

CFS_LS Reference Manual

by

Michael L. Johnson
Departments of Pharmacology and Internal Medicine
University of Virginia Health Sciences Center
Charlottesville, VA 22908

NON-PROFIT SOFTWARE LICENSE AGREEMENT

Licensor: Michael L. Johnson

Address : P.O. Box 260
Keswick, VA
22947

Software: Non-Linear Least-Squares program

Software is furnished to the Licensee free, or for a nominal charge, and may only be copied, in whole or in part, for use by the Licensee and his/her employees. The Licensee acknowledges that a copyright exist on this software. Licensee shall not provide or otherwise make available the software or any part or copies thereof in any form to any third party, except as may be permitted in writing by the Licensor. No title to or ownership of the software or any modified or unmodified parts is hereby transferred to the Licensee. The Licensor shall have the right to terminate the license if Licensee fails to comply with these license terms and Licensee agrees, upon notice of such termination, to return immediately or destroy the software and all portions and copies thereof.

The Licensee further agrees that this software will not be used for profit by anyone. This license is extended to allow corporate (*i.e.*, for profit) users to use the software for internal research purposes only. It does not allow resale of the software.

This software is supplied with no stated or implied warranties as to its functionality. The Licensor cannot be held responsible, or liable, for any damages, including but not limited to special, indirect, or consequential, arising out of, or in connection with, the use or performance of the software.

This agreement shall be governed by and interpreted according to the laws of the Commonwealth of Virginia.

NO WARRANTIES ARE EITHER EXPRESSED OR IMPLIED

TABLE OF CONTENTS

1.0 CFS_LS: Parameter Estimation Program	6
1.1 Tutorial: An Example of the Use of CFS_LS	8
1.2 CFS_LS Output Files	19
CFS_LS.PRN	19
CFS_LS.BST	19
CFS_LS.CMD	20
CFS_LS.GRF	21
CFS_LS.DBG	21
?????.TMP	21
1.3 Experimental Data File Formats	22
FD-CFS	22
IBH4-ASC	23
IBH4-ASC/B	24
IBH4-BIN	24
IBH4-BIN/B	24
TD-PRA	25
TD-PRA/B	26
FD-SPEX	26
1.4 Fitting Functions: Intensity Decay Law Functions	28
Time-Domain Transformations	28
Frequency-Domain Transformations	30
HETANL	36
GAUTAU	38
GAUDIS	40
LORTAU	42
LORDIS	44
RECTAU	46
RECDIS	48
SKUTAU	50
ET1DET	52
ET2DET	53
ET3DET	54
HET_JUMP2	55
1.5 Individual CFS_LS Commands	58
HELP	58
SHELL	58
QUIT	58
#	58
@	59
!	59
COOKIE	59

DATA	59
FUNCTION	59
GUESS	59
BROWSE	60
FIT	60
PLOT	60
SMOOTH	60
RESIDUALS	60
SIMULATE	60
GLOBAL	61
1.6 Minimal Computer System Requirements	62
IBM PC Clones	62
1.8 How to Report A Problem with the Programs	63
2.0 Numerical Methods	64
2.1 Parameter Estimation	66
Nelder-Mead method	67
2.2 Uniqueness of Parameters	69
2.3 Goodness-of-Fit Criterion	69
Scatter Diagram Residual Plots	69
Runs Test: Quantifying Trends in Residuals	70
Autocorrelation: Detecting Serial Correlation	72
Outliers: Identifying Bad Points	74
2.4 Determination of Confidence Intervals	75
Asymptotic Standard Errors	76
Monte-Carlo method	79
Bootstrap method	79
Grid search methods	80
Support Plane method	82
2.5 REFERENCES	83
3.0 VIEWGRAF: Examine and Print a Graph	85
4.0 MAKEFONT: Roll Your Own Character Font	87
6.0 INTERNAL FILE FORMATS	88
5.1 CFS_LS.INI : The initialization file	88
CMD_FILE	88
GRF_FILE	88
DISPLAY	89
HIGHLIGHT_BACKGROUND	89
HIGHLIGHT_FOREGROUND	89
NORMAL_BACKGROUND	89
NORMAL_FOREGROUND	89

MOUSE	89
PRN_FILE	89
PRINTER	89
PRN_CLASS	89
PLOTTER	90
WINDOW_INPUT	90
MODE10, MODE11, MODE12	90
5.2 ASCII graphics files	90
5.3 Software Character Set Source Data Files	91
7.0 ACKNOWLEDGMENTS	94
8.0 APPENDIX: Supported Hardware	95
TABLE I: Graphics Output File Types	95
TABLE II: IBM PC Clone VGA and SVGA Graphics Modes	95
9.0 INDEX	99

1.0 CFS_LS: Parameter Estimation Program

The CFS_LS program performs global nonlinear least-squares parameter estimations to determine fluorescence lifetimes from either frequency- or time-domain data. CFS_LS can also simultaneously analyze frequency- and time-domain data.

The CFS_LS software provides a user friendly windowing interface or the user can select a command line interface (with the TAB key). The windowing interface includes multiple windows such as a menu of available file names that is automatically generated if the user uses a file name that includes a wild card character. The user can generally use either the cursor keys or the mouse to navigate the menus. Most menus provide a dynamic help line which describes the currently highlighted option. At any time the user can obtain more detailed online help with the F1 key. Furthermore care has been taken to be sure that the program selects reasonable default answers so that the most common user response is simply to press the enter key.

When in the command line mode the user can bring back (with the up and down arrow keys) and edit the last few commands that were entered. This is particularly useful when entering several sequential nearly identical commands such as those encountered when multiple data sets with nearly identical file names are input for a global analysis.

The CFS_LS program can also be executed in "batch" mode. Furthermore at any time the user can execute a series of "batch" commands (*i.e.*, ASCII commands read from a .cmd file). When the CFS_LS program execute it automatically creates an

ASCII file, CFS_LS.CMD, that contains all of the commands used in the current session. The user can easily modify this file to repeat the analysis in a "batch" mode. Thus the CFS_LS program is user programmable.

A printer output file is created that contains the results of the analysis and publication quality graphics. The program currently supports 30 different types of printers and about 100 different types of displays and/or terminals. It can create graphics types HP-GL, Tektronix, Postscript, Encapsulated PostScript, PCX, BMP, and various type of raster graphics for printers by several manufacturers including Hewlett Packard and Epson.

In addition when CFS_LS generates a graph it also creates an ASCII graphics file (.grf) that can be viewed later with a secondary program, VIEWGRAF. The purpose of this is to allow the user to combined and/or edited these .grf files and thereby to create custom presentation quality figures with VIEWGRAF. Also, the .grf file is in ASCII format and contains both the experimental data and the calculated best fit curve so the .grf file can also be used to import the results of an analysis in other graphics packages such as SigmaPlot or Harvard Graphics.

Possibly the most serious shortcoming of most data analysis packages is that the user is not provided with realistic estimates of the precision or accuracy of the determined parameters. Most programs, including CFS_LS, provide the user with the asymptotic standard errors of the determined parameters. However, CFS_LS also provides the user with the option of determining the confidence intervals of the determined parameters by either a Monte-Carlo method or a Support Plane method.

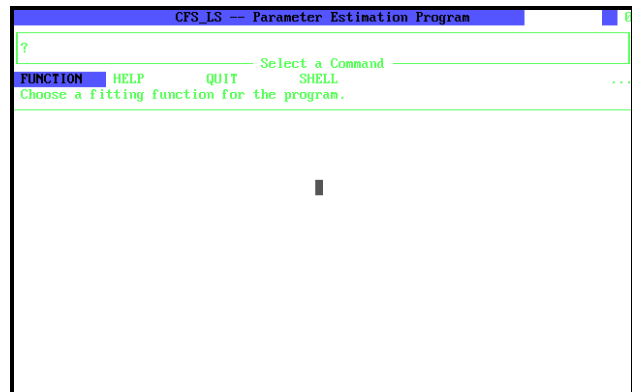
These two methods are superior to the conventional asymptotic standard errors because they consider the covariance between the parameters being estimated and the nonlinearity of the fitting equation. A graphical presentation is included for both the Monte-Carlo method and the Support Plane method (Chi-Squared plots).

The user is asked to choose from a menu of different fitting equations. The fitting equations are simply the intensity decay laws combined with the time-domain convolution integral or the frequency-domain sine and cosine transforms. After a parameter estimation has been completed the user has the option of performing a series of goodness-of-fit analyses on the residuals of the fit. These include autocorrelations, runs test, etc. These allows the user to ask the question, "Does the particular intensity decay law actually describe the experimental observations?" Or in other words to do hypothesis testing.

1.1 Tutorial: An Example of the Use of CFS_LS

The first step is, of course, to install the software. This procedure is covered in a later section called CFSSETUP.

Once the installation is complete the program can be started by simply entering **CFS_LS**. The program will then present the user with a screen that looks approximately like the figure to the right.



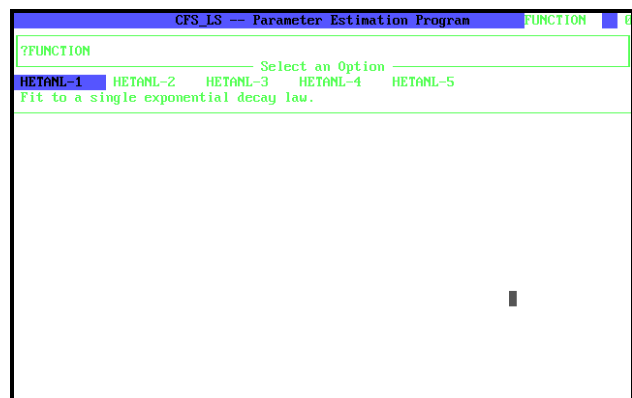
This is the general "what next" input screen for the program. At this stage the number of available options is quite low. Most probably the user will want to select the **FUNCTION** command. This command specifies that the user wishes to select a particular intensity decay law for the fitting process. Since the **FUNCTION** command is the default at this stage all that the user need do is press the **enter** key on the keyboard. The user can also select a command with the mouse. The square in the middle of the screen is actually the mouse cursor. The user can move to a different command with the left and right cursor keys or by entering the first letter of the command. The three dots . . . to the right of the fifth text line indicates that more options are available than are shown on the screen. This is an end-around menu so either selecting the three dots (. . .) with the mouse or the left and right cursor keys will move to other options.

For example, at this time the user could type **Q** to go to the **QUIT** command, the **enter** key, and then a **Y** to confirm and the program will terminate.

The user can also, at any time, press the **<F1>** key to obtain help. The **HELP** command will present a similar menu of the various items that help is available for.

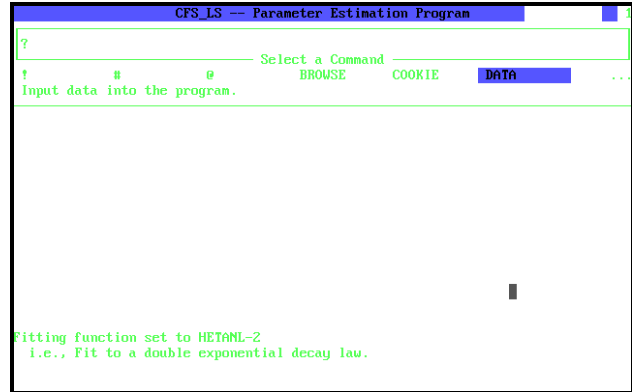
Also note that as the user moves from option to option a one line help message is updated with information about the current option.

After selecting the **FUNCTION** command the screen will look something

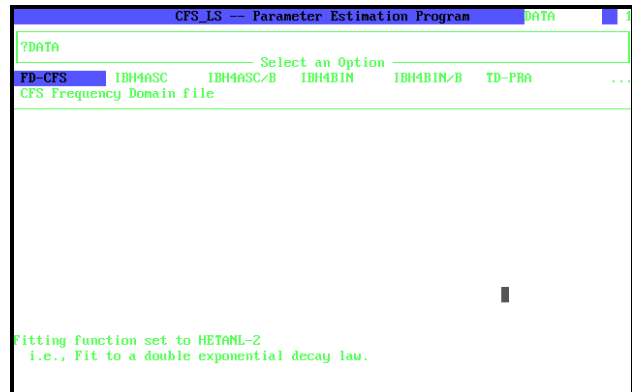


like the figure to the right. Move to the **HETANL-2** option with the right cursor key or the mouse and press the **enter** key.

The screen now looks like the image to the right. To accomplish a global fit simply execute the **DATA** multiple times. The default option is now the **DATA** command. Press the **enter** key.

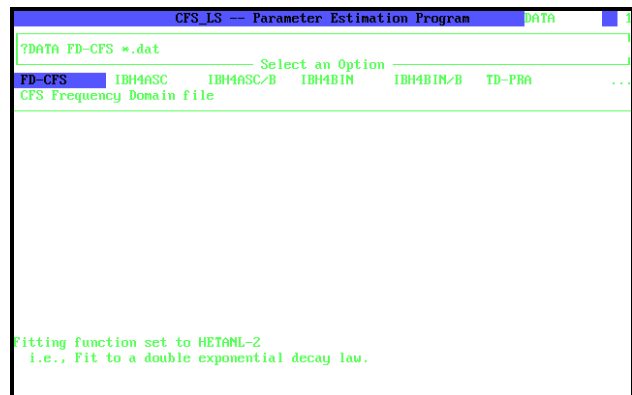


The screen will change to: Move to the **FD-CFS** option with the cursor keys or the mouse and press the **enter** key.



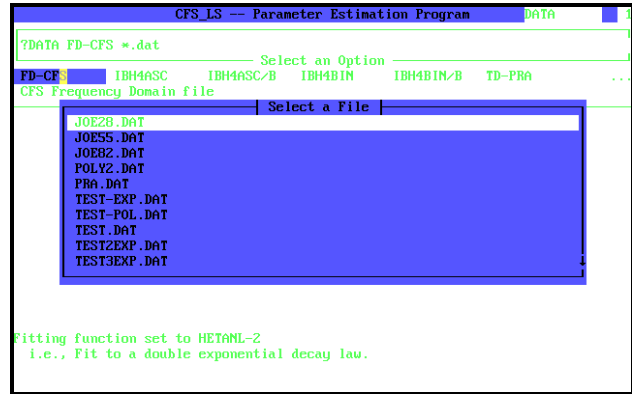
The screen now looks like:

Notice the third line of the text screen is **?DATA FD-CFS *.dat**. The previous entered responses are creating a command for the program. The general form for the **DATA** command is

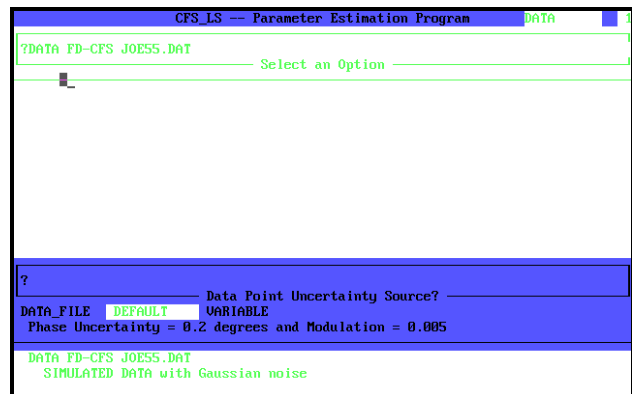


DATA <type> <file name>. Currently the file name contains a wild card character * so if the user presses the **enter** key a menu of available data files is opened. Press the **enter** key.

The screen now looks something like the figure to the right. Move to the **JOE55.DAT** line. Note the up and/or down arrows in the right margin of the menu. These indicate that the menu contains more file names either before or after those displayed on the screen.

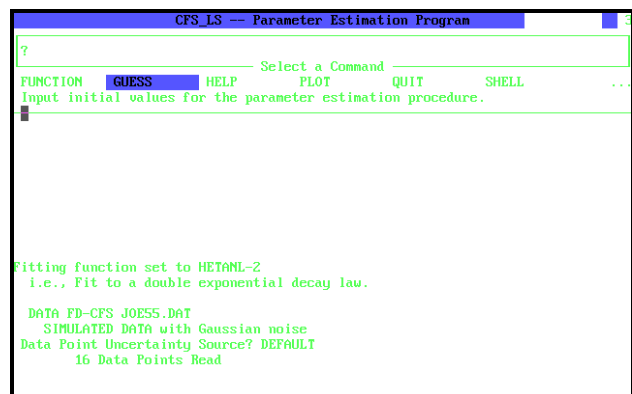


Press the **enter** key and the screen will change. This new menu wants you to specify if the individual data point standard errors are to come from the **DATA_FILE**, the **DEFAULT** values, or be specified by the user **VARIABLE**.



Select **DEFAULT** and press the **enter** key.

The display now looks something like that to the right. The next step is to enter the initial values for the parameters to be estimated by the least-squares



procedure. Select **GUESS** and press the **enter** key.

The display will change to:

You now need to specify the source for the initial values, *i.e.*, either the **KEYBOARD** or a data **FILE**. Select **KEYBOARD** and press the **enter** key.

```

CFS_LS -- Parameter Estimation Program  GUESS
?GUESS
----- Select an Option -----
KEYBOARD  FILE
Enter initial values from the keyboard.

Fitting function set to HETANL-2
i.e., Fit to a double exponential decay law.

DATA FD-CFS JOE55.DAT
SIMULATED DATA with Gaussian noise
Data Point Uncertainty Source? DEFAULT
16 Data Points Read
  
```

The display now looks like:

This menu requests the actual numerical values. You can edit the individual values (be sure to enter decimal points when needed). You can move between parameters with the **enter** key, the cursor keys, or the mouse. The menu may (but not in this simple example) show up and down arrows in the right margin indicating move values above or below those being displayed. The menu terminates when the user rolls off the bottom of the menu or selects **QUIT** with the mouse or by

```

CFS_LS -- Parameter Estimation Program  GUESS
?GUESS KEYBOARD
----- Select an Option -----

Edit Initial Values
[JOE55.DAT]TAU-1  ? 1.000000
[JOE55.DAT]AMPLITUDE-1  1.000000
[JOE55.DAT]TAU-2      2.000000
[JOE55.DAT]AMPLITUDE-2  1.000000
QUIT TOP BTM HELP UP DOWN
  
```

pressing the **Q** key. Press the **5** key, the **.** key and the **enter** key (for TAU-1). The **down cursor**, then **20**. (for TAU-2) and a **Q** key.

The display will now look like:

```

CFS_LS -- Parameter Estimation Program  GUESS
|BROWSE the Results of the Guess Command|

Resulting Sum of Squared Residuals = 27565.55
Current Number of Observations = 32

Corresponding Reduced Chi-Squared = 861.4236
Assuming no parameters being estimated.

Square Root of Reduced Chi-Squared = 29.35002

File Name = JOE55.DAT
SIMULATED DATA with Gaussian noise
Parameter Value
CONSTANT 0.0000000
Quit Help Top Btm Find Next Goto Print Line: 1
RESULTING SSR = 27565.55
  
```

This BROWSE menu allows the user to examine information such as the sum of the squared residuals and the current individual parameter values. Note that the right and bottom margins of the menu may contain arrows indicating more information is available in this directions. Use the cursor keys to examine the additional information. If you wish this information to appear in the output printer file press the **P** key. When finished examining the information press the **Q** key.

The screen now looks like :

The default option is now **FIT** indicating that the next step is to perform a nonlinear least-squares parameter estimation. Press the **enter** key.

```

CFS_LS -- Parameter Estimation Program
?
----- Select a Command -----
FIT  FUNCTION  GUESS  HELP  PLOT  QUIT  ...
Fit the function to the data.

Fitting function set to HETANL-2
i.e., Fit to a double exponential decay law.

DATA FD-CFS JOE55.DAT
SIMULATED DATA with Gaussian noise
Data Point Uncertainty Source? DEFAULT
16 Data Points Read

RESULTING SSR = 27565.55

```

The display now looks like:

The user is being requested to select the numerical method for the parameter estimation. Select **NM-Simplex** (for the Nelder-Mead Simplex algorithm) and press the **enter** key.

```

CFS_LS -- Parameter Estimation Program  FIT
?FIT
----- Select an Option -----
Gradient  NM-Simplex  Grid
Parameter Estimation using a Nelder-Mead Simplex Method.

Fitting function set to HETANL-2
i.e., Fit to a double exponential decay law.

DATA FD-CFS JOE55.DAT
SIMULATED DATA with Gaussian noise
Data Point Uncertainty Source? DEFAULT
16 Data Points Read

RESULTING SSR = 27565.55

```

The screen changes to:

The user is now being prompted for the method to be used to evaluate the confidence intervals of the estimated parameters. Select the **SuprtPlane** option to obtain confidence intervals by the support plane algorithm.

```
CFS_LS -- Parameter Estimation Program FIT
?FIT NM-Simplex
----- Select an Option -----
None      Grid      MonteCarlo  SuprtPlane  AsymStdErr
Confidence intervals by a Support Plane Method.

Fitting function set to HETANL-2
i.e., Fit to a double exponential decay law.

DATA FD-CFS JOE55.DAT
SIMULATED DATA with Gaussian noise
Data Point Uncertainty Source? DEFAULT
16 Data Points Read

RESULTING SSR = 27565.55
```

The screen will change to:

The user is now being requested to specify which parameters are to be estimated. Proper responses are either **Y**, **N**, **UP** and **DOWN** cursor, the mouse, etc. For the present example enter three **Y** keys. The program now performs the least squares parameter estimation (in may take a few moments).

```
CFS_LS -- Parameter Estimation Program FIT
?FIT NM-Simplex SuprtPlane
----- Select an Option -----

Estimate Parameter
IJOE55_DATITAU-1  ? NO
IJOE55_DATITAU-2  NO
IJOE55_DATIAMPLITUDE-2  NO
QUIT TOP BTM HELP
```

The screen should now look approximately like:

The user is now being ask which of the parameter confidence intervals are to be estimated by the support plane method. Proper responses are either **Y**, **N**, **UP** and **DOWN** cursor, the mouse, etc. Press three **Y** keys.

```
CFS_LS -- Parameter Estimation Program FIT
?FIT NM-Simplex SuprtPlane
----- Select an Option -----

Support Plane Parameter
IJOE55_DATITAU-1  ? YES
IJOE55_DATITAU-2  YES
IJOE55_DATIAMPLITUDE-2  YES
QUIT TOP BTM HELP
```

The screen now changes to another BROWSE screen containing the answers from the parameter estimation and the confidence interval evaluation. Move around with the cursor keys, print it if so desired (**P** key), and then press the

```

CFS_LS -- Parameter Estimation Program
|BROWSE the Answers|
RESULTS OF PARAMETER ESTIMATION
CONFIDENCE PROBABILITY = 0.6826
CONSTANT = 0.000000
DELTA-TIME = 0.000000
TAU-1 = 4.95868 { 4.864216 ,5.047520 }
              { -9.1652254E-02 ,9.1652254E-02 } -/+ diff
              { -1.85% ,1.85% } -/+ %
AMPLITUDE-1 = 1.000000
FRACT-AMPL-1 = 0.7992178 { 0.7882339 ,0.8098669 }
                   { -1.0983833E-02 ,1.0649189E-02 } -/+ diff
                   { -1.37% ,1.33% } -/+ %
SS-INTENS-1 = 3.960818 { 3.842139 ,4.079844 }
                   { -0.1186787 ,0.1190262 } -/+ diff
                   { -3.00% ,3.01% } -/+ %
TAU-2 = 20.07235 { 19.50389 ,20.66957 }
                { -0.5684631 ,0.5372168 } -/+ diff
                { -2.83% ,2.98% } -/+ %
AMPLITUDE-2 = 0.2512234 { 0.2347788 ,0.2686589 }
                   { -0.2347788 ,0.2686589 } -/+ diff
                   { -0.2347788 ,0.2686589 } -/+ %
Quit Help Top Btm Find Next Goto Print Line:
Support Plane on 3 Parameters

```

Q key. A second BROWSE menu appears with more information. Press **Q** again.

The screen should look something like:

Select the **PLOT** command and then the **Data** option to examine a plot of the data and the fitted function.

```

CFS_LS -- Parameter Estimation Program
?
Select a Command
FIT      FUNCTION  GUESS  HELP  PLOT  QUIT  ...
Create plots of the data and results.

Support Plane C.I.: Parm = 0, Distance = -0.7944413 ; 3/ 3
33.08671    37.19516    39.12070    35.04955

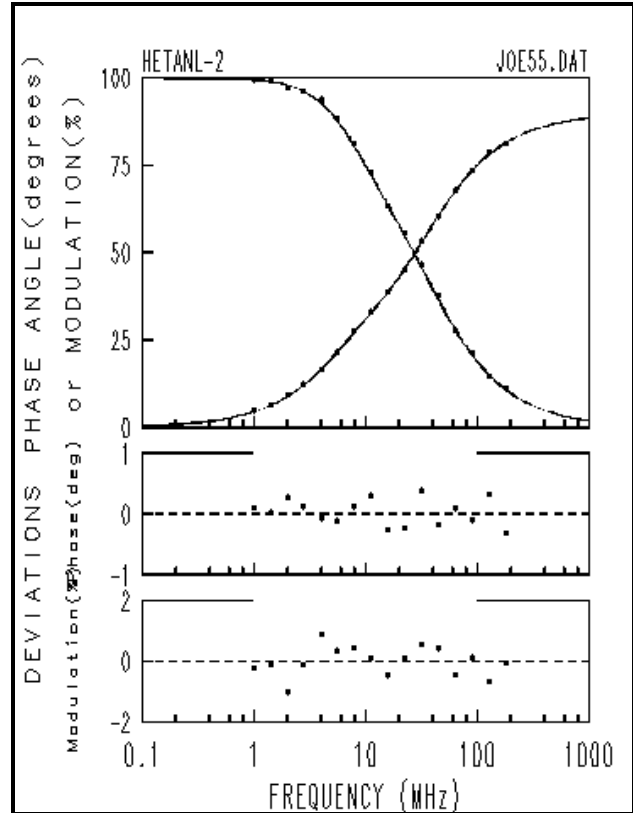
Support Plane on 3 Parameters

```

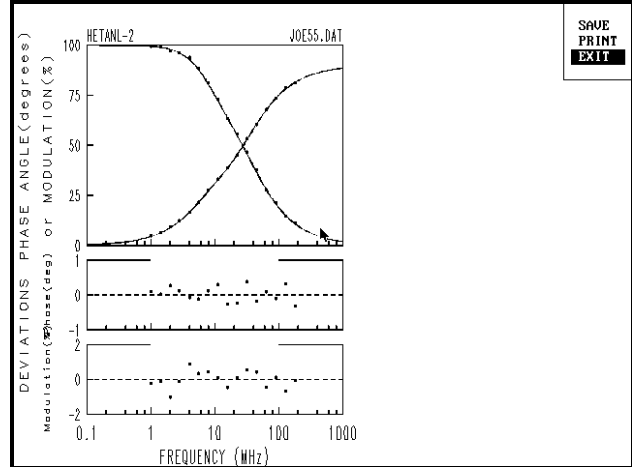
The upper panel of this figure contains the original data points and the best calculated curves based upon the estimated parameters. As the frequency increases the angle of the phase shift increases and the modulation (%) decreases.

The middle panel is the deviations (residuals) of the calculated phase angle from the data.

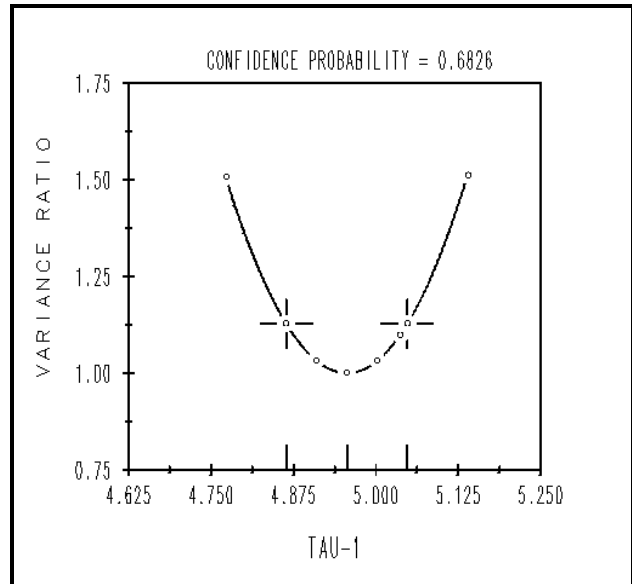
The lower panel is the modulation deviations.



Please note that when you are finished looking at a plot you press the **enter** key. You will now see a menu that allows you **SAVE** the image, **PRINT** the image in the output file, or **EXIT**.



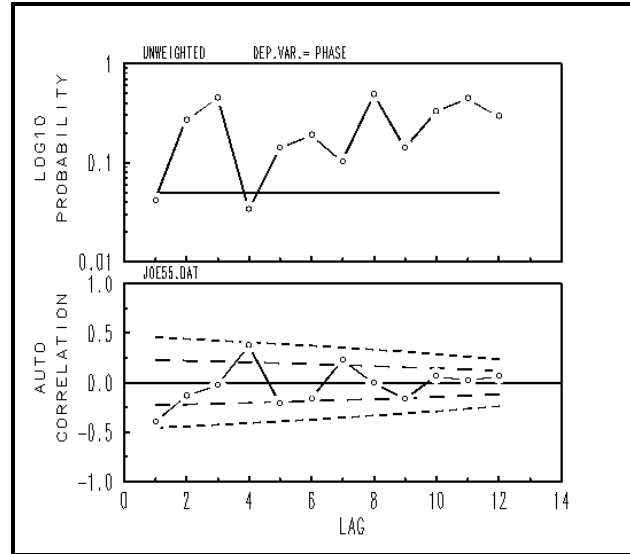
The **PLOT** command with the **SuprtPlane** option would show the results of the support plane confidence interval estimate. A further menu is used to specify which parameters are to be displayed. See also Figures 2.4.6 to 2.4.8.



The X-axis of this plot is a variation of one of the particular parameters while the Y-axis represents the observed variance ratio. This variance ratio is the variance-of-fit with the particular X value of the parameter divided by the variance-of-fit with the optimal (maximum likelihood) estimate of the parameter.

The **PLOT** command with the **AutoCorr** allows you to examine the autocorrelation function of the residuals.

The lower panel is the observed autocorrelation for various lag's (the open circles). The dashed lines correspond to \pm one and two standard errors (*i.e.*, significance). The upper panel is the



probability of observing the autocorrelations at random. The horizontal line in the upper panel corresponds to a 0.05 significance level. Autocorrelations that fall below this line appear to be significantly non-zero (*i.e.*, non-random).

To quit this tutorial select the **QUIT** option from the main menu and confirm with a **Y** key.

1.2 CFS_LS Output Files

During the normal operation of the CFS_LS program four output files are created: **CFS_LS.PRN**, **CFS_LS.BST**, **CFS_LS.CMD**, and **CFS_LS.GRF**.

CFS_LS.PRN

This is the general purpose output file created by the program. It generally contains the results of the analysis. Some items are automatically added to this file. Other items can be added by selecting the **PRINT** option when in the BROWSE mode of several of the CFS_LS commands: **BROWSE**, **FIT**, **GUESS**, or **RESIDUALS**. The **PRINT** option may also be selected during a **PLOT** to obtain a hard copy of a graph.

The name of this file can be changed with the **CFS_LS.INI** with the **PRN_FILE** option.

CFS_LS.BST

Whenever a nonlinear least-squares parameter estimation is performed the CFS_LS.BST file is automatically created and updated.

This is a no frills file that simply contains

```
0.0000000
0.0000000
0.4813371
1.438381
7.132469
0.2089052
```

Typical CFS_LS.BST file

a table of numbers that correspond to the set of parameter values that yielded the lowest variance-of-fit during the last parameter estimation.

The purpose of this file is to be read at a latter time by the **GUESS** with the **FILE** option. Note that since this file always has the same name it is rewritten every time a **FIT** is performed. However, the user can at anytime rename this file with a **SHELL** or ! command.

CFS_LS.CMD

Whenever the CFS_LS program executes all of the commands are stored in the **CFS_LS.CMD** file. A typical example is shown to the right.

There are two purposes of this file. First, it can be used as the basis for a series of commands to be used with the @ command. It can thus be used as a method to "program" a custom series of operations that can then be executed as a group. The second and most important use of this file is to aid in reporting and fixing problems with the program. A copy

```
FUNCTION HETANL-1
DATA TD-PRA PRA.DAT
POISSON
1
22
0.00000
1
19
0.00000
9
1024
GUESS KEYBOARD
0.00000      CONSTANT  :01
0.00000      DELTA-TIME :01
8.0000      TAU-1      :01
1.0000      AMPLITUDE-1 :01
FIT NM-Simplex None
NO
NO
YES
YES
QUIT
```

Typical CFS_LS.CMD file

of this file must be included with all reports or program errors and/or difficulties.

The name of this file can be changed with the **CFS_LS.INI** with the **CMD_FILE** option.

CFS_LS.GRF

This file is created whenever a **PLOT** is executed. It contains all of the information need to recreate the plot with the **VIEWGRAF** program. Since it is an ASCII file it can modified, and customized, by the user and then replotted with the **VIEWGRAF** program.

The name of this file can be changed with the **CFS_LS.INI** with the **GRF_FILE** option.

CFS_LS.DBG and **???????.TMP**

The are temporary files that may be generated under some error conditions. Periodically they should be deleted.

1.3 Experimental Data File Formats

FD-CFS: Center for Fluorescence Spectroscopy Frequency Domain Files

These data files are commonly used for time-resolved frequency domain fluorescence lifetime experimental data files. They are ASCII files so they can easily be printed and read by multiple programs and editors. An example of such a file is shown at the right.

The first line of the file is simply a comment. It can contain almost

```
SIMULATED DATA with Gaussian noise
END
16
16
16
16
CLOSE
1.00, 4.5912, 0.9921, 0.2000, 0.0050
1.40, 6.2786, 0.9884, 0.2000, 0.0050
2.00, 9.0974, 0.9690, 0.2000, 0.0050
2.80, 12.1845, 0.9603, 0.2000, 0.0050
4.00, 16.4100, 0.9369, 0.2000, 0.0050
5.60, 21.3471, 0.8817, 0.2000, 0.0050
8.00, 27.3156, 0.8099, 0.2000, 0.0050
11.20, 33.0000, 0.7260, 0.2000, 0.0050
16.00, 38.4803, 0.6318, 0.2000, 0.0050
22.40, 44.8766, 0.5529, 0.2000, 0.0050
32.00, 53.1361, 0.4645, 0.2000, 0.0050
44.80, 60.0696, 0.3740, 0.2000, 0.0050
64.00, 67.6666, 0.2767, 0.2000, 0.0050
89.60, 73.2304, 0.2115, 0.2000, 0.0050
128.00, 78.3863, 0.1444, 0.2000, 0.0050
179.20, 81.0592, 0.1088, 0.2000, 0.0050
```

JOE55.DAT FD-CFS data file

anything. The actual experimental data starts after the **CLOSE** line. The data consists of five columns per row separated by either spaces or commas. The first column is the frequency in MHZ. The second column is the observed phase angle. The third column is the observed modulation. The fourth column is the standard error of the phase angle. The fifth column is the standard error of the modulation.

FD-CFS data files are read by the **DATA** command. For example a **DATA** command to read the above file would be:

DATA FD-CFS JOE55.DAT

IBH4-ASC: Time Domain Files

The box at the right shows a typical IBH4-ASC data file.

Of the first eight numbers (INTEGER*2), only three are examined by this program. The first must be zero, indicating that this is a “raw” data file. The seventh is the starting data point number and the eighth is the ending data point number.

Of the next eight numbers (REAL*4), only the second is examined by this program. It contains the time increment between data channels. In the present example it is 1.835×10^{-2} nanoseconds per channel.

Immediately following the above sixteen is the data values (REAL*4). The number of these is specified by the starting data point number and the ending data point number specified above. These files contain either the data channel or the instrument response function channel, not both.

```
0
0
0
0
0
0
1
1023
2.68000E+0001
1.83500E-0002
0.00000E+0000
0.00000E+0000
0.00000E+0000
0.00000E+0000
0.00000E+0000
0.00000E+0000
0.00000E+0000
1.00000E+0000
1.00000E+0000
1.00000E+0000
1.00000E+0000
1.00000E+0000
2.00000E+0000
2.00000E+0000
2.00000E+0000
```

IBH4-ASC files are read by the **DATA** command. For example a **DATA**

command to read this data might look like:

DATA IBH4ASC D.DAT L.DAT

where **D.DAT** is the file containing the data observations and **L.DAT** is the file containing the instrument response (lamp) function.

IBH4-ASC/B: Time Domain Files

These are the same as **IBH4-ASC** files except that a third file name is read to be used as a background file. This file is simply subtracted from the first before any other processing takes place.

IBH4-BIN and IBH4-BIN/B: Time Domain Files

These files are the same as the corresponding **IBH4-ASC** files except that they are stored as “binary” files to conserve space.

These files also contain “binary file headers” that must also be processed. For example, the first byte (INTEGER*1) of the file always contains the number 75 and the last byte always contains -126. In addition, the information is contained in blocks with leading and trailing byte counts. Typically the byte counts will be -127 (actually 128 as a unsigned number). This the data blocks will typically be 130 bytes long, the first byte being -127, then 128 bytes of data, and the final byte again containing the -127. The last block will contain the actual byte counts needed for the remainder of the data.

TD-PRA: Time Domain Files

The box at the right contains a portion of a typical TD-PRA time domain data file.

This is also an ASCII data file so it can be easily printed and read by computer programs.

Only a few of the fields of this data file are actually read and processed by the CFS_LS program.

```
09-26-94          14:38:08
Data File Comment
      0          2      1024      40
      .0000          .0000          .0000
      .0000          .0267
      .000000E+00      .000000E+00
      .000000E+00      .000000E+00
      .000000E+00      .000000E+00

      0.      0.      0.      0.      0.      0.      0.      0.
      1.      2.      0.      2.      3.      2.      2.      1.
      1.      4.      0.      20.      40.      40.      60.      100.
      80.      150.      90.      10.      10.      18.      9.      11.
      18.      19.      23.      27.      26.      21.      25.      27.
      34.      37.      36.      44.      57.      53.      48.      45.
      64.      75.      70.      63.      91.      112.      115.      98.
      114.      125.      131.      157.      161.      155.      188.      210.
      240.      239.      254.      285.      314.      339.      320.      358.
      360.      439.      459.      463.      483.      485.      568.      644.
      628.      713.      750.      829.      859.      820.      895.      983.
      1028.      1119.      1207.      1270.      1305.      1377.      1450.      1535.
      1558.      1711.      1750.      1870.      2001.      2085.      2243.      2299.

      . . .
```

PRA.DAT Time Domain Data File

The first four lines are comments and all are processed.

The next line contains four numbers separated by commas or spaces. The third number is the number of data channels (N, the number of data points). The other three numbers are ignored.

The next line is ignored.

The second number on the next line is the time between data channels in nanoseconds. The first number on this line is ignored.

The next three lines are ignored.

Now comes the actual data. This program will read one or more numbers per line separated by either commas or spaces. All should contain a decimal point. The first N numbers correspond to the N channels of the instrument response function

(Lamp function). These are then followed by N actual data points.

TD-PRA data files are read by the **DATA** command. For example a **DATA** command to read the above file would be:

```
DATA TD-PRA PRA.DAT
```

TD-PRA/B: Time Domain Files

These are the same as **TD-PRA** files except that a second data file is read to be used as a background file. The second file is simply subtracted from the first before any other processing takes place.

TD-PRA/B data files are read by the **DATA** command. For example a **DATA** command to read the above file would be:

```
DATA TD-PRA/B PRA.DAT PRA2.DAT
```

FD-SPEX: SPEX Frequency Domain Files

These data files are commonly used for time-resolved frequency domain fluorescence lifetime experimental data files. They are ASCII files so they can easily be printed and read by multiple programs and editors.

The first line of the file is simply a comment. It can contain almost anything. The second line is simply a header for the data columns and is simply ignored. The data consists of five columns per row separated by either spaces or commas. The first column is the frequency in MHZ. The second column is the observed phase angle.

The third column is the observed modulation. The fourth column is the standard error of the phase angle. The fifth column is the standard error of the modulation.

FD-SPEX data files are read by the **DATA** command. For example a **DATA** command to read the above file would be:

```
DATA FD-SPEX SPEX.DAT
```

The reader might also be interested in the **SPEX2CFS.EXE** data conversion program. This program converts SPEX format frequency domain data files into CFS formatted data files. To use the program simply execute a command line like:

```
SPEX2CFS <spex_data_file> <cfs_data_file>
```

1.4 Fitting Functions: Intensity Decay Law Functions

The CFS_LS program contains multiple intensity decay laws (*i.e.*, fitting functions) that are selected with the **FUNCTION** command.

The following description of the fitting functions in terms of the intensity decay laws. The reader should be aware that the actual parameter estimation is performed on the original time- or Frequency-domain data. Therefore the actual fitting function is the intensity decay law, $I(t)$, with either the time- or frequency-domain transformations outlined below.

Time-Domain Transformations

The observed time-domain data at some time t is a convolution integral of the actual intensity decay law, $I(t)$, and an instrument response function, $L(t)$. This instrument response function is sometimes referred to as the lamp function. This instrument response function is actually a convolution integral of the lamp intensity as a function of time and the actual response of the instrument.

$$\text{Observed-Data}(t) = \int_{-\infty}^t L(T) I(t - T) dT$$

In some time domain instruments it is possible that a small difference could exist between the zero time value for the observed data and the instrument response function. Thus an additional parameter, **DELTATIME**, is included in all time domain

fitting functions to accommodate this instrumentation problem. This **DELTA***TIME* parameter will be either zero, or near zero. Thus in a lot of cases it can be simply set to zero and held constant by the parameter estimation procedure.

$$Observed-Data(t) = \int_{-\infty}^t L(T) I(t - T - DELTATIME) dT$$

It is also possible that some instruments may have a significant amount of background counts or dark current. Thus a second additional parameter, **CONSTANT**, is included in all time domain fitting functions to accommodate this instrumentation problem. This **CONSTANT** parameter will be either zero, or near zero. Thus in a lot of cases it can be simply set to zero and held constant by the parameter estimation procedure.

$$Observed-Data(t) = CONSTANT + \int_{-\infty}^t L(T) I(t - T - DELTATIME) dT$$

Consequently all time-domain fitting functions include the parameters of the actual intensity decay law, **I(t)**, and two additional parameters, **DELTA***TIME* and **CONSTANT**.

The unit of time in nanoseconds.

Frequency-Domain Transformations

Time resolved frequency domain data consists of two dependent variables, phase and modulation, as a function of modulation frequency, f . These dependent variables are related to the sine and cosine transforms of the intensity decay laws, $I(t)$. Specifically,

$$\omega = \frac{2 \pi f}{1000}$$

$$N_{\omega} = \frac{\int_0^{\infty} I(t) \sin(\omega t) dt}{\int_0^{\infty} I(t) dt}$$

$$D_{\omega} = \frac{\int_0^{\infty} I(t) \cos(\omega t) dt}{\int_0^{\infty} I(t) dt}$$

$$phase = \tan^{-1} \left(\frac{N_{\omega}}{D_{\omega}} \right)$$

$$\text{modulation} = \sqrt{N_{\omega}^2 + D_{\omega}^2}$$

The units of modulation frequency are megahertz. Modulation is dimensionless. The units of phase shift is degrees.

If the intensity decay law can be correctly expressed as a simple sum of exponential decays, as in **HETANL**, then the above integrals can be greatly simplified. Specifically, the explicit integration for the sum of exponentials is given by,

$$N_{\omega} = \frac{\sum_{i=1}^k \frac{\text{AMPLITUDE}_i \omega \tau_i^2}{1 + (\omega \tau_i)^2}}{\sum_{i=1}^k \text{AMPLITUDE}_i \tau_i}$$

$$D_{\omega} = \frac{\sum_{i=1}^k \frac{\text{AMPLITUDE}_i \tau_i}{1 + (\omega \tau_i)^2}}{\sum_{i=1}^k \text{AMPLITUDE}_i \tau_i}$$

However, not all intensity decay laws can be expressed as a simple sum of exponential decays. For example for **ET1DET** the intensity decay law is of the form

$$I(t) = \sum A_i e^{-Q_i(t)} ; \text{ where } Q_i(t) = a_i t + b_i t^{\frac{1}{6}}$$

For these cases the calculation of the integrals for N_{ω} and D_{ω} is approximated by a

single-exponential approximation numerical integral method.

The first step in this numerical integration is to explicitly tabulate the intensity decay law $I(t)$ at approximately 100 distinct values of time, t . The specific choice of the particular values of time is critical! The choice of time is itself performed as a multiple approximation process. The first approximation to the distinct time values is done by determining the lowest and highest frequency (in megahertz) contained within the data set. These are used to approximate the second time point (in nanoseconds) as;

$$t_2 = \frac{12.5}{\text{highest frequency}}$$

The eleventh time point is,

$$t_{11} = \frac{500}{\text{lowest frequency}}$$

The first time point is zero and points three to ten are logarithmically spaced between points two and eleven. The intensity decay law, $I(t)$, is then evaluated at these eleven time values. It should be noted that these eleven time values are only a gross approximation of the actual time values that will be used. They are only used to evaluate the approximate behavior of the intensity decay law.

These eleven values are then transformed such that,

$$X'_i = -Ln \left(\frac{I(t_i)}{I(0)} \right)$$

$$Y'_i = (t_i)^{\frac{1}{r}}$$

where r is the highest root of time that occurs in the intensity decay law. The following table list the value of r used for some of the decay laws.

Decay Law	r
HETANL	1
ET1DET	6
ET2DET	3
ET3DET	2

A natural cubic spline (G. E. Forsythe, M. A. Malcolm, and C. B. Moler, "Computer Methods for Mathematical Computations", Prentice-Hall, Englewood Cliffs, N.J., see problem P4-3 on page 80, 1977) is then evaluated using these eleven values of X' and Y' . A natural cubic spline is one where the second derivatives at the endpoints are zero. This spline is then evaluated at a series of points to obtain the actual distribution of time values to be used for the numerical integrations. For example, to evaluate the time value that corresponds approximately to a normalized intensity decay law value of 0.8 the program first calculates the negative natural logarithm of 0.8, i.e., 0.22314. The spline is then evaluated at an X' value of 0.22314. The resulting Y' value is the r^{th} root of the approximate time value, t , where $I(t)$ will equal 0.8.

To obtain the final tabulated values of the intensity decay law to be used for the

integration the natural cubic spline is used to evaluate approximately a hundred values of time. Within specific segments of the decay curve the $I(t)$ are logarithmically spaced. The specific segments of the distribution of $I(t)$ values used to generate the values of t is shown in the following table.

Number of Values	$I(t)$ Range
43	e^0 to e^{-1}
20	$>e^{-1}$ to e^{-2}
10	$>e^{-2}$ to e^{-3}
5	$>e^{-3}$ to e^{-4}
4	$>e^{-4}$ to e^{-5}
22	$>e^{-5}$ to e^{-15}

For example, the first ten desired values of the negative natural logarithm of $I(t)$ are:

0.000, 0.00625, 0.0125, 0.025, 0.050,
0.075, 0.100, 0.125, 0.150 and 0.175

The resulting 104 values of time, t , are then used to re-evaluate and tabulate the intensity decay law, $I(t)$, which is used for a numerical integration to evaluate N_ω and D_ω .

The numerical integrations are performed by integrating single-exponential approximations of the intensity decay law between time values of t_k and t_{k+1} (J. Kusba and J. R. Lakowicz, *Methods in Enzymology* 240, 216-262, 1994). The intensity decay law for t between t_k and t_{k+1} is approximated as,

$$I(t) \approx I(t_k)e^{-\beta_k(t-t_k)}$$

The value of β_k is (see Kusba and Lakowicz Equation 14),

$$\beta_k = \frac{1}{t_{k+1} - t_k} \ln \left(\frac{I(t_k)}{I(t_{k+1})} \right)$$

Using this single-exponential approximation (these include a corrected version of Equation 19 of Kusba and Lakowicz) the values of N_ω and D_ω are evaluated as,

$$N_\omega = \frac{1}{J_\omega(\beta_k^2 + \omega^2)} \sum_k I(t_{k+1})[-\beta_k \sin(\omega t_{k+1}) - \omega \cos(\omega t_{k+1})] - I(t_k)[- \beta_k \sin(\omega t_k) - \omega \cos(\omega t_k)]$$

$$D_\omega = \frac{1}{J_\omega(\beta_k^2 + \omega^2)} \sum_k I(t_{k+1})[-\beta_k \cos(\omega t_{k+1}) + \omega \sin(\omega t_{k+1})] - I(t_k)[- \beta_k \cos(\omega t_k) + \omega \sin(\omega t_k)]$$

$$J_\omega = \sum_k \frac{I(t_{k+1}) - I(t_k)}{-\beta_k}$$

One significant advantage of this approach is that the tabulation of the intensity decay law, $I(t)$, is used for all of the frequencies in the particular data set. Thus, it provides a significant time savings. This is particularly true for the $I(t)$ functions that require significant amounts of computer time to calculate.

HETANL

This is the most commonly used and simplest of all intensity decay laws. For this function the intensity decays is simply the sum of one or more exponentials:

$$I(t) = \sum_{i=1}^k \text{AMPLITUDE}_i e^{-\frac{t}{\tau_i}}$$

The **TAU-i** are the individual fluorescence lifetimes and the **AMPLITUDES-i** are the un-normalized relative amplitudes. The CFS_LS program actually has six different forms of the **HETANL** type of intensity decay law: **HETANL-1**, **HETANL-2**, **HETANL-3**, **HETANL-4**, **HETANL-5**, and **HETANL_NI**. These correspond to different numbers of exponential decays, *i.e.*, different values of k in the above equation.

HETANL-1 has four fitting parameters and two derived parameters. The fitting parameters are:

CONSTANT, see time-domain transformation discussion above
DELTATIME, see time-domain transformation discussion above
AMPLITUDE-1, amplitude corresponding to **TAU-1**
TAU-1, the fluorescence lifetime

For time-domain data all four of these parameters can be estimated by the least-squares procedure. However, **CONSTANT** and **DELTATIME** only apply to time-domain data. Furthermore, since frequency-domain data is always normalized **AMPLITUDE-1** is always unity. Therefore for frequency-domain data only **TAU-1** can be estimated with **HETANL-1**. For the more complex versions, **HETANL-2** etc., each additional term in the summation will introduce two additional parameters,

AMPLITUDE-2 TAU-2 etc., both of which can be estimated for either time- or frequency-domain data.

The two derived parameters are:

FRACT-AMPL-1, the fraction amplitude corresponding to **TAU-1**
SS-INTENS-1, the steady state intensity corresponding to **TAU-1**

For the more complex versions, **HETANL-2** etc., additional derived parameters are also included, **FRACT-AMPL-2** and **SS-INTENS-2** etc.

$$FRACT-AMPL-i = \frac{AMPLITUDE_i}{\sum_{i=1}^k AMPLITUDE_i}$$

$$SS-INTENS-i = AMPLITUDE-i \cdot T-i$$

HETANL_NI is the same as **HETANL-5** except that N_ω and D_ω , and thus $I(t)$, are calculated by the single-exponential approximation numerical integral method.

Please note: This program is coded for up to five fluorescence lifetimes. However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve all of these.

GAUTAU

For this function the intensity decay law is simply multiple exponentials decays with amplitudes defined by the sum of up to five Gaussian distributions:

$$I(t) = \sum_{j=1}^5 \alpha_j(\tau) e^{-\frac{t}{\tau}}$$

Where the amplitudes, $\alpha_j(\tau)$, are given by individual Gaussian distributions.

$$\alpha_j(\tau) = \text{AMPLITUDE}_j \frac{1}{\sigma_j \sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{\tau - \bar{\tau}_j}{\sigma_j} \right)^2}$$

GAUTAU has a total of 32 parameters. The first two are for time-domain data only:

CONSTANT,	see time-domain transformation discussion above
DELTATIME,	see time-domain transformation discussion above

There are also three parameters that can be estimated by the least-squares procedure for each of the five distributions:

MEAN TAU-j,	the mean value of the j^{th} distribution
HALF-WIDTH-j,	the full width at half height of the distribution, note that HALF-WIDTH-j is equal to 2.3545 times σ_j
AMPLITUDE-j,	the amplitude of the distribution

There are also three derived parameters for each distribution:

NORM-AMP-j,	the normalized amplitude of the j^{th} distribution
NORM-AREA-j,	the normalized area under the j^{th} distribution
SS-INTENS-j,	the steady state intensity of the j^{th} distribution

$$NORM-AMP-i = \frac{AMPLITUDE_i}{\sum_{i=1}^k AMPLITUDE_i}$$

$$NORM-AREA-i = \frac{AMPLITUDE_i HALF-WIDTH_j}{\sum_{i=1}^k AMPLITUDE_i HALF-WIDTH_j}$$

$$SS-INTEN_j = \frac{\int_0^{\infty} \alpha_j(\tau) d\tau}{\sum_{j=1}^5 \int_0^{\infty} \alpha_j(\tau) d\tau}$$

The smooth Gaussian probability distributions are approximated as 33 distinct rectangles equally spaced between \pm two half-widths. Thus, the calculation of **I(t)** is by the explicit integration method.

Please note: This program is coded for up to five distribution functions. However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve that number of distributions.

GAUDIS

For this function the intensity decay law is in terms of up to three Gaussian distance distributions in energy transfer between donor and acceptor (one acceptor per donor).

$$I(t) = \sum_{j=1}^3 \int_{r=0}^{\infty} \alpha_j(r) \sum_{i=1}^5 \alpha_{Di} e^{-\left[\frac{t}{\tau_{Di}} \left(1 + \left(\frac{R_0}{r} \right)^6 \right) \right]} dr$$

Where the amplitudes, $\alpha_j(r)$, are given by individual Gaussian distributions. α_{Di} is the amplitude of the i^{th} donor relaxation, τ_{Di} .

$$\alpha_j(r) = \text{AMPLITUDE}_j \frac{1}{\sigma_j \sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{r - \bar{r}_j}{\sigma_j} \right)^2}$$

GAUDIS has a total of 25 parameters. The first two are for time-domain data only:

CONSTANT, see time-domain transformation discussion above
DELTATIME, see time-domain transformation discussion above

The donor decay has two parameters for each of the five donor decays:

α_{Di} , the amplitude of the i^{th} donor decay
 τ_{Di} , the relaxation time of the i^{th} donor decay

Also

R₀, the Forster distance
R_{min}, the minimum radius to consider
R_{max}, the maximum radius to consider
FRACTLAB, the fraction labeled

There are also three parameters that can be estimated by the least-squares procedure for each of the three distributions:

MEAN RADIUS-j, the mean value of the j^{th} distribution
HALF-WIDTH-j, the full width at half height of the distribution, note
that **HALF-WIDTH-j** is equal to 2.3545 times σ_j
AMPLITUDE-j, the amplitude of the distribution

The smooth Gaussian probability distributions are approximated as 33 distinct rectangles equally spaced between \pm two half-widths. Thus, the calculation of $I(t)$ is by the explicit integration method.

Please note: This program is coded for up to three distribution functions and five donor relaxations. However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve all of these.

LORTAU

For this function the intensity decay law is simply multiple exponentials decays with amplitudes defined by the sum of up to five Lorentzian distributions:

$$I(t) = \sum_{j=1}^5 \alpha_j(\tau) e^{-\frac{t}{\tau}}$$

Where the amplitudes, $\alpha_j(\tau)$, are given by individual Lorentzian distributions.

$$\alpha_j(\tau) = \text{AMPLITUDE}_j \frac{1}{\pi} \frac{\text{HALF-WIDTH}_j / 2}{(\tau - \bar{\tau}_j)^2 + (\text{HALF-WIDTH}_j / 2)^2}$$

LORTAU has a total of 32 parameters. The first two are for time-domain data only:

CONSTANT,	see time-domain transformation discussion above
DELTATIME,	see time-domain transformation discussion above

There are also three parameters that can be estimated by the least-squares procedure for each of the five distributions:

MEAN TAU-j,	the mean value of the j^{th} distribution
HALF-WIDTH-j,	the full width at half height of the distribution
AMPLITUDE-j,	the amplitude of the distribution

There are also three derived parameters for each distribution:

NORM-AMP-j,	the normalized amplitude of the j^{th} distribution
NORM-AREA-j,	the normalized area under the j^{th} distribution
SS-INTENS-j,	the steady state intensity of the j^{th} distribution

$$\text{NORM-AMP-}i = \frac{\text{AMPLITUDE}_i}{\sum_{i=1}^k \text{AMPLITUDE}_i}$$

$$NORM-AREA-i = \frac{AMPLITUDE_i HALF-WIDTH_j}{\sum_{i=1}^k AMPLITUDE_i HALF-WIDTH_j}$$

$$SS-INTEN_j = \frac{\int_0^{\infty} \alpha_j(\tau) d\tau}{\sum_{j=1}^5 \int_0^{\infty} \alpha_j(\tau) d\tau}$$

The smooth Lorentzian probability distributions are approximated as 416 distinct rectangles spaced between \pm two thousand half-widths. Thus, the calculation of $I(\mathbf{t})$ is by the explicit integration method.

Please note: This program is coded for up to five distribution functions. However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve that number of distributions.

LORDIS

For this function the intensity decay law is in terms of up to three Lorentzian distance distributions in energy transfer between donor and acceptor (one acceptor per donor).

$$I(t) = \sum_{j=1}^3 \int_{r=0}^{\infty} \alpha_j(r) \sum_{i=1}^5 \alpha_{Di} e^{-\left[\frac{t}{\tau_{Di}} \left(1 + \left(\frac{R_0}{r} \right)^6 \right) \right]} dr$$

Where the amplitudes, $\alpha_j(r)$, are given by individual Lorentzian distributions. α_{Di} is the amplitude of the i^{th} donor relaxation, τ_{Di} .

$$\alpha_j(r) = \text{AMPLITUDE}_j \frac{1}{\pi} \frac{\text{HALF-WIDTH}_j / 2}{(r - \bar{r}_j)^2 + (\text{HALF-WIDTH}_j / 2)^2}$$

LORDIS has a total of 25 parameters. The first two are for time-domain data only:

CONSTANT, see time-domain transformation discussion above
DELTATIME, see time-domain transformation discussion above

The donor decay has two parameters for each of the five donor decays:

α_{Di} , the amplitude of the i^{th} donor decay
 τ_{Di} , the relaxation time of the i^{th} donor decay

Also

R₀, the Forster distance
R_{min}, the minimum radius to consider
R_{max}, the maximum radius to consider
FRACTLAB, the fraction labeled

There are also three parameters that can be estimated by the least-squares procedure for each of the three distributions:

MEAN RADIUS-j, the mean value of the j^{th} distribution
HALF-WIDTH-j, the full width at half height of the distribution
AMPLITUDE-j, the amplitude of the distribution

The smooth Lorentzian probability distributions are approximated as 416 distinct rectangles spaced between \pm two thousand half-widths. Thus, the calculation of $I(t)$ is by the explicit integration method.

Please note: This program is coded for up to three distribution functions and five donor relaxations. However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve all of these.

RECTAU

For this function the intensity decay law is simply multiple exponentials decays with amplitudes defined by the sum of up to five rectangular probability distributions:

$$I(t) = \sum_{j=1}^5 \alpha_j(\tau) e^{-\frac{t}{\tau}}$$

Where the amplitudes, $\alpha_j(\tau)$, are given by individual rectangular probability distributions.

$$\alpha_j(\tau) = \text{AMPLITUDE}_j \begin{cases} 1, & \text{if } |z_j| \leq 0.5 \\ 0, & \text{if } |z_j| > 0.5 \end{cases}$$

$$z_j = \frac{\tau - \bar{\tau}_j}{\text{HALFWIDTH}_j}$$

RECTAU has a total of 32 parameters. The first two are for time-domain data only:

CONSTANT,	see time-domain transformation discussion above
DELTATIME,	see time-domain transformation discussion above

There are also three parameters that can be estimated by the least-squares procedure

for each of the five distributions:

MEAN TAU-j,	the mean value of the j^{th} distribution
HALFWIDTH-j,	the full width at half height of the distribution
AMPLITUDE-j,	the amplitude of the distribution

There are also three derived parameters for each distribution:

NORM-AMP-j,	the normalized amplitude of the j^{th} distribution
NORM-AREA-j,	the normalized area under the j^{th} distribution
SS-INTENS-j,	the steady state intensity of the j^{th} distribution

$$NORM-AMP-i = \frac{AMPLITUDE_i}{\sum_{i=1}^k AMPLITUDE_i}$$

$$NORM-AREA-i = \frac{AMPLITUDE_i HALF-WIDTH_j}{\sum_{i=1}^k AMPLITUDE_i HALF-WIDTH_j}$$

$$SS-INTEN_j = \frac{\int_0^{\infty} \alpha_j(\tau) d\tau}{\sum_{j=1}^5 \int_0^{\infty} \alpha_j(\tau) d\tau}$$

The smooth rectangular probability distributions are approximated as 41 distinct rectangles equally spaced between ± 0.5 half-widths. Thus, the calculation of $I(t)$ is by the explicit integration method.

Please note: This program is coded for up to five distribution functions. However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve that number of distributions.

RECDIS

For this function the intensity decay law is in terms of up to three rectangular distance distributions in energy transfer between donor and acceptor (one acceptor per donor).

$$I(t) = \sum_{j=1}^3 \int_{r=0}^{\infty} \alpha_j(r) \sum_{i=1}^5 \alpha_{Di} e^{\left[-\frac{t}{\tau_{Di}} \left(1 + \left(\frac{R_0}{r} \right)^6 \right) \right]} dr$$

Where the amplitudes, $\alpha_j(r)$, are given by individual rectangular distributions. α_{Di} is the amplitude of the i^{th} donor relaxation, τ_{Di} .

$$\alpha_j(r) = \text{AMPLITUDE}_j \begin{cases} 1, & \text{if } |z_j| \leq 0.5 \\ 0, & \text{if } |z_j| > 0.5 \end{cases}$$

$$z_j = \frac{r - \bar{r}_j}{\text{HALFWIDTH}_j}$$

RECDIS has a total of 25 parameters. The first two are for time-domain data only:

CONSTANT, see time-domain transformation discussion above
DELTATIME, see time-domain transformation discussion above

The donor decay has two parameters for each of the five donor decays:

α_{Di} , the amplitude of the i^{th} donor decay
 τ_{Di} , the relaxation time of the i^{th} donor decay

Also

R_0 , the Forster distance
 R_{min} , the minimum radius to consider

R_{max} , the maximum radius to consider
FRACTLAB, the fraction labeled

There are also three parameters that can be estimated by the least-squares procedure for each of the three distributions:

MEAN RADIUS-j, the mean value of the j^{th} distribution
HALFWIDTH-j, the full width at half height of the distribution
AMPLITUDE-j, the amplitude of the distribution

The smooth rectangular probability distributions are approximated as 41 distinct rectangles equally spaced between ± 0.5 half-widths. Thus, the calculation of $I(t)$ is by the explicit integration method.

Please note: This program is coded for up to three distribution functions and five donor relaxations. However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve all of these.

SKUTAU

For this function the intensity decay law is simply multiple exponentials decays with amplitudes defined by the sum of up to three skewed Gaussian distributions:

$$I(t) = \sum_{j=1}^5 \alpha_j(\tau) e^{-\frac{t}{\tau}}$$

Where the amplitudes, $\alpha_j(\tau)$, are given by individual Gaussian distributions.

$$\alpha_j(\tau) = \text{AMPLITUDE}_j \frac{1}{\sigma_j \sqrt{2\pi}} e^{-\frac{1}{2} \left(\frac{\tau - \bar{\tau}_j}{\sigma_q} \right)^2}$$

$$\sigma_q = \begin{cases} 2\sigma_j z_j, & \text{if } (\tau - \bar{\tau}_j) \geq 0 \\ 2\sigma_j(1-z_j), & \text{if } (\tau - \bar{\tau}_j) < 0 \end{cases}$$

$$z_j = \frac{e^{\text{SKEWNESS}_j}}{1 + e^{\text{SKEWNESS}_j}}$$

SKUTAU has a total of 23 parameters. The first two are for time-domain data only:

CONSTANT,	see time-domain transformation discussion above
DELTATIME,	see time-domain transformation discussion above

There are also four parameters that can be estimated by the least-squares procedure for each of the three distributions:

MEAN TAU-j,	the mean value of the j^{th} distribution
HALF-WIDTH-j,	the full width at half height of the distribution, note that HALF-WIDTH-j is equal to 2.3545 times σ_j

AMPLITUDE-j, the amplitude of the distribution
SKEWNESS-j, the skewness of the distribution

There are also three derived parameters for each distribution:

NORM-AMP-j, the normalized amplitude of the jth distribution

NORM-AREA-j, the normalized area under the jth distribution

SS-INTENS-j, the steady state intensity of the jth distribution

$$NORM-AMP-i = \frac{AMPLITUDE_i}{\sum_{i=1}^k AMPLITUDE_i}$$

$$NORM-AREA-i = \frac{AMPLITUDE_i \cdot HALF-WIDTH_j}{\sum_{i=1}^k AMPLITUDE_i \cdot HALF-WIDTH_j}$$

$$SS-INTEN_j = \frac{\int_0^{\infty} \alpha_j(\tau) d\tau}{\sum_{j=1}^5 \int_0^{\infty} \alpha_j(\tau) d\tau}$$

The smooth skewed Gaussian probability distributions are approximated as 33 distinct rectangles equally spaced between \pm two half-widths. Thus, the calculation of **I(t)** is by the explicit integration method.

Please note: This program is coded for up to five distribution functions.

However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve that number of distributions.

ET1DET

This intensity decay law is for energy transfer to a random distribution of acceptors in one dimension (*Hauser, Klein, and Gosele, Zeitschrift fur Physikalische Chemi Neue Folge 101, 255-266, 1976*). The actual decay law is of the form

$$I(t) = \sum \text{amplitude}_i \exp \left[- \left(\frac{t}{\tau_i} \right) - b \left(\frac{t}{\tau_i} \right)^{\frac{1}{6}} \right]$$
$$b = (2.26 \times 10^{-8}) R_0 \frac{\text{Acc/Base}}{\text{Cm/Base}}$$

This intensity decay law contains 28 variables.

CONSTANT, see time-domain transformation discussion above
DELTATIME, see time-domain transformation discussion above
TAU-i, the donor fluorescence lifetimes for i equals 1 to 5
AMPLITUDE-i, amplitude corresponding to **TAU-i**
FRACT-AMPL-i, the derived fraction amplitude corresponding to **TAU-i**
SS-INTENS- i the derived steady state intensity corresponding to **TAU-I**
R₀
Acc/Base, the concentration of acceptor per DNA/RNA “base”
Cm/Base, the dimension of a DNA/RNA “base” in Cm
F/F₀ or **I/I₀**, the derived fractional energy transfer
F₀/F or **I₀/I**, the derived inverse fractional energy transfer
Molec/Cm, the derived number of molecules per Cm.

For **ET1DET** the values of **N_ω** and **D_ω**, and thus **I(t)**, are calculated by the single-exponential approximation numerical integral method.

Please note: This program is coded for up to five fluorescence lifetimes.

However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve all of these.

ET2DET

This intensity decay law is for energy transfer to a random distribution of acceptors in two dimensions (*Hauser, Klein, and Gosele, Zeitschrift fur Physikalische Chemi Neue Folge 101, 255-266, 1976*). The actual decay law is of the form

$$I(t) = \sum \text{amplitude}_i \exp \left[- \left(\frac{t}{\tau_i} \right) - b \left(\frac{t}{\tau_i} \right)^{\frac{1}{3}} \right]$$
$$b = \frac{4}{3} \pi (10^{-8})^2 (R_0)^2 \frac{\text{Acc/Lipid}}{\text{Area/Lipid}}$$

This intensity decay law contains 28 variables.

CONSTANT, see time-domain transformation discussion above
DELTATIME, see time-domain transformation discussion above
TAU-i, the donor fluorescence lifetimes for i equals 1 to 5
AMPLITUDE-i, amplitude corresponding to **TAU-i**
FRACT-AMPL-i, the derived fraction amplitude corresponding to **TAU-i**
SS-INTENS- i the derived steady state intensity corresponding to **TAU-I**
R₀
Acc/Lipid, the concentration of acceptor per "lipid"
Cm/Base, the area of a "lipid" in Cm^2
F/F₀ or **I/I₀**, the derived fractional energy transfer
F₀/F or **I₀/I**, the derived inverse fractional energy transfer
Molec/Cm², the derived number of molecules per Cm^2

For **ET2DET** the values of **N_ω** and **D_ω**, and thus **I(t)**, are calculated by the single-exponential approximation numerical integral method.

Please note: This program is coded for up to five fluorescence lifetimes.

However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve all of these.

ET3DET

This intensity decay law is for energy transfer to a random distribution of acceptors in three dimensions (*Hauser, Klein, and Gosele, Zeitschrift fur Physikalische Chemi Neue Folge 101, 255-266, 1976*). The actual decay law is of the form

$$I(t) = \sum \text{amplitude}_i \exp \left[- \left(\frac{t}{\tau_i} \right) - b \left(\frac{t}{\tau_i} \right)^{\frac{1}{2}} \right]$$
$$b = N \frac{4}{3} \pi^{1.5} (10^{-8})^3 R_0^3 \text{ Acceptor}$$

This intensity decay law contains 26 variables.

CONSTANT, see time-domain transformation discussion above
DELTATIME, see time-domain transformation discussion above
TAU-i, the donor fluorescence lifetimes for i equals 1 to 5
AMPLITUDE-i, amplitude corresponding to **TAU-i**
FRACT-AMPL-i, the derived fraction amplitude corresponding to **TAU-i**
SS-INTENS- i the derived steady state intensity corresponding to **TAU-I**
R₀
Acceptor, the molar concentration of acceptor
F/F₀ or **I/I₀**, the derived fractional energy transfer
F₀/F or **I₀/I**, the derived inverse fractional energy transfer

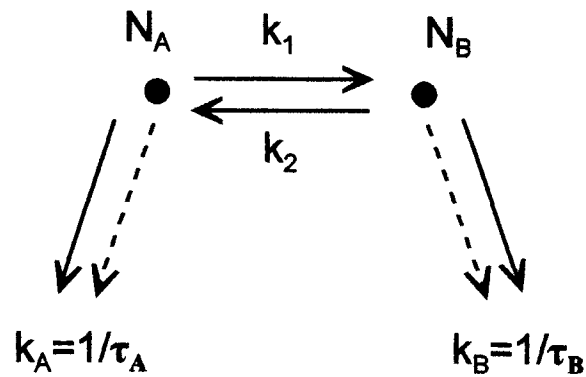
For **ET3DET** the values of **N_w** and **D_w**, and thus **I(t)**, are calculated by the single-exponential approximation numerical integral method.

Please note: This program is coded for up to five fluorescence lifetimes.

However, it is very unlikely that there is enough information within any particular group of data set to actually be able to resolve all of these.

HET_JUMP2

This is an implementation of a two state jump model where a macromolecule can exist in two state, N_A and N_B . These undergo an isomerization reaction with rate constants k_1 and k_2 while the intensity decay of each is given by k_A and k_B .



$$\begin{aligned}\frac{dN_A(t)}{dt} &= -(k_A + k_1)N_A(t) + k_2N_B(t) \\ \frac{dN_B(t)}{dt} &= -(k_B + k_2)N_B(t) + k_1N_A(t)\end{aligned}$$

The solution of this system of differential equations is:

$$\begin{vmatrix} -(k_A + k_1) - \lambda & k_2 \\ k_1 & -(k_B + k_2) - \lambda \end{vmatrix} = 0$$

where

$$\begin{aligned}2 \lambda_1 &= b - \sqrt{b^2 - 4c} \\2 \lambda_2 &= b + \sqrt{b^2 - 4c} \\b &= k_A + k_1 + k_B + k_2 \\c &= (k_A + k_1)(k_B + k_2) - k_1 k_2\end{aligned}$$

The solution being given as:

$$\begin{aligned}N_A(t) &= A_1 e^{-\lambda_1 t} + A_2 e^{-\lambda_2 t} \\N_B(t) &= B_1 e^{-\lambda_1 t} + B_2 e^{-\lambda_2 t}\end{aligned}$$

with boundary conditions

$$\begin{aligned}N_A(t=0) &= f_A^0 \\N_B(t=0) &= f_B^0 \\f_A^0 + f_B^0 &= 1 \\A &= A_1 \\A_2 &= f_A^0 - A \\B &= B_1 \\B_2 &= f_B^0 - B\end{aligned}$$

Which in turn becomes

$$\begin{aligned}N_A(t) &= A e^{-\lambda_1 t} + (f_A^0 - A) e^{-\lambda_2 t} \\N_B(t) &= B e^{-\lambda_1 t} + (f_B^0 - B) e^{-\lambda_2 t} \\A &= \frac{f_A^0 (\lambda_2 - k_A - k_1) + k_2 f_B^0}{\lambda_2 - \lambda_1} \\B &= \frac{f_B^0 (\lambda_2 - k_B - k_2) + k_1 f_A^0}{\lambda_2 - \lambda_1}\end{aligned}$$

The intensity decay law is the weighted sum of the two states

$$I(t) = N_A(t) + N_B(t) = e^{-t/\tau_A} + (1 - \frac{f_A^0 k_A}{f_A^0 k_A + f_B^0 k_B}) e^{-t/\tau_B}$$

The above solution is normalized as is needed for the frequency-domain experiments.

For the time-domain experiments the $I(t)$ is multiplied by the sum of the un-normalized f_A^0 and f_B^0 .

Fitting parameters:

K , the equilibrium constant which is defined as k_1/k_2

k_1 , one of the rate constants

τ_A , the lifetime of state A

τ_B , the lifetime of state B

f_A^0 , the *un-normalized* initial concentration of state A

f_B^0 , the *un-normalized* initial concentration of state B

1.5 Individual CFS_LS Commands

HELP

Obtain help about a command, as in **HELP DATA**. Help is available for all commands. Help is also available for **COM_LINE** and **CHARACTERS**.

HELP COM_LINE

Displays information about command line arguments.

HELP CHARACTERS

Displays information about special flow control characters.

SHELL

Opens a system shell for commands. Be careful to restore the state of the computer before returning from the shell command. For example, a CD command from within a shell will change the default directory for the remainder of the execution of the program. In fact, this is the recommended method to change the directory while running the program.

QUIT

Stops the program!

#

Enter a comment line.

@

Executes a command file, as in **@TEST.CMD**.

!

Execute a system command, as in **!DIR**. Be careful to restore the state of the computer before returning from the shell command. For example, a CD command from within a shell will change the default directory for the remainder of the execution of the program. In fact, this is the recommended method to change the directory while running the program.

COOKIE

Display a Cookie.

DATA

Input data into the program.

FUNCTION

Choose a fitting function for the program.

GUESS

Input initial values for the parameter estimation procedure.

BROWSE

Browse thru the internal arrays of this program. Not useful for most users.

FIT

Fit the function to the data. *I.e.*, performs a nonlinear least-squares parameter estimation procedure.

PLOT

Creates plots of the data and results. See for example Figures 2.4.2 to 2.4.8.

SMOOTH

Smooth the data. NOT RECOMMENDED!

RESIDUALS

Analysis of Residuals.

SIMULATE

Simulate Data. The simulated data is used for all further calculations. It is also stored separately in a data file(s).

NOTE! If you wish to simulate **IBH4** data files you should be sure to use the entire range of data contained in the data files. Otherwise, the data file may not correspond properly with the desired instrument response (lamp) function file.

GLOBAL

This command is used to redefine which of the parameters are global and which are not global. The user is presented with a menu and is expected to respond with YES or NO, indicating whether the particular parameter is to be assumed to be global.

1.6 Minimal Computer System Requirements

IBM PC Clones

The minimal system to execute these programs is:

- 80386 processor with 80387 math coprocessor
(Pentium Pro recommended)
- 8 Mbytes of RAM (DOS)
 - 16 Mbytes for Windows 3.1
 - 24 Mbytes for windows 95 and NT
- Hard Disk Drive with 20 Mbytes available

However, the programs will run very slow on the above system. As for any numerical analysis program the fastest possible processor is recommended. The program uses a significant amount of protected mode memory and if required does virtual memory swapping. 16 Mbytes of RAM is recommended. The program also uses the disk drive. Therefore the more available hard disk space the better. Do not attempt to execute these programs with the default disk drive being a floppy (A: or B:) or a read only device (CD-ROM)!

The PC should be running a current version of MS DOS (5.1, por later). If you must, it should work in a DOS window under any current version of MS Windows (3.1, 95, NT). Note, display mode 13 does not work correctly under Windows NT.

1.8 How to Report A Problem with the Programs

Incomplete trouble reports are, at best, frustrating and time consuming. It is impossible to locate and correct a problem without detailed and specific documentation about the problem. To fix a problem I need to be able to reproduce the problem. Therefore I need to know exactly what was done and in what order. I also need all of the appropriate data files that were used when the problem occurred

Consequently, to report a problem please send (by U.S. Mail) a written description of the problem and any pertinent printer output and graphs. Include a description of your computer, type of processor, amount of memory, type of display, type of printer, etc. Also specify the operating system in use, e.g., DOS Window under Windows 95. Also send a diskette containing all of the data files that were used. The diskette should also contain the CFS_LS output files **CFS_LS.BST**, **CFS_LS.CMD**, **CFS_LS.GRF** (if plots are involved), **CFS_LS.DBG** and **CFS_LS.PRN**. The diskette should also contain the current CFS_LS initialization file, **CFS_LS.INI**. Lastly, the diskette should contain a copy of your PC initialization files, **C:\CONFIG.SYS** and **C:\AUTOEXEC.BAT**.

Given the above information I can attempt to reproduce and fix the problem.

Trouble reports **CANNOT, and WILL NOT**, be taken by phone or FAX!

Please mail trouble reports to:

Michael Johnson
P.O. Box 448
UVA-HSC / Pharmacology
Charlottesville, VA 22908

2.0 Numerical Methods

Whenever a researcher applies nonlinear regression techniques for the analysis of experimental data, several objectives must be accomplished¹⁻². Specifically the researcher needs:

- 1) The numerical values of the various parameters of the fitted equation with the highest probability, or maximum likelihood, of being correct.
- 2) Information about the uniqueness of these parameter values.
- 3) Measures of the goodness-of-fit of the regression model and parameters.
- 4) A realistic measure, or estimate, of the precision of the parameter values based upon the regression model and the experimental data.

The primary objective of most commercially available regression analysis software packages is to provide the values of the parameters with the highest probability, or maximum likelihood, of being correct given a specific form of the fitting equation and a series of data points. In most cases this is accomplished by a least-squares fit of a functional form to some experimental data. In general, most regression analysis packages are very good at accomplishing this objective.

For nonlinear models it is important to know that the determined parameters are unique, *i.e.*, are there multiple sets of parameter values that correspond to a minimum

in the variance? However, very few commercially available regression analysis software packages provide the researcher with information about the uniqueness of the determined parameter values.

It is also important to address the question, "Did the regression equation actually fit the experimental data?" This question is usually addressed by one, or more, goodness-of-fit criteria³. If the fitted regression function is a good description of the data points then the residuals, the differences between the data points and the fitted function, should be random. In general, most goodness-of-fit criteria simply test one, or more, aspect of the randomness of the residuals.

In addition it is critical that the researcher obtain a reliable measure of the precision of the parameter values as determined by the nonlinear regression procedures. In practice, this is more important than the determination of the actual parameter values. For example, if a researcher has determined that the molecular weight of a protein is 90,000 daltons just how much is actually known about the protein? In reality it might have been $90,000 \pm 80,000$ daltons, or it might have been $90,000 \pm 1,000$ daltons. In the first case we know very little about the protein and in the latter case we know a great deal about the protein. In most instances the important information is that the molecular weight is likely to be between 89,000 and 91,000 daltons rather than that the most probable value is 90,000 daltons.

2.1 Parameter Estimation

Nonlinear models are defined by the form of the fitting function. When the second, and higher, derivatives of the fitting function with respect to the parameters being estimated are not zero the model is nonlinear. Polynomials, as in Equation 2.1.1, are linear fitting equations.

$$\begin{aligned} Y &= a + bX \\ Y &= a + bX + cX^2 \end{aligned} \tag{2.1.1}$$

However, most functions of interest to basic science researchers are actually nonlinear. For example, for a single exponential decay the second and higher derivatives are not equal to zero (see Equations 2.1.2).

$$\begin{aligned} Y &= a e^{-bX} \\ \frac{\partial Y}{\partial a} &= e^{-bX} \\ \frac{\partial Y}{\partial b} &= -a X e^{-bX} \\ \frac{\partial^2 Y}{\partial a \partial a} &= 0 \\ \frac{\partial^2 Y}{\partial a \partial b} &= -X e^{-bX} \\ \frac{\partial^2 Y}{\partial b \partial b} &= a X^2 e^{-bX} \end{aligned} \tag{2.1.2}$$

Least-squares regression methods all require a number of assumptions predominately pertaining to the nature of the experimental and observational uncertainties contained within the data. These assumption are are^{1-2,4-7}:

- 1) All uncertainties are contained in the dependent variable, *i.e.*, the Y-axis.
- 2) The uncertainties follow a Gaussian distribution.
- 3) No systematic errors occur within the data.
- 4) The fitting function is the correct mathematical description of the data.
- 5) Each data point is an independent observation.

The sample variance-of-fit is evaluated as:

$$s^2 = \frac{\sum_{i=1}^n \left(\frac{Y_i - G(X_i, \text{parameters})}{\sigma_i} \right)^2}{n-j} = \frac{SSR}{n-j} \quad 2.1.3$$

where X_i and Y_i correspond to the i^{th} data point; σ_i is the standard error of the mean corresponding to the i^{th} value of Y ; G is the fitting function evaluated at X_i and the current parameters; n is the number of data points; and j is the number of parameters being simultaneously estimated.

Complete discussions of several nonlinear regression (least-squares) methods occur elsewhere in the literature^{1-2,4-7} and will only be briefly repeated here. An example of the Nelder-Mead method for the evaluation of least-squares (maximum likelihood) parameter estimates follows.

2.2 Uniqueness of Parameters

2.3 Goodness-of-Fit Criterion

Scatter Diagram Residual Plots

Visualizing residuals is commonly performed by generating scatter diagrams (P. Armitage, "Statistical Methods in Medical Research," 4th Printing, p. 316. Blackwell, Oxford, 1977). Residuals may be plotted as a function of various experimental variables to permit convenient identification of trends that may not have been accounted for by the analytical model. Residuals are most commonly plotted as a function of either the values of the independent variable(s) (e.g., time in kinetic experiments or ligand concentration in ligand binding experiments) or the calculated values of the dependent variable (i.e., the values of the experimental observable calculated from the model). However, residual plots versus some other functional relationship of the independent variable(s) or some other potentially significant variable that was not explicitly considered in the original model may also provide information about important relationships that have not been previously identified.

An examination of the trend in residuals as a function of some particular variable space may even provide information about the type of quantitative relationship that must be accommodated by the analytical model that currently is not. However, correctly accounting for any newly incorporated relationships into currently existing analytical

models requires reevaluation of the data set(s) originally considered. This is necessary so as to simultaneously estimate values for all of the model parameters characteristic of the model, both previously existing and newly incorporated. Quantifying phenomena originally omitted from consideration by a model must not be attempted by analyzing the resulting residuals. Correlation among parameters must be accommodated during a parameter-estimation procedure so as to produce the true best-fit parameter values that accurately characterize the interdependence between parameters of the model and between these parameters and the properties of the data being analyzed (the dependence of the experimental observable on the independent experimental variables as well as on the distribution of experimental uncertainty in the data).

Runs Test: Quantifying Trends in Residuals

The existence of trends in residuals with respect to either the independent (i.e., experimental) or dependent (i.e., the experimental observable) variables suggests that some systematic behavior is present in the data that is not accounted for by the analytical model. Trends in residuals will often manifest themselves as causing too few runs (consecutive residual values of the same sign) or, in cases where negative serial correlation occurs, causing too many runs. A convenient way to assess quantitatively this quality of a distribution of residuals is to perform a runs test (Y. Bard, "Nonlinear Parameter Estimation," p. 201, Academic Press, 1974). The method involves calculating the expected number of runs given the total number of residuals as well as an estimate of variance in this expected number of runs. A run is simply one, or more,

residuals in a row of the same sign.

The expected number of runs, R , may be calculated from the total number of positive and negative valued residuals, n_p , and n_n , as

$$R = [2 n_p n_n / (n_p + n_n)] + 1$$

The variance in the expected number of runs, σ_R^2 , is then calculated as

$$\sigma_R^2 = \frac{2 n_p n_n (2 n_p n_n - n_p - n_n)}{(n_p + n_n)^2 (n_p + n_n - 1)}$$

A quantitative comparison is then made between the expected number of runs, R , and the observed number of runs, n_R , by calculating an estimate for the standard normal deviate as

$$Z = \left| \frac{n_R - R \pm 0.5}{\sigma_R} \right|$$

When n_p and n_n are both greater than 10, Z will be distributed approximately as a standard normal deviate. In other words, the calculated value of Z is the number of standard deviations that the observed number of runs is from the expected number of runs for a randomly distributed set of residuals of the number being considered. The value of 0.5 is a continuity correction to account for biases introduced by approximating a discrete distribution with a continuous one. This correction is + 0.5 (as above) when testing for too few runs and is -0.5 when testing for too many runs. The test is therefore

estimating the probability that the number of runs observed is different from that expected from randomly distributed residuals. The greater the value of Z , the greater the likelihood that there exists some form of correlation in the residuals relative to the particular variable being considered.

Autocorrelation: Detecting Serial Correlation

Experimental data collected as a time series typically exhibit serial correlations. These serial correlations arise when the random uncertainties superimposed on the experimental data tend to have values related to the uncertainties of other data points that are close temporally. For example, if one is measuring the weight of a test animal once a month and the data are expressed as a weight gain per month, negative serial correlation may be expected. This negative serial correlation is expected because a positive experimental error in an estimated weight gain for one month (i.e., an overestimate) would cause the weight gain for the next month to be underestimated.

A basic assumption of parameter-estimation procedures is that the experimental data points are independent observations. Therefore, if the weighted differences between experimental data points and the fitted function (the residuals) exhibit such a serial correlation, then either the observations are not independent or the mathematical model did not correctly describe the experimental data. Thus, the serial correlation of the residuals for adjacent and nearby points provides a measure of the quality of the fit.

The autocorrelation function provides a simple method to present this serial correlation for a series of different lags, k (E. P. Box and G. M. Jenkins, "Time Series

Analysis Forecasting and Control,” p. 33. Holden-Day, Oakland, California, 1976). The lag refers to the number of data points between the observations for a particular autocorrelation. For a series of N observations, Y_t , with a mean value of μ , the autocorrelation function is defined as

$$\beta_k = \frac{\hat{\sigma}_k}{\hat{\sigma}_0}$$

for $k = 0, 1, 2, \dots, K$, where the autocovariance function is

$$\hat{\sigma}_k = \frac{1}{n} \sum_{t=1}^{n-k} (Y_t - \mu)(Y_{t+k} - \mu)$$

For $k = 0, 1, 2, \dots, K$. In these equations, K is a maximal lag less than n . The autocorrelation function has a value between -1 and +1. Note that the autocorrelation function for a zero lag is equal to 1 by definition.

The expected variance (A. P. Moran, *Biometrika* 34, 281 (1947)) of the autocorrelation coefficient of a random process with independent, identically distributed random (normal) errors is

$$\text{var}(\beta_k) = \frac{n-k}{n(n+2)}$$

where μ is assumed to be zero.

Autocorrelations are presented graphically as a function of k . This allows an

investigator to compare easily the autocorrelation at a large series of lags k with the corresponding associated standard errors (square root of the variance) to decide if any significant autocorrelations exist.

Outliers: Identifying Bad Points

In any experimental measurement, occasionally values may be observed that produce an unusually large residual value after an analysis of the data is performed. The existence of an outlier (or “bad point”) suggests that some aberration may have occurred with the measurement of the point. The presence of such a point in the data set being analyzed may influence the derived model parameter values significantly relative to those that would be obtained from an analysis without the apparent outliers. It is therefore important to identify such “bad points” and perhaps reconsider the data set(s) being analyzed without these suspect points.

Visual inspection of residual scatter diagrams often reveals the presence of obvious outliers. Cumulative frequency plots will also indicate the presence of outliers, although perhaps in a less direct manner. Visualization methods may suggest the presence of such points, but what method should be used to decide whether a point is an outlier or just a point with a low probability of being valid? A method to provide a quantitative basis for making this decision derives from estimating the apparent standard deviation of the points after analysis. This is calculated as the square root of the variance of fit obtained from analysis of an unweighted data set. The variance of fit is defined as the sum of the squared residuals divided by the number of degrees of

freedom (the number of data points minus the number of parameters being estimated). In the case that the model employed is capable of reliably characterizing the data (i.e., capable of giving a “good fit”), the distribution of residuals will, in principle, represent the distribution of experimental uncertainty. Any residuals possessing values that are more than approximately 2.5 to 3 standard deviations from the mean have only a 1 to 0.25% chance of being valid. When considering relatively large data sets (of the order of hundreds of points or more), the statistical probability of a residual possessing a value 3 standard deviations from the mean suggests that such a point should be expected about once in every 400 data points.

2.4 Determination of Confidence Intervals

There is no closed form method for the evaluation of the precision of parameter estimates for nonlinear regression problems. There are, however, a number of approximate methods which require different restrictive assumptions and require different amounts of computer time. While none of these estimates are exactly correct for nonlinear problems, some provide substantially better results than others. This section discusses some of these methods with specific application to frequency domain fluorescence lifetime measurements. A significant conclusion being that the commonly used asymptotic standard errors do not provide reasonable estimates of the precision of parameter values for nonlinear regression problems.

It should be noted that there is not an exact formula, or theory, to predict the precision of the determined parameters for nonlinear models. For regression problems

with nonlinear models there are several approximate methods for the estimation of determined parameter precision. When evaluating the precision of determined parameters there is an inverse relationship between accuracy and the required amount of computer time. The worse methods are very easy to evaluate but have a large number of assumptions. As the required assumptions decrease the accuracy increases but so does the amount of computer time required for the calculations. The most precise methods are rarely utilized because of the large amount of computer time required. Some are listed in the Table 2.4.1.

Table 2.4.1: Methods for the Evaluation of Parameter Precision

	Usage	Precision	Required CPU Time	Assumptions
Asymptotic Standard Errors ^{1-2,4}	Common	Worst	Least	Most
Joint Confidence Intervals ^{1-2,5}	Sometimes	Improved	↓	↑
Nonlinear Joint CI. ^{1-2,4-6}	Sometimes	Good	↓	↑
Profile Trace Plots ⁷	Rarely	Better	↓	↑
Support Plane ^{1-2,4-5}	Rarely	Better	↓	↑
Grid Search ^{1-2,5,4}	Rarely	Better	↓	↑
Monte-Carlo ^{1-2,5,4,8}	Rarely	Best	Most	↑
Bootstrap ⁹	Rarely	Best	Most	Least

The most commonly used method for the estimation of the precision of the determined parameters is to utilize the asymptotic standard errors (ASE). These are the simplest to calculate. In the process of performing the parameter estimation most least-squares methods must evaluate a matrix of partial derivatives of the fitting function with respect to the fitting parameters at every data point. The transpose of this matrix times the matrix is known as the Hessian, or information, matrix. The jk element of the Hessian matrix is shown in Equation 2.4.1. The summation is over the l data points.

$$H_{jk} = \sum_{i=1}^n \frac{1}{\sigma_i^2} \left[\frac{\partial G(X_i, \text{parameters})}{\partial \text{parameter}_j} \frac{\partial G(X_i, \text{parameters})}{\partial \text{parameter}_k} \right] \quad 2.4.1$$

For most nonlinear least-squares methods the Hessian matrix is inverted and used to predict the direction to change the parameter values to obtain a decrease in the variance-of-fit. The asymptotic standard errors are simply the square root of the diagonal elements of the inverse of the Hessian matrix times the variance of the parent population of the experimental uncertainties contained within the data (see Equation 2.4.2 where k refers to a particular parameter).

$$ASE_k = \sqrt{\sigma^2 H_{kk}^{-1}} \quad 2.4.2$$

The asymptotic standard error method requires all of the general assumptions for least-squares regression and in addition assumes that:

- 6) The parent population variance of the experimental uncertainties, σ^2 , can be approximated by the sample variance-of-fit, s^2 , *i.e.*, that there is a near infinite number of data points.
- 7) The fitting function is linear near the minimum of the variance.
- 8) The fitting parameters are orthogonal, *i.e.*, all of the off diagonal elements of the inverse of the Hessian matrix are zero. This implies that the covariances between all the parameters are equal to zero.
- 9) The confidence intervals about the most probable parameter values are symmetrical.

All four of these assumptions are invalid for an arbitrary nonlinear fitting function.

However, it is usually possible to collect enough data points so that assumption 6 above is reasonable. Assumption 7 is commonly, but not always, reasonable. It is interesting to note that assumptions 6 and 8 are also invalid for an arbitrary linear fitting function (*i.e.*, 7 and 9 are valid for linear models). Therefore, asymptotic standard errors should probably not be used for any nonlinear or linear least-squares problems.

The Monte-Carlo method is one of the simplest to program, has almost the least assumptions, but requires approximately 1000 times as much computer time as the original least-squares regression to evaluate the probability distributions for the estimated parameters. The actual procedure is simple. First, perform the nonlinear regression to evaluate the parameters of the fitting equation with the highest probability of being correct. Second, the optimal model parameters are used to simulate a set of experimental data. This simulated set of data must contain observations at exactly the same independent variables, *i.e.*, the X-axis. The simulated data set must also contain a realistic amount of random noise (*e.g.*, experimental uncertainties). The major assumption of the Monte Carlo method is that the experimental uncertainties can be simulated with a realistic magnitude and distribution function. Third, the nonlinear regression procedure is repeated on the simulated data to obtain apparent values for the parameters being estimated. Steps two and three are repeated a large number of times (*e.g.*, 500-1000 times) and the apparent parameter values from each cycle saved. The sets of apparent values of the parameters, from the multiple cycles of steps two and three, are then used to create distributions of apparent parameter values. These distribution correspond to the complete probability distributions of the estimated parameters.

The Bootstrap method is extremely analogous to the Monte-Carlo method. The only difference being how the pseudo-random experimental noise is generated. For the Monte-Carlo method a Gaussian distributed pseudo-random number generator is used. For the “experimental noise” is generated by randomly selecting from the observed

weighted residuals. Approximately 63% of the residuals are randomly selected. The remaining 37% of the residuals are resampled from the first 63%. These are applied to the data and the procedure is repeated as in the Monte-Carlo method.

Grid search methods are the simplest to program, but have more restrictive assumptions than the Monte-Carlo method and the amount of computer time required increases as the power of the number of parameters being simultaneously estimated. As an example consider the fitting equation, or model, to be a single exponential decay.

$$I(t) = AMP e^{-t/\tau} \tag{2.4.3}$$

This model contains two parameters, an amplitude AMP and a fluorescence lifetime τ . In the grid search method the apparent variance-of-fit (Equation 2.1.3) is evaluated for each of several hundred different values of AMP, several hundred different values of τ , and all of the cross combinations. The optimal value is simply the one

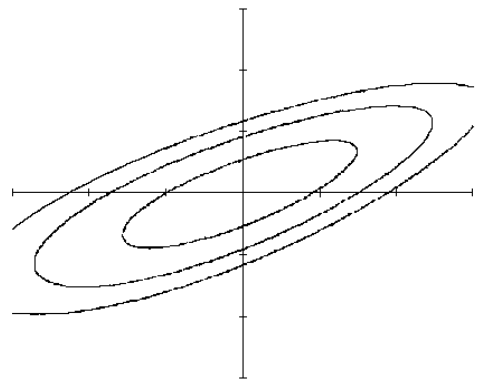


Figure 2.4.1

with the lowest sample variance-of-fit. A typical contour plot of the sample variance-of-fit is plotted as a function of AMP and τ , in the Figure 2.4.1. The individual contours correspond to levels of constant sample variance-of-fit. The probability corresponding to any particular contour can be evaluated from:

$$\frac{SSR}{SSR_{\min}} = 1 + \frac{j}{n-j} F(j, n-j, 1-P) \tag{2.4.4}$$

where SSR_{\min} is the sum of the weighted squared residuals (as defined in Equation 2.1.3) at the minimum of the variance space; SSR is the sum of the weighted squared residuals for a particular contour level; n is the number of data points; j is the number of parameters being simultaneously estimated; F is the upper 1-P quantile for the Fisher's F distribution with j and $n-j$ degrees of freedom⁶. Please note that this equation assumes that the fitting equation is a linear equation. The contour lines in Figure 2.4.1 correspond to one, two, and three standard deviations of confidence. Note that the parameters are partially correlated. If one parameter is moved from the optimal value then the value of SSR increases. However, if the other parameter is altered along the major axis of the elliptical region the change in SSR is, in part, compensated.

This correlation is also indicated by the off diagonal elements of the inverse of the Hessian matrix (Equation 2.4.1) being non-zero. The correlation coefficient between the j and k parameters is given by Equation 2.4.5.

$$CORRELATION_{jk} = \frac{H_{jk}^{-1}}{\sqrt{H_{jj}^{-1} H_{kk}^{-1}}} \quad 2.4.5$$

Joint Confidence Intervals model the confidence regions, as in Figure 2.4.1 as an off axis multi-dimensional ellipse. Joint Confidence Intervals thus assume that the fitting parameters are not orthogonal. Therefore the third assumption need not be made. Thus, Joint Confidence Intervals are always preferable to Asymptotic Standard Errors. However, for nonlinear fitting equations assumptions 7 and 9 are still invalid.

The Nonlinear Joint Confidence Interval, the Profile Trace Plots, and the Support

Plane methods are similar to the Grid Search method in that they search the space looking for significant increases in the sample variance-of-fit above the minimum value. For nonlinear problems the confidence regions are not, in general, elliptically shaped. Therefore these search procedures are preferred. They are different from the Grid Search method in that they employ more rational search algorithms than the brute force grid search method. For example, Nonlinear Joint Confidence Intervals search only along the axes of the multi-dimensional ellipse defined by the Joint Confidence Intervals. These search methods are not as good as a complete Grid Search, but they usually work very well and require substantially less computer time.

The Support Plane method searches each of the estimated parameters independently of the others. Thus, they require an amount of computer time that increases proportional to the number of parameters being estimated instead of a power as in the grid search method. The first step is to perform the nonlinear regression to evaluate the optimal parameter values that correspond to the minimum of the sample variance-of-fit. Second, a plot of apparent sample variance-of-fit as a function of each parameter value is generated. This is accomplished by a series of nonlinear regressions where a particular parameter is fixed at series of values near its optimal value. All other parameter values and the apparent sample variances-of-fit being reevaluated by the regression procedure. The second step is repeated for each of the parameters in turn. Thus, a range of acceptable values for every parameter can be evaluated for any desired confidence level. However, please note that only one of the parameters is being specified. Thus, the equation 2.4.4 takes the form,

No exact theory exists for the evaluation of the precision of parameters estimated by nonlinear regression methods. This section has presented several approximate methods. Each of these methods make some limiting assumptions about the data and thus the "answers" obtained are dependent upon the method. It is, however, clear that Asymptotic Standard Errors should not be used to evaluate the precision of parameter values estimated by nonlinear regression methods.

2.5 REFERENCES

1. M. L. Johnson and L. M. Faunt, "Parameter Estimation by Least-Squares Methods", *Methods in Enzymology* 210, 1-37, 1992.
2. M. L. Johnson, "Use of Least-Squares Techniques in Biochemistry", *Methods in Enzymology* 240, 1-22, 1994.
3. M. Straume and M. L. Johnson, "Analysis of Residuals: Criteria for Determining Goodness-of-Fit", *Methods in Enzymology* 210, 117-129, 1992.
4. M. Straume, S. G. Frasier-Cadoret, and M. L. Johnson, "Least-Squares Analysis of Fluorescence Data", in *Topics in Fluorescence Spectroscopy* 2, 177-240, 1991.
5. G. E. P. Box, *Ann. N. Y. Acad. Sci.* 86, 792, 1960.
6. M. L. Johnson and S. G. Frasier, "Nonlinear Least-Squares Analysis", *Methods in Enzymology* 117, 301-342, 1985.
7. D. G. Watts, "Parameter Estimates from Nonlinear Models", *Methods in Enzymology* 240, 23-36, 1994.

8. M. Straume and M. L. Johnson, "Monte Carlo Method for Determining Complete Confidence Probability Distributions of Estimated Model Parameters", *Methods in Enzymology* 210, 117-129, 1992.
9. B. Efron and R. J. Tibshirani, "An Introduction to the Bootstrap", Chapman Hall, New York, 1993.

3.0 VIEWGRAF: Examine and Print a Graph

VIEWGRAF is a program to examine and print the ASCII files created by the CFS_LS least-squares parameter estimation program. These ASCII files contain a series of "commands" that instruct the VIEWGRAF program to either display a graph (or figure) on the screen, print a graph to a dot-matrix or postscript printer, and/or save a secondary file in any, or all, of several "output file types." The objective being to allow the user to create custom graphs by first editing the ASCII file with the users favorite ASCII text editor and subsequently redraw the customized graph with the VIEWGRAF program.

The allowed "output file types" and the default file extensions are shown in the first table in the appendix. Virtually any type of graphics device can be accommodated within one of these general classes of output. Some of the various supported hardware is shown in the tables of the CFSSETUP manual.

The use of the VIEWGRAF program is simple. A single operating system command of the form

```
VIEWGRAF input_file [output_file]
```

is used to start the program and define the input_file name. The input_file must already exist and be of the CFS_LS / VIEWGRAF ASCII graphics files (.grf). The output file is optional and probably not be needed. The default output file for an IBM PC is PRN. This file format is described in the file format section. Very little error checking is

performed on these files so it is possible that the VIEWGRAF program may die if the file format is not correct.

The first program output will be a graph on the screen. Note, the hardcopy devices have much better resolution than does the computer screen. Thus a ragged looking graph on the screen will look better in real life. To proceed enter a carriage return <CR>. You will now see a menu with SAVE, PRINT and EXIT options. You may use the mouse and or cursor keys to select an option. You may see further menus for the file type, file name and (dots per inch) resolution. Just select the appropriate options when the menu appears.

4.0 MAKEFONT: Roll Your Own Character Font

Some users may find it of use to be able to define their own software graphics fonts. To that end, the MAKEFONT program is included within this package.

The use of the MAKEFONT program is quite simple. A single line operating system command is required.

```
MAKEFONT input_file output_file
```

The input_file must be of the format of the Software Character Set Source Data Files described in the file format section. The output_file file must not already exist.

6.0 INTERNAL FILE FORMATS

5.1 CFS_LS.INI : The initialization file.

This file is normally generated by the CFSSETUP program. The user will not normally be required to manually alter this file. A typical program initialization file is of the form to the right.

A line by line description of this file follows:

```
CMD_FILE=CFS_LS.CMD
PRN_FILE=LPT1
GRF_FILE=CFS_LS.GRF
DISPLAY=10
HIGHLIGHT_BACKGROUND=BLUE
HIGHLIGHT_FOREGROUND=WHITE
MOUSE=2
NORMAL_BACKGROUND=BLACK
NORMAL_FOREGROUND=GREEN
PRINTER=14
PRN_CLASS=4
PLOTTER=1
WINDOW_INPUT=.TRUE.
MODE11 = 61 0 0 0 1 1280 1024
```

Typical CFS_LS.INI file

```
CMD_FILE=CFS_LS.CMD
```

When the CFS_LS program runs it always creates a batch mode command file of all of the commands that were used during the program execution. These are useful when used with the "@" command of the CFS_LS program. The name of the automatically generated command file is given by the CMD_FILE line of the initialization file.

```
GRF_FILE=CFS_LS.GRF
```

When the CFS_LS program creates a graph it also creates an ASCII text file containing the graphics information for the last graph generated. The name of the automatically generated command file is given by the GRF_FILE line of the initialization

file. This file can be used as a input to the VIEWGRAF program.

```
DISPLAY=10
```

The DISPLAY line specifies the display type to be used. For this example a IBM PC VGA monitor with analog color monitor and the DOS operating system. The possible choices are shown Appendix Tables IV, V and VI of the **CFSSETUP**

Reference manual.

```
HIGHLIGHT_BACKGROUND=BLUE  
HIGHLIGHT_FOREGROUND=WHITE  
NORMAL_BACKGROUND=BLACK  
NORMAL_FOREGROUND=GREEN
```

These lines specify the "foreground" and "background" colors for the "normal" and "highlight" modes of the CFS_LS program. The optional choices are Black, Cyan, Red, Blue, Yellow, Magenta, Green and White.

```
MOUSE=2
```

The MOUSE option specifies the type of mouse being utilized. In this case a two button mouse. See Appendix Table II of the **CFSSETUP Reference Manual**.

```
PRN_FILE=LPT1
```

When the CFS_LS program runs it always creates a listing file. The name of the automatically generated command file is given by the PRN_FILE line of the initialization file.

```
PRINTER=14
```

This specifies the type of printer to generate raster (.ras) graphics files for. Refer to Appendix Table I of the **CFSSETUP Reference Manual** for more information.

```
PRN_CLASS=4
```

This specifies the general class of printer in use: 0=no printer, 2=PostScript and 4=Dot-Matrix printer.

PLOTTER=1

This specifies the type of plotter to generate HP-GL (.gl) graphics files for. Refer to Appendix Table III for more information.

WINDOW_INPUT=.TRUE.

This specifies that the initial user interface of the CFS_LS program is a windows (not Microsoft Windows) interface. .FALSE. implies a command line interface. Batch commands such as "@" always use the command line interface.

MODE10, MODE11, MODE12

The mode commands are PC specific and are used to specify the VGA and SVGA graphics mode to be used by CFS_LS. These options are optional. They are discussed with examples in TABLE II of this manual and TABLE VI of the **CFSSETUP Reference Manual**.

5.2 ASCII graphics files [.grf]

This file consists of a series of "commands" starting in column one on successive lines of the file. The objective of these commands is to allow the user to create virtually any publication quality plot as a simple series of commands. These are discussed in detail in the **VIEWGRAF** manual.

5.3 Software Character Set Source Data Files¹

This section describes the format of the ASCII source data file required to generate your own software character sets. Character sets can be converted from ASCII to binary format using the MAKEFONT utility.

The source data file defines the shapes of all the printable ASCII characters, in the form of a series of vectors. The first line of the file is a comment, followed by a single line defining the character grid size and character space:

`n_x_cell n_y_cell x_space y_space` in (213,2F8.2) format

Each following line defines one and only one character shape, in strict ascending order from character 33 (!) up to character 126 (~). Including the initial comment and grid size lines, there should therefore be a total of 96 lines in the file.

Each character definition line consists of a series of digits and upper case letters, interspersed with the occasional lower case 'u,' all starting in column 3. Columns 1 and 2 are not

```
'Standard' software character set
 7 12  1.28  1.08
! 3233u363B
" 1B19u4B49
# 222Bu525Bu0565u0868
$ 0413536466571708091A5A69u323B
```

used, but it is useful to include the character defined by that line in column 1.

The sequences of numbers and upper case letters describe the character

¹Note that the software character generation routines are from the INTERACTER package by Interacter Software Services LTD, Stafford, United Kingdom. This section of this manual closely copies appendix D of the INTERACTER Users Guide.

be used as a 'pen-up' character. An equals sign, for instance, must be drawn as two separate lines. This can be defined as 0464u0868, which decodes as 'draw from (0,4) to (6,4), then move to (0,8) and draw to (6,8)'.
'

As noted earlier, the second line of the file not only defines the character size, but also the character space. The two parameters `x_space` and `y_space` control the amount of space left around each character when it is printed. A value of 1.0 causes characters to 'butt' together. Values greater than one leave a proportionate amount of space around the character and values less than one cause characters to overlap (the latter is useful for Script type character sets). The Standard character set uses an `x_space` value of 1.28 giving a neat separation between characters. The two character space parameters can be varied independently of the character definitions themselves.

7.0 ACKNOWLEDGMENTS

This software uses INTERACTER from Interactive Software Services Ltd., 25 St. Michaels Close, Penkridge, Stafford ST19 5AD, United Kingdom. It uses the cubic spline routines from G.E. Forsythe, M.A. Malcolm, and C.B. Moler "Computer Methods for Mathematical Computations." It also uses the RAN3 random number generator from W. H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling "Numerical Recipes, The Art of Scientific Computing."

MS DOS and Microsoft Windows are trademarks of Microsoft Corporation.

8.0 APPENDIX: Supported Hardware

TABLE I: Graphics Output File Types

Extension	Type
.gl	Hewlett-Packard Graphics Language (HP-GL)
.ps	PostScript
.ad	Acorn Draw format
.ras	Raster Graphics (dot-matrix/laser/inkjet/etc. printers)
.tek	Tektronix 4014 (DEC LN03+)
.pcx	PCX
.eps	Encapsulated PostScript
.grf	CFS_LS ASCII graphics format

TABLE II: IBM PC Clone VGA and SVGA Graphics Modes
copyright Interactive Software Services 1989-1994

INTERACTER (and thus CFS_LS) supports 4 non-standard screen modes numbered 9 to 12 which can be activated by inserting suitable 'MODEnn' keywords in the INTERACTER (CFS_LS.INI) initialization file. If you have a suitable combination of monitor/adapter, entries from the appropriate table for your particular display should be added to your CFS_LS.INI initialization file using a text editor (or word processor in ASCII mode).

The supported non-standard screen modes are as follows

INTERACTER Mode Number	Mode Type
9	640x400 'Olivetti/AT&T' monochrome graphics
10	SVGA extended 16 colour text mode
11	SVGA extended 16 colour graphics mode
12	SVGA extended 256 colour graphics mode

Notes :

- (1) The dimensions of modes 10-12 are definable as part of the MODE10, MODE11 and MODE12 keywords. The default dimensions are 132x25, 800x600 and 640x480 respectively.
- (2) Resolutions higher than 640x480 require a multisync or Super-VGA type monitor. On some (but not all) boards, 132 column mode also requires such a monitor. Be sure to use a suitable monitor/adapter combination.

- (3) The vast majority of MOUSE.COM drivers do not operate correctly in these non-standard modes. In particular, movement can be erratic and/or the mouse cursor is not maintained correctly (if at all). In this situation, INTERACTER can be instructed to work round these problems by adding a space-delimited C at the VERY END of the line which begins with the appropriate MODEnn keyword. e.g.
Correct : mode11 = 84 0 0 0 5 800 600 (800x600x16 graphics) C
Correct : mode11 = 84 0 0 0 5 800 600 C
Incorrect : mode11 = 84 0 0 0 5 800 600 C (800x600x16 graphics)
- (4) The precise mode numbers used by particular video board manufacturers can sometimes vary. This is rare, but users should check the documentation supplied with their video board before including the supplied example MODEnn entries in their CFS_LS.INI initialization file.
- (5) The 640x400 mono graphics mode (MODE9) rarely turns up on modern video cards, having been superseded by the VGA standard. However, some older sub-VGA hardware may support this mode. Most commonly, this is available on older Toshiba portables.

VESA Support

Many of the above cards are also VESA compatible and should therefore also be usable with the VESA BIOS option ('chipset' number 18).

One of the supported chipsets (number 18) is designed to be used with any SVGA which provides a VESA BIOS (either in hardware or in a TSR).

Provided you know which type of chipset a given VGA is based on, you will commonly have a choice of using manufacturer-specific MODEnn settings or VESA-compatible equivalents.

When selecting MODE10 settings, to identify an extended 16-colour text mode, there is no particular advantage in using VESA-compatible settings over manufacturer-specific ones. The VESA standard defines a smaller number of extended text modes than some individual manufacturers, so required mode dimensions are probably the only consideration here. i.e. Use whichever MODE10 settings give you the text mode size you require. Performance is not affected.

Manufacturer-specific MODE11/MODE12 settings cause the video hardware to be accessed directly for bank-switching purposes. This is normally quicker than the VESA BIOS (except in 800x600x16 mode) and generally more reliable. Manufacturer-specific

settings should therefore always be used as a first preference, where they are found to work. However, the VESA option provides a useful fallback option allowing operation on an even wider range of otherwise unsupported SVGA's.

The main additional benefit of using the VESA BIOS option is that you do not need to identify the chipset which a VGA is based on, to get up and running.

VESAINFO Program

A program called VESAINFO.EXE is supplied which performs two useful tasks :

- (a) It identifies whether a VESA compatible video BIOS is currently installed. If not, a VESA TSR may have been supplied with the PC or SVGA adapter on a utilities disk. Alternatively, the 'Universal VESA VBE' video BIOS extension from SciTech Software is recommended.
- (b) If a VESA compatible video BIOS is found, VESAINFO will list the MODEnn keywords for the screen modes which the BIOS claims to support. Add the required MODEnn records to your initialization file. However, see also the comments below.

Unfortunately, some VESA BIOS's incorrectly report the availability of modes which are supported by the current SVGA adapter, but not by the current display. This can occur on some notebooks where a VESA BIOS may report the availability of 800x600x16 mode even though the internal display only supports 640x480 resolution. Most VESA BIOS's do not suffer from this problem and correctly adapt their mode availability reports according to the type of monitor which is currently attached. This lack of consistency in what is meant to be a 'standard' is, to say the least, regrettable. The only useful advice which can be offered here is : Don't try to use screen modes which the display obviously cannot support, even if the VESA BIOS says they're available!

Sample initialization file entries for VESA compatible VGA's. (VESA = Video Electronics Standards Association). Add one or more of the following MODEnn keywords to your CFS_LS.INI initialization file, according to the settings reported by VESAINFO.EXE. If you experience mouse problems, add a C at the very end of the MODEnn record, to force INTERACTER to maintain its own mouse cursor.

Notes :

- (1) Some SVGA's require a TSR to be loaded to provide VESA compatibility, whilst others include VESA support in hardware.
- (2) A program called VESAINFO.EXE is supplied which will report the MODEnn settings required to select the screen modes which a VESA compatible SVGA claims to support. If a VESA BIOS is not present, VESAINFO will report this.

Some VESA BIOS's (but not all) inaccurately report modes as being available even though the current display cannot possibly support them, e.g. they may report 800x600 modes as being available on a 640x480 notebook screen. Don't try to use screen modes which the display obviously cannot support, even if the VESA BIOS says they're available !

- (3) Always try manufacturer-specific MODEnn keywords first. It is normally better to use VESA as a second choice, if manufacturer-specific settings do not appear to work.
- (4) 1280x1024x256 mode requires 2mb of video RAM.

For an extended 16-colour text mode select :

mode10 = 20226 265 0 0 18 132 25 (132x25 16-colour text)

The following larger text modes are also supported under 386/INTERACTER :

mode10 = 20226 264 0 0 18 80 60 (80x60 16-colour text)

mode10 = 20226 266 0 0 18 132 43 (132x43 16-colour text)

mode10 = 20226 267 0 0 18 132 50 (132x50 16-colour text)

mode10 = 20226 268 0 0 18 132 60 (132x60 16-colour text)

For Super VGA 16 colour graphics select one of :

mode11 = 20226 258 0 0 18 800 600 (800x600 16-colour graphics)

mode11 = 20226 260 0 0 18 1024 768 (1024x768 16-colour graphics)

mode11 = 20226 262 0 0 18 1280 1024 (1280x1024 16-colour graphics)

For Super VGA 256 colour graphics select one of :

mode12 = 20226 256 0 0 18 640 400 (640x400 256-colour graphics)

mode12 = 20226 257 0 0 18 640 480 (640x480 256-colour graphics)

mode12 = 20226 259 0 0 18 800 600 (800x600 256-colour graphics)

mode12 = 20226 261 0 0 18 1024 768 (1024x768 256-colour graphics)

mode12 = 20226 263 0 0 18 1280 1024 (1280x1024 256-colour graphics)

More information can be found in the **VIEWGRAF** and **CFSSETUP** manuals.

9.0 INDEX

.bst	see also CFS_LS.BST	19
.chr	see also MAKEFONT	87
	see also fonts	87
.cmd	6
	see also CFS_LS.CMD	20
.exe	CFS_LS.EXE	6
	MAKEFONT.EXE	87
	VIEWGRAF.EXE	85
.grf	7, 21, 85, 88, 90, 95
	see also CFS_LS.GRF	21
	see also VIEWGRAF	90
.ini	88
	see also CFS_LS.INI	88
	see also INTERACTER initialization files	95
	CFS_LS.INI	19, 21, 88
	vesa.ini	97
.prn	see also CFS_LS.PRN	19
asymptotic standard errors	76
Bootstrap Method	80
CD, change directory		
CD	58, 59
CFS_LS	6, 58, 85, 88
CFS_LS Commands		
!	20, 59
#	58
@	20, 59
BROWSE	19, 60
commands	20
COOKIE	59
DATA	10, 23, 26, 27, 59
FIT	13, 19, 20, 60
FUNCTION	9, 28, 59
GLOBAL	61
GUESS	12, 19, 20, 59
HELP	9, 58
PLOT	15, 17-19, 21, 60
QUIT	9, 12, 18, 58
RESIDUALS	19, 60
SHELL	20, 58

SIMULATE	60
SMOOTH	60
CFS_LS Data Files	22
FD-CFS	22
FD-SPEX	26
IBH4-ASC	23
IBH4-ASC/B	24
IBH4-BIN	24
IBH4-BIN/B	24
TD-PRA	25
TD-PRA/B	26
CFS_LS Output Files	19
?????.TMP	21
CFS_LS.BST	19
CFS_LS.CMD	20
CFS_LS.DBG	21, 63
CFS_LS.GRF	21
CFS_LS.PRN	19
CFS_LS.CMD	7
CFS_LS.INI	88, 95
CFSSETUP	8, 88
see also CFS_LS.INI	88
change directory	
CD	58, 59
Colors, screen	89
Confidence Intervals	75
Displays	
SVGA	95
VESA	96
VGA	95
FAQ	63
files	
.grf	85, 88, 90, 95
?????.TMP	21
CFS_LS.BST	19
CFS_LS.CMD	20
CFS_LS.DBG	21, 63
CFS_LS.GRF	21
CFS_LS.INI	19, 21, 88
CFS_LS.PRN	19
INTERACTER initialization file	95
Fitting Functions	28
ET1DET	31, 52
ET2DET	53

ET3DET	54
Frequency-Domain Transformations	30
GAUDIS	40
GAUTAU	38
HET_JUMP2	55
HETANL	10, 31, 36
LORDIS	44
LORTAU	42
RECDIS	48
RECTAU	46
SKUTAU	50
Time-Domain Transformations	28
fonts	87
see also MAKEFONT	87
Frequency-Domain	
FD-CFS Files	22
FD-SPEX Files	26
Global Fits	10
Goodness-of-Fit Criterion	69
Grid Search Methods	80
Hardware	95
Intensity Decay Laws	28
ET1DET	31, 52
ET2DET	53
ET3DET	54
Frequency-Domain Transformations	30
GAUDIS	40
GAUTAU	38
HET_JUMP2	55
HETANL	10, 31, 36
LORDIS	44
LORTAU	42
RECDIS	48
RECTAU	46
SKUTAU	50
Time-Domain Transformations	28
INTERACTER	94, 95
INTERACTER Initialization file	
see also CFS_LS.INI and CFSSETUP	88
MAKEFONT	87
Monte-Carlo Method	79
Nelder-Mead Method	67
Parameter Estimation Methods	66
README	63

Reports of Trouble	63
Screen colors	89
SPEX2CFS.EXE	27
Standard Errors of Parameters	75
Support Plane Method	82
SVGA	95
System Requirements	62
IBM PC Clones	62
Time-Domain	
IBH4-ASC	23
IBH4-ASC/B	24
IBH4-BIN	24
IBH4-BIN/B	24
TD-PRA Files	25
TD-PRA/B Files	26
Tutorial	8
Uniqueness of Parameters	69
VESA	96, 97
VESAINFO	97
VGA	95
VIEWGRAF	7, 21, 85, 90