The REGRESS Program
Version 4.26 – March 30, 2009

Developed by Prof. John R. Wolberg
TECHNION - Israel Institute of Technology
Haifa, ISRAEL
www.technion.ac.il/wolberg

CONTENTS:

1. OVERVIEW:

The **REGRESS** program is a general-purpose tool for least squares analysis of data. (A detailed description of the method of least squares is included in my book: *Data Analysis Using the Method of Least Squares*. The book was published by Springer in 2006.) The program input includes data and functions used to fit the data. **REGRESS** includes the following features:

1) The fitting functions may be linear or nonlinear (see the section on **FITTING FUNCTIONS**).

2) The fitting functions may be multi-dimensional.

3) Basic functions such as EXP, LOG, SIN, SQRT, etc. may be used in the formulation of the fitting functions.

4) The dependent variable may be a scalar or a vector quantity.

5) A variety of methods for weighting the data points are available (see the section on **DATA WEIGHTING**).

6) The data can be partitioned into modeling and evaluation data sets (see the section on the **EVALUATION DATA SET**).

7) The program output includes the values of the unknown parameters of the fitting functions and estimates of their standard deviations.

8) Optionally, the program output may include an interpolation table with values of the fitting function included for user specified values of the independent variable (or variables) and estimated standard deviations of these values (see the section on the **INTERPOLATION TABLE**).

9) For problems in which there is difficulty in achieving convergence, a number of convergence enhancement features are available (see the section on **CONVERGENCE PROBLEMS AND ADVICE**).

10) An integral operator is available for problems requiring solution of initial valued differential equations (see the section on **DIFFERENTIAL EQUATIONS**).

11) The program supports Bayesian estimators for the unknown parameters of the fitting functions (see the section on **BAYESIAN ESTIMATORS**).

12) The program supports recursive models (see the section on **RECURSIVE MODELS**).

13) The program can be used to simulate experiments in order to estimate the uncertainly that one can expect from a proposed experiment (see the section on **PREDICTION ANALYSIS**)

14) The program can accept text data or Excel data (see **USING EXCEL DATA FILES**).

2. FITTING FUNCTIONS:

A number of examples of usage of REGRESS with various fitting functions are included on the website `www.technion.ac.il/wolberg` , The functions can be based upon a variety of operators and built-in basic functions. Each fitting function includes one or more unknown coefficients. **REGRESS** attempts to determine the values of the unknown coefficients that minimize the least-squares criterion (see the section on THE **METHOD OF LEAST SQUARES**). Functions are specified as follows in **REGRESS**:

```
F=' A1*EXP(-A2*X) + A3'
```

or alternatively:

```
Y=' A1*EXP(-A2*X) + A3'
```

In Version 4 of REGRESS, alias names can be used in place of the X's, Y's or F's, and A's. For example consider the following:

```
DEPENDENT     RATE;
INDEPENDENT   TIME;
UNKNOWN       amplitude, time_constant, backround;
RATE = ' amplitude*EXP(-time_constant*TIME) + backround'
```

See the section on **ALIASES** for details and for additional examples.

Detailed rules for specifying functions are included in the section on **FUNCTION SPECIFICATION**. As a second example, the following is a function with two unknowns (i.e., A1 and A2) and two independent variables (i.e., X1 and X2):

F='A1 + A2*X1 + A3*X2'

The program is limited to functions with a maximum of 20 unknowns and 9 independent variables. Functions may also include constants:

F='17.3 + A1*SIN(PI*X1) - COS(X2/(PI*2)) / (1-A2)'

The program recognizes PI as a constant. This function contains two unknowns and two independent variables. Functions can also be specified with symbolic constants:

F='Q1 + A1*SIN(PI*X1) - COS(X2/(PI*2))/(1-A2)'

If Q1 is specified as 17.3 then this function is exactly the same as the previous function and the resulting values of A1 and A2 will be exactly the same. However, by changing Q1, we can change the function slightly without having to rewrite it. Up to 9 symbolic constants can be used to specify a function. . In Version 4 constants can also be specified using aliases.

For problems in which the dependent variable is a vector and not a scalar, the program allows the user to include up to nine elements of the dependent variable. Each variable element requires a separate function (i.e., F1, F2, ... or Y1, Y2, ... or alias names). See the section on **FUNCTION SPECIFICATION** for examples of models requiring more than one function.

The method of least squares is sensitive to whether or not the fitting function (or functions) is linear or nonlinear with respect to the unknown parameters A_k. For example, the first function considered above is non-linear and the second function is linear. The problem with nonlinear functions is that they do not always converge to a solution. **REGRESS** includes a number of features for enhancing convergence.

When a solution has been obtained, several output tables are displayed including the values of A's and their standard deviations (SIGA's), the values of the Y's calculated at the input X values and the value of the least squares criterion: S / (N - P). If an interpolation table has been specified, this is also displayed. All output is displayed on the screen and also sent to the output file. The program then allows the user to alter parameters and repeat the calculation. The fitting function can be changed at this point so the user can repeat the analysis for a variety of functions using the same data.

3. SPECIFYING A REGRESS RUN:

REGRESS runs are specified by a parameter file. Data may be included within the parameter file or as a separate file. The syntax for running the **REGRESS** program from a DOS prompt is **REGRESS** followed by one, two or three file names:

REGRESS parameter_file [data_file] [output_file]

For example:

REGRESS EXP1.PAR SAMPLE.DAT

The first name specifies the name of the Parameter file, the second name (if specified) specifies the name of the Data file, and the third name (if specified) is the name of the Output file. If the 2nd and 3rd names are not specified, the program assumes that the data is in the parameter file. If the data is included in the

parameter file it must start on a new line after a semicolon at the end of the list of parameters. If the Output file is not specified it is assumed to be Name_of_Parms_file.OUT. For the above example, the program output is directed to the file EXP1.OUT (as well as the screen). Full names do not have to be specified. If the filetype of the Parameter file is not specified it is assumed to be PAR or PARMS. If the filetype of the Data file is not specified it is assumed to be TXT or DAT. If the filetype of the Output file is not specified it is assumed to be OUT. Thus the above example could have been specified as follows:

REGRESS EXP1 SAMPLE

A number of examples are included on the website www.technion.ac.il/wolberg,

4. THE PARAMETER FILE:

A variety of parameters can be specified in this file and their order is unimportant. Some parameters must be specified (NCOL and F or Y or a suitable "alias") and others may be specified or they will assume their default values. The file is case insensitive (i.e., the user may use upper and lower case letters at his or her discretion). A list of REGRESS parameters is as follows:

PARAMETER	OPTIONAL	DEFAULT-VALUE	EXPLANATION
A0(k)	Yes	0.0	Initial guess of A(k)
A(k)	Yes	0.0	Alternative form of A0(k)
AMIN(k)	Yes	Not specified	Lower search limit for A(k)
AMAX(k)	Yes	Not specified	Upper search limit for A(k)
CAF	Yes	1.0	Convergence acceleration factor
CONDITION	Yes	0	If 1 CONDITION NUMBER in output
CT(i)	Yes	1.0	Synonym for CX(i)
CX(i)	Yes	1.0	Constant used in Sig X(i) calc (see below)
CY	Yes	1.0	Constant used in Sig Y calc (see below)
CY(i)	Yes	1.0	Constant used in Sig Y(i) calc (see below)
DEBUG	Yes	0	0 or 1: 1 gives C, V & DEL_A on OUT file
DISPLAY	Yes	2	0-3 Display levels (see below)
DT(i)	Yes	Not specified	Synonym for DX(i)
DX(i)	Yes	Not specified	Interpolation step for X(i)
EPS	Yes	0.001	Convergence criterion
F	No		Fitting Function (see below)
GROUP	Yes	Not specified	See Evaluation Data Set
F(i)	No		Fitting Function (see below)
FTYPE	Yes	Not specified	'E' : Excel tab delimited; 'P' : PRISM
M	Yes	'Y'	Number of independent variables
MAXDEPTH	Yes	(See below)	Maximum depth in integration scheme
MODE	Yes	Not specified	MODE='P' initiates prediction analysis
MODEL_FIRST	Yes	'Y'	See Evaluation Data Set section
N	**	(See below)	Number of data records
NY	Yes	From F(i)'s	Number of dependent variables
NCOL	*	(See below)	Number of data columns per data record
NEVL	Yes	Not specified	See Evaluation Data Set section
NP(i)	Yes	Not specified	Number of interpolation steps for X(i)
NREC	**	(See below)	Synonym for N
NUMITMAX	Yes	15	Maximum number of iterations
NUMRECIT	Yes	10	Max num iterations in recursion calculation
NUMXCOLS	***	(See below)	Number of X columns

PARM1[i]	Yes	Not specified	Distribution parm – Prediction Analysis
PARM2[i]	Yes	Not specified	Distribution parm – Prediction Analysis
PRINT_INVERSE	Yes	0	Print Inverse C matrix if > 0
Q(i)	Yes	Not specified	Value of symbolic constant Q(i)
RAF	Yes	1.0	Recursion acceleration factor
RECEP	Yes	EPS	Convergence criterion: recursion calculation
REL_ERROR	Yes	'N'	Include relative error in output
SEED	Yes	13	Seed for rand generator Prediction Analysis
SET	Yes	1	Set number for the first analysis
SIGA0(k)	Yes	Not specified	Sigma of Bayesian estimator for A(k)
SIGX	Yes	Z	Alternative to SXTYPE (Z, U, C, F or S)
SIGX(i)	Yes	Z	Alternative to SXTYPE(i)
SIGY	Yes	U	Alternative to SYTYPE (Z, U, C, F or S)
SIGY(i)	Yes	U	Alternative to SYTYPE(i)
STARTEVAL	Yes	Not specified	First record used in Evaluation Data Set
STARTREC	Yes	1	First record used
STARTXCOL	Yes	1	XCOL(1)
STCOL	Yes		Synonym for SXCOL
STCOL(i)	Yes		Synonym for SXCOL(i)
STTYPE	Yes		Synonym for SXTYPE
STTYPE(i)	Yes		Synonym for SXTYPE(i)
SXCOL	Yes		Synonym for SXCOL(1)
SXCOL(i)	Yes	Not specified	Sigma X(i) column for data record
SXTYPE	Yes	0	Synonym for SXTYPE(1)
SXTYPE(i)	Yes	0	Sigma type for X(i) (See below)
SYCOL	Yes	Not specified	Synonym for SYCOL(1)
SYCOL(i)	Yes	Not specified	Sigma Y(i) column for data record
SYTYPE	Yes	1	Synonym for SYTYPE(1)
SYTYPE(i)	Yes	1	Sigma type for Y(i) (See below)
T0(i)	Yes		Synonym for X0(i)
TCOL	Yes	1	Synonym for XCOL
X0(i)	Yes	Not specified	Initial value for X(i) interpolation
XCOL	Yes	1	Synonym for XCOL(1)
XCOL(i)	Yes	i	X(i) column for data record
Y	No		Alternative form of F
Y(i)	No		Alternative form of F(i)
YCOL	Yes	NCOL	Y column for data record
YCOL(i)	Yes	M + i	Y(i) column for data record

* - NCOL must be specified if the input data is in *ascii* format. If the data file has been specified as an Excel file (FTYPE = 'e'**)** then NCOL does not have to be specified as it can be determined from the file. If the data is in PRISM format (i.e., FTYPE = 'p'**,** a specific binary format), the value of NCOL will be read directly from the file (even if it is specified).

**- NREC (or N) does not have to be specified. If the data is in *ascii* format, the total number of numerical values (num_vals) is counted and then the number of records is computed by dividing by NCOL. If the remainder of num_vals/NCOL is not zero, then an error message is issued. If the data is in PRISM format, then the total number of records is determined from the file header. For both cases NREC is set equal to the total number of records - NEVL - STARTREC + 1. If both NREC and NEVL or STARTREC are specified, and if the sum is greater than the total number of records in the file, an error message is issued.

*** - If NUMXCOLS is not defined the program first checks to see if an INDEPENDENT list has been specified (i.e., an *alias* list for the independent variables). If so then NUMXCOLS is set equal to the number in the list. If neither have been specified then the default is one.

The parentheses for all parameters are optional. Thus in the Parameter file, the initial guess for the 2nd unknown can be specified as A0(2)=1 or A02=1. In addition, square brackets may also be used (e.g., A0[2]=1).

The DISPLAY parameter sets the level of output. If DISPLAY=0, then the program displays the minimum output (i.e., only the table of A(k)'s plus summary results). The table summarizes the data regarding the A(k)'s including the initial guesses, SIGA0(k) if any Bayesian estimators have been specified, AMIN(k), AMAX(k), the least square values of A(k) and SIGA(i). Setting DISPLAY=1 includes the table of A(k)'s and S / (n-p) from iteration to iteration. Setting DISPLAY=2 causes an additional output table: all N points including the values of X,Y, SIGY and YCALC. If DISPLAY is 3, then the results are almost the same as DISPLAY=2 except the column YCALC is replaced by Y - YCALC. If REL_ERROR='Y' then the relative error is also included (where REL_ERROR is defined as (Y – YCALC)/ SIGY). If DEBUG is specified as 1, then the program prints the matrix C and the vectors V and DEL_A from iteration to iteration. Higher levels of DEBUG are only meaningful for the program development team.

The parameters SYTYPE, SYTYPE(i), SXTYPE and SXTYPE(i) require further explanation. These parameters define the type of calculations used to determine the uncertainties (i.e., SIGMA) of the data. Alternatives to these parameters are SIGY, SIGY(i), SIGX and SIGX(i). The alternatives use characters rather than numbers for the input values: Z is zero, U is unit (i.e., one), C is constant. F is constant fraction and S is square root. The values of SIGMA are computed as follows:

TYPE	SIGMA Y or Y(i)	SIGMA X(i)
0 or Z	Read: from col SYCOL or SYCOL(i)	0.0
1 or U	1.0	1.0
2 or C	CY or CY(i)	CX(i)
3 or F	CY*Y or CY(i)*Y(i)	CX(i)*X(i)
4 or S	CY*sqrt(Y),CY(i)*sqrt(Y(i))	CX*sqrt(X(i))

If the user prefers to use the letter types (i.e., Z, U, C, F, or S) then the correct parameter names are SIGX, or SIGX(i) or SIGY. If the user specifies the function using T instead of X, the values of CX(i) may be replaced using the notation CT(i). If the values of SIGMA X(i) (or T(i)) are to be read from the data file, specify SXCOL(i) (or STCOL(i)). The SIGMA values are used to determine their weighting for each data point. If no information regarding the uncertainties of the data points is specified, the program defaults to unit weighting (i.e., SIGMA Y(i) = 1 and SIGMA X(i) = 0).

The rules for specifying the fitting function F are described in the Function Specification section. As an example of a parameter file (that also includes the data) consider the following file USERDOC.PAR:

```
!  Example 1: prepared by J. Wolberg, Oct 30, 2003
NCOL=3      ! note comments can be included on any line
A01 = 1     A02 = 1
A0(3)=0.5   AMIN3=-4     AMAX3=4
F='A1 + A2*EXP(A3*X1)*SIN(PI*X2)';
//start of data
1    0.5   10
2    0.9    5
2.5  1.5  -18
3    2.3   10
5    2.7  5.5
10   3.8   -8
```

```
12   5.0    2
15   7.0   -2;
//a new function
F='A1 + A2*EXP(A3*X1)*SIN(PI*X2+A4)'
```

Note that this example includes comments. The comments are denoted using either the exclamation mark ! or double slash //. The comment is terminated by a new line or end-of-file.

For this example, the columns have not been specified, so the defaults are used (X1 in column 1, X2 in column 2 and Y in column 3). Note that minimum and maximum values are specified for A3. The limits for this case assume that the search is for an exponent in the range -4 to 4. The number of data records N has not been specified so all data records are used. Note also that blanks around the = sign (e.g. A01 =) are optional as well as parens around subscripts (e.g., A0(3)).

Following the first semi-colon (after the function specification), the data is included and terminated by a semi-colon. A 2nd function is then included. This second function is treated as a separate case. The entire analysis is repeated using the same parameters but using the new specification of F and results are also included for this analysis on the same output file. The program pauses after the first analysis and issues a query if the user wishes to continue. If the response is Y (i.e., yes), and if there is additional data in the parameter file after a terminating semi- colon for the first analysis, the data is read from the parameter file. If there is no terminating semi-colon, or if there is no data beyond the semicolon, the program reads the response from the standard input device. The user may include any of the following parameters: A0, AMIN, AMAX, CAF, EPS, F and/or NUMITMAX after each semi-colon. The NUMITMAX parameter sets the maximum number of iterations used to determine values for the unknown parameters that satisfies the convergence criterion. If this number of iterations is completed without achieving convergence, the user is prompted to elect to continue or discontinue the search for an additional NUMITMAX iterations. This process can be continued indefinitely. The user can add as many separate cases as desired by adding a semi-colon after each case. Results for this analysis are as follows:

```
PARAMETERS USED IN REGRESS ANALYSIS: Sun Oct 16 13:14:02 2005
  INPUT PARMS FILE: userdoc.par
  INPUT DATA  FILE: userdoc.par
  REGRESS  VERSION: 4.13, Oct 16, 2005

  STARTREC - First record used            :     1
  N - Number of recs used to build model  :     8
  NO_DATA - Code for dependent variable    -999.0
  NCOL - Number of data columns           :     3
  NY   - Number of dependent variables    :     1
  YCOL1 - Column for dep var Y               :  3
  SYTYPE1 - Sigma type for Y              :     1
     TYPE 1:  SIGMA Y = 1
  M - Number of independent variables     :     2
  Column for X1                           :     1
  SXTYPE1 - Sigma type for X1             :     0
     TYPE 0:  SIGMA X1 = 0
  Column for X2                           :     2
  SXTYPE2 - Sigma type for X2             :     0
     TYPE 0:  SIGMA X2 = 0

Analysis for Set 1
  Function Y:  A1 + A2*EXP(A3*X1)*SIN(PI*X2)

  EPS - Convergence criterion           : 0.00100
  CAF - Convergence acceleration factor :   1.000
```

ITERATION	A1	A2	A3	S/(N.D.F.)
0	1.00000	1.00000	0.50000	1298.58
1	-1.44967	3.12800	0.19431	62.96751
2	-2.08590	10.59570	-0.10537	31.54384
3	-1.92748	14.20584	0.02789	17.39242
4	-1.96913	14.56456	-0.02408	8.74594
5	-1.92455	14.95635	-0.03989	8.50676
6	-1.91684	14.94462	-0.03986	8.50665

POINT	X1	X2	Y	SIGY	YCALC
1	1.00000	0.50000	10.00000	1.00000	12.44380
2	2.00000	0.90000	5.00000	1.00000	2.34740
3	2.50000	1.50000	-18.00000	1.00000	-15.44394
4	3.00000	2.30000	10.00000	1.00000	8.81082
5	5.00000	2.70000	5.50000	1.00000	7.98871
6	10.00000	3.80000	-8.00000	1.00000	-7.81308
7	12.00000	5.00000	2.00000	1.00000	-1.91685
8	15.00000	7.00000	-2.00000	1.00000	-1.91685

K	A0(K)	AMIN(K)	AMAX(K)	A(K)	SIGA(K)
1	1.00000	Not Spec	Not Spec	-1.91685	1.12568
2	1.00000	Not Spec	Not Spec	14.94470	3.11798
3	0.50000	-4.00000	4.00000	-0.03987	0.05615

Variance Reduction: 93.44
S/(N - P) : 8.50665
RMS (Y - Ycalc) : 2.30579

Analysis for Set 2
Function F: A1 + A2*EXP(A3*X1)*SIN(PI*X2+A4)

EPS - Convergence criterion : 0.00100
CAF - Convergence acceleration factor : 1.000

ITERATION	A1	A2	A3	A4	S/(N.D.F.)
0	1.00000	1.00000	0.50000	0.00000	1623.22
1	-1.50581	3.13829	0.19319	-0.00014684	78.84858
2	-2.80767	11.09816	-0.13283	-0.04783	45.07255
3	-2.35486	13.77303	-0.03629	-0.13474	10.57484
4	-2.32592	15.72227	-0.05815	-0.13219	9.33854
5	-2.28781	15.62974	-0.05450	-0.11938	9.32376
6	-2.29822	15.68605	-0.05567	-0.12233	9.32363
7	-2.29499	15.67068	-0.05535	-0.12148	9.32360
8	-2.29588	15.67502	-0.05544	-0.12172	9.32360

POINT	X1	X2	Y	SIGY	YCALC
1	1.00000	0.50000	10.00000	1.00000	12.42369
2	2.00000	0.90000	5.00000	1.00000	3.62687
3	2.50000	1.50000	-18.00000	1.00000	-15.84095
4	3.00000	2.30000	10.00000	1.00000	7.41656
5	5.00000	2.70000	5.50000	1.00000	8.09258
6	10.00000	3.80000	-8.00000	1.00000	-8.43410
7	12.00000	5.00000	2.00000	1.00000	-1.31742
8	15.00000	7.00000	-2.00000	1.00000	-1.46723

```
  K       A0(K)      AMIN(K)      AMAX(K)         A(K)      SIGA(K)
  1     1.00000    Not Spec     Not Spec     -2.29563      1.27148
  2     1.00000    Not Spec     Not Spec     15.67380      3.56094
  3     0.50000    -4.00000      4.00000     -0.05541      0.06273
  4     0.00000    Not Spec     Not Spec     -0.12165      0.17988

 Variance Reduction:          94.25
 S/(N - P)          :        9.32360
 RMS (Y - Ycalc)    :        2.15912
 Runs Test: Number of points much be >= 10 to perform test.
```

The following Parameter File (USERDOC2.PAR) is for a model with two dependent variables (Y1 and Y2) as functions of a single independent variable (X):

```
      !  Example 2: a 2 dimensional Y model, Oct 30, 2003
      NCOL=5  display=0  numitmax=100
      XCOL(1)=1  YCOL(1)=3    SYCOL(1)=4   YCOL(2)=5
      A01=1 A02=1 A03=1 A04=1
      AMIN2=0.01  AMAX2=4  AMIN4=0.01  AMAX4=4
      Y1 = ' A1*exp(-A2*X)'
      Y2 = ' A3*(1 - exp(-A2*X)*exp(-A4*X))' ;
      // start of data
      1     100   5007  34    42
      1.5   110   4532  30.5  103
      2.0   107   4117  28.6  150
      2.5   113   3760  25    202
      3.0   112   3303  23    267
      4.0   120   3027  21.5  265
      5.0   118   2786  20    215
      6.0   105   2515  19.2  183 ;
      Y1 = ' A1*exp(-A2*X) + A5';  // A 2nd function for Y1
```

In this example XCOL(1) is specified as 1. Columns 3 and 4 are YCOL(1) and SYCOL(1). Thus the uncertainties (sigma's) associated with the Y1 values in column 3 are specified in column 4. The values of Y2 are included in column 5 (i.e., YCOL(2)).). Neither SYCOL(2) nor SYTYPE(2) is specified, so the default values of SigmaY(2)=1 are used. Note that column 2 is not being used. After the data has been read and the analysis performed, Y1 is changed. Note, however, since Y2 is not changed, the original specification for Y2 is used in the second analysis.

Results for this example are as follows:

```
PARAMETERS USED IN REGRESS ANALYSIS: Sun Oct 16 13:40:44 2005

  INPUT PARMS FILE: userdoc2.par
  INPUT DATA  FILE: userdoc2.par
  REGRESS  VERSION: 4.13, Oct 16, 2005

  STARTREC - First record used            :     1
  N - Number of recs used to build model  :     8
  NO_DATA - Code for dependent variable    -999.0
  NCOL - Number of data columns           :     5
  NY   - Number of dependent variables    :     2
  YCOL1 - Column for dep var Y1              :  3
  YCOL2 - Column for dep var Y2              :  5
  SYCOL1 - Column for sigma Y1            :     4
  SYTYPE2 - Sigma type for Y2             :     1
     TYPE 1:  SIGMA Y2 = 1
  M - Number of independent variables     :     1
```

```
  Column for X1                              :     1
  SXTYPE1 - Sigma type for X1                :     0
     TYPE 0:  SIGMA X1 = 0

Analysis for Set 1
  Function Y1:   A1*EXP(-A2*X)
  Function Y2:   A3*(1 - EXP(-A2*X)*EXP(-A4*X))

  EPS - Convergence criterion           : 0.00100
  CAF - Convergence acceleration factor :   1.000


   K       A0(K)     AMIN(K)     AMAX(K)        A(K)     SIGA(K)
   1     1.00000    Not Spec    Not Spec     5420.99     1072.63
   2     1.00000     0.01000     4.00000     0.13771     0.05861
   3     1.00000    Not Spec    Not Spec   248.89815    37.81764
   4     1.00000     0.01000     4.00000     0.37870     0.20586

 Variance Reduction:         79.73 (Average)
    VR:           Y1         95.99
    VR:           Y2         63.47
 S/(N - P)         :       1317.42
 RMS (Y - Ycalc)   :     120.34751 (all data)
     RMS(Y1-Ycalc):     164.41612
 RMS((Y1-Ycalc)/Sy):       6.46067
     RMS(Y2-Ycalc):      43.98168
 Runs Test: Number of points much be >= 10 to perform test.


Analysis for Set 2
  Function Y1:   A1*EXP(-A2*X) + A5
  Function Y2:   A3*(1 - EXP(-A2*X)*EXP(-A4*X))

  EPS - Convergence criterion           : 0.00100
  CAF - Convergence acceleration factor :   1.000

   K       A0(K)     AMIN(K)     AMAX(K)        A(K)     SIGA(K)
   1     1.00000    Not Spec    Not Spec     4327.29     2321.36
   2     1.00000     0.01000     4.00000     0.41949     0.64363
   3     1.00000    Not Spec    Not Spec   248.92927    39.15811
   4     1.00000     0.01000     4.00000     0.09674     0.67525
   5     0.00000    Not Spec    Not Spec     2197.73     1709.70

 Variance Reduction:         81.49 (Average)
    VR:           Y1         99.50
    VR:           Y2         63.47
 S/(N - P)         :       1411.30
 RMS (Y - Ycalc)   :      51.33103 (all data)
     RMS(Y1-Ycalc):      57.75258
 RMS((Y1-Ycalc)/Sy):       2.48076
     RMS(Y2-Ycalc):      43.98168
 Runs Test: Number of points much be >= 10 to perform test.
```

5. FUNCTION SPECIFICATION:

The fitting function is specified in the parameter file as follows:

```
F = '...' or  Y = '...' or F1 = '...' or Y1 = '...'
```

There is no difference in the use of F or Y. It is a matter of user preference. If the model requires more than one function they must be numbered:

```
F1 = ' . . .'      F2 = ' . . . '           etc.
```

The blanks around the = sign are optional. The following operators can be used in F: +, -, *, / and ^. The ^ operator means raising to a power. Three types of parentheses may be used: ( ), { }, and [ ]. The program recognizes a mismatch if a left parens of one type is closed by a right parens of another type. The variables that may be included in F are: A1, A2, …, A20, Q1, Q2, …, Q9, T1, T2, … T9, X1, X2, … X9, Y1, Y2, ... Y9. The parameter specification is case insensitive, so lower case letters may also be used. The numbering of the unknown A's need not be inclusive but the X's must be numbered inclusively. For example, the program accepts the following function specification:

```
F='A1 - A3*X1'
```

but will not except:

```
F='A1 - A3*X2'
```

Note that blanks within the apostrophes are disregarded, so the user can add blanks for aesthetic purposes. If there is only one independent variable, then X or T can be substituted for X1 or T1. The user is free to choose X or T as the symbol denoting the independent variable (or variables). A variety of standard functions can be used in the specification of the function (or functions). These include ABS, ATAN, COS, COSH, EXP, LOG, LOG10, SIN, SINH, SQR, SQRT, and TAN. The program also recognizes the unary plus and minus if the first character in the function is + or - . The program recognizes PI as the constant 3.14159265. Standard precedence rules are used in **REGRESS**, but when in doubt, the user can add parens.

The following are valid specifications of F:

```
F = ' A1 + A2*(X^3)'
F = ' X1 / SINH(A2*[X1 - X2/A3])'
F = ' -X1/A1 - X2/A2 + {X1*X2}/A3'
F = ' a1 * exp(-a2*t) * sin(a3*t/pi) '
```

Note that **REGRESS** is insensitive to case in the function specification and indeed, in the entire parameter file.

REGRESS also includes the INT (integral) operator. This operator requires three parameters. For example:

```
F = ' A1 * INT(COS(LOG(X^2)), 0, X) + A2'
```

The first parameter (that is, COS(LOG(X^2))) is the integrand of the INT operation. The second parameter, in this example 0, is the lower limit of the integration and the third parameter, X, is the independent variable. The result of this INT operation is the integral of COS(LOG(X^2)) from 0 to X. The INT operator uses a numerical integration scheme based upon a 4th order Runge-Kuta step. The basic step size is dependent upon the input data but within each step a halving and doubling strategy is used until the desired accuracy criterion is achieved. The integration scheme attempts to satisfy an accuracy criterion (INTEPS) and failure is noted if this criterion is not satisfied. A message is then issued suggesting that the user might try increasing the MAXDEPTH and INTEPS parameters in order to satisfy the integration

accuracy criterion. The default values of MAXDEPTH and INTEPS are 10 and EPS. The default value of EPS is 0.001.

It should be emphasized that the user does not have to supply derivatives of F to **REGRESS**. The program performs symbolic differentiation to obtain all required derivatives. In addition, upon completion of an analysis, the user can change the function without having to re-edit the parameter file.

An additional feature of **REGRESS** is the ability to specify a vector rather than a scalar for the dependent variable Y. For such cases the user must specify F(i) for all the terms in the Y vector. For example, assume that for each X the dependent variable is a three-element vector. We might choose the following model:

```
F1 = ' A1* exp(-A2*x)'
F2 = ' A3*(1-exp(-A2*x)*exp(-A4*x) + Y1*x'
F3 = ' A1*A2*x + A3*A4*x'
```

Fi can be specified in several different ways: Fi, F(i) or F[i]. F1 can also be specified as merely F (without a subscript). Clearly, for every data point for this case, there must be a value for X, Y1, Y2 and Y3. The program checks to see that the data specification agrees with the functional specification. An alternative is to use Y instead of F to specify the functions.

In Version 4 of REGRESS, the use of aliases for the parameters is permitted. For example, the following is a valid alternative to Y = 'A1 + A2 * X' :

```
dependent pressure;
independent temperature;
unknown alpha, beta;

pressure = alpha + beta * temperature;
```

The aliases are then used throughout the output report. See the Section on **ALIASES** for additional examples.

6. RECURSIVE MODELS:

A model is recursive if the functions defining the dependent variables Y(i) are interdependent. For example, the following model is recursive:

```
Y1 = 'A1*X*sqrt(Y2) + A2'
Y2 = 'A3*X*sqrt(Y1) + A4'
```

In other words, the value of Y2 is required to compute Y1, and Y1 is needed to compute Y2. If a model requires several functions, the recursive relationship may be subtler. For example:

```
Y1 = 'A1*X*sqrt(Y4+Y2)'
Y2 = 'A2*Y3 + Q1'
Y3 = 'A3 * INT(COSH(TAN(X)/LOG(X^2), 0, X)'
Y4 = 'Y1 + A4*Y2/X'
```

There is a recursive relationship between Y1, Y2 and Y4 but Y3 can be computed directly.

REGRESS first determines if the functions include recursive relationships and if so, the calculated values of those functions that are related recursively are determined iteratively. Solving a set of coupled non-linear equations is not trivial and sometimes the iterative process fails. If it fails for more than a third of the

data points, the program aborts with an appropriate error message. There are several parameters that can be changed which might enhance convergence. These include NUMRECIT (the maximum number of iterations tried before failure is noted), RECEPS (the recursive calculation convergence criterion), and RAF (the recursive acceleration factor). The calculated change per iteration for each of the Y's is multiplied by RAF, so by decreasing RAF from its default value of one, a non-converging model can sometimes be turned into one in which the convergence is successful.

The user should be aware that for some models (and/or) combinations of parameters, the iterative process will fail regardless of the choice of NUMRECIT, RECEPS and RAF. For example, consider the following deceptively simple recursive model:

```
F1 = 'A1*X*Y2 + A2'
F2 = 'A3*X*Y1 + A4'
```

If the choice of initial values of the A(K)'s is such that for any X the two curves in the plane Y1-Y2 do not intersect, the recursive calculation must fail. If, for example, we initially choose A01=1, A02=5, A03=2 and A04=10 the program will note failure immediately.

7. DIFFERENTIAL EQUATIONS:

REGRESS can be used to find the unknown parameters for models based upon initial value ordinary differential equations. However, the equations must first be recast into a form based upon the INT operator. For example, consider the following 2nd order differential equation and boundary conditions:

$$\frac{\partial^2 \boldsymbol{Y}}{\partial \boldsymbol{X}^2} = \boldsymbol{A}1 * \boldsymbol{X}^2 + \boldsymbol{A}2 * \boldsymbol{Y}^2$$

$$\boldsymbol{Y}=0.5 \quad \boldsymbol{\&} \quad \frac{\partial \boldsymbol{Y}}{\partial \boldsymbol{X}} = 0.1 \qquad \text{at } \boldsymbol{X}=2$$

The 2nd order differential equation is first expressed as 2 first order differential equations and then put into suitable integral forms. In the following equations, Y1 is the original Y variable and Y2 is its derivative with respect to X:

```
Y1 = ' INT(Y2, 2, X) + 0.5 '
Y2 = ' INT(A1*X^2 + A2*Y1^2), 2, X) + 0.1 '
```

If the data consists of only values of Y and X, then there are no data points associated with Y2. The program determines which of the Yi's are true data points and which are merely intermediate variables by considering the values of YCOL(i). For this problem, if we specify YCOL(2)=0 the program treats Y2 as an intermediate variable. If on the other hand, YCOL(2) is not zero, then for each value of X there are two data points: Y1 and Y2.

8. THE METHOD OF LEAST SQUARES:

The method of least squares is described in detail in the books listed in the Reference section of this document. The purpose of the method is to find values for the coefficients **A(k)** which minimize ***S***. For the simple case in which there is one dependent variable ***Y*** and the values of the independent variables **X(1), X(2), . . . X(M)** are all error free, ***S*** is defined as follows:

$$S = \sum_{i=1}^{n} W(i) * R(i)^2$$

where ***n*** is the number of data points and ***R(i)*** is called the residual at point ***i*** and is the difference between the actual value of ***Y*** at that point and the calculated value.

When there are several dependent variables (i.e., **Y1, Y2**, etc.), the theory is complicated and is best explained using matrix notation. For all cases, ***S*** is specified and the method of least-squares attempts to find the value of the unknown parameters (i.e., **A1, A2**, ...) which minimize ***S***. The search for the unknown **A(k)**'s is not always successful if the equation (or equations) specifying the **Y**'s are highly non-linear.

For cases in which the uncertainties associated with the **X(j)**'s are not negligible, the equation for ***S*** must also include the residuals associated with the **X(j)**'s. The **REGRESS** program treats the general case in which uncertainty may be specified in both the **X** and **Y** vectors.

The **F(i)**'s are the values of **F** computed using the functions specified in the parameter file. The values of the **A(k)**'s included in the function (or functions) are assumed to be the initial guesses **A0(k)**. The ***W(i)***'s are the "weights" given to each point and this subject is discussed in the section on **DATA WEIGHTING**. The solution is obtained by starting from initial guesses **A0(k)** and then determining **DEL_A(k)** by solving a set of **P** linear equations of the form:

C * DEL_A = V

where **P** is the number of A(k) coefficients. The new guesses for the next iteration are computed as follows:

A0(k) = A0(k) + DEL_A(k) k = 1 to P

This procedure is continued until all the values of **DEL_A(k) / A0(k)** are less than **EPS** (which by default is 0.001). Actually, **REGRESS** uses a modified form of the previous equation:

A0(k) = A0(k) + CAF * DEL_A(k) k = 1 to P

where **CAF** is the Convergence-Acceleration-Factor. The default value of **CAF** is one, but it is useful to reduce **CAF** for problems in which the user experiences difficulty in obtaining convergence. Another device used by the program is to set limits on the values of **A0(k)** (i.e., **AMIN(k)** and **AMAX(k)**). Judicious use of these limits can also increase the probability of convergence. The resulting **A(k)**'s are the values of **A0(k)** after the program has converged to a solution.

When the program detects that after the **A0(k)**'s are changed ***S*** increases, a special procedure is followed in an attempt to increase the probability and speed of convergence to a solution. See the section on **CONVERGENCE** for a discussion of the algorithm upon which this procedure is based.

The terms of the C matrix are computed as follows:

$$C(j,k) = \sum_{i=1}^{n} W(i) * \frac{\partial F}{\partial A(j)} * \frac{\partial F}{\partial A(k)}$$

The terms of the V vector are computed as follows:

$$V(j) = \sum_{i=1}^{n} W(i) * \frac{\partial F}{\partial A(j)} * (Y(i) - F(i))$$

In these equations **Y(i)** represents the actual value of **Y** and **F(i)** is the calculated value. For the more complicated cases in which there are more than one dependent variable and the **X** (or **X**'s) include uncertainty, **W(i)** is actually a matrix with dimensions **NY** by **NY** (where **NY** is the number of dependent variables (i.e., **Y**'s).

An added feature of the method of least squares is that it yields unbiased estimates of the standard deviations of **A(k)** (i.e., **σ(k**)). These values are computed as follows:

$$\sigma(k) = \sqrt{\frac{S}{(n-p)} * CINV(k,k)}$$

where ***n*** is the number of data points and **CINV(k,k)** is the term (k,k) of the inverse matrix of **C** and ***p*** is the number of unknown parameters **A(k)**.

The method also yields unbiased estimates of **SIGYCALC** which is the standard deviation of the calculated value of the function **F** at any point in the **X** space.

9. CONVERGENCE PROBLEMS AND ADVICE:

The **REGRESS** program uses iterative methods for seeking convergence in three separate areas:

1) Searching for the values of **A(k)** which are the least-squares solution.

2) Determining the calculated values of **Y** for recursive problems. For details, see the section on **RECURSION**.

3) Determining the values associated with the **INT** operator. The **INT** operator performs numerical integration. For details, see the discussion of the **INT** operator in the section on **FUNCTION SPECIFICATION**.

The following discussion is limited to the search for the least squares values of **A(k).** The algorithm used to change the **A0(k)**'s from iteration to iteration was discussed briefly in the section on the **LEAST SQUARES METHOD**. The following is a more detailed description of the algorithm:

1) After computation of the **DEL_A(k)**'s, the initial guesses **A0(k)** are recomputed using the following equation: `A0(k) = A0(k) + CAF * DEL_A(k)  k = 1 to P`

2) When the new value of **S** is computed using the new values of **A0(k),** they are accepted as long as **Snew < 0.2 * S**.

3) If the value of **Snew** if between **0.2*S** and **0.99*S** then **DEL_A** is multiplied by a constant factor and the calculation is repeated as long as there is improvement in **Snew** or the multiplication constant does not exceeds a maximum value.

4) If **Snew** exceeds **0.99 * S**, the Marquardt algorithm is applied. This algorithm is explained in my book (***Data Analysis Using the Method of Least Squares***) and changes not only the magnitude of the **DEL_A** vector but also its direction.

When a particular case doesn't converge regardless of the choice of parameters, the trouble may be fundamental. The following discussion describes some of the reasons why some problems are difficult to solve due to an inability to achieve convergence:

For both linear and non-linear problems, the source of trouble might be in either the data or the choice of the function. There must be sufficient "variety" in the data so that the C matrix (see the section on the **LEAST SQUARES METHOD**) is not ill-conditioned. Similarly, a bad choice for the function can also lead to an ill-conditioned or even a singular C matrix. If the parameter **CONDITION** is specified then the condition number of the C matrix is included in the program output. Convergence is rarely a problem for linear functions. If however, a linear problem fails to converge, there are several possible reasons for the trouble:

1) Values of AMIN, AMAX and/or CAF are still in effect from a previous function. If the solution lies outside the limits set by AMIN and AMAX the program will not converge. If the function is linear and CAF is one, the solution is obtained on the first iteration. However if CAF is not one, the program might never reach the solution.

2) A linear function has been specified but it results in a singular matrix. For example, F='A1+2*A2+A3*X1' will cause a singular matrix because the ratio of the derivatives dF/dA1 and dF/dA2 is constant for all points.

3) The following deceptive example is a variation of (2). Consider the function F='A1+A2*X1+A3*X2'. It looks as though the ratio of the derivatives will vary from point to point, however if all the data points have a single value of X1 or X2, then we revert to a singular matrix.

4) If the function leads to a C matrix that is so ill-conditioned, then numerical problems can be the cause of the lack of convergence. This can happen when higher-order polynomials are used to fit the data.

Convergence for non-linear functions is often a problem but there are a number of techniques useful for enhancing convergence:

1) The most obvious technique is to allow additional iterations. One can usually see if the value of S/(N-P) is decreasing from iteration to iteration. If so, just respond Y to the "Continue?" query after the program issues the "Fails to converge" message.

2) Attempt to use better values for A0(k). The closer the A0's are to the least square solution, the greater the probability of convergence.

3) Use AMIN(k) and AMAX(k) to limit the search to a reasonable range of values. In particular, use these limits to avoid regions that will cause problems such as negative values of a LOG or SQRT, or overflows and underflows.

4) Reduce CAF (the convergence acceleration factor) to a value below the default value of 1. This is often sufficient to turn a diverging problem into a converging problem.

5) Perhaps the convergence criterion EPS is too "tight". Try increasing EPS above its default value of 0.001.

6) Try using another function, preferably a function with fewer unknown A(k)'s or perhaps a more linear function. This alternative might not be feasible if the entire purpose of the analysis is to determine parameters of a specific model.

7) If all else fails, turn the non-linear problem into a series of linear problems by changing an A into a Q and then repeat the analysis for a range of values of Q. Once you determine the best region

for the Q, you might try changing it back to an A but use your new knowledge to set A0 and the range AMIN and AMAX for this A term. The fact that an A is treated as a Q will affect the SIGMA calculations.

.

10. ILL-CONDITIONED AND SINGULAR MATRICES:

An ill-conditioned or singular matrix condition refers to the C matrix (see the section on the Least Squares Method). These conditions can be the result of several different causes:

1) The number of different values of the independent variable must be at least as large as the number of unknown variables. An example of a case that can cause problems is the following:

X	Y
0.5	13.2
0.5	15.3
1.0	18.2
1.0	20.1

If we attempt to fit a simple parabola (F=' A1 + A2*X1 + A3*X1^2 ') to this data, we will get the Singular Matrix message. However, if the fit is to a straight line (F=' A1 + A2*X1 ') we will get a least squares solution.

2) The function must be such that the partial derivations dF/dA(k) are independent. In other words, we do not want a function for which one partial derivative is a linear combination of any of the others. The following example is a function that will lead to the Singular Matrix condition regardless of the data:

```
F = 'A1 + 2*A2 +X1*A3'
```

The partial derivative dF/dA2 = 2 which is 2 * dF/dA1. Sometimes the combination of data and function can cause the problem:

X1	X2	Y
1.3	0.7	-3.4
1.3	0.9	0.2
1.3	1.6	4.1
1.3	2.4	5.7

This data will cause an ill-conditioned or singular matrix condition if we try to fit it using the following function:

```
F = ' A1 + A2*X1 + A2*X2 '
```

The ratio of the derivatives dF/dA2 and dF/dA1 is X1 but for this data this ratio is constant for all points!

3) The initial values used for the unknowns can cause matrix problems. For example, consider the following function:

```
F = ' A1 * EXP(A2*X1)'
```

The relevant derivatives are:

```
dF/dA1 = EXP(A2*X1)
dF/dA2 = X1 * A1 * EXP(A2*X1)
```

We see that the choice of A0(1)=0 will cause df/dA2 to be zero at all points and so we will end up with a singular C matrix. Since the user does not have to supply the derivatives to **REGRESS**, this type of error might not always be obvious. It is a good idea to avoid initial guesses of zero if the function is non-linear.

Matrices that are close to being ill-conditioned can cause problems. If the CONDITION NUMBER of the C matrix is large, then the results can be very sensitive to small changes in the data. The log10 of CONDITION NUMBER is an estimate of the number of decimal digits of accuracy that are lost due to rounding errors in the numerical calculations associated with obtaining the least squares solution. For nonlinear problems a high CONDITION NUMBER often leads to difficulties in achieving convergence to a solution.

11. INTERPOLATION TABLE:

One of the purposes of least squares analysis is the generation of a table of results. Such tables can be used for interpolation. **REGRESS** allows generation of interpolation tables by specifying three parameters for each independent variable: NP(i), X0(i) and DX(i). These are the Number of Points for X(i), the starting value and the change from point to point in the table. For example, assume that the least squares equation has a single independent variable and we have specified the following:

NP=5 X0=1 DX=0.5

The program will generate the following table:

POINT	X1	YCALC	SIGYCALC
1	1.0000	…	…
2	1.5000	…	…
3	2.0000	…	…
4	2.5000	…	…
5	3.0000	…	…

The values of **YCALC** and **SIGYCALC** are computed using the least squares equation generated with the input data. If the equation includes two independent variables we can specify a table as shown with the following example:

```
NP1=3   X01=5   DX1=2.5        NP2=2   X02=-2   DX2=3
```

These parameters specify a table that includes all combinations of 3 values of X1 and two values of X2. The program will generate the following table:

POINT	X1	X2	YCALC	SIGYCALC
1	5.0000	-2.0000	…	…
2	5.0000	1.0000	…	…
3	7.5000	-2.0000	…	…
4	7.5000	1.0000	…	…
5	10.0000	-2.0000	…	…
6	10.0000	1.0000	…	…

The parameters NPi, X0i and DXi can also be used to specify the values of Xi for prediction analyses. If all the M sets of these 3 parameters are specified then an interpolation table will be included in the prediction analysis output.

12. BAYESIAN ESTIMATORS:

A modification to the method of least squares is to consider the initial guesses A0(k) as additional data points. This has several advantages:

1) Previous knowledge of a parameter is used to obtain a "better" value.

2) The total number of data points (including the Bayesian estimates of the A(k)'s will exceed the number of unknown parameters if all parameters are estimated. This is true even if only one regular data point (i.e., Y(1), X(1,1), . . . X(m,1)) is available.

3) Use of Bayesian estimators makes the C matrix more diagonally dominant and thus enhances convergence.

An initial guess A0(k) becomes a Bayesian estimator of A(k) if the parameter SIGA0(k) is specified. The equation for S is modified by the addition of a summation term:

$$S = \sum_{i=1}^{n} W(i) * R(i)^2 + \sum_{k=1}^{p} \left(\frac{(A(k) - A0(k)}{SIGA0(k)} \right)^2$$

If SIGA0(k) is not specified, **REGRESS** assumes that A0(k) is not a Bayesian estimator. Thus if no SIGA0(k)'s are specified, the summation term on *p* is zero.

13. DATA WEIGHTING:

Data weighting is important in least squares if there is a significant difference in the uncertainties associated with the various data points. We define SIGY(l,i) as the standard deviation associated with the i-th value of Y(l) and SIGX(j,i) as the standard deviation associated with the i-th value of X(j). If we have no knowledge of the SIG values, we can choose the default values of SIGY=1 and SIGX=0. We call this choice "unit weighting" and each point is weighted equally. For the general case of a scalar Y, the program gives the following weight to each point:

$$W(i) = \frac{1}{SIGY(i)^2 + \sum_{j=1}^{m} \{ SIGX(j,i) * \frac{dF}{dX(j)} \}^2}$$

However, for the most general case (several Y(l)'s and non-zero values of SIGX(j,i)), W(i) is a matrix. We see for the case of unit weighting, W(i)=1 for all points. For all cases but unit weighting, this weighting function ensures that data that is more accurate is given greater weight than less accurate data. This method of weighting is called "statistical weighting". Using this equation it is possible to have an infinite weight. For example, this can happen if SIGX(j,i)=0 and SIGY(i)=CY*Y(i) and the value of Y at the i-th point is zero. If the weight for one or more of the data points is infinite, the C matrix will be singular. To avoid this problem, REGRESS disregards all points in which W(i) is infinite by assigning them weights of 0.

REGRESS has a number of methods for specifying SIGY and SIGX. See the **PARAMETER FILE** section for details.

14. VARIANCE REDUCTION:

Variance reduction (VR) is a measure of the "power" of a model. How much of the variance in the Y values of the data is explained by the model? The equation for VR is:

$$VR = 100 * \left(1 - \frac{\sum_{i=1}^{n} (Y(i) - Y\_calc(i))^2}{\sum_{i=1}^{n} (Y(i) - Y\_avg)^2}\right)$$

Since the method of least squares minimizes the weighted sum of all the $(Y - Y\_calc)^2$ values, we expect a large VR for the data used to determine the model. However, if we use the "Evaluation" option in **REGRESS** (by specifying the parameter NEVL), we also obtain Variance Reduction for an independent data set. This measure of VR can help us determine the value of the model (i.e., the fitted function) as a tool for predicting Y as a function of X (or X's). See the section on the **EVALUATION DATA SET**.

15. EVALUATION DATA SET:

If NEVL is specified, **REGRESS** reads a total of NREC + NEVL records from the data file. If NREC is not specified, then NREC is set equal to the total number of records in the data file minus NEVL - STARTREC + 1. Several parameters are available for selecting which records are used for modeling and which are used for testing: STARTEVAL, MODEL_FIRST and GROUP. If none of these are specified, then the first NREC records are used to determine the model (i.e., the function f(X1,..Xm) fitted to the data), and the next NEVL records are used to evaluate the model. If only the parameter STARTEVAL is specified, then the NEVL records are taken from this point. If only MODEL_FIRST is specified as 'N' then STARTEVAL is set to STARTREC. If GROUP is specified, then the data is put into the modeling and evaluations sets by groups. For example, if GROUP=5 and MODEL_FIRST is 'Y' and STARTREC=6, then records 6 thru 10 are put into the modeling set, 11 thru 15 into the evaluation set, 16 thru 20 into the modeling set, etc. The evaluation report includes the Variance Reduction (VR), the Root Mean Square (RMS) value for Y - Ycalc and the Fraction Same Sign (FSS) in the Evaluation Data set. VR is defined in a separate section. FSS is the fraction of the NEVL records in which the sign of Y and Y_calc are the same. Clearly this measure is only relevant if the Y data includes both positive and negative values. In addition, for cases in which all the values of Y do not have the same sign, results for a significance test for FSS is included. For cases in which more than one Y is specified, the values of VR and RMS are included for each Y.

16. PREDICTION ANALYSIS:

Prediction Analysis is a method of predicting the results of a least squares analysis. More specifically, the values of SIGA(K) and SIGYCALC are predicted. The method is described in **Data Analysis using the Method of Least Squares,** by J. R. Wolberg (see the section on References). I am in the process of writing a new book to be called **Prediction Analysis: A method for Designing Quantitative Experiments.** This book will be published by Springer in 2010. To specify a Prediction Analysis set MODE='P' in the parameter file. The program then assumes that the ratio S/(N-P) or S/(N+NBAYES-P) is equal to one. The resulting values of SIGA(K) and SIGYCALC are the predicted values. In the tables in which these values appear, the heading are changed to PRED_SA(K) and PRED_SIGY.

The values of X (or Xi if there are more than 1 independent variable) can be inputted in the standard manner (i.e., using XCOLi) however for prediction analyses several alternative methods are available. If NCOL is not specified and MODE='P', **REGRESS** computes the values of X using the interpolation table parameters (i.e., X0i, NPi, and DXi). For MODE='P' the values of Y are always computed using the values of X and the values of the initial guesses A01, . . . A0p. For example, assume the function F is A1*EXP(A2*X), X01=1, NP1=5, DX1=0.5, A01=10 and A02=-1. If MODE='P' and NCOL is not specified, the program assumes that NREC equal 5 and that the five values are 1, 1.5, 2, 2.5 and 3. The five corresponding values of Y are 3.679 (i.e., 10*exp (-1)), 2.231, 1.353, 0.821 and 0.498.

Two alternative methods of specifying the values of X are available: if GRNUM=1 (or GRNUMi=1) then the values of Xi are computed from a normal distribution with mean of PARM1i and standard deviation of PARM2i. If RNUM=1 (or RNUMi=1) then the values of Xi are computed for a random distribution in the range from PARM1i to PARM1i+PARM2i. If one is interested in an interpolation table, then the values of X0i, NPi and DXi must also be specified for this independent variable. If NCOL is specified then for all columns in which GRNUM or RNUM are not specified, NREC records are taken from the NCOL columns of data. If NREC is larger than the number of records, then an error message is issued. If NCOL is not specified, then for all columns in which GRNUM or RNUM are not specified, the values of X0i, NPi and DXi must be specified. Unless the parameter SEED is specified, the random numbers that are generated will be the same for all runs.

17. ALIASES

To make the **REGRESS** output easier to interpret, the "alias" concept was introduced in Version 4.01 of the program. Alias names can be used in place of the dependent variable (F or Y) or variables (F1, Y1,…), the independent variable (X or T) or variables (X1m T1,…), the unknowns (A1, A2, . . .) and the symbolic constants (Q1, Q2,…). Aliases can also be used for cases in which there are multiple dependent and/or independent variables. Consider the following example in which aliases are not used:

```
Y = 'A1 + A2*T'
```

This example could be respecified as follows:

```
independent temperature;
dependent pressure;
unknown alpha, beta;

pressure = alpha + beta * temperature;
```

These specifications must precede the remainder of the parameter file. The aliases will appear throughout the output report in place of Y, A1, A2 and T. If we wished to treat alpha as a known constant rather than as an unknown, we could alter the specifications as follows:

```
independent temperature;
dependent pressure;
unknown beta;
constant alpha;
```

Clearly the value of alpha would have to be included in the parameter list. .The following is an example in which there are two dependent variables (GROWTH and PRESSURE), one dependent variable (TEMPERATURE) and four unknown parameters. The parameter file for this example is as follows:

```
// a two dependent variable example using aliases
      dependent growth, pressure;
      independent temperature;
      unknown growthcoeff, pressure_c1, pressure_c2, p_c3;

      growth = 'growthcoeff + pressure_c1 * pressure'
      pressure = 'pressure_c2 * int(temperature ^ 0.5, 0, temperature) + p_c3'
      a0[2] = 2.4  pressure_c20 = 3  growthcoeff = 1.2
      pressure_c2min=-1000
      growthcoeffmax = 10000   ! max value
      pressure_c2sig = 0.5     ! Bayesian estimator
      ncol=3  xcol = 1  ycol1=2  ycol2=3;
//    temperature  growth  pressure
              1.0     2.0       3.0
              1.1     2.2       3.7
              1.2     2.4       3.9
              1.3     2.7       4.8
              1.4     3.0       6.0
              1.5     3.5       9.0
```

Several points should be noted in this example:

1) Initial values can be specified by setting a value to the alias (e.g., growthcoeff=1.2), by appending a 0 to the alias (e.g., pressure_c20=3), or by using the original mode of specification (e.g., a0[2]=2.4). The specification a0[2]=2.4 or a0(2)=2.4 or a02=2.4 means that the initial value of the 2nd unknown (i.e., pressure_c1) is 2.4. For this example, there is no initial value specification for the 4th unknown so it is set to the default value of 0.

2) Min and max value of the unknown can be specified by appending min or max to the relevant alias (e.g., growthcoeffmax=10000). Alternatively this max value could have been specified in the usual manner (i.e., AMAX(1)=10000).

3) Bayesian estimators can be specified by appending sig to the alias (e.g., pressure_c2sig=0.5). The alternative specification without use of the alias would be siga0(3)=0.5.

The output for this example is as follows:

```
PARAMETERS USED IN REGRESS ANALYSIS: Tue Nov 27 12:08:55 2001

  INPUT PARMS FILE: growth.par
  INPUT DATA  FILE: growth.par
  REGRESS  VERSION: 4.02, Nov 25, 2001

  STARTREC - First record used            :     1
  N - Number of recs used to build model  :     6
  NO_DATA - Code for dependent variable    -999.0
  NCOL - Number of data columns           :     3
  NY   - Number of dependent variables    :     2
  YCOL1 - Column for dependent variable 1 :     2
  YCOL2 - Column for dependent variable 2 :     3
  SYTYPE1 - Sigma type for Y1             :     1
     TYPE 1:  SIGMA Y1 = 1
  SYTYPE2 - Sigma type for Y2             :     1
     TYPE 1:  SIGMA Y2 = 1
  M - Number of independent variables     :     1
  XCOL1 - Column for X1                   :     1
  SXTYPE1 - Sigma type for X1             :     0
     TYPE 0:  SIGMA X1 = 0
  MAXDEPTH - Max depth in INT scheme      :    10
```

```
  INTEPS - Integration converge criterion 0.00100

Analysis for Set 1
  Function GROWTH:  GROWTHCOEFF + PRESSURE_C1 * PRESSURE
  Function PRESSURE:  PRESSURE_C2 * INT(TEMPERATURE ^ 0.5, 0, TEMPERATURE) + P_C3

  EPS - Convergence criterion           : 0.00100
  CAF - Convergence acceleration factor :   1.000

 ITERATION GROWTHCOEFF PRESSURE_C1 PRESSURE_C2        P_C3  S/(N.D.F.)
         0     1.20000     2.40000     3.00000     0.00000     7.70317
         1    -1.11561     1.57278     3.29712     0.40361     3.67504
         2    -2.52621     0.76947     3.35041     1.92323     2.06379
         3    -1.33007     0.78225     3.35041     1.92323     1.39801

 REC Y-INDEXTEMPERATURE      GROWTH   SIGGROWTH  CALC_VALUE
   1       1    1.00000     2.00000     1.00000     1.92160
   2       1    1.10000     2.20000     1.00000     2.19014
   3       1    1.20000     2.40000     1.00000     2.47117
   4       1    1.30000     2.70000     1.00000     2.76417
   5       1    1.40000     3.00000     1.00000     3.06867
   6       1    1.50000     3.50000     1.00000     3.38425


 REC Y-INDEXTEMPERATURE    PRESSURE SIGPRESSURE  CALC_VALUE
   1       2    1.00000     3.00000     1.00000     4.15682
   2       2    1.10000     3.70000     1.00000     4.50010
   3       2    1.20000     3.90000     1.00000     4.85936
   4       2    1.30000     4.80000     1.00000     5.23393
   5       2    1.40000     6.00000     1.00000     5.62319
   6       2    1.50000     9.00000     1.00000     6.02661



       ALIAS  INIT_VALUE   SIG_BAYES     MINIMUM     MAXIMUM       VALUE       SIGMA
 GROWTHCOEFF     1.20000    Not Spec    Not Spec    10000.00    -1.33007     4.95826
 PRESSURE_C1     2.40000    Not Spec    Not Spec    Not Spec     0.78225     0.96824
 PRESSURE_C2     3.00000     0.50000    -1000.00    Not Spec     3.35041     0.57568
        P_C3     0.00000    Not Spec    Not Spec    Not Spec     1.92323     0.72438

 Variance Reduction:         73.67 (Average)
    VR:       GROWTH         97.81
    VR:     PRESSURE         49.53

 S/(N+NBAYES-P)    :       1.39801

 RMS (Y - Ycalc)   :       1.00431 (all data)
      RMS(Y1-Ycalc):       0.07477
      RMS(Y2-Ycalc):       1.41834
```

18. USING EXCEL DATA FILES

The **REGRESS** program can use data from files created with Excel. The file must be saved as a **Text file (Tab delimited)**. A file with extension **txt** is created unless otherwise specified. When Excel files are used, specify **ftype = 'e'** in the parameter file. If the file includes alpha-numeric columns, then as long as

these columns are not specified as data columns, the program will skip over them. For example, assume that the following file **xyz.txt** is created using Excel:

10	a b c	21
11.2	xyz	33
-5	###	100

The **REGRESS** parameter file might include the following: **xcol=1 ycol=3.** Only the first and third columns will be used as input data to the program. Note that **ncol** does not have to be specified for Excel files because the program counts the number of columns if Excel files are used.

19. THE RUNS TEST:

The runs test is only performed if there are at least 10 data points, a single independent variable and the values of the independent variable either increase or decrease monotonically. If there are more than one dependent variable then the runs test is performed on the residuals of each of the dependent variables separately. The test examines the "runs" in the residuals to test for randomness. If the proposed model is a reasonable representation of the data, the residuals should be randomly distributed about the calculated curve. The number of runs is the number of times the sign of the residual changes as the value of the independent variable increases. The number of runs is observed and a lower limit is computed based upon a 2.5% confidence level. If the number of runs is less than or equal to this limit then it can be concluded that there is a lack of randomness in the residuals. To see the residuals on the output table, set DISPLAY=3. The results are almost the same as the default value of DISPLAY=2 except the column YCALC is replaced by Y - YCALC. If REL_ERROR-'Y' then the relative error is also included (where REL_ERROR is defined as (Y – YCALC)/ SIGY). For a detailed explanation of the Runs test see Section 3.9 of my book [John Wolberg, *Data Analysis Using the Method of Least Squares*, Springer, 2006].

20. GRAPHICS INTERFACE:

The **REGRESS** program does not include graphics. The **REGRESS** program is written in standard C and is therefore highly portable. To enhance its portability, no attempt has been made to marry it to any particular graphics package. However, some users find it necessary to display their results graphically. This should be a fairly simple process because all output from **REGRESS** is saved in an ascii file. For example, if the input file is xyz.par, the output ascii file is xyz.out. It is left to the user to write an interface between the .out file and the graphics package of interest.

21. REFERENCES:

The primary reference for the REGRESS program is my book:

John R. Wolberg
Data Analysis Using the Method of Least Squares
Springer, 2006

The general method of least squares is also described in:

N. R. Draper & H. Smith
Applied Regression Analysis
Wiley & Sons, 1966

John R. Wolberg
Prediction Analysis
Van Nostrand – Reinhold, 1967

Peter Gans
Data Fitting in the Chemical Sciences
Wiley & Sons, 1992

A further reference on non-linear parameter estimation is:

Yonathan Bard
Non-linear Parameter Estimation
Academic Press, 1974

The method of least squares is described in many books on numerical methods but usually, the discussion is limited to single linear function. Typically the discussion is also limited to unit weighting.