VLBI Software Documentation
Field System

logpl: Plot Log Data

Fredrik Lindenau
NVI, Inc./GSFC

Program Reference Manual

NASA/Goddard Space Flight Center
Space Geodesy Program

Version 9.3
September 1, 1997

Typographical Conventions

The typographical conventions used in all the manuals for Field System version 9.0 are as follows:

Times New Roman	All normal text.
Arial bold	Field System manual names.
`Courier`	Any computer-related items, such as prompts, example screen displays and printed output, file names and directory names, program names, and Linux utilities.
Arial italics	A variable quantity for which a specific value must be specified, such as the parameters to a command.
Arial bold italics	A variable quantity to be typed in by the user in a commad.
`Courier bold`	Commands and verbatim type-ins by the user.
Times New Roman Italics	Used for emphasis in the text.

Table of Contents

1.0 Introduction

`logpl` is a program to examine Field System log files. This includes making selections of the data to be examined, the type of data as well as the time period of interest. Data can also be plotted, using an interactive graphical user interface, and printed, in encapsulated postscript format. The horizontal axis of the plot is time, the vertical axis is selected data values from the log file.

Although `logpl` supports a graphical user interface, the program can also be started in a text-based command-line mode. This enables the possibility of making script files, for example to make standard plots after experiments. However, when started in command-line mode, plot output can only be made to a postscript printer, or to a file in encapsulated postscript format. The file can then be viewed by postscript viewers, such as `ghostview`. Also, selections of data from the log file can be printed to standard output.

1.1 On Field System log files

The Field System makes log entries in the format *TimeTypeCommandData*, where:

Time A 13-digit ASCII string in the format *yydddhhmmssss*. For example, **`9720517224415`** means year 97, day 205, time 17:22:44.15.

Type A single character indicating the type of entry.

Command A string that defines what type of data will follow in the *Data* section. For example, `wx/` means that weather data will follow.

Data A set of comma-separated data.

`logpl` extracts data from log files by letting the user specify a selection. A selection is defined by a command, a parameter and an optional matching string. The command is the Field System log file command that should be extracted and the parameter is the index in the comma-separated data list following the command. The optional search string allows the user to specify another criteria for selecting data, since the log file entry must also contain the search string to be selected.

1.2 Definitions

Log file	Field System log file
Log directory	directory for log files, usually `/usr2/log`
Command	The string following directly after the time entry on the log file.
Selection	An extraction of data from a log file, defined by its Command, Parameter and String.

2.0 User's guide—Graphical User interface

2.1 Getting started: the `File` menu and the main screen

To start `logpl`, just type **`logpl`** at the command prompt. A window will open on your screen, from where you can run `logpl` interactively with your pointing device. If no window opens, please go to section 7.0 Installation for further reference.

2.1.1 The `New` menu item

To specify a new log file for input, select the `New` menu item in the `File` menu. A window will open, labeled `Open new log file`, in which you should enter the filename of the log file you want to examine. If you do not enter a directory path, `logpl` defaults to the log directory. In order to open a file in the current directory (from which `logpl` was started), you must enter ***`./filename`***, to specify that the log file is in the current directory.

When hitting return in the `Open new log file` window, `logpl` tells you if the filename was valid or not. If the file did not exist, the extension `.log` is assumed and checked. If the filename was a valid log file, the station name and starting day number is displayed. Hit return again, or press `OK`, to accept this log file. (NB! When running `logpl` under Tk 4.0 and higher, `logpl` may open the log file without first displaying the day number for confirmation.)

Two error messages are possible in the `Open new FS log file` window. The first, `The file specified could not be opened` is displayed if thesystem call to open the file generated an error. The most probable reason for this is that the file did not exist. Please note that `logpl` defaults to the Field System log directory, not the current directory, unless otherwise specified.

The second error message, `The file specified did not start with a valid day no.` is displayed if the file did exist, but did not start with five numeric (range 0–9) characters. All Field System log files should start with a day number of five numeric characters. If the file does not, it may be corrupt and must be repaired before it can be opened by the `logpl` program.

2.1.2 The Print menu item

You can print any plot by selecting the Print menu item in the File menu. Plots are printed in encapsulated postscript (eps) format

When you select the menu item, a window will open where you can select the print destination. If you select destination Printer the lpr or psprint command will be used to print the plot. The command or script to be used for printing can also be selected in the print window.

If you select destination File the postscript will be saved to the file specified in the Filename entry field. If you do not specify a directory, the current directory is used as default. Also, you may select whether the file should be overwritten or if the data should be appended to the end of the file.

2.1.3 The I/O setup menu item

After selecting the I/O setup menu, a window will open that contains three entry fields. The entry fields contain filenames and paths for logpl's disk I/O operations. They can be changed, but please note that the changes are only temporary. To change the default values, the initialization section of the source code must be changed. See section 6.0 for information on how to do that.

The first entry field, labeled Default directory for FS log files is where logpl looks for log files when the user tries to open a log file without specifying a directory path. This field should give the path to the FS log file directory, usually /usr2/log.

The second entry field is labeled Base filename for logpl temp files. logpl creates a number of temporary files during runtime. However, all of them use the same base filename. logpl automatically adds an extension to the base filename specified in order to keep track of its temporary files. All temporary files are removed when terminating logpl normally, that is, using the Exit command.

The third entry field gives the filename of the current logpl control file used. The control file can be used to create a number of pre-defined data selections. See section 4.2 for information on how to use the control files. If no control file was found at start-up, the Chan menus will not have any entries. In that case, the Edit selections menu item in the Options menu must be used to specify something to plot. However, the entry field in the I/O setup window will still tell you where logpl looked for the control file.

You may specify another control file in the entry field. However, this will destroy all selections previously made, replacing them with the selections defined in the new control file. If you specify

an invalid file name, the previous selections will be preserved, and the control file name will not change.

2.1.4 The `Exit` menu item

To terminate `logpl`, select the `Exit` menu item in the `File` menu. `logpl` will remove any temporary files it created and then terminate.

2.1.5 The `Time` box on the main screen

The time box has three entry fields, `Tmin` (start time), `Tmax` (stop time) and `#Pts` (number of data points). These fields let the user specify different time ranges to examine. The time data must be entered in the form *YYDDDHHMMSSSS*. However, only the *YYDDD* part is required. If hours, minutes, seconds or fractions of seconds are left out, they will be set to zero. That is, zeros are appended to the string until it is 13 characters long, provided that the original length was greater than, or equal to, 5.

You may also specify a number of data points to plot, after the current start time. What happens is that `logpl` calculates a stop time based on that number of data points. `logpl` then plots the new time period.

The screen is re-plotted when you hit return in any of the entry fields of the time box. Now, if you for example want to specify a new start time as well as a new stop time, you should use the mouse to move the cursor between the fields, without hitting return, to avoid unnecessary re-plotting.

2.1.6 The `Y-axis` box on the main screen

The `Y-axis` box has two entry fields and four radio buttons. The radio buttons, labeled `1`–`4`, are used to change the current data channel. `logpl` can plot a maximum of four channels of data, superimposed or non-superimposed. When changing the current data channel, the values in the entry fields are changed to the maximum and minimum values of the current plot of that data channel. Also, the `#Pts` field in the time box is changed to the number of data points on-screen for that channel.

The entry fields, labeled `Ymin` and `Ymax`, let the user specify a new range of Y-axis values to plot. Any floating point value may be entered here. For example, the format "**`3.000e+3`**" for "**`3000`**" is also allowed for input. As with the time entry fields, the screen is re-plotted when you hit return in one of these entry fields. The new Y-axis range is read, and the plot updated.

2.2 Viewing and editing data: the `Edit` menu

`logpl` supports manual editing of data. You can click on data points to mark them as "bad". What happens is that the data point's symbol will be outlined on the screen, and it will be disregarded in any plotting or statistical functions.

When editing data, there are some things to notice. The `#Pts` entry field of the time box on the right part of the screen tells you how many active data points are currently plotted on-screen. As you edit data points out by clicking on them, this value will decrease. You may notice that sometimes this value will decrease by more than one point. The data editing in fact removes all data points enclosed in a small box around the coordinates you clicked. This is useful, because sometimes you have two data points with the same coordinates, and usually you want to mark both of them as bad. If you do not, you can always re-add one of them with the `Undo` option, as described in section 2.2.1.

2.2.1 The `Undo last remove` menu item

When selecting this menu item, the data point you most recently removed, in the current data channel, will be re-added as active. If the data point is within the time range of the current plot, the data point will also be filled solid on the screen, and the number of active points on-screen in the time box will be incremented. If the data point was outside the time range of the current plot, it will seem like nothing has happened. However, when you click the `Reset` button in the time box, or specify a time range including the re-added data point, it will be displayed as active.

The `Undo last remove` operation is a continuous process. You can re-add all removed data points at any time, by selecting the menu item repeatedly. When no more data points remain to be re-added, selecting the menu item will have no effect.

2.2.2 The `View source` menu item

This menu item lets you view the source data for the current plot. All the floating point values that were successfully extracted from the FS log file will be listed. The source list is displayed in a separate window. You can print this window, but please note that it is printed in ASCII text format, not in postscript format as is the case with plots.

Also, the list will only contain data points within the current specified time range. To view all data points, you should first hit the `Reset` button in the time box.

2.2.3 The `View log file` menu item

This menu item lets you view the log file entries that match the current selection. All matching log file entries are listed, not only the ones from which a floating point value was successfully converted. This allows you to make selections for comments that cannot be plotted (because they do not contain any floating point values). However, the comments can be displayed by selecting the `View log file` menu item.

The list is displayed in a separate window. You can print this window, but please note that it is printed in ASCII text format, not in postscript format as is the case with plots. Also, the list will only contain log file entries within the current specified time range. To view all current selection entries, you should first hit the `Reset` button in the time box.

2.3 Preparing plots: the `Options` menu

To plot data, you must first define a data selection, for extracting data from the log file. This can be done in two ways. You can pre-define selections by using `logpl` control files. See section 4.2 for information on how to do that. The second way is to create or modify selections using the ineractive `Edit selections` menu item in the options menu. When a selection is created, it will become available in the `Chan` menus, from where it can be selected and plotted.

2.3.1 The `Edit selections` menu item

When selecting this menu item, a new window will open on your screen. In this window, you may define your selections. The window has the following entry fields:

`Selection number` This is a unique number for the selection. It cannot be changed.

`Command` This is the log file command to search for. For example, `wx/` means weather data will be selected, and `cable/` means that cable-length data will be selected.

`Parm` This is the index of comma separated data that will be used. For example, for the `wx/` command, `1` means temperature data, and `2` means pressure data.

`String` This argument is optional. If it is left empty, all log file entries matching the current command will be selected. However, sometimes it is useful to restrict data selection to only those log file entries that contain a certain string anywhere in the entry. That string is specified in this entry field.

`Description` The selection description will be used in the `Chan` menu, as well as on the plots. It should be easy to understand and unique. It will also be on the postscript prints when a plot is printed.

The window also has five buttons. The three buttons on the upper line are for navigating among the available selections. You can examine all available selections by clicking on the `<Prev` and `Next>` buttons. To create a new selection, click the `New` button. The next available selection number will be displayed, and the cursor will go to the command window for input.

When you have changed any selection, you must click `Update` to accept the changes. If you close the window or go to another selection without updating, the changes will be lost. When you are done editing the selections, click the `Close` button to close the window. After that, your selections will be available in the `Chan` menus.

2.3.2 The `Superimpose` option

If `Superimpose` is on, which is shown by the checkbox in the menu, the plots selected on the different data channels will be superimposed. Since `logpl` supports four separate data channels, up to four plots can be superimposed. The option is useful for discovering similarities between different data selections. If `Superimpose` is off, the plot area will be divided into different areas for each of the plots.

Note that all data channels are always re-plotted when toggling `Superimpose` on and off.

2.3.3 The `Connecting line` option

This option adds a connecting line between all active data points. This will sometimes make plots easier to read, especially superimposed plots. The lines can be toggled on/off by selecting the menu item repeatedly. Note that creating the lines might take some time on older computers. Also, please note that when the connecting line option is on, the lines are not removed when data points are edited out. For the lines to those data points to disappear, you would have to re-plot the lines. The simplest way to do this is to toggle the connecting lines off and on once.

2.4 Plotting data: the `Chan` menus

As described in section 2.3, there are two ways to create data selections to plot, by the `Edit selections` menu item in the `Options` menu, and by `logpl` control files. Please see sections 2.3.1 and 4.2, respectively, for information on how to create selections.

When data selections have been created, they are available in all the `Chan` menus. `logpl` supports four different data channels. They are separated in the plots by having different data point symbols (small circles, rectangles or triangles) and different colors on the screen.

2.4.1 Plot a data selection: the `Chan` selection menu items

When you start `logpl`, all `Chan` menus are set to display `No data`. If you have created a selection labeled, for example, `My selection`, just select this menu item in one of the `Chan` menus to plot it on the screen. If the data selection is not already in memory, it will be read from the current log file. By default, `logpl` plots all data points in the selection, from the start of the log file to the end of it, unless otherwise specified in the `Time` box on the right part of the screen. When new selection is plotted for the first time, it is always auto-scaled. Both the time and scale can be changed later, see section 2.1.4 and 2.1.5 for information on re-scaling.

Plotting may take some time, especially on older computers. When the number of active channels (that is, the number of channels set to show other than `No data`) is changed, all active data channels are always re-plotted (unless the `Superimpose` option in the `Options` menu is on). Therefore, going from three to four data channels takes more time than changing an already active channel to plot another data selection.

For further information on how plotting is done, see section 6.0 of this manual.

2.4.2 The `Invert` option

To see similarities between plots, it can sometimes be useful to invert a plot, i.e. having the minimum value on top and the maximum value on the bottom, instead of the other way around. This can be done by selecting the `Invert` option in the channel menu. When toggling `Invert` on and off, that data channel is automatically re-plotted.

2.4.3 The `Log scale` option

A plot with a logarithmic scale can sometimes be useful for detecting small variations in data values. You can have the Y-axis of a plot scaled with a logarithmic scale by selecting the `Log scale` menu option in the respective channel menu.

Please be aware of that negative values cannot be log-scaled. If the current Y-axis scale in the Y-axis box on the right part of the screen is not all positive, the `Log scale` command will be ignored. The channel will be re-plotted, but without log scale. If the Y-axis scale is positive,

the plot will be log-scaled. However, if the current selection contained one or more negative values, these values will be ignored. Also, if the negative data points were active, i.e. not edited out, a warning message will appear, informing the user that these values have been ignored.

3.0 User's Guide—The Command-line Interface

`logpl` also supports a command-line interface. However, some features that are available in the interactive interface are not available using the command-line interface. For one thing, you can only have output to a printer or a file. Neither plots nor data can be viewed on the screen. Also, data editing is not supported. However, the command-line interface can be useful for printing standard plots, since the command-line interpreter also can read script files. This enables you to set a number data selections, and their scales, and print them to the printer.

3.1 Getting started

To start `logpl` in command-line mode, start the program by typing

`logpl -cmd`

on the command line. You may see a window appearing for a short time on your screen. This is because `logpl` must open a window to be able to create any plots. However, the window will soon be iconified, and you will get a prompt, saying `logpl>` .

To help you get started, there is a `help` command, giving a short description of every command available. Just type **`help`** and press return at the `logpl` prompt to view the help text. If you have been running `logpl` in the normal graphical interactive mode, you will find that many commands are very similar to menu selections in that interface. For example, the **`log`** command triggers selection of the `New log file` menu item, the **`channel`** command triggers a click on the channel selection buttons in the Y-axis box, etc.

3.1.1 On command recognition

Although the command for setting the output filename is **`output=`*filename***, it would have been enough to just type **`out=`*filename***. In fact, for `logpl` to recognize a command you only are required to type enough characters before the **`=`** character to make the command uniquely recognized. However, at least two characters must be typed for any command. For example, the command **`ou=`*filename*** would work, but **`o=`*filename*** would not.

Also, all commands are converted internally to lower-case characters. The case of anything typed before the **`=`** character does not matter.

All but a few of the commands are used to set values. The syntax for these commands is always **command=*value,value...***. To query a value, without setting a new one, you can always type the command without the **=** character. The current value(s) will be printed to the screen. To clear a value (for those commands that allows clearing of values), the syntax is **command=**, that is, nothing should follow the **=** character. Invoking this syntax on commands that do not allow clearing of values will have the same effect as if the **=** character was omitted. That is, the current value will be printed.

3.1.2 Some examples

To print a temperature data plot from the log file `station.log` in the Field System log file directory, you should type the following commands:

```
log=station.log
output=printer
command=wx/
parm=1
plot
```

If you also had a control file `logpl.ctl` in the Field System control file directory, that had a temperature selection on line 5, the following commands would give the same result:

```
log=station.log
output=printer
select=5
plot
```

Please see section 3.3 for further examples on `logpl` scripts.

3.2 Command reference

This section describes each of the commands available in the `logpl` command-line interface, in alphabetical order.

3.2.1 ***cfile***

Syntax: **cfile=*filename***

The **cfile** command causes logpl to read a file named ***filename***, which contains logpl commands to execute, one per line. The commands are executed in the order in which they appear in the file.

If logpl was started with the **-cmd** flag, control will be returned to the user when the end of the command file is reached. However, the **cfile** command can also be invoked from the command line when starting logpl. If logpl is started by typing **logpl -cfile *filename* -log station.log**, the log file station.log will be used for input, and the command file *filename* will be executed. When started in this fashion, logpl will terminate when the end of the command file is reached. See section 4.1 for more information on the command line flags for logpl.

If an error is encountered when running a logpl command file, the execution of commands is halted. After printing an error message, logpl will behave as if the end of the command file had been reached.

*3.2.2 **channel***

Syntax: **channel**
channel=*channel_number*

This command is used for setting the active channel. The active channel is used by a number of other commands, like **command**, **parm**, **list**, **invert** and more.

logpl supports four channels of data, which can be plotted superimposed, or non-superimposed. The **channel** command is used to inform logpl of which of the channels you want data to appear on, when creating selections with the **command** and **parm** commands. The ***channel_number*** must be an integer in the range **1**–**4**. The default is **1**.

For example, to set the parameter for data channel four, that channel must be made active, by using the command **channel=4**. Then the parameter for that channel can be set with the **parm** command.

*3.2.3 **command***

Syntax: **command**
command=
command=*command*

With this command you can create a new data selection of log entries from a log file. Only log entries that include the command specified will be selected for listing or plotting. When searching

for a command, `logpl` only searches the string immediately after the time field and type-character in the log file. However, a data selection for plotting is defined by a command and a parameter. To make plots (i.e. not only log file entry lists with the **`glist`** command) the **`parm`** command must be used in addition to the **`command`** command.

The command search can be cleared by typing **`command=`**. This will cause all log entries to be selected, provided that the optional matching string (see **`string`** command) is also cleared.

3.2.4 *`control`*

Syntax: **`control`**
`control=`*`filename`*

This command is used for reading a new `logpl` control file specifying a number of pre-defined data selections. When a new control file is read, the pre-defined selections of the control file previously used are destroyed. They are replaced with the selections of the new control file. However, if the file specified does not exist, the previous selections will be preserved.

3.2.5 *`exit (quit, ::)`*

Syntax: **`exit`**
`quit`
`::`

This command causes `logpl` to remove all temporary files created and terminate.

3.2.6 *`glist`*

Syntax: **`glist`**
`glist=reset`
`glist=`*`start_time`*
`glist=`*`start_time,stop_time`*
`glist=`*`start_time,#number_of_entries`*

This command is used to list log file entries. Also, the minimum and maximum time values of the current plot may be set. If **`glist`** is invoked with no parameters, the current values will be used. The default is between the lowest and highest time found in the log file. To reset the time values to their defaults, type **`glist=reset`** at the command prompt.

The time data must be entered in the form ***YYDDDHHMMSSSS***. However, only the ***YYDDD*** part is required. If hours, minutes, seconds or fractions of seconds are left out, they will be set

to zero. That is, zeros are appended to the string until it is 13 characters long, provided that the original length was greater than, or equal to, 5. Note that the **#** character must be used to specify a number of log entries instead of a stop time. If the **#** character is specified, `logpl` will calculate a stop time based on the number of log entries specified.

`Logpl` will print a list of all log entries that match the current selection, as specified with the **command**, **parm** and **string**, or **select** commands. The list will be printed in ASCII text format to the output device specified with the **output** command (file or printer).

3.2.7 ***help***

Syntax: **help**

This command prints a help text on the screen, giving a short description of `logpl`'s command-line commands.

3.2.8 ***invert***

Syntax: **invert**
invert=0
invert=1

To see similarities between plots, it can sometimes be useful to invert one of them, i.e. having the minimum value on top and the maximum value on the bottom, instead of the other way around. The **invert** command sets that option on (**1**) or off (**0**), for the current data channel.

3.2.9 ***line***

Syntax: **line**
line=0
line=1

This option adds a connecting line between all active data points in plots created. This will sometimes make plots easier to read, especially superimposed plots. This option affects all data channels, and is turned on (**1**) or off (**0**) with the **line** command.

3.2.10 ***list***

Syntax: **list**
list=reset
list=*start_time*
list=*start_time,stop_time*

list=*start_time*,#*number_of_points*

This command is used to list the source values for the current data selection. Also, the minimum and maximum time values of the current plot may be set. If **glist** is invoked with no parameters, the current values will be used. The default is between the lowest and highest time found in the log file. To reset the time values to their defaults, type **glist=reset** at the command prompt.

The time data must be entered in the form ***YYDDDHHMMSSSS***. However, only the ***YYDDD*** part is required. If hours, minutes, seconds or fractions of seconds are left out, they will be set to zero. That is, zeros are appended to the string until it is 13 characters long, provided that the original length was greater than, or equal to, 5. Note that the **#** character must be used to specify a number of log entries instead of a stop time. If the **#** character is specified, `logpl` will calculate a stop time based on the number of log entries specified.

`logpl` will print a list of all floating point values that match the current selection, as specified with the **command**, **parm** and **string**, or **select**, commands. The list will be printed in ASCII text format to the output device specified with the **output** command (file or printer).

3.2.11 ***log***

Syntax: **log**
log=*filename*

This command is used to specify a new FS log file for input. If ***filename*** does not contain a directory path, `logpl` defaults to the FS log directory. In order to open a file in the current directory (from which `logpl` was started), you must enter **./*filename***, to specify that the log file is in the current directory. If the file did not exist, an extension `.log` is assumed and checked.

If the filename was a valid log file, the station name and first day number is printed on the screen. If something went wrong, an error message is printed. Two error messages are possible. The first, `The file specified could not be opened` is displayed if the system call to open the file generated an error. The most probable reason for this is that the file did not exist. Please note that `logpl` defaults to the log directory, not the current directory, unless otherwise specified.

The second error message, `The file specified did not start with a valid day no.` is displayed if the file did exist, but did not start with five numeric (range 0–9) characters. All Field System log files should start with a day number of five numeric

characters. If the file does not, it may be corrupt and must be repaired before it can be opened by the `logpl` program.

As with the **`cfile`** command, the **`log`** command can be invoked from the command line when starting `logpl`. If `logpl` is started by typing **`logpl -cfile`** ***`filename`*** **`-log station.log`**, the log file `station.log` will be used for input, and the command file ***`filename`*** will be executed. See section 4.1 for more information on the command line flags for `logpl`.

3.2.12 ***`lscale`***

Syntax: **`lscale`**
`lscale=0`
`lscale=1`

A plot with a logarithmic scale can sometimes be useful for detecting small variations in data values. You can have a plot log-scaled by invoking the **`lscale=1`** command. The **`lscale`** command only affects the current data channel.

Please be aware that negative values cannot be log-scaled. If the current Y-axis scale, as specified by the **`scale`** command, is not all positive, the log scale will be ignored when plotting. The channel will be plotted, but without log scale. If the Y-axis scale is positive, the plot will be log scaled. However, if the current selection contained one or more negative values, these values will be ignored.

3.2.13 ***`output`***

Syntax: **`output`**
`output=printer`
`output=printer,lpr`
`output=printer,psprint`
`output=`*`filename`*
`output=`*`filename`*`,overwrite`
`output=`*`filename`*`,append`

The output command is used for specifying the current output setting. When invoking a **`list`**, **`glist`**, or **`plot`** command, data is written to whatever destination is specified with the **`output`** command.

If the output destination is specified as **`printer`**, output will be written to a temporary file, and then printed, using the print command specified after the comma. The default is **`lpr`**. The

temporary file will be removed after having been sent to the printer. Note that, since "printer" is a reserved word, you cannot name an output file `printer`.

If the output destination is specified as a filename, output will be printed to that filename. You may also choose whether the output file should be overwritten or appended to, by specifying that after a comma. The default is **`append`**. Please note that the **`plot`** command prints postscript plots, while the **`list`** and **`glist`** commands prints text in ASCII format. You may not want to append postscript to an ASCII text file, or vice versa, but this will not be checked by `logpl`. You will have to specify a different filename before invoking a command that prints a different kind of output.

3.2.14 ***`parm`***

Syntax: **`parm`**
`parm=`*`n`* (***`n`*** is an integer number greater than zero)

With this command you can specify a new data selection of log entries from a log file. The parameter is the index in the list of comma-separated data following the command in log files. However, a data selection for plotting is defined by a parameter and a command. To make plots, the **`command`** command must be used in addition to the **`parm`** command.

See section 1.1 for further information on commands and parameters in Field System log files.

3.2.15 ***`plot`***

Syntax: **`plot`**
`plot=reset`
`plot=`*`start_time`*
`plot=`*`start_time,stop_time`*
`plot=`*`start_time,#number_of_points`*

This command prints a plot in postscript format to the output device specified with the **`output`** command. All channels that are not set to display `No data`, which is the default, are plotted. If a selection has been specified with any of the **`command`**, **`parm`**, or **`string`** commands, the current data channel is automatically set to plot that selection. Otherwise, the **`select`** command must be used to set the current data channel to plot a selection.

The minimum and maximum time values for the plot may also be set. If **`plot`** is invoked with no parameters, the current values are used. The default time range is between the lowest and highest time found in the log file. To reset the time values to their defaults, use the **`reset`** parameter to the plot command.

The time data must be entered in the form ***YYDDDHHMMSSSS***. However, only the ***YYDDD*** part is required. If hours, minutes, seconds or fractions of seconds are left out, they will be set to zero. That is, zeros are appended to the string until it is 13 characters long, provided that the original length was greater than, or equal to, 5. Note that the **#** character must be used to specify a number of log entries instead of a stop time. If the **#** character is specified, `logpl` will calculate a stop time based on the number of log entries specified.

The current scale values for the current data channel are used to determine the Y-axis scale of the plot. These values can be set using the **`scale`** command. Also, when a channel is set to display a new selection, the Y-axis scale values are always reset, so that the plot will be autoscaled.

3.2.16 ***`query`***

Syntax: **`query=`*`variable_name`**

This command is only used for system maintenance. You can display the value of any internal Tcl variable used in `logpl`, by invoking the query command. Note that array variables must be given the correct number of indices or `logpl` might bail out.

3.2.17 ***`scale`***

Syntax: **`scale`**
`scale=0,0`
`scale=`*`minimum,maximum`

The **`scale`** command let the user specify a new range of Y-axis values to plot. Any floating point value may be specified. For example, the format **`3.000e+3`** for **`3000`** is allowed for input. If the scale is set to **`0,0`**, the plot will be autoscaled.

The **`scale`** command only affects the current data channel. To specify a new scale for another data channel, the **`channel`** command must first be used to set that channel as active. Also, the scale is always reset to **`0,0`** (autoscale) when the current selection for the data channel is changed. The current selection can be changed with any of the **`command`**, **`parm`**, **`string`**, or **`select`** commands.

3.2.18 ***`select`***

Syntax: **`select`**
`select=0`

select=*n* (***n*** is an integer value greater than zero)

This command is used to determine what data selection should be plotted on the current data channel. Note that, if either a **command**, **parm**, or **string** command is invoked, the channel is always set to display the selection defined by those commands. However, you can set the channel to show a pre-defined selection, from a `logpl` control file, by invoking the **select** command with the line number of the selection in the control file. For example, if your control file's third line has a selection labeled `Pressure`, invoking the command **select=3** will set the current channel to display that selection.

Invoking the command **select=0** will cause the current channel not to display any data at all.

3.2.19 ***string***

Syntax: **string**
string=
string=*search_string*

With this command you can specify a new data selection of log entries from a log file. Only log entries that contain the search string specified will be selected for listing or plotting. However, a data selection for plotting is defined by a command and a parameter. The **string** command is only optional. If the no search string is specified, all log file entries matching the command set by the **command** command will be selected. To make plots, the **command** and **parm** commands must also be used.

The search string can be cleared by invoking the command **string=**. This will cause all log entries matching the command set by the **command** command to be selected. If both the command and the search string are cleared, all log file entries will be selected.

3.2.20 ***super***

Syntax: **super**
super=0
super=1

If superimpose is on (**super=1**), the plots that are selected on the different data channels will be superimposed. Since `logpl` supports four separate data channels, up to four plots can be superimposed. The option is useful for discovering similarities between different data selections. If superimpose is off (**super=0**), the plot area will be divided into different areas for each of the active data channels (that is, channels set to plot data selections).

3.3 Writing scripts for `logpl`

As explained in section 3.2.1, `logpl` can read a script, containing `logpl` commands, one command per line. The commands are executed in the order in which they appear in the file. This can be useful to print standard plots from log files. If `logpl` detects an unrecognized command in the script, it will stop executing the file and print an error message on the screen.

3.3.1 Suggestions for scripts

When developing scripts, it is often best to start the script with the **`cfile`** command, as explained in section 3.2.1. However, to start a pre-written script, it may be better to start `logpl` with both the **`-log`** and **`-cfile`** command-line flags. If `logpl` is started by typing **`logpl -cfile`** ***filename*** **`-log station.log`**, the log file `station.log` will be used for input, and the command file *filename* will be executed. See section 4.1 for more information on the command line flags for `logpl`.

3.3.2 An example

This is an example of a script to print a plot of temperature, pressure, humidity and cable-length data. The plots are printed separately on one single piece of paper. The name of the log file must be specified on the command line.

```
output=printer
command=wx/
channel=1
parm=1
channel=2
command=wx/
parm=2
channel=3
command=wx/
parm=3
channel=4
command=cable/
parm=1
plot
```

4.0 More on Starting logpl

4.1 The command line

logpl accepts four optional flags on the command line when started. To start logpl in anything else than its normal, graphical interactive mode, some of these flags must be used. For example, the **-cmd** flag starts logpl in its command-line mode.

4.1.1 ***-cfile***

Syntax: **-cfile** ***filename***

This flag starts the command-line interpreter of logpl. However, the first command issued will be the **cfile** command, which transfers control to a pre-written script containing logpl commands. See section 3.3 for more information on these scripts. When started with the **-cfile** flag, logpl will terminate when the end of the script is reached or when an error occurs.

4.1.2 ***-cmd***

Syntax: **-cmd**

This flag makes logpl start in the text based command-line mode, instead of the normal, graphical interactive, mode. If starting logpl in this mode, the prompt logpl> should appear on the screen, after an initialization process has been completed. Note the initialization process also includes opening a window, since this is necessary for making plots in graphical mode. However, the window will be immediately iconified.

4.1.3 ***-control***

Syntax: **-control** ***filename***

As default, logpl starts with loading the control file named logpl.ctl in the /usr2/control/ directory, if the file exists. However, the user can start logpl with another control file, by giving the path to the control file after the **-control** flag on the command-line.

4.1.4 ***-log***

Syntax: **-log *filename***

This flag makes `logpl` start with ***filename*** as the log file for input. The filename is treated as a filename entered in the `Open log file` window. That is, if no directory is specified, the Field System log file directory is used as default, instead of the current directory.

4.2 Using control files

`logpl` control files can be used to set up the `Chan` menus to show pre-defined selections of data. These selections are also available in command-line mode with the **`select`** command. A selection of data is defined by a command, a parameter and an optional search string. Also, a description of the selection should be specified. The description will be on the plots created using the selection, and in the `Chan` menus.

The following is an example of a control file to set up the `Chan` menus to show selections for extraction of temperature, pressure, humidity and cable-length data:

```
* LOGPL.CTL - Control file for LOGPL
*
* 1. Command, the command logpl will search the log file for.
* 2. Parameter, the column of comma-separated data after the command.
* 3. Description, the menu label logpl will use for the command.
* 4. String, an optional search string for selecting data.
*
* NB! This file is space-separated. No field may contain spaces.
*
* 1:Command     2:Parameter   3:Description     4:String
* ------------------------------------------------------------------
*
  wx/           1             Temperature
  wx/           2             Pressure
  wx/           3             Humidity
  cable/        1             Cable-length
*
```

You may have several control files available on the disk. The control file used can be changed at startup, using the **`-control`** command-line flag, as described in section 4.1.3. Or it can be changed at run-time by selecting the `I/O setup` menu item in the `File` menu. If you are using the command-line mode, the control file used can be changed by invoking the **`control`** command.

5.0 Implementation

`logpl` is an event-driven program. When started, it does some initialization, the sleeps and waits for user input. When user input is received, `logpl` performs some tasks, and then sleeps again, waiting for another user event.

Data is not read from the log file until necessary. When the user invokes a plot or list command, `logpl` checks whether the data is already in memory or not. If not, the log file is opened and data is extracted. Memory is dynamically allocated for the read data. `logpl` then closes the log file again, so that the file is not open while not in use.

6.0 Hints for Further Development

6.1 On Tcl/Tk

`logpl` was developed using Tcl 7.3 and Tk 3.6. Tcl stands for "Tool Command Language", and Tk for "Toolkit", which is an extension to Tcl for writing graphical user interfaces. The language is normally interpreted, although compilers are now available on the World Wide Web. To start the interpreter, type **`wish -f`** ***sourcefile*** at the UNIX command prompt, where ***sourcefile*** is the Tcl/Tk script you want to execute. To be able to run the script without first starting the interpreter, the first line of the script should give the path to the interpreter, which is the case for `logpl`, where it reads:

```
#!/usr/local/bin/wish
```

since this is the location of the interpreter on the GSFC workstations.

Since the language is interpreted it is fairly easy to port it to another platform, i.e. Windows 3.1/95/NT. Theoretically, all you would have to do is to install the Windows version of the interpreter, and start the same script as you did in the UNIX environment. However, some things must be changed, i.e. file names and operating system calls. Please refer to the "Tcl/Tk for Windows" manual pages for further information on porting Tcl/Tk scripts to this platform.

One of the main features of the Tk extension to Tcl is the ability to bind commands to X (or Windows) events. An X event occurs, for example, when the user moves the mouse, clicks the mouse buttons, or presses a key on the keyboard. This makes it very convenient to write interactive programs like `logpl`. However, in `logpl`, very few application-specific X event bindings are used. Most Tk widgets created keep their default bindings as specified in the Tcl/Tk manual pages. The application-specific bindings made can be found by searching for `bind` commands in the source code.

6.2 On the source code of `logpl`

The source code is divided into sections, each of them containing related Tcl/Tk procedures. This is a short description of each of the sections.

6.2.1 The initialization section

This section is executed only once, at startup. In this section, some parts of the Tk widgets section are also executed at startup.

Since there is no `#DEFINE` statement in Tcl, `logpl` starts with setting a number of global variables. These are later made accessible in sub-procedures by use of the `global` command. Those variables that are not trivial have comments in the source code, like `Default directory for log files`, etc.

All variables that have default values are assigned those values in this section. If you want to change a default value, for example the default location of the control file, this is the place to look. The comments should be useful.

6.2.2 The time converting section

This section contains the procedures used for converting time in log file entry (*YYDDDHHMMSSSS*) format to a numerical value. The time converting procedures convert time into 100ths of seconds after 00:00:00, January 1, 1989. This also means that they will not work longer than until December 31, 2088, since the program will think it is back in the 1980s. During the period of 1989-2088, leap years are handled correctly, but not leap seconds.

6.2.3 The disk I/O section

This section contains procedures whose main task is to perform disk I/O, for example to extract data from the input log file, according to a specified selection criterion, and to read `logpl` control files, specifying such selections. The section also contains some child procedures to those, for example to set the `Chan` menus.

However, note that some procedures in the new-file section also perform disk I/O.

6.2.4 The command-line section

This section contains the procedures for decoding commands given at the command line. It works as an interface between the user and the rest of the program. First, the command-line is decoded. Then, a number of other procedures are called to execute the command, before control is returned to the user.

6.2.5 The plot section

This is the core of the program. All mathematics required for calculating screen coordinates for the plots is implemented in this section. Also, the procedures for marking points as “bad” are

here, as well as the scaling procedures. The parent plot procedure is named `Replot`, which calls a number of child procedures to do the actual work.

6.2.6 The selection section

This section contains the code for handling the selection window, where the user can change existing selections, and make new ones.

6.2.7 The new-file section

This section is directly related to the `New log file` dialog box. The first procedure is the procedure called when the menu item is selected. It creates the dialog box by calling a procedure in the Tk widgets section. The second procedure is called when the user has clicked the `OK` button. It takes care of opening the file selected, and extracting some data. There is also a procedure for checking if a filename is a valid log file or not.

6.2.8 The preferences and print sections

These sections are controlling dialog boxes. They also contain the code that is executed when the `OK` button of the corresponding dialog box is pressed.

6.2.9 The Tk widgets section

This section contains everything that has to do with creating the graphical user interface. Creating the interface is a two-step procedure. First, the widgets (entry fields, labels, menus etc.) are created and given unique names. Then they are displayed on the screen, using the Tk `pack` geometry manager.

Some of the code in the Tk widgets section is executed at startup. That is the code for creating the widgets for the plot area. The widgets used for dialog boxes are created when their menu item is selected, and destroyed when the `OK` or `Cancel` button is pressed. Note that the `Chan` menu item widgets are not created until a control file is read, which is done in the Disk I/O section.

Creating the menu widgets also includes specifying which commands should be executed when the menu items are selected. For example, the line:

```
.mbar.edit.menu add command –label “Reprocess” –command runError
```

tells the interpreter to run the procedure named `runError` when the user selects the `Reprocess` menu item. To understand in which order the source code is executed, it is often

a good start to search for the `add command` statement and following the procedures called. This can be a useful method for maintenance.

6.3 Windows 95 converting issues

From what I know, there are only three things that must be changed for `logpl` to work in a Windows 95 environment:

1. All file nams must be in DOS FAT (xxxxxxxx.xxx) format. However, the "\" character cannot be used as a directory separator in Tcl/Tk, since it is a reserved character. The "/" character is converted to "\" by the interpreter when performing I/O operations.

2. `Logpl` uses the UNIX `grep` command. There are several DOS-based `grep` commands freely available on the web, but the flags are usually somewhat different. You may have to send an extra flag to the DOS `grep` command to make the output look like the output of the UNIX `grep`.

3. Printing can not be done with the `lpr` command. Instead, maybe executing the following DOS command would work for printing to the parallel port: `type filename > LPT1:`.

4. There may be some trouble with the alignment of the widgets on the screen, since the Windows 95 interpreter for Tcl/Tk obviously uses some different default spacing criteria. However, this should not affect the functionality of the program.

7.0 Installation

In order to work, `logpl` requires the following installation procedure:

The Tcl/Tk interpreter `wish` must be installed. For information on how to do this, please read the Tcl/Tk information available on the Wold Wide Web. For example, Yahoo has a page on Tcl/Tk, from where software as well as documentation may be downloaded. Please note that `logpl` was developed using Tcl version 7.2 and Tk version 3.6. If you are installing the latest version of Tcl/Tk (as of July 24, 1997, Tcl 7.6 and Tk 4.2), `logpl` may require some conversion to work. As long as the major version number (Tcl 7.x and Tk 3.x) is the same, the version is backwards compatible. However, the upgrade to Tk version 4.x is not completely backwards compatible.

The first line of `logpl` must be changed to give the path to the interpreter. For example, on the GSFC workstations, it reads `#!/usr/local/bin/wish`.

The UNIX `grep` command must be available and in the search path.

7.1 Licensing

This is the license statement for the Tcl/Tk interpreter:

"This software is copyrighted by the Regents of the University of California, Sun Microsystems, Inc., and other parties. The following terms apply to all files associated with the software unless explicitly disclaimed in individual files.

"The authors hereby grant permission to use, copy, modify, distribute, and license this software and its documentation for any purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions. No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this software may be copyrighted by their authors and need not follow the licensing terms described here, provided that the new terms are clearly indicated on the first page of each file where they apply.

"In no event shall the authors or distributors be liable to any party for direct, indirect, special incidental, or consequential damages arising out of the use of this software, its documentation, or any derivatives thereof, even if the authors have been advised of the possibility of such damage.

"The authors and distributors specifically disclaim any warranties, including, but not limited to, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. This software is provided on an "as is" basis, and the authors and distributors have no obligation to provide maintenance, support, updates, enhancements, or modifications.

"Government use: If you are acquiring this software on behalf of the U.S. government, the Government shall have only "Restricted Rights" in the software and related documentation as defined in the Federal Acquisition Regulations (FARs) in Clause 52.227.19 (c) (2). If you are acquiring the software on behalf of the Department of Defense, the software shall be classified as "Commercial Computer Software" and the Government shall have only "Restricted Rights" as defined in the Clause 252.227-7013 (c) (1) of DFARs. Notwithstanding the foregoing, the authors grant the U.S. Government and others acting in its behalf permission to use and distribute the software in accordance with the terms specified in this license."