Contents

List of Figures

1 Introduction

REPLAY is an interactive command language program that displays data in various ways; performs some algebraic operations on data; and does least squares curve–fitting.

To use the program, it is necessary to make the following assignments:

```
$  ASSIGN  LIB0:[FIOWA.REPLAY]REPLAY.HLB     REPLAYHLP
$  ASSIGN  LIB0:[FIOWA.REPLAY]AMPERSAND.HLB  AMPERHLP
$  ASSIGN  LIB0:[FIOWA.REPLAY]REPLAY.TXT     REPLAYTXT
```

To run the **REPLAY** program, enter:

```
$ RUN LIB0:[FIOWA.REPLAY]REPLAY
```

REPLAY makes it easy to examine and display binned, or count, data. In particular, one may readily examine data accumulated with the **FIOWA** histogramming subroutines. The data analysis program must save the common blocks of **FIOWA** in a disk file according to a specified format. This file may then be read by **REPLAY**, giving the user access to all the histograms and scatterplots accumulated by **FIOWA**. A file is produced automatically, called `REPLAY.REC`, which is a record of the commands you have entered and the responses that were displayed by **REPLAY**.

There are three classes of commands that are recognized by **REPLAY**:

1. Commands relating to data operations. These begin with a slash, `/`
2. Commands dealing with graphics control. These begin with an ampersand, `&`
3. Expressions permit the user to perform algebraic operations on data. These have no special preceding symbol.

Commands may be user–defined by means of the assignment ":==", i.e., it is possible to define a new **REPLAY** command, which *must* begin with a slash, /, that is equivalent to an existing **REPLAY** command *and its associated argument*. For example, the following two assignments:

```
/R:==/RESTORE [MYID]FILE.SAV
/S:==&SET YLOG 10
```

defines the command `/R` to be equivalent to `/RESTORE [MYID]FILE.SAV` and defines the command `/S` to be equivalent to `&SET YLOG 10`.

The commands that have been defined may be displayed by means of the command `/SHOW`.

Data entities recognized by **REPLAY** may be categorized by dimension as follows:

Data Type	Description
Scalar	A single number
Vector	A one dimensional array
Matrix	A two dimensional array
H–set	A set of two or three vectors of the same length: an independent array, a dependent array and an optional error array
S–set	A set of two vectors of any length, say n_1 and n_2, and one or two matrices of dimension $n_1 \times n_2$: two independent arrays, a matrix of dependent values and an optional error matrix

Each data entity is assigned a name, by which it will be recognized by **REPLAY**. All the histograms and scatterplots of a **FIOWA** save file that has been restored have predetermined names. Other data which results from operations on data, must be explicitly given a name by the user.

When using data names in commands or expressions, the name may usually be followed by a *modifier*, which allows the user to specify that only part of the data be considered and/or, in the case of binned data, that the data be rebinned.

2 Modifiers

The modifier allows one to access only part of the data referred to by the data name. It follows immediately after the data name, enclosed in parentheses, `( )`, and contains a number of fields. The fields are separated by slashes, `/`, and contain information on which part of the data you would like to reference. The fields have the general form:

```
/KEY=key1,key2/
```

With the modifier, you may specify data subsets or increment/rebinning factors.

2.1 "Reduced" Data Types

One of the most useful aspects of modifiers is their ability to "reduce" a data type. For example, the `/PLOT` and `/HISTOGRAM` commands will accept only one dimensional arrays as arguments. However if you chose a data subset such that a scatterplot is reduced to a one dimensional histogram, then the reduced scatterplot becomes a valid argument for the `/PLOT` and `/HISTOGRAM` commands.

For the purpose of the **REPLAY** expression evaluator, the "reduced" data type is the only one considered. For example, if `V` is a vector, then `V(1)` is considered a scalar by the expression evaluator. However `V(I=1,10)` is a vector of length 10.

2.2 Data Subsets

A data subset is a continuous subset of the data. You may specify a window in the x–direction for vectors, matrices, H–sets or S–sets and a window in the y–direction for matrices or S–sets.

1. Data Subsets Defined by Array Index Value

   Every **REPLAY** array is an indexed array. For a vector of length n, the index numbers run from 1 to n. For a matrix of dimension $n \times m$, there is a pair of index numbers, the first running from 1 to n and the second from 1 to m. The keywords for indicating a data subset by array index value are `I` and `J`. The `I` keyword is used for vectors and H–sets and the x–direction of matrices and S–sets. The `J` keyword is used for the y–direction of matrices and S–sets.

   Consider the example of a vector of length n. Then the following configurations of the `I` field are permitted:

Field	Low Index	High Index	Error Conditions
/I=i_1, i_2/	i_1	i_2	$i_1 \leq 0$ or $i_2 \leq 0$ $i_1 > n$ or $i_2 > n$ $i_1 > i_2$
/I=i_1,/	i_2	n	$i_1 \leq 0$ or $i_1 > n$
/I=,i_2/	1	i_2	$i_2 \leq 0$ or $i_2 > n$
/I=i/	i	i	$i \leq 0$ or $i > n$
/i/	i	i	$i \leq 0$ or $i > n$

i_1, i_2 and i may be either explicit integers, as in `/I=1,50/`, or they may be a scalar data names, as in `/I=N,M/` where `N` and `M` are use–defined scalars. These may be mixed, as in `/I=1,M/`. *Only* scalar data names are permitted here. "Reduced" scalars are not permitted.

The same configurations and restrictions apply to the `J` field. An error is generated if you use a `J` field in a modifier following a vector or H–set.

The last configuration, /i/, is of special note. In this case the keyword `I` is missing. When reading the modifier, the first such field, a single integer without keyword, encountered will be considered an `I` field. The second such field encountered will be considered a `J` field, and thus invalid for vectors and H–sets. This configuration allows you to quickly refer to the n^{th} element of vector `V` with the familiar notation `V(n)`.

2. Data Subsets Defined by Data Value

In the case of H–sets and S–sets it is always true that the independent array(s) of the set are strictly increasing, from the definition of sets. Thus the data subset limits can be expressed as data values, rather than array index numbers. The data values are converted internally to array index values. The keywords for specifying data subsets by data value are `X` and `Y`. `X` is used for H–sets and the x–direction of S–sets; `Y` is used for the y–direction of S–sets. *Data subsets defined by data value are permitted only for sets!*

In the case of binned data, the array index number is assigned by calculating into which bin the data value falls. If the data values lies below the first bin it is assigned to the first bin; if it lies above the highest bin, it is assigned to the highest bin. In the case of non–binned data, the independent array is divided into a number of zones, and the zone into which the data value falls is calculated. Let us represent this function which converts data value to array index value by $\mathcal{L}$, where $\mathcal{L}(x)$ is the index number corresponding to value x.

Consider the following example of an H–set of length n, whose independent vector ranges from x_{lo} to x_{hi}. The following configurations are permitted:

Field	Low Limit	High Limit	Error Conditions
/X=x_1, x_2/	$\mathcal{L}(x_1)$	$\mathcal{L}(x_2)$	$x_1 > x_{hi}$ $x_2 < x_{lo}$ $x_1 > x_2$
/X=x_1,/	$\mathcal{L}(x_1)$	n	$x_1 > x_{hi}$
/X=,x_2/	1	$\mathcal{L}(x_2)$	$x_2 < x_{lo}$
/X=x/	$\mathcal{L}(x)$	$\mathcal{L}(x)$	$x < x_{lo}$ or $x < x_{hi}$

x, x_1, x_2 may be either explicit integers or reals, integers will be converted to reals, or scalar data names. The same configurations apply to the `Y` field, except that `Y` is not permitted for H–sets.

3. Rebinning/Increment Factor

These are the `IXINC` and `IYINC`, abbreviated `IX` and `IY`, keywords. These mean two quite different things, depending on whether or not the data is binned or general. Only sets can have the binned classification; all individual arrays are categorized as general.

If a data subset *and* a rebinning/increment factor have been specified in the modifier, the data subset is *always* taken first and the rebinning or increments taken from the subset's low value.

For binned data, this is the rebinning factor. If `IX=`n, then the data in rebinned by a factor of n, i.e., adjacent bins are added together in groups of n. If the length of the x–direction is `N` and $n >$ `N`, an error is generated. If $n = 0$, then the rebinning factor is calculated to be the exact size necessary to rebin this direction to length 1, that is, a projection . This is particularly useful when you do not know exactly how many bins you have, as will usually be the case when you form a data subset by data value. If `N` is not an even multiple of n, the remainder bins are discarded.

For non–binned data, this is the increment factor. If `IX=`n, this means every n^{th} element, starting at the data subset's low value, will be considered, $n = 0$ is invalid for non–binned data. If n is greater than the x–direction length `N` this does not generate an error because it is always possible to get at least one value, this first if $n >$ `N`. For both cases, $n < 0$ generates an error.

3 Data Names

The data name identifies the data for **REPLAY**. User defined data names may contain up to 12 alphanumeric characters, but must start with an alphabetic character (`A` — `Z`). **FIOWA** data names always start with `%`, followed by either an `H`, for a histogram, or an `S`, for a scatterplot, and then an integer between 1 and 100, referring to the index number. If the histogram or scatterplot referred to is not defined, an error message will be printed out to this effect.

In the case where the data is a set, then the elements of the set may be accessed by using *extensions* to the data name. The following extensions are recognized:

H–set			**S–set**		
`.U`	$\Longleftarrow$	independent vector	`.UX,.U1`	$\Longleftarrow$	X–independent vector
			`.UY,.U2`	$\Longleftarrow$	Y–independent vector
`.V`	$\Longleftarrow$	dependent vector	`.V`	$\Longleftarrow$	dependent matrix
`.E`	$\Longleftarrow$	error vector	`.E`	$\Longleftarrow$	error matrix

For example, `%H45.V` would be the array of bin counts of **FIOWA** Histogram #45. If you refer to the error array and it does not exist an error message will be printed out to that effect. The error array for **FIOWA** data is calculated assuming Poisson statistics.

4 Commands Beginning with /

4.1 Executing Files of REPLAY Commands

Command: `/@`
Argument: filename
Qualifiers: none
Examples: `/@FILEONE`
`/@FILETWO.EXAMPLE`

This command allows the user to read in a set of **REPLAY** commands from a file, rather than entering commands interactively from the terminal keyboard. When an end of file is detected, control is returned to the terminal keyboard. If the filename is entered without an extension, the extension `.RCM` is assumed.

4.2 Clearing the Alpha–numeric Terminal Screen

Command: `/AClear`
Argument: none
Qualifiers: none
Example: `/ACLEAR`

This command clears the alpha–numeric, or transparent, portion of the terminal screen only. This command has no affect on the graphics.

4.3 Centroid Calculation

Command: `/CEntroid`
Argument: any 2 one–dimensional arrays of the same length
Qualifiers: none
Example: `/CE %H1`

This command calculates and prints the centroid of a histogram.

Suppose the histogram contains the values x_i and y_i for $i = 1, 2, \ldots, n$. The total number of points in the histogram

$$y_{sum} = \sum_{i=1}^{n} y_i$$

is output first, followed by the peak location and value. The true mean,

$$y_{mean} = \frac{\sum_{i=1}^{n} y_i x_i}{y_{sum}}$$

and the standard deviation,

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} y_i x_i^2 - y_{sum} y_{mean}^2}{y_{sum} - 1}}$$

are listed next, and then the standard deviation of the mean,

$$\sigma_{mean} = \frac{\sigma}{\sqrt{y_{sum}}}$$

is displayed.

If $y_{sum} < 2$ the centroid calculation is not done, and a message to that effect is displayed.

4.4 Clearing Graphics

Command: `/CLear`
Argument: none
Qualifiers: none
Example: `/CLEAR`

This command clears the graphics screen, the hardcopy plot buffers and the `/REPLOT` option.

By default, the graphics is cleared between successive plots. Use the `/CURVE` command to overlay a plot on a current one. To change the default so that successive plots will be overlaid on a current plot, use the `&PLOTCLEAR OFF` command.

This is true even if the plots are being plotted in `&NOSCREEN` mode, i.e., the hardcopy plot buffers only.

4.5 Contour Plotting

Command: `/COntour`
Argument: 2 one–dimensional arrays and a two–dimensional array whose dimensions match the lengths of the x– and y– axis arrays
Qualifiers: `/Hard`
Example: `/CONTOUR %S1`

This command produces a contour plot of a two dimensional array, for example, see Figure 1. Currently there is no control on the number of contours. The default is to have ten contour levels.

If the contours lines are very noisy, try rebinning the data, with the rebinning modifier.

When the `/HARD` qualifier is entered, a hardcopy of the contour plot will automatically be produced on the assigned bitmap device. See the `/OUTPUT` command.

4.6 Entering DCL Commands

Command: `/DCL`
Argument: none
Qualifiers: none
Example: `/DCL`

This command spawns a subprocess, which allows the user to enter DCL commands, without stopping the execution of **REPLAY**. There is no restriction on what can be done in the subprocess, except that files currently open by **REPLAY** will not be accessible, they are locked. To return to **REPLAY**, enter the DCL command `RETURN`.

Figure 1: Contour Plot Example:

Figure 2: Density Plot Example:

4.7 Density Plots

Commands:	`/DENSity, /DENA, /DENN`
Argument:	2 one–dimensional arrays and a two–dimensional array whose dimensions match the lengths of the x– and y– axis arrays
Qualifiers:	`/Hard, /ATARI, /NORMAL`
Example:	`/DENSITY %S1`

This command produces a density plot of a two dimensional array, for example, see Figure 2. Each data point of the array will be represented by a rectangle, where the size of the rectangle represents the relative amplitude at that point.

The `/DENA` command plots the density in fast atari mode, while the `/DENN` command plots the density in normal mode.

When the `/HARD` qualifier is entered, a hardcopy of the density plot will automatically be produced on the assigned bitmap device. See the `/OUTPUT` command.

When the `/ATARI` qualifier is entered, the plot is done in fast Atari mode, overriding the default mode, as set by the `&ATARI` comamnd. When the `/NORMAL` qualifier is entered, the plot is done in normal mode, again, overriding the default mode.

4.8 Describing User Defined Variables

Command: `/DEscribe`
Argument: none
Qualifiers: none
Example: `/DESCRIBE`

This command produces a table of all user defined variables and their attributes.

4.9 Displaying User–defined Variables

Command: `/DImension`
Argument: none
Qualifiers: none
Example: `/DIMENSION`

This command displays any user–defined arrays, giving the name, data type, x–length, y–length, and way the array was defined.

4.10 Emptying Record File

Command: `/EMpty`
Argument: none
Qualifiers: none
Example: `/EMPTY`

This command deletes the current `REPLAY.REC` file and opens a new one. This `REPLAY.REC` file is a record of the **REPLAY** commands you enter and the responses that are displayed by **REPLAY**.

4.11 Quiting REPLAY

Command: `/QUIT, /EXIT`
Argument: none
Qualifiers: none
Example: `/QUIT`

This command stops program execution.

4.12 Fitting

Command: `/FIT`
Argument: an expression
Qualifiers: `/E=phrase`, `/W=phrase`
Examples: `/FIT Y=A*X+B`
`/FIT Y=A*SIN(X)+B*COS(X)`

This command does weighted or unweighted least–squares curve fitting of functions which are linear or non–linear in the varying parameters. Fitting is restricted to one dimensional arrays. There may or may not be an array of associated errors. The function to be fitted must be typed in explicitly.

You can use the `/E` qualifier to specify an array of errors. It will be converted internally to weights according to the formula $WEIGHT = 1 \div ERROR^2$

You can use the `/W` qualifier to specify an array of weights.

For example, suppose `Y` is a one dimensional array of data values functionally dependent on `X`, the corresponding array of independent values. If you wish to see how closely a function, say $\mathcal{F}$, can match the relationship between `X` and `Y`. $\mathcal{F}$ has as its arguments, `X` and one or more parameters, that is, scalars with values which will vary in the fit.

All arguments of $\mathcal{F}$ must be defined. The scalars which are to vary must be freed with the `/FREE` command. They may later be fixed with the `/FIX` command. The command takes the form:

`/FIT/qualifier Y=` $\mathcal{F}$`(X,A1,A2,A3,....)`

The χ^2 and new parameter values are displayed with each iteration. You may interrupt the fit at any time by typing `CTRL-O` and then typing `CTRL-C`. If the fit does not terminate normally, that is, there is an error return or you interrupt the fit, the parameter values remain at what they were before the fit began. Only if the fit terminates successfully are the parameters values reassigned.

4.13 Fixing Scalar Values

Commands:	`/FIX, /FIX?`
Argument:	a list of scalars
Qualifiers:	none
Examples:	`/FIX A,B,C` `/FIX?`

This command fixes the scalar values in any future `/FIT` commands. Scalars are by default fixed until they are freed with the `/FREE` command. The command `/FIX?` generates a list of all fixed scalars.

4.14 Free Scalar Values

Commands:	`/FREE, /FREE?`
Argument:	a list of scalars
Qualifiers:	none
Examples:	`/FREE A,B,C` `/FREE?`

This command frees the scalars to vary in any future `/FIT` commands. Scalars may later be fixed with the `/FIX` command. Scalars are fixed by default. The command `/FREE?` generates a list of all free scalars.

4.15 Help

Command:	`/HELP`
Argument:	command name
Qualifiers:	none
Examples:	`/HELP PLOT` `/HELP`

This command invokes the on–line help facility. Each of the commands is explained. To get help on a specific command, you may enter the command name after `/HELP`.

4.16 Histograms

Command:	`/HIstogram`
Argument:	a pair of one–dimensional arrays of the same length
Qualifiers:	`/Hard`

Examples:	`/HISTOGRAM %H1`
	`/HISTOGRAM %S1(J=1)`
	`/HISTOGRAM %S1(IY=0)`

This command is used to histogram a pair of one dimensional arrays. The output appearing on the screen is compressed so that the entire histogram will fit on the screen. The hardcopy output is full–size, including bin counts for each channel.

When the `/HARD` qualifier is entered, a hardcopy of the histogram will automatically be produced on the assigned bitmap device. See the `/OUTPUT` command.

The first example above histograms **FIOWA** histogram #1, while the second example histograms the first row of **FIOWA** scatter plot #1. The third example histograms the x–axis projection of scatter plot #1.

4.17 Listing Data

Command:	`/LIst`
Argument:	1 or more one–dimensional arrays of any length
Qualifiers:	`/Error, /NOError`
Examples:	`/LIST %H1,%H2(X=-10,10)` `/LIST %S1(I=50)`

This command lists the values and index numbers of one or more one dimensional arrays.

If the qualifier `/ERROR` is entered, then the errors, if any, are included in the listing. Errors on **FIOWA** data are calculated assuming Poisson statistics.

The first example above lists all of histogram #1 and histogram #2 in the range $x = -10.0$ to $x = 10.0$. The second example lists the 50^{th} column of Scatterplot #1.

4.18 Assigning a Default Bitmap Device Queue

Command:	`/OUTput`
Argument:	bitmap device queue name
Qualifiers:	none
Example:	`/OUTPUT TRMF`

This command assigns a default bitmap device queue to use when the `/HARD` qualifier is entered with a command. If the `/OUTPUT` command has not been entered, the default printer queue is `SYS$PRINT`.

The commands `/HISTOGRAM`, `/DENSITY`, `/SCATTERPLOT`, `/PLOT` and `/CONTOUR` generate hardcopies automatically on the default bitmap device queue when the `/HARD` qualifier is included. In addition, if the `/RELEASE` command is given without the `/OUTPUT` qualifier, printing will be done on the default bitmap device queue.

The example above directs bitmap output to the queue named `TRMF`.

4.19 Plotting a Graph

Commands:	`/PLot, /CUrve, /PA, /PN`
Argument:	2 or 3 one–dimensional arrays of the same length
Qualifiers:	`/Error, /Hard, /HIstogram, /NOHistogram,` `/CENTROID, /CLEAR, /ATARI, /NORMAL`

Examples:	`/PLOT %H1`
	`/PLOT %S1(J=2)`
	`/CURVE %H1`

This command produces a graph of a pair of one dimensional arrays. See Figure 3.

The `/PLOT` command draws axes and the data curve. By default, the screen is cleared before the plot is done. There are two ways to overlay another data curve on the current set of axes:

1. use the `/CURVE` command instead of `/PLOT`
2. use the command `&PLOTCLEAR OFF` and then use the `/PLOT` command

Errors may be specified for the second array. To obtain error bars on the plot, you can enter the array of errors as the third argument, or, if an H–set contains an array of errors, these errors may be plotted by including the `/ERROR` qualifier.

When the `/HARD` qualifier is entered, a hardcopy of the graph will automatically be produced on the assigned bitmap device. See the `/OUTPUT` command.

When the `/HISTOGRAM` qualifier is entered, the plot is done in bar–graph fashion. If the `/NOHISTOGRAM` qualifier is entered, the plot is done in normal line graph form. If neither qualifier is entered, the current value of the graph parameter `HISTYP` is used. A value for `HISTYP` is assigned with the `&SET HISTYP` command.

When the `/CENTROID` qualifier is entered then the centroid will be calculated and displayed.

Figure 3: Plot Example:

When the `/CLEAR` qualifier is entered, the screen is cleared before plotting.

When the `/ATARI` qualifier is entered, the plot is done in fast atari mode. This overrides default mode, as set by the `&ATARI` command. The command `/PA` is equivalent to the command `/PLOT/ATARI`. When the `/NORMAL` qualifier is entered, the plot is done in normal mode, overriding the default mode. The command `/PN` is equivalent to the command `/PLOT/NORMAL`.

The first example above plots histogram #1, while the second example plots the 2^{nd} row of scatterplot #1.

4.20 Display of Histograms and Scatterplots

Commands:	`/PRint, /TYpe`
Argument:	a list of the histograms and scatterplots and the `INDEX` keyword
Qualifiers:	`/Hard, /Echo`
Examples:	See below

This command outputs **FIOWA** histograms and scatterplots using the **FIOWA** output subroutine **PRTOUT**. The output from these commands will be identical to that which would have been obtained from a **FIOWA** run. Since a great many pages of output may result, these commands do not automatically write to the screen and the hardcopy file. `/PRINT` directs output to the `REPLAY.REC` file while `/TYPE` directs output to the terminal screen.

When the `/HARD` qualifier is used with `/TYPE`, it causes the output to appear in the hardcopy file as well as on the screen. It has no effect with the `/PRINT` command. *Note:* `/PRINT/ECHO` is the same as `/TYPE/HARD`.

When the `/ECHO` qualifier is used with `/PRINT`, it causes the output to appear on the screen as well as in the hardcopy file. It has no effect with the `/TYPE` command. *Note:* `/PRINT/ECHO` is the same as `/TYPE/HARD`.

Consider the following examples:

Command:	Result
`/TYPE H 1`	Display Histogram #1
`/TYPE H 1,4,6`	Display Histograms #1, 4 and 6
`/TYPE H 1,3-6,8`	Display Histograms #1, 8, and 3 to 6 inclusive
`/TYPE H 1,2,S 3,4`	Display Histograms #1 and 2 and Scatterplots #3 and 4
`/TYPE S,H1-10`	Display ALL defined scatterplots and Histograms #1–10 inclusive
`/TYPE H,INDEX,S1`	Display ALL defined histograms, Scatterplot #1 and the index
`/TYPE INDEX`	Display the index, i.e., list of histograms and scatterplots
`/TYPE`	Display the index, and all defined histograms and scatterplots

Note: No data subset or rebinning modifiers are permitted with `/PRINT` and `/TYPE`. These commands are handled internally quite differently from `/HIST` and `/PLOT`. This is also why the `%` sign is not necessary, and why you may specify more that one histogram and/or scatterplot in the argument.

4.21 Graphics Hardcopies

Commands:	`/PX, /HArdcopy`
Argument:	none
Qualifiers:	none
Examples:	`/HARDCOPY`

This command produces hardcopies of the current plot. The plot will be directed to the bitmap device specified with the `/OUTPUT` command or the pen plotter assigned with the `&DEVICE` command.

4.22 Releasing Print Files

Command:	`/RELease`
Argument:	logical unit number or filename
Qualifiers:	`/OUTPUT=queue`

Examples:	`/RELEASE 1`
	`/RELEASE REPLAY.REC`
	`/RELEASE/OUTPUT=TXC2:  1`

This command prints files that are attached to **REPLAY** on the specified printer. The file may be specified as either a logical unit number corresponding to the desired file attached to **REPLAY**, or it may be specified by the name of the file. The file is closed after printing.

Use the `/OUTPUT` qualifier to specify a printer queue for the output. If this is not done, the output is directed to the queue assigned with the `/OUTPUT` command, which defaults to `SYS$PRINT`.

The first example above prints the file attached to unit 1 on the queue previously assigned with the `/OUTPUT` command. The second example prints the `REPLAY.REC` file on the queue previously assigned with the `/OUTPUT` command. The third example prints the file attached to to unit 1 on the queue `TXC2:`.

4.23 Replotting data

Command:	`/REPlot`
Argument:	none
Qualifiers:	none
Examples:	`/REPLOT`

This command produces a new plot of all arrays currently in the plot buffer. Each time the `/PLOT` command is given, the specified arrays are added to the plot buffer. The plot produced by `/REPLOT` will be rescaled to include all data in the plot buffer. When the `/CLEAR` command is given, the plot buffer is emptied.

4.24 Restoring Saved FIOWA Data Files

Command:	`/REStore`
Argument:	filename
Qualifiers:	none
Example:	`/RESTORE SCR3:[MYID]FILE.SAV`

This command reads in a save–file of **FIOWA** common areas.

4.25 Scatterplots

Command:	`/SCatterplot`
Argument:	2 one–dimensional arrays and a two–dimensional array whose dimensions match the lengths of the x– and y– axes
Qualifiers:	`/Hard`
Example:	`/SCATTERPLOT %S1(X=0,100/Y=-.05,.05)`

This command produces a non–graphics density plot of a two dimensional array.

When the `/HARD` qualifier is entered, a hardcopy of the scatterplot will automatically be produced on the assigned bitmap device. See the `/OUTPUT` command.

The above example plots Scatterplot #1 in the range $0.0 \leq x \leq 100.0$ and $-.05 \leq y \leq .05$.

4.26 Displaying User–defined Command Symbols

Command:	`/SHow`
Argument:	none
Qualifiers:	none
Examples:	`/SHOW`

Commands may be user–defined by means of the assignment ":==" in the following way. It is possible to define a new command, which *must* begin with a slash, `/`, which is equivalent to another command and its associated arguments. For example, the following two assignments:

```
/R:==/RESTORE [MYID]FILE.SAV
/S:==&SET YLOG 10
```

defines the command `R` to be equivalent to `/RESTORE [MYID]FILE.SAV` and defines the command `/S` to be equivalent to `&SET YLOG 10`.

The commands that you have defined can be displayed by means of the command `/SHOW`.

4.27 Surface plots

Command:	`/SURFace`
Argument:	2 one–dimensional arrays and a two–dimensional array whose dimensions match the lengths of the x– and y–axis arrays
Qualifiers:	`/Hard`
Example:	`/SURFACE %S1`

This command produces a graphics surface plot of a two dimensional array, for example, see Figure 4. The data array will be represented by a surface, scaled to fit on the terminal screen.

When the `/HARD` qualifier is entered, a hardcopy of the surface plot will automatically be produced on the assigned bitmap device. See the `/OUTPUT` command.

Figure 4: Surface Plot Example:

4.28 Asynchronous Error Trapping

Commands: `/TRap, /NOTRap`
Argument: none
Qualifiers: none
Examples: `/TRAP`
`/NOTRAP`

The exception handler is automatically enabled when **REPLAY** begins. This traps all exceptions, for example, divide by zero or square root of a negative number. Thus, one need not fear crashing the program by attempting to evaluate $\sqrt{-1}$! After an exception, an explanatory message is printed and the program returns to the `REPLAY` > prompt.

The exception handler also usually traps `CTRL-C`. Thus, you may interrupt any command, e.g., to interrupt a long listing, and return to the `REPLAY` > prompt by typing `CTRL-O` and then `CTRL-C`. *There are some problems with this and it is not recommended.*

4.29 Unfitting

Command: `/UNFit`
Argument: none
Qualifiers: none
Example: `/UNFIT`

This command restores the parameters of the most recent fit to the values they had before the fit began. A list of the scalars and their old values is first printed, and you are queried as to whether you want to do the restoration.

4.30 Screen Width

Command: `/WIdth`
Argument: 80 or 132
Qualifiers: none
Examples: `/WIDTH 132`
`/WIDTH 80`

This command adjusts the alphanumeric, or transparent mode, terminal screen width, in columns.

4.31 Writing to a Disk File

Command: `/WRite`
Argument: a filename and a list of arrays
Qualifiers: `/Error, /NOError, /Append`
Example: `/WRITE/ERROR MYFILE %H1(I=1,10)`

This command writes a list of arrays to a disk file. If you do not in include an extension for the filename, the `.DAT` extension will be automatically added by **REPLAY**. The arrays to be written to the disk file must be all the same length!

Use the `/ERROR` qualifier to include the error array, in the case of sets.

If the `/APPEND` qualifier is entered, the program looks for an existing file with the same filename. The arrays are written onto the end of that file.

The above example writes the first ten bins of Histogram #1 and the error array to a file named `MYFILE.DAT`, which is opened as a new file.

4.32 Zero

Command:	`/ZEro`
Argument:	list of histograms and scatterplots
Qualifiers:	none
Example:	`/ZERO H 1,3-6,8,S 3,4`

This command zeros some or all of the **FIOWA** histograms and scatterplots. This command may *only* be used when **REPLAY** is running in a special mode that shares the **FIOWA** common areas with another program. In general it is not possible to overwrite the **FIOWA** common areas except with a `/RESTORE` command.

The above example zeros histograms # 1, 3, 4, 5, 6, and 8 and scatterplots # 3 and 4.

5 Commands Beginning with &

5.1 Setting ATARI plot mode

Command:	`&ATARI`
Argument:	`ON, OFF`
Examples:	`&ATARI ON` `&ATARI OFF`

This command controls whether a plot produced by the `/PLOT` command or the `/DENSITY` command is done with the normal **GPLOT** graphics, `OFF`, or with the fast Atari graphics, `ON`.

5.2 Autoscaling a Graph

Command:	`&AUTOscale`
Argument:	`ON, OFF, X, Y`
Examples:	`&AUTOSCALE ON` `&AUTOSCALE X`

This command controls the autoscaling feature for plotting future graphs. If autoscaling is `ON`, the both the x and y axes will be autoscaled. If autoscaling is `OFF`, then the current axis scales will be frozen, or they can be fixed with the `&SCALES` command. If you wish to autoscale the x axis, but fix the y axis, enter `&AUTOSCALE X`. And finally, if you wish to autoscale the y axis, but fix the x axis, enter `&AUTOSCALE Y`.

5.3 Automatic Centroid Calculation

Command: `&CENTROID`
Argument: `ON, OFF`
Examples: `&CENTROID ON`
`&CENTROID OFF`

This command controls whether the centroid is calculated automatically when an H–set is plotted with the `/PLOT` command.

5.4 Plotting Symbol

Command: `&CHar`
Argument: an integer n, $-16 \leq n \leq 16$
Examples: `&CHAR 0`
`&CHAR=-5`

This command defines the plotting symbol. If the plotting symbol is set to 0 (the default) then points are joined with straight lines and no symbol is plotted at the data points. If the plotting symbol is set > 0, then a symbol is plotted for each point and the data points are joined with straight lines. If the plotting symbol is set < 0, then a symbol is plotted for each point and the points are not joined with lines.

The currently available plotting symbols are:

Plotting symbol number	
0	no character
1	square
2	cross
3	square superimposed on cross
4	plus sign
5	diamond
6	diamond superimposed on plus sign
7	asterisk
8	filled square
9	filled diamond
10	triangle
11	filled triangle
12	circle
13	filled circle
14	five point star
15	filled star
16	point

5.5 Terminal Colour

Command: `&COLour`
Argument: colour name
Examples: `&COLOUR`
`&COLOUR=RED`
`&COLOUR RED`

This command sets the colour on a colour terminal. It has no affect on the pen that is used on a multiple pen plotter, for that see `&PEN`.

If just the command is entered, with no colour name, then the valid colours for the current terminal are displayed.

The valid colours for the currently supported colour terminals are:

CIT–467	Black	Red	Blue	Magenta
	Green	Yellow	Cyan	White
VS11	Black	Blue	Red	Magenta
	Dark_green	Light_blue	Orange	Pink
	Green	Sky_blue	Ochre	Ivory
	Yellow_green	Cyan	Yellow	White
TK4107	Black	White	Red	Green
	Blue	Cyan	Magenta	Yellow
	Orange	Green_yellow	Green_cyan	Blue_cyan
	Blue_magenta	Red_magenta	Dark_grey	Light_grey

5.6 Selecting Plotter and Terminal Type

Command: `&DEVice`
Argument: a plotter name and a terminal name
Example: `&DEVICE HP CIT-467`

This command selects the plotter type and the terminal type. The currently available plotters are:

`HP ZETA QMS`

The currently available terminal types are:

`CIT-467 VT100 VT640 VT241 TK4010 TK4107 VS11 PT100G`

The `HP` and `ZETA` devices are 8 pen plotters, while the `QMS` device is a vector plotter with no colour capability. The `VT100` terminal is a non-graphics terminal and the `VS11` is not a terminal, but a colour graphics output device. The rest are graphics terminals, and the `CIT-467` and the `TK4107` are colour graphics terminals.

5.7 Accessing the Graphics Editor, EDGR

Command: `&EDGR`
Argument: `OPEN, CLOSE, EDIT, FRAME`
Examples: See below

This command gives the user access to the graphics editor, **EDGR**. There is a manual available that describes how to use the graphics editor, the GRAPHICS EDITOR USER'S GUIDE. To create an **EDGR** editable drawing, use the command `&EDGR OPEN` *before* doing the graphics that you would like to edit later. To edit a drawing, use the command `&EDGR EDIT` from within **REPLAY**, or you can `$ RUN EDGR` as a stand alone program outside **REPLAY**.

See the **EDGR** manual for a description of a `FRAME`. You do not need to close an **EDGR** drawing, when you `OPEN` a new drawing the current one is closed automatically, or when you `/EXIT` or `/QUIT` from **REPLAY** the current drawing is closed.

5.8 Turning Off the Terminal Graphics Monitoring

Command: `&NOSCreen`
Argument: none
Example: `&NOSCREEN`

When the `&NOSCREEN` command is given, any future commands that involve graphics will produce the graphics in the internal buffers only. This allows one to produce hardcopy plots when using a non–graphics terminal. `&SCREEN` resets the screen mode.

5.9 Selecting the Plotter Pen Number

Command: `&PEN`
Argument: an integer n where $1 \leq n \leq 8$
Examples: `&PEN 3`
`&PEN=2`

This command selects the pen number to be used next on the pen plotter. The `HP` and the `ZETA` pen plotters have 8 pens.

5.10 Setting Automatic Clearing Mode

Command: `&PLOTCLEAR`
Argument: `ON, OFF`
Examples: `&PLOTCLEAR ON`
`&PLOTCLEAR OFF`

Use this command to determine whether or not successive `/PLOT` commands will automatically clear the graphics. If `ON`, then successive `/PLOT` commands will clear the graphics before plotting, so each graph has its own scale, assuming `&AUTOSCALE` is `ON`. If `PLOTCLEAR` is `OFF`, then successive `/PLOT` commands will overlay plots on the current graph.

5.11 Fixing the Graphical Scales

Command: `&SCALes`
Argument: `xmin, xmax, ymin, ymax`
Example: `&SCALES 3 10 1.E1 5.E1`

This command sets the axis scales on future graphs. It also turns off the autoscaling. This applies to the `/PLOT, /DENSITY` and the `/CONTOUR` commands. To turn on the autoscaling again, use the `&AUTOSCALE` command.

5.12 Turning On the Terminal Graphics Monitor

Command: `&SCreen`
Argument: none
Example: `&SCREEN`

If the `&NOSCREEN` command is given, any future commands to plot will be produced in the internal buffers only. This allows one to produce hardcopy plots when using a non–graphics terminal. `&SCREEN` resets the screen mode.

5.13 Setting or Querying of a Graph Characteristic

Command:	`&SET`
Argument:	a **GPLOT** plot characteristic name and a value
Examples:	See below

This command sets those graph characteristics that are associated with the **GPLOT** graph plotting package. The plot characteristics that can be changed with the `&SET` command are those listed in the LOW LEVEL GRAPHICS AND GRAPH PLOTTING AT TRIUMF manual under the heading "Descriptions of the Plot Characteristics".

If just the `&SET` command is entered, with no arguments, then the menu of all **GPLOT** graph characteristics will be displayed.

If a valid graph characteristic name is entered with the `&SET` command but no value is entered, then the current value of that name is displayed. The current value is unchanged if the user types the `RETURN` key without entering a new value.

For example, suppose you enter: `&SET` then **REPLAY** will display the **GPLOT** menu of all graph characteristic names and values.

If you enter:

```
&SET XMIN
Currently:  XMIN = 0.00000E+00, New value ?
```

Then **REPLAY** will display the current value of `XMIN` and ask for a new value. If you type `RETURN` without entering a value, `XMIN` will remain unchanged.

If you enter:

```
&SET XMIN -1        or
&SET XMIN=-1
```

Then **REPLAY** will simply set `XMIN` to the value -1.

5.14 Drawing Text Strings

Command:	`&TEXT`
Argument:	a character string
Examples:	`&TEXT This is a line to be plotted.`

This command draws character strings using the **GPLOT** routine **SETLAB**. See the LOW LEVEL GRAPHICS AND GRAPH PLOTTING AT TRIUMF manual for a complete description of the text drawing capabilities. When the `&TEXT` command is given, if `CURSOR` > 0, the default, the crosshairs will be used to locate the reference position for the text string, and the text justification will be determined interactively. If `CURSOR` ≤ 0 the values of `XLOC` and `YLOC` will be used to locate this reference position and the text justification will be determined by the value of `CURSOR`. The height and angle of the text will be determined by `TXTHIT` and `TXTANG`. The values of `CURSOR, XLOC, YLOC, TXTHIT`, and `TXTANG` can be changed with the `&SET` command. The text string may have imbedded formatting commands, which include many special character names and can control the font, super– and sub–scripting, etc. Again, please see the above mentioned manual for complete descriptions of these formatting commands.

5.15 Plot Windows

Command: `&WINDow`

Argument: total number of windows desired and the initial window

Examples: `&WINDOW 4`
`&WINDOW 4 2`

This command controls the windowing feature for plotting future graphs. Currently you can set up 6 different sets of windows: 1 window (the default), 2 windows, 4 windows, 8 windows, 16 windows, or 24 windows, see Figure 5. For example, `&WINDOW 4` sets up 4 windows. The next `/PLOT`, `/DENSITY`, or `/CONTOUR` command will automatically go into the first window. The next `/PLOT`, `/DENSITY`, or `/CONTOUR` command will automatically go into the second window, etc., up to the fourth window. If you want a graph to go into a specific window in a set, use the optional initial window parameter. For example, if you want the next `/PLOT` to go into the second window out of four, use the command `&WINDOW 4 2` before entering the `/PLOT` command. If a graph exists in a window already, a new graph will be overlayed on the old with the same scales. The default is to have one window.

6 Arithmetic Expressions

REPLAY may be used to evaluate arithmetic expressions. The expressions are entered in FORTRAN–like code and may or may not contain an equality. Any line entered to **REPLAY** which does not begin with a / or and & will be treated as an expression. For example,

```
REPLAY > EXP(1-2*SQRT(3))/4          <== The expression is evaluated and
         .212713E-01                 <== the result is printed on the screen.
```

If the expression contains an equality,

```
REPLAY > A=EXP(1-2*SQRT(3))/4        <== The expression is evaluated and the result
                                         is assigned to a variable called A.
                                         Nothing is printed on the screen.
```

The above example illustrates how the user may create variables. In this case, if `A` does not already exist, it will be created, added to the user–defined variable name table, assigned the value 0.02128 and classified as a scalar, a single value. If `A` already exists in the name table, and it is classified as a scalar, its value will be reassigned to .02128. If it exists but is not a scalar, an error message will be printed and no reassignment made.

The right hand side of an expression may contain numerical constants, user–defined variables or references to **FIOWA** histograms or scatterplots or parts of scatterplots.

The left hand side of an expression, if it exists, must be a user–defined variable. The user is not permitted to redefine the **FIOWA** data, nor numerical constants!.

Figure 5: The six window configurations

REPLAY recognizes five different types of data: scalar, vector, matrix, H–set and S–set. Expressions may contain any of these data types, although there are restrictions on the way data types may be combined. Most of these are common sense restrictions. For example, you cannot add together vectors of differing lengths, nor can you add a matrix to a vector.

All data in the expression is evaluated as REAL*4. If you enter numerical constants as integers, they will be converted to real.

6.1 Functions

REPLAY allows functions names to appear in expressions, with the syntax being as in FORTRAN. All the standard FORTRAN functions are recognized, plus others. The intrinsic functions are:

`LOG(X)` `LOGE(X)`	$\ln X$
`LOG10(X)`	$\log X$
`EXP(X)`	e^X
`SIN(X), COS(X), TAN(X)`	$\sin(X)$, $\cos(X)$, $\tan(X)$

`ASIN(X), ARSIN(X)`
`ACOS(X), ARCOS(X)`
`ATAN(X), ARTAN(X)`
	$\arcsin(X)$
	$\arccos(X)$
	$\arctan(X)$

`SINH(X), COSH(X), TANH(X)`	$\sinh(X)$, $\cosh(X)$, $\tanh(X)$
`ERF(X)`	$\frac{2}{\sqrt{\pi}}\int_0^x e^{-\xi^2}d\xi$
`INT(X)` `IFIX(X)`	Truncate decimal part
`NINT(X)`	Nearest integer
`ABS(X)`	$\lvert X\rvert$
`MAX(X,Y)`	Maximum value of X and Y
`MIN(X,Y)`	Minimum value of X and Y
`MOD(X,Y)`	$X - Y(\frac{X}{Y})$
`SIGN(X,Y)`	$\lvert X\rvert$ if $Y > 0$ or $-\lvert X\rvert$ if $Y < 0$
`RAD(X)`	Convert X from degrees to radians
`DEG(X)`	Convert X from radians to degrees
`NORMAL(X,A,B)`	$\frac{1}{B\sqrt{2\pi}}e^{-\left[\frac{1}{2}\left(\frac{X-A}{B}\right)^2\right]}$ (Normalized Gaussian distribution)
`TINA(X,B,C,D)`	$e^{\left(\frac{X-B}{D}\right)}\left[1-\frac{2}{\sqrt{\pi}}\int_0^{\frac{X-B}{C}} e^{-\xi^2}d\xi\right] / \left[2De^{\frac{C^2}{4D^2}}\right]$ (Normalized Tina resolution function)

6.2 Operators

1. Scalar Operators: $+ \quad - \quad * \quad / \quad **$

   The standard scalar FORTRAN operators $+$ $-$ $*$ $/$ $**$ are permitted. The unary operators $+$ and $-$ may be applied to any type of data of any size. The non–scalar operands of the binary operators $+ \quad - \quad * \quad /$ $**$ must match in length and type. For non–scalar operands, the operation is carried out on each element of the array. For example, `2 * I` would multiply each element of the vector `I` by 2. `I * I` would multiply each element of `I` by itself (this being equivalent to $\mathtt{I}^2$). It is thus readily apparent why the operands must match in length and type. To evaluate `A+B` where `A` and `B` are different lengths would not be possible because there would not be an element of `A` corresponding to every element of `B`.

2. Non–Scalar Operators: $\sim$ `&`

   $\sim$ is a unary operator that evaluates the transpose of a matrix. The operand must be a matrix.

   `&` is a concatenation operator. This permits you to extend arrays in the x–direction. For example, if `A` and `B` are two one dimensional arrays of lengths a and b, then `A&B` will be a one dimensional array of length $a + b$. Scalars may be concatenated to form a vector, e.g., `1&2&3&4&5` produces a vector of length five whose elements are { 1,2,3,4,5 }. Matrices may be concatenated provided their y–dimensions match. A vector may be concatenated to a matrix where its length matches the y–dimension of the matrix.

6.3 Sets

You may evaluate expressions containing sets. Set arithmetic is restricted to linear combinations of sets. Function names and non–scalar operators may not occur in the expression. The sets in an expression must match in length and type.

The operations are performed on the dependent arrays and the error array is automatically adjusted. The independent array of the result will be taken to be the independent array of the *first* set in the expression. For example, if `A` and `B` are H–sets and you wish to evaluate:

REPLAY $>$ `C=`a`*A+`b`*B`

then `C` will be an H–set whose elements are defined as follows:

$$\begin{aligned}
\texttt{C.U} &= \texttt{A.U} \\
\texttt{C.V} &= a\texttt{A.V} + b\texttt{B.V} \\
\texttt{C.E} &= \sqrt{(a\texttt{A.V})^2 + (b\texttt{B.V})^2}
\end{aligned}$$

If all the sets in an expression were of the binned type, any **FIOWA** data is assumed to be binned, then the result is also classified as binned.