal amount of numbers is present.

5. The execution section of the input system is activated by a call of the procedure "execute" which, among other things, simulates the basic cycle of

a computer:

```
next: get the order, indicated by the ordercounter
      increase the ordercounter by 1
      isolate the order- and address-part
      execute the order
      goto next
```

This cycle ends when the ordercounter tries to leave the (translated macro) program. The (macro) orders itself are coded via the switch lists "macro" and "macro2". In an errorsituation a call of "endrun" updates the administration of the mass storage section.

6. After the execution of the input- and modelorders, the (transformed) design matrix is delivered in the array "resultlist". In case of buffer overflow, successive column buffer images are written to mass storage.

When a residual analysis or a process submodels option is requested, the current result list must be saved to mass storage, regardless of buffer overflow. A call of "restore resultlist" resets all administration and buffers in such a way that the following piece of program can act as if nothing has happened.

The actual computation of the regression coefficients is done via a call of "lsqortdec" followed by a call of "lsqsol" and of "lsqinv". These are slightly modified version of lsqdec and lsqsol, described in [2] pp.65-69. The correlation matrix is computed via calls of "tammat", described in [2] pp.8-9.

All other computations are straightforward.

REFERENCES

[1] AFIFI, A.A. & S.P. AZEN, *Statistical Analysis; A Computer Oriented Approach*, Academic Press, (1972).

[2] DEKKER, T.J., *ALGOL 60 procedures in numerical algebra, part 1*, MC Tract 22, Mathematisch Centrum, (1970).

[3] DRAPER, N.R. & H. SMITH, *Applied Regression Analysis*, John Wiley & Sons, (1966).

[4] GRUNE, D. (ed.), *LR 1.1, Handleiding MILLI-systeem voor de EL X8*, Mathematisch Centrum, (1971).

[5] JONGE, H. DE, *Inleiding tot de Medische Statistiek, deel II*, Nederlands Instituut voor Praeventieve Geneeskunde, (1960).

[6] KRUSEMAN ARETZ, F.E.J., *ALGOL 60 translation for everybody*, Elektronische Datenverarbeitung, 6 (1964), 6, p.233-244.

[7] KRUSEMAN ARETZ, F.E.J., *Programmeren voor rekenautomaten, (De MC ALGOL 60 vertaler voor de EL X8)*, MC Syllabus 13, Mathematisch Centrum, (1972).

[8] KRUSEMAN ARETZ, F.E.J., P.J.W. TEN HAGEN & H.L. OUDSHOORN, *An ALGOL 60 compiler in ALGOL 60*, MC Tract 48, Mathematisch Centrum, (1973).

[9] NAUR, P. (ed.), *Revised report on the algorithm language ALGOL 60*, Regnecentralen, Copenhagen, (1964).

[10] ROSEN, S. (ed.), *Programming systems and languages*, McGraw-Hill Book Company, (1967).

APPENDIX 3

ADDITIONS

1. The estimate of σ: $s = \sqrt{\frac{Y'Y - b'X'Y}{n - p}}$, which is frequently called "standard error of estimate", but which is indicated more properly here as "standard deviation of the error term", is given as a separate entry in the computer output.

2. If the disturbances are mutually independent and normally distributed, we can perform a significance test for a particular regression coefficient, or more specifically:
   For a particular $i \leq p$ the null hypothesis is:

   H_0: $\beta_i = 0$ (given that all β_j for $j \neq i$ are in the model)

   which is tested against the alternative hypothesis:

   H_1: $\beta_i \neq 0$,

   by treating the ratio $FR_i = \frac{b_i^2}{sd_i^2}$ as a realisation of a $\underline{F}_i = \underline{F}(1,n-p)$ variate.

   However this test must be used with caution, because with a (preset) level of significance α only <u>one</u> coefficient can be tested properly this way, while the computer output lists statistics for all coefficients.
   It seems very tempting to test the coefficients serially one at a time, but one must keep in mind that in doing so the level of significance of the whole test rises above the nominal value.

3. Besides the multiple correlation coefficient R, the adjusted multiple correlation coefficient $\bar{R}$, which is defined by:

$$\bar{R}^2 = 1 - \frac{n-1}{n-p}(1 - R^2) = 1 - \frac{n-1}{n-p}\left(\frac{Y'Y - b'X'Y}{Y'Y - n\bar{y}^2}\right),$$

is also given in the computer output. A corresponding adjustment is made to the proportion of variation explained.

4. The vector of fitted values is obtained by using the regression equation: $\hat{Y} = Xb$.
The estimated standard deviation of $\hat{y}_i$ at $x_i = (x_{i1},\ldots,x_{ip})$ is:

$$s\sqrt{x_i'\,(X'X)^{-1}\,x_i}\,,$$

and is given as a separate (column) entry in the computer output of the residual analysis.
It can be used to construct a confidence interval for the expected value of $\underline{y}_i$: $E(\underline{y}_i)$ at $x_i = (x_{i1},\ldots,x_{ip})$, or to construct a prediction interval for the mean of g new observations at this point.
In the first case the confidence interval is:

$$\hat{y}_i \pm t\{n-p-1,\ 1-\tfrac{1}{2}\alpha\}\ s\sqrt{x_i'\,(X'X)^{-1}\,x_i}\,,$$

and in the second case the prediction interval is:

$$\hat{y}_i \pm t\{n-p-1,\ 1-\tfrac{1}{2}\alpha\}\ s\sqrt{\frac{1}{g} + x_i'\,(X'X)^{-1}\,x_i}\,.$$

5. To the process submodel's option a so called "submodel specification list" can be appended, to prevent the production of waste output for unwanted submodels. In this list the number of terms to be omitted must be given enclosed in round brackets. For example the option process submodels (6, 10) instructs the program to process only two submodels, one with the last six terms omitted and one with the last ten terms omitted (from the complete model). If the user asks for more terms to be omitted than are present in the complete model, an errormessage is supplied and the execution of that job is terminated. Moreover if in addition to the process submodels option no submodel specification list is given, options 2 and 3 yield no effect (even if specified!) for the output of the submodels. So if the user wants option 2 and/or 3 to have effect for (some of) his submodels, an exhaustive submodel specification list of those submodels must be given.

   The following errormessages have been added:

   633) is missing after a submodel specification list.
   634 submodel specification starts with an inadmissible symbol.
   635 submodel specified with less than one term left.

6. The correlation matrix of the variables and, instead of the correlation matrix of the estimates, the corresponding variance-covariance matrix are written to file (via channel 66) if options 2 and 6 are requested.

   An input statement to describe one record of data produced and written to file when options 1, 2, 3, 6 and 7 are requested, could read:

```
"input"  n, m, n * (m * [data element]),
         k, <k * (k+1) // 2> * [correl element],
         p, p * (1, <1 * (1+1) // 2> * [covar element],
                n, n * [obs, fitval, standdev, resid, standres])
```

7. Errormessage 806 has been renumbered to 810, while a new errormessage with errornumber 806 has been introduced, so

806	argument of "arcsin" is in absolute value larger than one.
810	given, read or computed replication factor is not an integer.

8. An option with option number 8 and option identifier "print datalist" has been introduced. It provides the possibility to echo <u>all</u> the data back to the lineprinter, instead of just the transformed data matrix. However no special layout is imposed, because the "data" consists of an unstructured series of numbers (c.f. section 2.3).