Essential RS/1
Release 3

Beverly Chew
Pam Weizenbaum
Mark W. Moormans
MIT Information Service
for Project Athena
Revision E
February 14, 1988

Revision E, July 1988: M. W. Moormans, E. N.Prabhakar, Brandy Bechtel. Updated to Release 3.0, standard for workstation release 6.0C.

Revision D, January 1988: E. N. Prabhakar. Updated to X Version 11 and PostScript.

Revision C, September 1987: K. L. Lieben and Joshua Q. Lubarr. Revised and edited document for the workstation environment and incorporated changes from "RS/1 on VAXstation 100 Terminals" by L.B. Merims.

Revision B, April 1985: Pam Weizenbaum. Added VT100 support documentation.

1. Introduction

RS/1 is a powerful electronic laboratory tool for data storage, management, and analysis, developed and donated by BBN Research Systems. With RS/1 you will find it easy to enter and store data as tables, and graph that data into pie charts, x-y plots, bargraphs, and three-dimensional plots. Interactive graph and table editors allow easy modification of your graphs. RS/1 offers an impressive collection of functions, to be used in commands or in your own RPL procedures. A variety of statistical tests are also available.

RS/1 offers its own PL/I-like programming language, RPL, which permits advanced users to program their own extensions and tailor the existing system. Advanced users can also take advantage of the RS/1 modeling capability.

The basic structure within RS/1 is the two-dimensional table, which is just a set of columns and rows. All data is stored in this structure. The intersection of a column and a row is called a cell. Each cell of a table contains some form of data, such as text or numeric values, equations, or variables. Tables are ‘‘data objects’’: collections of data that can be manipulated. RS/1 is unique in that data objects may range from single variables to tables to three-dimensional graphs. All data objects can be manipulated in the same ways.

The new version of RS/1, Release 3, is incompatible with the old version fo RS/1, so data created by it cannot be used with the old version (version 12). There is a program to convert data between them, however (see Section 2.2). The new version has a number of features such as:

- Improved GRAPH editor
- Permanent GRAPH display modifiers
- Data Smoothing
- Enhanced THREED graphics
- More robust directory structure
- Reorganized HELP facilities
- Expanded manual set

1.1 Documentation

This short manual is designed for any Athena user doing coursework that involves graphing and analysing data. It introduces you to the fundamentals of RS/1 Release 3 by analyzing a small dataset with the basic table-creation and manipulation commands. Simple examples of graphing, curve fitting, and statistical analysis are included. This manual assumes that you are familiar with the concepts presented in ***Athena Workstation***.

BBN provides full documentation of RS/1 in a three-volume ***RS/1 User's Guide***, referred to often in this manual. When you are comfortable with the basic commands presented here, refer to the ***User's Guide*** (especially Books 1 and 2) for information on more advanced commands and options. Reference copies are available in all the Athena terminal cluster manual racks; it is also sold at the Information Services Publications Office in Room 11-209. An excellent pocket reference card can also be purchased at very low cost.

1.2 Manual Conventions

Throughout this manual, the following conventions have been observed:

Within text, RS/1 commands are presented in upper case (DIRECTORY), while Unix commands are italicized in lower case (*setenv*). Occasionally, an example of what to type is given within text; these commands are simply quoted (''erase''). In actual use, however, RS/1 does not care whether commands are in upper or lower case (as opposed to UNIX, which is case sensitive).

Sample command lines and user-computer interactions are indented. User-typed portions are printed in **boldface**, and *<placeholders>* names and values that you supply, are italicized within angle brackets. RS/1 responses appear in an courier font. From time to time, annotations are given to the right of examples; these are in italics.

2. To Begin

2.1 Accessing RS\1

The two versions of RS\1 are now kept in lockers called **rs1.old** and **rs1.new**, respectively. To use RS\1 Version 3 you should:

% attach *rs1.new*

2.2 Using RS\11 for the First Time

Before you use RS/1 on Project Athena for the first time, you need to login and make a separate directory to contain your RS/1 files, and tell Unix to use this directory whenever you invoke RS/1 Release 3. To do this, issue this command:

host% mkdir ~/rsuserhome

The prototype .login file contains the command *setenv RS1HOME ~/rs1home*. To use the new release of RS/1, you will need to replace that line with *setenv RSUSERHOME ~/rsuserhome* in your .login. To do this, enter the emacs editor with your .login file, delete the old command and add the new line:

```
setenv RSUSERHOME ~/rsuserhome         # RS/1 Home directory (NEW)
```

In your *.cshrc* file you should edit your path variable setting. BEFORE the /usr/athena entry in your path add:

```
/mit/rs1.new
```

The best way to use RS/1 on a workstation is to place it in a separate window. Rather than type in a long command to do it each time, you can create a new command that will do it for you. This is called an alias, and you place it at the end of your *~/.cshrc* file. The following command will create a window in the upper left corner that runs RS/1. It will also create a invisible Tektronix window of the same size which you can use later for drawing graphs. Type:

1. **% emacs ~/.cshrc &** *start up Emacs*
2. **ESC >** *move to the end of the file*
3. **alias xrs1 'xterm -fn 9x15 =84x38-0+0 %-0+0 -n RS/1 -e rs1 &'** *set it*

Then exit emacs, and, if you wish to use RS/1 before you logout, type:

% source ~/.login
% source ~/.cshrc

2.3 Updating from the Old Version of RS/1

If you have used RS/1 before you will need to consider whether or not you wish to keep your old files. If you don't want to save any data from before everything is simple. If you *do* need your data, you must convert it to a format compatible with version 3 or RS/1.

First you should create a new directory for RS/1 files:

% mkdir ~/rsuserhome

and add the following line to your .login file:

```
setenv RSUSERHOME ~/rsuserhome   # RS/1 Home directory (NEW)
```

Then modify your path in *.cshrc* to use the new RS/1 by placing the following line BEFORE /usr/athena in your path:

```
/mit/rs1.new
```

2.3.1 Deleting Old Data

If you do not wish to save any of your data delete it by deleting your RS1HOME.

```
rm -r ~/rs1home
```

You can delete individual data files from within RS/1 with the delete command.

2.3.2 Converting Data

If you do have data which you wish to convert, you need to run a conversion program. The full procedure is outlined in th ***RS/1 Release 3 Features*** document. Fortunately, there is a shell script which automates the conversion process. *Be sure 'rs1' gives you the correct version of RS/1.* The conversion script, like RS/1, can only run on the VAX. It takes from 10 to 20 minutes, depending on how large your RS/1 directory is. It will create a 'compressed' backup copy of your old directory, so make sure you aren't near your quota.

ARE THERE WAYS AROUND A NEAR TO QUOTA PROBLEM...

To use it type:

`attach enprabha` *WHAT SHOULD THIS BE....*

`/mit/enprabha/Bin/rs1update`

2.4 Using RS/1

You are now ready to start up RS/1 Release 3.

The command to start up RS/1 is simply **rs1**. However, as we said before, you will probably want create it in a separate window. [1] Type:

`host%` **xrs1** *or*
`host%` **xterm -fn 9x15 =84x38-0+0 %-0+0 -n RS/1 -e rs1 &**

RS/1 responds with a message like this:

```
RS/1 Release 3.0

    Used under license from BBN Software Products Corporation.
    Copyright (c) 1987 by BBN Software Products Corporation.
    All rights reserved.

    With RS/Explore and RS/Discover Version 1.2


[setting TERMINAL to xterm ... Done]
#
```

Now that you have started RS/1, you may begin giving commands. RS/1 always lets you know it is ready for a command by showing you its command-level prompt: **#**.

2.5 Setting Variables

There are a number of variables you will want to set during your first RS/1 session. RS/1 will save their values, so you only need to reset them if your directory gets corrupted. To do this type:

 # call $set_athena_env

This sets ....

Since there are currently no default printers for Athena clusters, you need to specify a

[1] The main exception is when RS/1 will not let you enter. Then you need to type **rs1** in your window so you can see the error message.

PostScript printer for RS/1 to queue output to. You do this using:

 # call $set_printer_queue

Hitting return will give the previous printer used, if any. This is not necessarily the most appropriate printer to use!

You may also want to tell RS/1 to use an Emacs in the upper corner of the screen.

 # set editor_name to '/usr/athena/emacs -q -w =80x24-1+1'

This is for editing procedures and variables.

2.6 Getting Out

To leave RS/1 and return to Unix command level, type ''logout'' after an RS/1 prompt. This will kill off the RS/1 window.

3. Going Further

3.1 Help

To receive on-line help from RS/1, type HELP and a carriage return (<RET>) for information on commands and topics. When HELP displays its menu of help topics, type a topic number and <RET>. You will see a message about the topic plus a list of subtopics.

In RS/1 dialogs, type ''?<RET>'' for information on responding to prompts.

In addition, student consultants are waiting to help you in the Athena clusters. If they cannot answer a question, they will refer it to an RS/1 expert.

3.2 Typing RS/1 Commands

With a few exceptions, case doesn't matter to RS/1: you can type commands and responses to prompts in upper case, lower case, or both. Text strings do reflect case, however. One or more spaces must separate words, names, and values in a command line.

Variable names must begin with a letter, which may be followed by any number of letters, digits, and underscores (_). Text values (column names, table titles, and so on) must be enclosed in single (') or double (") quotes.

You may use a semicolon (;) to separate multiple commands typed on one line. To continue one command onto a second line, end the first line with a hyphen (-). The next line will then begin with a prompt for you to continue:

```
# display col 12 of table lab4 -
Continuation# col 12 < 17.25
```

In an RS/1 dialog, default values are shown in square brackets (for example: [yes]). Press <RET> to choose the default value. If a choice of answers is offered, you need type only enough letters for a unique match.

3.3 Correcting Mistakes

If you make a mistake in typing, you can delete the last character typed with the DELETE key. Type CTRL-W to delete the last word, and CTRL-U to delete the entire line you have just typed.

3.4 Interrupting

You can stop terminal output (for example, when data is scrolling past your screen) by typing CTRL-S. To resume terminal output, type CTRL-Q. If you ever need to stop RS/1 in the middle of a command execution, just type CTRL-C.

3.5 When RS/1 breaks...

Because of RS/1's complexity, it is prone to a variety of errors. When RS/1 doesn't start up, the most common problem is a 'locked' home directory. This is usually caused by logging out of the workstation without exiting RS/1. The solution is to type:

`host%` **rs1fix**

If ''rs1fix'' gives you an error message, such as

`rm: ./0.ULK nonexistent` *or*
`rm: /rs1home/0.ULK nonexistent`

it is usually because the RSUSERHOME environment variable isn't set properly. If your directory name is ''~/rs1home'' type:

`host%` **setenv RSUSERHOME ~/rs1home**

If you don't have an RS/1 home directory, or if this line ins't in your ''.login'' file, refer to Section 2.1 on how to set things up.

If you are still having problems, there may be other 'lock files' floating around. You can delete them by typing:

`host%` **rm -i $RSUSERHOME/*LK**

This will ask you for each file you wish to delete. Just type 'y' for yes. All files ending in 'LK' are lock files, which are designed to keep other people from writing into your RS/1 directory when you are using it. You can delete those whenever you want, since they aren't really necessary in the Athena environment. Sometimes they backfire and prevent YOU from using it. Just be careful not to delete anything else.

If RS/1 still isn't working, it means that something more serious is the problem. Try using OLC or calling the Consulting Hotline (x3-4435). Be sure to give a full description of the error, as well as what you tried to fix it. You may need to run RS/1 in your current window by typing **rs1** to get a full description of the error.

4. A First Session

This tutorial is based on your hypothetical study of the effects of a new reading program on third-graders' reading speed and comprehension. In this first session, you will input and edit the data. For each child, the following information has been recorded: the student's name; birthdate; sex; reading group (1 = new program, 2 = standard program); verbal IQ; ''error'' (number of errors on comprehension questions); and ''speed'' (number of minutes to finish reading a given story).

Here is an example set of data (missing data leaves an empty space):

student:	bday:	sex:	group:	viq:	error:	speed:
wilma	01-jul-77	f	1	98	0	18.30
betty	02-jan-77	f	1	104	4	15.80
andrea	26-nov-76	f	1	102	2	16.4
jay	05-mar-77	m	1		9	14.3
bill	17-sep-77	m	1	95	5	14.6
andrew	10-jul-77	m	1	100	10	135
sue	02-oct-77	f	2	103	6	15.2
sue	02-oct-77	f	2	100	1	15.2
kathy	29-apr-77	f	2	100	1	25.5
theo	06-feb-78	m	2	101	5	16.8
tom	13-apr-77	m	2	95	10	15.1

Now you want to construct an RS/1 table to store your data.

4.1 Making A Table

Within RS/1, all data is stored as tables. A table is a set of rows and columns containing data in each cell. In addition to the data, an RS/1 table includes a table name, an optional table title (displayed at the top), optional notes (displayed at the bottom), a count of the number of rows and columns, row numbers and optional row names, and column numbers and optional column names.

To enter the student data into a table called ''reading'', you can either use the table editor or enter into a ''dialog'' with RS/1, in which you respond to a series of prompts. First, there is the table editor:

 # make table reading

```
Make the table through a DIALOGUE (D) or with EDITOR (E)? [D] e
```

Typing ''E'' means that RS/1 will repaint the screen as a blank, two-dimensional table. You can fill in the table by moving the cursor to the cell you want to fill with the arrow keys and typing. Type /HELP to the editor, or see pp. 74-83 of the *User's Guide*, Book 1. As you fill in the cells of the table, only eight characters will appear; if you display the table after you have written it, the full names will be there.

Below is a sample of an interactive session with MAKE TABLE using ‘‘dialog’’ option. Start the table-making dialog by giving the command MAKE TABLE followed by the name of the table you want to create. The data is from the study discussed earlier.

```
# make table reading

Make the table through a DIALOG (D) or with EDIT TABLE (E)? [D] d
Title of table: READING DATA
Are there row names? [Yes] N
Are there column headings? [Yes] <--Press <RET> for default response.
Enter columnwise? [Yes]
WATCH the table as you create it? [Yes]
 Heading of Col 1=student
  Row 1 Col 1=wilma
  Row 2 Col 1=betty
  Row 3 Col 1=andrea
  Row 4 Col 1=jay
  Row 5 Col 1=bill
  Row 6 Col 1=andrew
  Row 7 Col 1=sue
  Row 8 Col 1=sue
  Row 9 Col 1=kathy
  Row 10 Col 1=theo
  Row 11 Col 1=tom
  Row 12 Col 1=next          <--Type ‘‘next’’ to indicate end of data
   Heading of Col 2=bday     for first column - go to next column.
  Row 1 Col 2=1-jul-77
  Row 2 Col 2=2-jan-77
      .             .        <--Continue adding data.
      .             .
  Row 11 Col 4=2             <--After the first column, RS/1
   Heading of Col 5=viq      advances to each column automatically
  Row 1 Col 5=98             after the correct number of rows.
  Row 2 Col 5=104
  Row 3 Col 5=102
  Row 4 Col 5=<RET>          <--Press <RET> to enter the
      .             .        EMPTY value for missing data.
      .             .
   Heading of Col 7=speed
  Row 1 Col 7=18.3
      .             .
      .             .
  Row 11 Col 12=15.1
   Heading of Col 8=exit     <--Type ‘‘exit’’ to leave the dialog.
```

If you enter data incorrectly, you may either finish entering the data and then use the

Table Editor (see Section 3.4 below) to make corrections; or type the interrupt character **(!)** and a SET command, to place the correct value in the cell. For example:

 # !set col 6 row 5 = 10

Book 1, Section 3 of the ***User's Guide*** contains detailed information on the MAKE TABLE command.

4.2 Displaying A Table

To display the table you have created, simply type:

 # display table reading

```
0      1 student  2 bday       3 sex  4 group  5 viq    6 error   7 speed
-------------------------------------------------------------------------
1      wilma      01-jul-77    f      1.00      98.00   0.00      18.30
2      betty      02-jan-77    f      1.00     104.00   4.00      15.80
3      andrea     26-nov-76    f      1.00     102.00   2.00      16.4
4      jay        05-mar-77    m      1.00              9.00      14.3
5      bill       17-sep-77    m      1.00      95.00   5.00      14.6
6      andrew     10-jul-77    m      1.00     100.00  10.00     135
7      sue        02-oct-77    f      2.00     103.00   6.00      15.2
8      sue        02-oct-77    f      2.00     100.00   1.00      15.2
9      kathy      29-apr-77    f      2.00     100.00   1.00      25.5
10     theo       06-feb-78    m      2.00     101.00   5.00      16.8
11     tom        13-apr-77    m      2.00      95.00  10.00      15.1
```

For hardcopy output, see Chapter 6 of this manual.

4.3 Editing A Table

The easiest way to make changes to a table is to use the Table Editor. Table Editor commands allow you to change values of individual cells; insert or delete cells, rows, or columns; copy data between cells; change the table layout; add or change the table title and table notes; and more: see the ***User's Guide***, Book 1, Section 3.6.

To enter the Table Editor with the table you have just created, type:

 # edit table reading
`Copy READING so you can RESTORE it if necessary? [Yes] <RET>`

Always answer yes to the response; you can then abandon these changes and RESTORE the original table, or SAVE the changes and replace the original data.

Within the Editor, you enter into another dialog, in which you are prompted for the part of the table you want to change, and for the new value. You will see these changes as they are made. To get a list of the editing commands, type ''/<RET>''. The slash (/) always precedes editing commands.

RS/1 now replaces the screen with two separate regions: the upper screen, which displays your table, and the lower screen, which provides a scrolling command region for Table Editor prompts. All the commands and data you type appear in the lower screen. When you give a command that changes the table, or when you access a part of the table not visible on the screen, the display portion of the screen is updated.

In order to tell RS/1 which cells you want to edit, you need to know how to move around in the display portion of the Editor. There are two ways:

arrow keys	move the cursor cell by cell, within the display screen. By moving the cursor, any offscreen parts of your table become visible. This method is often fastest. Arrows are shown in examples as ‘‘<ARROW-NAME>’’. /GOTO prompts for a specific row and column.

Look at the data displayed on the screen (or see Section 3.3 above). You'll notice two entries which do not seem to make sense. The data for Sue is duplicated (rows 7, 8), and the speed variable for Andrew should be 13.5 rather that 135 (row 6).

We now correct these errors within an EDIT TABLE dialog. In this example, we demonstrate how to change individual cell values, how to delete and insert lines, and how to add table notes. Because we cannot show the dynamic changes in the display and command regions, we show the RS/1 prompts and your typed responses on the left, with annotations on the right. First, delete row 8:

```
New heading for Column 0: /goto          <--Use /GOTO command to move to a
Enter row number: [0] 8                  new cell. You are prompted for row
Enter column number: [0]<RET>            and column of new cell.
New name for Row 8: /delete              <--Use /DELETE to delete the row.
Delete a SINGLE CELL (S), a COLUMN (C), a ROW (R),
    or skip to NEXT prompt (NEXT)? r
OK to delete ROW 8?: yes
```

Second, change value of Row 6, Col 7 from 135 to 13.5:

```
New name for Row 8: /goto          <--Use /goto to move to incorrect
Enter row number: [8] 6            value in Row 6, Col 7 cell.
Enter column number: [0] 7
Row 6 Column 7: [135] 13.5         <--Type in correct value over old value.
```

Third, add extra observation: rose 01-aug-77 f 2 101 * 15.0

```
Row 6 Column 7: [13.5] <down-arrow>
Row 7 Column 7: [15.2] <left-arrow>
Row 7 Column 6: [6.00]
    .          .
    .          .
Row 8 Column 1: [6] 9
Enter Column number: [7] 0
New name for Row 9: /insert             <--Insert a blank row above current row.
Insert a SINGLE CELL (S), a COLUMN (C), a ROW (R),
 or skip to NEXT prompt (NEXT)? r
New name for Row 9: <right-arrow>       <--Use arrow keys (but not <RET>)
Row 9 Column 1: rose <right-arrow>      to enter values.
Row 9 Column 2: 01-aug-77 <right-arrow>
Row 9 Column 3: f <right-arrow>
Row 9 Column 4: 2 <right-arrow>
Row 9 Column 5: 101 <right-arrow>
Row 9 Column 6: EMPTY <right-arrow>     <--Type ''EMPTY'' for missing value.
Row 9 Column 7: 15.0
Row 10 Column 7 [16.8]: /notes          <--Use /NOTES command to add note
                                        information to the table.
Notes: Reading Data Codes: Group        <--Use hyphen to continue long line.
Continuation# (1=New, 2=Old)
Row 12 Column 7 [15.0]: /exit           <--Leave the editor.
```

Now view your corrected table with the DISPLAY command.

5. More On Tables

5.1 Table-Portions: Subsetting And Manipulating Data

Many times you will want to display or operate on only a part of the data in a table. A table-portion is a subset of a table consisting of cells that lie in the intersection of specified columns and rows. You display a table-portion by specifying the rows and columns that define it. You can use column/row numbers or names, and you may specify lists or ranges of row and column numbers and names. For example:

 # display cols 1,2,5 of reading

shows the name, birthday, and viq of all the students. To display the boys from the new reading program, type (note order of rows):

 # display rows 5,4,6 *<--RS/1 will prompt for table name.*

Typing the command:

 # display col 'viq' of rows 1 to 6 of reading

shows the first six IQ scores in the table; since these are all scores from students in the new reading program, we can use this command to look at the scores of that group. Unfortunately, most databases are not as tidy as ours, which is why the WHERE command can be handy.

WHERE makes a conditional test of the values in the table. For example, you can display the same student records shown in the example above with this command:

 # display reading where col 'group' = 1

To find the viq of only those children born before January 1, 1977, type:

 # display col 'student' to 5 of reading where col 'bday' < '1-jan-77'

You can use complex conditional expressions in WHERE expressions, including those using arithmetic and logical operators.

5.2 New Tables from Old

There are several ways to create new tables from parts of existing ones. The easiest way is to issue the MAKE TABLE command, specifying those parts of the existing table you want included in the new one. For example, to make two new tables containing the girls' and the boys' data from the table ''reading'', type:

 # make table girls from reading where col 'sex' = 'f'
 # make table boys from reading where col 'sex' = 'm'

You can add new columns to tables by using the SET command in conjunction with table-portions. For example, to create and store a new variable, the logarithm of the speed variable, in the ''reading'' table, type:

 # set col 8 of reading to log(col 'speed')

When SET is used this way, you must refer to the added columns by their numbers. To add column names, enter the Table Editor, or give the SET command:

 # set heading of col 8 of table reading to 'logspeed'

In combination, the use of arithmetic and logical expressions with the SET command and WHERE gives you considerable power and flexibility for data transformation and manipulation. Additional examples of these commands are given in Book 1 of the ***User's Guide***, pages 56 through 66. Several other useful table commands are SORT, MERGE, and TRANSPOSE (see the ***User's Guide***, Book 1, pages 84 to 88).

5.3 Functions

In the example above, we showed you a log function. A function is a procedure that performs a calculation on an argument and returns a single value; for example, the function SQRT(10) calculates the square root of the number ten. RS/1 offers an impressive collection of functions, to be used in commands or in your own RPL procedures. These functions include arithmetic, Boolean, conversion, file I/O, graphics, text, trigonometric, and type testing operators. Explanations of all functions are given in Book 3 of the ***User's Guide***, Section 13. Some examples:

# **type abs(-10.7)**	*displays the absolute value of a number.*
# **set col 3 = ASIN (SQRT(col 2))**	*sets contents of column 3 equal to arcsine of square root of data in column 2.*
# **if EMPTY(x) THEN x = 0**	*tests for EMPTY value.*

5.4 Table Maintenance

When you have been using RS/1 for some time, you will have created a number of tables and graphs. RS/1 stores these data objects in a private directory named ''$RS1HOME''. RS/1 directories are distinct from Unix directories, and their contents should be manipulated only from within RS/1. It is possible to create subdirectories to help organize your RS/1 data objects. You can also create group directories so that data may be shared among several people at one time. Section 4 of Book 1 of the ***User's Guide*** describes the structure of RS/1 directories in detail.

To display the contents of your RS/1 directory from within RS/1, give the DIRECTORY command (which may be abbreviated as DIR):

dir

If you try the DIRECTORY command before creating any RS/1 tables, you will be told that no files exist.

To list the names of all your data files, issue the DIRECTORY command. Other useful housekeeping commands are RENAME and DELETE:

rename table girls to 'female' *<--Include required quotes.*

delete table temp1

When a table is no longer of current use yet must be saved long-term, move it out of your current RS/1 directory with the ARCHIVE command (***User's Guide***, Book 1).

5.5 Creating Tables From Unix Files

For data already in a Unix file, you may enter it directly into an RS/1 table with the $EZ_READFILE command, *if the file has delimiters* (spaces, commas, etc.) between data values. When your data files do not have such delimiters, use the $READ_FILE command (***User's Guide***, Book 1).

$EZ_READFILE is a public procedure that must be called in a special way:

call public $ez_readfile

Once called, it prompts you for the name of the Unix file containing your data, the name of the RS/1 table you want to create, whether to enter the data columnwise or rowwise, and what delimiter you are using. (See Book 3 of the ***User's Guide***, page 158 for a description of public commands.)

Unix files may be created from RS/1 tables by following the procedure for using the PRINTOUT command discussed in Section 7 of this manual, or by using the public procedure $WRITE_FILE.

6. Graphs

One of the most powerful aspects of RS/1 is its graphics capability. RS/1 offers commands that create, edit, and store four types of graphs: BARGRAPH, GRAPH (x-y graphs: scatter plots, trend plots or line graphs), PIECHART, and THREED (three-dimensional graphs). You can also display and plot with a variety of graphics symbols, connect data points, add or delete curves, change labels and titles, and add error bars. Axes are automatically scaled with appropriate units. In this section, we have space to show only one simple example. Refer to Book 2 of the ***User's Guide*** for many more ways to create graphs with RS/1.

The four types of graphs are data objects, just as tables are. They can be manipulated with all the basic RS/1 commands. RS/1 also provides a Graph Editor for editing graphical data objects. Like the Table Editor, it will prompt you for the information it needs. Section 5.2 below provides a brief overview of this feature.

6.1 Graphing

Here is one way of creating an x-y plot, using the interactive form of the MAKE GRAPH command. (The non-interactive form includes default scaling, symbols, and titles.) The example displays and analyzes an x-y plot of the relationship between number of errors and the logarithm of reading speed in our sample dataset. (Logspeed rather than speed is used because its distribution is more normal.) To graph logspeed versus error from the ''reading'' table:

```
# make graph logspeed_error
Title of graph? Logspeed vs Error
Label for x-axis? Log speed
Scale of x-axis: LINEAR(LI) or LOGARITHMIC (LO)? [LI]
Label for y-axis? number of errors
Scale of y-axis: LINEAR(LI) or LOGARITHMIC (LO)? [LI]
Do you want two y-axes? [No] <RET>
Source of x-values for curve 1: ENTER (E), TABLE (T),
   or FUNCTION (F)? t
Enter tableportion: col 8 of reading
Source of y-values for curve 1: ENTER (E), TABLE (T),
   or FUNCTION (F)? t
Enter tableportion: col 6 of reading
Symbol (enter ? to see choices) for curve 1: *
Label for curve: error curve
Connect points on curve? [No] no
Add curve 2? [No] n
DISPLAY graph (D), EDIT graph (E), or return
    to COMMAND level (C)? [COMMAND]
Graph LOGSPEEDS_ERROR created.
#
```

The resulting graph, ‘‘logspeed_error’’, is stored as an RS/1 file. To see it, type:

```
# display graph logspeed_error

                             Logspeed versus Error
n
u 10*          *
m   |     *
b   |
e   |
r   |           *                                                      error curve
    |       *             *                                                      *
o  5|
f   |               *
    |
e   |                   *
r   |                                                                 *
r  0|______________________________*_________________________________________
o  2.6       2.7       2.8       2.9       3.0       3.1       3.2       3.3
r
s                                  Log speed
#
```

6.1.1 Adding A Curve

To add a curve to a graph, you re-enter the MAKE GRAPH dialogue at the point where RS/1 asks for the X values for the curve by typing:

 # add curves to *<graph-name>*

You are given the option of using the same X values used for the previous curves, or you can specify new ones; RS/1 automatically adjusts the graph axes.

6.2 Graphing Gets Better

When you first entered RS/1, you set your terminal to be a vt100. The vt100-style terminal is fine for making and editing tables, but a Tektronix 4010 terminal is better for graphics. When you want to display a graph, you need to switch to Tektronix mode. While holding down the **CTRL** key, click the **MIDDLE** mouse button while in your RS/1 window. This will pop up the ''Modes'' menu. Move the mouse cursor to the item ''Select Tek mode'', then click any button. This will *pop up* the Tektronix window if it exists, or create it if it doesn't. You may need to specify its location by clicking the mouse on the upper left corner of where you wish it to be. Then, you must tell RS/1 what you have done.

 # terminal ='t4010'

Do the same thing with ''Select VT mode'' when you wish to switch back, then reset the terminal type to 'vt100'.

By the way, you may be wondering why you can't just leave the xterm as a t4010. Here's why: a real Tektronix 4010 has a so-called storage tube phosphorescent green graphics display. Although it is a nice graphics terminal, it is impractical (and annoying) to use for simple text work. You can't back up the cursor and it has a tendency to scribble over existing lines. Project Athena's Tektronics xterm, too, does these things. With this in mind, we suggest that you switch back and forth: vt100 for text and t4010 for graphics.

6.3 Curve Fitting

To fit a line to the ''logspeed_error'' graph, type the command:

 # fit line to curve 1 of graph logspeed_error

```
THE EQUATION OF THE BEST FIT IS:     <--RS/1 responds.
          -14.677698*X +46.156958
Do you want to compute GOODNESS-OF-FIT STATISTICS? [Yes]
       .     .
       .     .
       .     .                  <--Output continues.
```

The FIT LINE command performs a linear fit, and will display the regression results including residual, parameter, and ANOVA tables.

FIT POLYNOMIAL, FIT FUNCTION (non-linear regression), and FIT MULTIPLE (stepwise multiple regression) commands are also available, non-interactively or interactively, and the functions may be fit from a graph (as in our example) or from a table-portion. See ***User's Guide***, Book 2, Section 7, pages 165 to 194.

6.4 Error Bars

You can add error bars to your graphs by constructing a table which contains the high and low values for each data point in the graph, and then calling an interactive public procedure:

 # call public $add_error_bars

An example of this command is presented in the ***User's Guide***, Book 2, pages 26 through 29.

6.5 Graph Editor

After you have constructed a graph, you may want to change some of its features to improve its presentation. The easiest way to edit a graph is to use the Graph Editor. The Graph Editor is an interactive procedure that prompts you for the parts of the graph you wish to change. Various editing options are available for the four types of graphical objects. To invoke the graph editor on an existing graph, type:

 # edit graph *<graph-name>*

See Book 2 of the ***User's Guide***, Section 6.2 for detailed examples.

6.6 Storing Graphs

All the graphical objects you create are listed in your RS/1 directory. You can list, rename, copy, and delete them, in the same ways that you manage tables. Book 2 of the ***User's Guide*** presents detailed information on graphical data objects.

7. Hardcopy Output

7.1 Printing Out Tables and Graphs

Printing out graphs and tables has never been easier! RS/1 Release 3 has several features that facilitate getting hardcopy output. Since there are currently no default printers for Athena clusters, you need to specify a PostScript printer for RS/1 to queue output to. You do this using:

```
# call $set_printer_queue
```

Hitting return will give the previous printer used, if any. This is not necessarily the most appropriate printer to use!

The general procedure for getting hardcopy output from RS/1 is to create a temporary file from within RS/1, then print it out. These commands are analogous to the DISPLAY command, and take all the same options (norownumbers, bottomkey, etc.). If you specify the argument on file, it will place the output in a permanent file but NOT send it to the printer. The file is a valid UNIX filename, hence it is case sensitive and also must be enclosed in quotes.

7.1.1 Standard Tables

To get a UNIX file of a table, use the ***printout*** command. It takes the usual arguments: noheader, nocolnumbers, norownumbers, etc.

```
# printout  table data norownumbers          <-- printer
                        or
# printout  table data noheader on file 'data'  <-- file
```

If you do *not* specify a file RS/1 will print your file on the printer specified by the **call $set_printer_queue** procedure.

7.1.2 Wide Tables

You can print extra large tables by specifying 132 columns and then using the enscript command from UNIX. Note that need to use an intermediate file, and the RS/1 command 'chain' to use UNIX commands.

```
# set printer  to 'lpt132'
# printout  table longfoo on file 'lfoo'
# chain  'enscript -r -Pprinter lfoo '
# set printer  to 'lpt80'
```

You need to change back to 'lpt80' for regular tables.

7.1.3 Graphs

Graphs are done in the same way as tables, except that you use the 'plot' command to print them out in PostScript.

```
# plot  graph qikpict norownumbers                <-- printer
                        or
# plot  graph qikpict noheader on file 'picture.PS' <-- file
```

Pictures saved in the second manner can be included in Scribe file with suitable modification.

Some of the display options there are very useful are:

nocolnumbers norownumbers noheader bottomkey nokey

For example:

```
# printout  table thesis_data nocolnumbers  norownumbers
# plot  graph mypix noheader  bottomkey
```

7.2 Including Graphs in Scribe Files

Since tables are text, you can insert them directly into a Scribe @Verbatim environment. There is now a way to include RS/1 graphs in Scribe as well. First, prepare the graph, and make sure it looks the way you want it to. Next, set the plotter variable:

```
# set plotter to 'ps_include'
```

This is a special format modified to allow PostScript output to be included in Scribe files. Because of this, files produced in this way can NOT be printed out by themselves.

Plot out the graph directly onto a ".PS" file:

```
# plot graph labdata on file 'labdata.PS'
```

Now you can include this graph in Scribe by using the ‘‘@Picture’’ command. This is best done in the context of an ‘‘@FullPageFigure’’, as described in "More Scribe: Reports". You should use something that looks like this:

```
@Begin(FullPageFigure)

@Picture(Width 6.5 inch, Height 8.25 inch, PostScript ‘copy.PS’)

@Caption(This is an RS/1 Graph)
@End(FullPageFigure)
```

8. Odds And Ends

RS/1's capabilities extend far beyond the scope of this manual. We urge you to look through the three-volume set of BBN documentation to familiarize yourself with the applications that might be especially useful to you. Meanwhile, we touch on a few more of the major features of RS/1 in this section.

8.1 Statistics

RS/1 statistics commands give you access to descriptive statistics, t-tests, F-tests, nonparametric tests, analysis of variance and covariance, and contingency table analysis. Here we illustrate a few simple examples. Many more functions are available; for more information, see ***User's Guide*** Book 2, Section 8.

The MEASURE command displays the complete set of descriptive statistics for a table-portion, in a table labeled ‘‘MEASURES’’:

 # measure col ’speed’ of reading

```
MEASURES  12R x 1C

0 Statistic           1 Value
_________________________________
 1 Count (N)           11.000000
 2 Sum                180.500000
 3 Mean                16.409091
 4 SEM (s.e. of mean)   0.991276
 5 Median              15.200000
 6 Variance            10.808909
 7 StDev (sd)           3.287691
 8 Maximum             25.500000
 9 Minimum             13.500000
10 Range               12.000000
11 Skewness             2.446000
12 Kurtosis             6.745871
```

The MEASURES table is overwritten when you re-issue the command. To save it, rename it with a unique name.

You can also request individual statistics on a row or column:

 # type maximum of col ’viq’ of reading

or display any row or column statistic for an entire table-portion:

 # display colstdev of cols 5 to 7 of reading

For more information on this procedure, see the *User’s Guide*, Book 2, pages 196 through 200.

The CORRELATE command calculates Pearson correlation coefficients:

 # correlate col 5 of reading with col 7

8.1.1 Distribution Tables

RS/1 allows you to categorize data and display its tabulation in a distribution table. This type of table shows the number of occurrences of observations in a given range or of a particular type. Either one-way or two-way classifications are possible. The results of one-way classifications are easily presented as bargraphs. To create a distribution table, use the MAKE TABLE command interactively:

 # make table reading as distribution

Examples of such tables, and details on this special form of the MAKE TABLE command, are given in Book 1 of the *User’s Guide*, pages 88 through 92.

8.2 Models

RS/1 allows you to create spreadsheet-like models of any system that can be described through a set of computational rules. You enter data values and formulas into a table with the MAKE MODEL command, and then run the model with the COMPUTE command. Detailed information about model construction and computation is found in the ***User's Guide***, Book 1, Section 5.

8.3 RPL Programming Language

You can develop your own RS/1 procedures, and package existing RS/1 commands to perform specially tailored analyses. For these tasks, you will need to learn RPL, RS/1's research programming language. RPL allows you to write simple or sophisticated procedures for terminal and file input/output, numeric manipulations, table manipulations, and terminal-independent graphics. A complete description of RPL is in Book 3 of the ***User's Guide***.

Table of Contents