FERRET

USERS GUIDE

Version 3.1

NOAA/PMEL/TMAP

Steve Hankin
Martha Denham
September 1994

ABOUT THE COVER

The cover of this Users Guide was produced by FERRET. From the top down the plots are: "Toga-Tao SST," time series from the Tropical Pacific Tao array; "Levitus Climatological SST," level one of the annual Climatological Atlas of the World Oceans by Sydney Levitus of NOAA/NODC; "Perturbation Solution," a visualization of abstract functions by Dr. Ping Chang; "Vents Megaplume Thermal Structure," vertical temperature profiles of undersea thermal vents from the NOAA Vents program.

CONTENTS

Chapter 1: INTRODUCTION

1 OVERVIEW

FERRET is an interactive computer visualization and analysis environment designed to meet the needs of oceanographers and meteorologists analyzing large and complex gridded data sets. "Gridded data sets" in the FERRET environment may be multi-dimensional model outputs, gridded data products (e.g. climatologies), singly dimensioned arrays such as time series and profiles, and for certain classes of analysis, scattered n-tuples (optionally, grid-able using an objective analysis procedure provided in FERRET). FERRET accepts data from ASCII and binary files, and from two standardized, self-describing formats. FERRET's gridded variables can be one to four dimensions - usually (but not necessarily) longitude, latitude, depth, and time. The coordinates along each axis may be regularly or irregularly spaced.

FERRET was developed by the Thermal Modeling and Analysis Project at NOAA/PMEL in Seattle to analyze the outputs of its numerical ocean models and compare them with gridded, observational data. Model data sets are often multi-gigabyte in size with mixed 3 and 4-dimensional variables defined on staggered grids. FERRET offers the ability to define new variables interactively as mathematical expressions involving data set variables. Calculations may be applied over arbitrarily shaped regions.

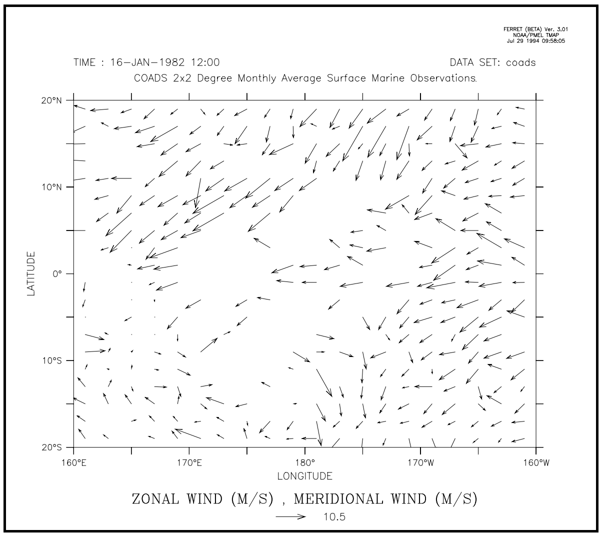
FERRET provides fully documented graphics, data listings, or extractions of data to files with a single command. Without leaving the FERRET environment, graphical output may be customized to produce publication-ready graphics. Animations are also available. FERRET is supported on a variety of Unix worstations with a version also available for Macintosh. A point-and-click interface is under development at the time of this writing. FERRET is available at no charge from anonymous FTP [node abyss.pmel.noaa.gov] or from Mosaic [URL http://www.pmel.noaa.gov/]. For users with access to NCSA Mosaic on the Internet, FERRET may be perused in hypertext.

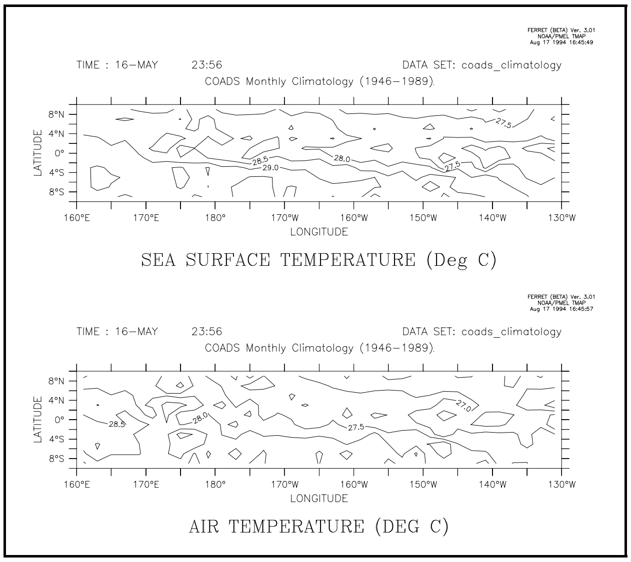
2 GETTING STARTED

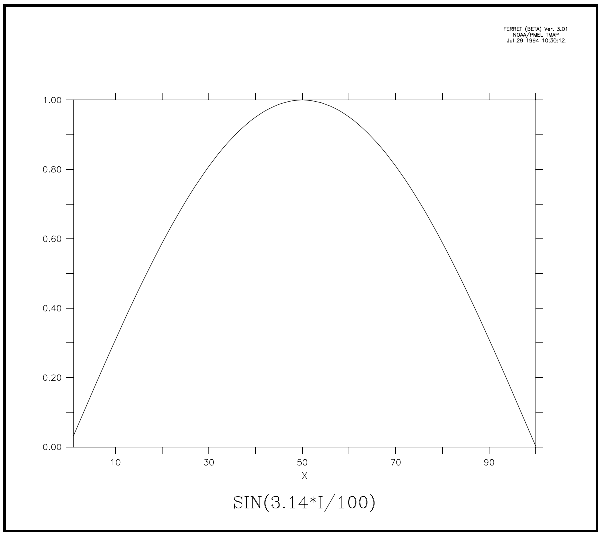
A quick way to get to know FERRET is to run the tutorial provided with the distribution.

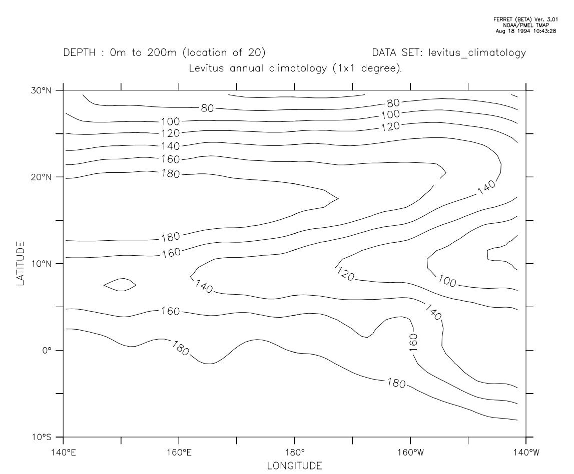
```
% ferret
yes? GO tutorial
```

If FERRET is not yet installed consult Chapter 7. (The tutorial is also available through Mosaic.) The tutorial demonstrates many of FERRET's features, showing the user both the commands given and FERRET's textual and graphical output. You may find the explanations, terms and examples in this manual easier to understand after running the tutorial.


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Jul 29 1994 09:58:05
TIME : 16-JAN-1982 12:00
DATA SET: coads
COADS 2x2 Degree Monthly Average Surface Marine Observations.
20°N
10°N
0°
10°S
20°S
LATITUDE
160°E
170°E
180°
170°W
160°W
LONGITUDE
ZONAL WIND (M/S) , MERIDIONAL WIND (M/S)
10.5


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 17 1994 16:45:49
TIME : 16-MAY 23:56
DATA SET: coads_climatology
COADS Monthly Climatology (1946-1989).
8°N
4°N
0°
4°S
8°S
LATITUDE
27.5
28.5
29.0
28.0
27.5
160°E
170°E
180°
170°W
160°W
150°W
140°W
130°W
LONGITUDE
SEA SURFACE TEMPERATURE (Deg C)
FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 17 1994 16:45:57
TIME : 16-MAY 23:56
DATA SET: coads_climatology
COADS Monthly Climatology (1946-1989).
8°N
4°N
0°
4°S
8°S
LATITUDE
28.5
28.0
27.5
27.0
160°E
170°E
180°
170°W
160°W
150°W
140°W
130°W
LONGITUDE
AIR TEMPERATURE (DEG C)


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Jul 29 1994 10:30:12
1.00
0.80
0.60
0.40
0.20
0.00
10
30
50
70
90
X
SIN(3.14*I/100)



DEPTH of 20 in TEMPERATURE (METERS)

3 COMMON COMMANDS

A quick reference to the most commonly used FERRET commands (typing "SHOW COMMANDS" at the FERRET prompt lists all commands):

Command	Description
SET DATA_SET	names the data set to be analyzed
SHOW DATA	produces a summary of variables in a data set
SHOW GRID	examines the coordinates of a grid
SET REGION	sets the region to be analyzed
LIST	produces a listing of data
PLOT	produces a plot
CONTOUR	produces a contour plot (/FILL for color-filled contour plot)
VECTOR	produces a vector arrow plot
SHADE	produces a shaded-area plot
STATISTICS	produces summary statistics about variables and expressions
LET	defines a new variable
SAVE	saves data in NetCDF format
GO	executes FERRET commands contained in a file

Information on all FERRET commands is available in Part II, Commands Reference, of this manual.

4 SYNTAX

Commands in program FERRET conform to the following template:

```
COMM [/Q1/Q2...] [SUBCOM[/S1/S2...]] [ARG1 ARG2 ...] [!comment]
```

where

COMM	is a command name	`yes? LIST`
Q1...	are qualifiers of the command	`yes? CONTOUR/SET_UP`
SUBCOM	is a subcommand name	`yes? SHOW MODE`
S1...	are qualifiers of the subcommand	`yes? SET LIST/APPEND`
ARG1...	are arguments of commands	`yes? CANCEL MODE INTERPOLATE`

notes...

- Items in square brackets are optional.
- One or more spaces or tabs must separate the command from the subcommand and from each of the arguments. Spaces and tabs are optional preceding qualifiers.
- Multiple commands, separated by semi-colons, can be given on the same line.
- Command names, subcommand names, and qualifiers require at most 4 characters.
  (e.g. `yes? CANCEL LIST/PRECISION` is equivalent to
  `yes? CANC LIST/PREC`)
- Some qualifiers take an argument following " = "
  (e.g. `yes? LIST/Y=10S:10N`).

5 GO FILES

GO files are files containing FERRET commands. They can be executed with the command "GO filename". Throughout this manual, these files are referred to as *GO scripts* or *journal files* (the file names end in *.jnl). There are two kinds of GO files provided with the distribution (differing in function, not form) - demos and tools.

5.1 Demonstration files

Demonstration GO files provide examples of various FERRET capabilities (the tutorial is such a script) . The demonstration GO files may be executed simply by typing the FERRET command

```
yes? GO demo_name
```

example: `yes? GO spirograph_demo`

Below is a list of the demo files provided (located in directory $FER_DIR/examples). The Unix command "Fgo demo" will list all GO scripts containing the string "demo".

Name	Description
tutorial	brief tour through FERRET capabilities
topographic_relief_demo	global topography
coads_demo	view of global climate using the Comprehensive Ocean-Atmosphere Data Set
levitus_demo	T-S relationships using Sydney Levitus' climatological Atlas of the World Oceans
fnoc_demo	Naval Fleet Numerical Oceanography Center data
vector_demo	vector plots
wire_frame_demo	3D wire frame representation
custom_contour_demo	customized contour plots
viewports_demo	output to viewports
multi_variable_demo	multiple variables with multiple dependent axes
objective_analysis_demo	interpolating scattered data to grids
polar_demo	polar stereographic projections
log_plot_demo	log plots using PPLUS in FERRET
depth_to_density_demo	contour with a user-defined variable as an axis
file_reading_demo	reading an ASCII file
regridding_demo	tutorial on regridding data
mathematics_demo	abstract function calculation
statistics_demo	probability distributions
spirograph_demo	for-fun plots from abstract functions
splash_demo	for-fun mathematical color shaded plots

5.2 GO tools

GO tools are scripts which contain FERRET commands and perform dataset-independent tasks. For example, "GO land" overlays the outline of the continents on your plot. (Note: In order for FERRET to locate the GO scripts, the environment variable FER_GO must be properly defined. See Chapter 7 "Computing Environment" for guidance.)

The Unix command Fgo has been provided to assist with locating tools within the Unix directory hierarchy. For example,

`% Fgo grid`	displays all tools with the substring "grid" in their names
`% Fgo '*'`	displays all GO tools and demonstrations

Below is a table of the tools provided with your FERRET installation. Some tools accept optional arguments to control details. Use `Fgo -more script_name` for details on a script.

Tool name	Description
OVERLAYS	
land [color]	overlays continental boundaries (color controls)
bold_land	overlays darker continental boundaries
fland [res color]	overlays filled continent masses (color and resolution controls)
graticule	sets the plot axis style to use a graticule (rather than tics)
tics	resets the plot style to use axis tics (rather than a graticule)
gridxy	overlays a "graticule" labelling the I,J subscripts
gridxz	overlays a "graticule" labelling the I,K subscripts
gridxt	overlays a "graticule" labelling the I,L subscripts
gridyz	overlays a "graticule" labelling the J,K subscripts
gridyt	overlays a "graticule" labelling the J,L subscripts
gridzt	overlays a "graticule" labelling the K,L subscripts
box	draws a box at the specified location on the plot
ellipse	draws an ellipse at the specified location on the plot
MATHEMATICAL	
frequency_histogram	makes a frequency distribution plot (histogram) of data
ts_frequency	creates a 2-variable histogram (typically an oceanographer's TS density diagram)
polar	defines R and THETA from X and Y to perform (limited) polar plots
regressx	defines variables for linear regression along X axis
regressy	defines variables for linear regression along Y axis
regressz	defines variables for linear regression along Z axis
regresst	defines variables for linear regression along T axis
unit_square	sets unit square as default for abstract variables

variance	defines variables to compute variances and covariances
var_n	refines TVARIANCE with corrected n/n+1 factors
dynamic_height	defines FERRET variables for dynamic height calculations

SAMPLE DISPLAYS

line_samples	draws specimens of the available line styles
line_thickness	draws examples of pen color/thickness styles in PPLUS
fill_samples	draws specimens of the available fill styles
show_symbols	draws specimens of the default symbols
show_88_syms	draws specimens of all 88 PPLUS symbols

GRAPHICS

bar_chart	makes a color-filled bar chart from a line of data
bar_chart2	makes a bar chart using hollow rectangles
centered_vectors	makes a vector plot with coords at vector midpoints
scattered_vectors	makes a vector plot from an ASCII file: x,y,u,v
stick_vectors	makes a stick vector plot of a line of U,V values

PLOT APPEARANCE

magnify [factor]	increases the data plotting area (area inside the axes)
unmagnify	restores the plot origin and axis lengths to default values
black	sets video background to black, foreground to white
white	sets video background to white, foreground to black
bold	sets up PLOT+ and FERRET to produce bolder-looking plots
unbold	resets plot environment to normal after "GO bold"
unlabel [label #]	removes a specified (numbered) PPLUS movable label
box_plot	produces a plot with "bare" axes (no tics, no labels)

COLOR

try_palette [pal]	displays palette appearance for various numbers of color levels
try_centered_palette	displays centered palette appearance for various numbers of levels
exact_colors	sets up FERRET and PPLUS to modify individual colors in a color palette
squeeze_colors	modifies a color palette by squeezing and stretching the color scale

MULTIPLE X AND Y AXES (run demo: `yes? GO multi_variable_plots`)

left_axis_plot	plots a single variable preparing for a 2nd axis on the right
right_axis_plot	overlays a plot of a single variable using an axis on the right
multi_xaxis_plot1	draws a plot formatted for later overlays using multiple X axes
multi_xaxis_overlay	overlays a variable with a distinct X axis
multi_yaxis_plot1	draws a plot formatted for later overlays using multiple Y axes
multi_yaxis_overlay	overlays a variable with a distinct Y axis

VIEWPORTS

landscape	set up for 8.5 x 11 page: landscape orientation

landscape1x2	8.5 x 11 page - two viewports stacked
landscape2x1	8.5 x 11 page - two viewports side-by-side
landscape2x2	8.5 x 11 page - 4 viewports/reduced labels
landscape3x2	8.5 x 11 page - 6 viewports/reduced labels
portrait	sets up for 8.5 x 11 page: portrait orientation
portrait1x2	sets up for two viewports on an 8.5 x 11 page
portrait1x3	sets up for three viewports on an 8.5 x 11 page
portrait1x4	sets up for four viewports on an 8.5 x 11 page

POLAR STEREOGRAPHIC PROJECTIONS (run demo: `yes? GO polar_demo`)

convert_to_polar_2d	extracts (sample) data for a 2D polar plot
polar_2d	produces a 2D polar stereographic plot
polar_fland	overlays solid-filled continents on a polar plot
polar_grid	overlays "quick" radial longitude and curved latitude lines
polar_grid_fancy	overlays "fancy" radial longitude and curved latitude lines
polar_land	overlays the continental outlines on a polar stereographic plot
polar_map_inv_eqns	defines the equations used for polar projections
polar_vector	produces a 2D polar stereographic vector plot
polar_vs	performs a polar PLOT/VS of 2 variables: lat long
projected_map_grid	defines the map grid for a projected plot

TESTS

test	tests proper functioning of FER_GO
ptest	produces a quick test plot
squares	creates a filled-area test plot

5.3 Writing GO Tools

A GO tool ("GO script", "journal file", ...) is simply a sequence of FERRET commands stored in a file and executed with the GO command. Writing a simple GO tool requires nothing more than typing normal commands into a file.

To write a robust GO tool that may be shared, however, certain guidelines should be followed:

1) the GO tool should be well documented
2) the GO tool should leave the FERRET context unmodified
3) the GO tool may need to run "silently"
4) the GO tool may need to accept arguments (parameters)

5.3.1 Documenting GO tools

Documentation consists primarily of well-chosen comment lines (lines beginning with an exclamation mark). In addition, a line of this form should be included:

```
! Description:  [one-line summary of your GO tool]
```

This line is displayed by the Fgo tool.

5.3.2 Preserving the FERRET context in GO tools

Often a complex GO tool requires setting data sets, modifying the current region, etc. But to a user executing this tool its behavior may seem erratic if the user's previous context is modified by running the tool. A tool can restore the previous state of FERRET by these means:

region: Save the current default region with the command DEFINE REGION/DEFAULT save. Restore it at the end of your GO tool with SET REGION save.

data set: Save the current default data set with SET DATA/SAVE. Restore it at the end of your GO tool with SET DATA/RESTORE.

grid: Save the current default grid set with SET GRID/SAVE. Restore it at the end of your GO tool with SET GRID/RESTORE.

modes: If you modify a mode inside your GO tool by issuing a SET MODE or a CANCEL MODE command the original state of that mode can be restored using SET MODE/LAST

5.3.3 Silent GO tools

If a user has set mode "verify" then by default every line of your GO tool, including comment lines, will be displayed at the screen as FERRET processes it. To make your GO tool run silently include the command CANCEL MODE VERIFY at the beginning of the GO tool and SET MODE/LAST VERIFY at the end. If the backslash character "\" is found at the beginning of any line that single line will not be displayed *regardless* of the state of MODE VERIFY. Thus the command "\CANCEL MODE VERIFY" is often the first line of a GO tool. Note also that the command LET/SILENT is useful in GO tools which need to define variables.

5.3.4 Arguments to GO tools

Arguments (parameters) may be passed to GO tools on the command line. For example,

```
yes? GO land red
```

passes the string "red" into the GO file named land.jnl. Inside the GO tool the argument string "red" is substituted for the string "$1" wherever it occurs. The "1" signifies that this is the first argument - similar logic can be applied to $1,... $9 or $0 where $0 is replaced by the name of the GO tool, itself. Similarly "$*" is replaced by all the arguments, 1-9 as a single string.

As FERRET performs the substitution of $1 (or other) arguments it offers a number of string processing and error processing options. For example, *without* these options, if a user failed to supply an argument to "GO land" then FERRET would not know what to substitute for $1 and it would have to issue an error message. A default value can be supplied by the GO tool writer using the syntax

```
$1%string%
```

for example,

```
$1%black%
```

inside land.jnl would default to "black" if no color were specified. Note that in the example percent signs were used to delimit the default string but any of the characters ! # $ % or & also work as delimiters.

In another case it might not be appropriate to supply a default string but instead it would be desirable to issue an instructional error message. The "<" character indicates an error message text -

```
$1"<you must supply an argument to this GO tool"
```

In still other cases there are a range of acceptable arguments but all other arguments are illegal. The allowable arguments can be specified following "|" (vertical bar) characters as in this example:

```
$1"|black|red|<You must specify black or red"
```

or a default of "black" could be specified together with the options as

```
$1"black|black|red|"
```

In the interest of "friendliness" a GO file may want to allow the user to specify a string other than the string actually needed by the GO tool. For example, a red plot line is actually obtained by the PLOT command qualifier /LINE=2 - the string "red" never appears in this command. To allow a user to specify "red" and yet have the string "2" substituted, FERRET has provided the replacement arrow ">". Thus

```
$1"1|red>2|"
```

specifies a default string of "1" if no argument is given but substitutes "2" if "red" is supplied. In a typical GO tool line, defaults, options, substitutions, and an error message are combined like this:

```
PLOT/LINE=$1"1|red>2|green>3|blue>4|<must be red, green, or blue"
```

Note that the error message will be issued only if some color other than "red", "green", or "blue" is specified; if no argument is specified then "1" is substituted.

A "*" (asterisk) can be used to designate that any text whatsoever is acceptable as an option.

```
PLOT/LINE=$1"1|red>2|green>3|blue>4|*>7"
```

would never generate an error and would use line style 7 (thick black) if an unrecognized argument string such as "purple" were given.

A final style note to keep in mind when writing GO tools that use arguments: providing error message feedback and appropriate documentation for the user is essential. In complex GO tools, all arguments should be checked at the beginning of the GO tool using the no-op command (has no effect) "QUERY/IGNORE". Thus the GO tool land.jnl might contain these lines at the beginning:

```
! check the argument
QUERY/IGNORE $1"1|red|green|blue|<must be red, green, or blue"
```

Once argument errors have been trapped and reported, the lengthy error text would not be needed again in the GO tool.

GO tools that use arguments should also be carefully documented. There are numerous examples provided with FERRET; try, for example, the Unix commands

```
% Fgo -more fland.jnl
% Fgo -more stick_vectors
```

or

```
% Fgo -more squeeze_colors
```

6 SAMPLE DATA SETS

A number of demonstration data sets are included with this distribution. Several of these data sets are used by the demonstration "GO" files, above. The data sets should be accessible simply by typing the FERRET command

```
yes? SET DATA data_set_name
```

for example,

```
yes? SET DATA coads_climatology
```

Demonstration data are located in directory $FER_DSETS/data. Grids are located in directory $FER_DSETS/grids. For GT and TS format datasets, descriptor files are located in directory $FER_DSETS/descr.

Data set	Description
etopo120	relief of the earth's surface at 120-minute resolution
etopo60	relief of the earth's surface at 60-minute resolution
levitus_climatology	subset of the Climatological Atlas of the World Oceans by Sydney Levitus
coads_climatology	12-month climatology derived from 1946-1989 of the

	Comprehensive Ocean/Atmosphere Data Set
monthly_navy_winds	Monthly-averaged Naval Fleet Numerical Oceanography Center global marine winds (1982-1990)
esku_heat_budget	Esbensen-Kushnir 4x5 degree monthly climatology of the global ocean heat budget (25 variables)

7 UNIX TOOLS

A number of tools are provided with FERRET to assist with Unix-level activities: on-line help, converting data to FERRET's formats, locating files, etc. They are located in the FERRET installation area - typically $FER_DIR/bin. See Chapter 7 "Setting Up an Account" if the tools are not available on-line. They are described below.

Faddpath Usage: Faddpath new_path

Faddpath will add a new path name to the default lists of directories that FERRET searches a) in response to the SET DATA command; b) when looking for grid definition files; c) when looking for data files.

Fapropos Usage: Fapropos string (i.e. `% Fapropos regridding`)

Fapropos searches the FERRET Users' Guide for all occurrences of the given word or string. The string is not case sensitive. If the string contains multiple words it must be enclosed in quotation marks. Fapropos will list all lines of the Users' Guide that contain the word or string and report their line numbers. The line numbers may be used with Fhelp to enter the Users' Guide at the desired location.

Fdata Usage: Fdata data_file_substring

Searches the list of directories contained in the environment variable FER_DATA to find the data files whose names contain the indicated substring. For example,

```
% Fdata coads
```

locates the data files containing "coads" in their names. (Use this command to locate NetCDF data sets by giving the string "cdf".)

Fdescr Usage: Fdescr des_name_substring

Searches the list of directories contained in the environment variable FER_DESCR to find the descriptor files whose names contain the indicated substring. For example,

```
% Fdescr coads
```

locates the descriptor files containing "coads" in their names. ("Fdescr .des" will list all accessible descriptors.)

Fenv Usage: Fenv

Prints the values of environment variables used by FERRET

Fgo Usage: Fgo name_substring

Searches the list of directories contained in the environment variable FER_GO to find the GO command files whose names contain the indicated substring. For example,

```
% Fgo grid
```

locates the FERRET tools that contain "grid".

Fgrids Usage: Fgrids gridfile_substring

Searches the list of directories contained in the environment variable FER_GRIDS to find the grid definition files whose names contain the indicated substring. For example,

```
% Fgrids fnoc
```

locates the grid definition files containing "fnoc" in their names. ("Fgrids .grd" will list all accessible grid files.)

Fhelp Usage: Fhelp line_number or Fhelp string

Fhelp enters the FERRET Users' Guide beginning at the indicated line number or at the first occurrence of the given string. The string, if used, is not case sensitive. The Unix "more" command is used to access the Users' Guide. The most commonly used "more" commands are documented under Ftoc.

Examples:

```
% Fhelp  1136
% Fhelp  "modulo axis"
```

Fman Usage: Fman

(Not yet implemented.) Enters the FERRET User's Guide as on-line, formatted hypertext.

Fpalette Usage: Fpalette name_substring

Searches the list of directories contained in the environment variable FER_PALETTE to find the palette files whose names contain the indicated substring. For example,

```
% Fpalette blue
```

locates the palette files containing "blue" in their names.

Fpurge Usage: Fpurge filename_template

Fpurge is a support routine to manage multiple versions of files created by FERRET - particularly journal files and graphic metafiles. Fpurge will remove all versions of a file except the current version. For example, `Fpurge ferret.jnl' will eliminate all past versions of ferret.jnl in the current directory.

Fread1 Usage: Fread1

Interactive program to examine individual records of the FERRET-formatted GT and TS files. The program allows the user to specify by record number the record to be examined. It is a valuable tool for debugging new codes that produce TMAP-formatted files. It is not necessary to use this routine when the data conversion routines provided with the FERRET distribution are used. Fread1 prompts the user interactively for required inputs.

Fslicer Usage: Fslicer
Interactive program to create new data sets that are subsets of existing data sets (reduced number of variables and/or reduced limits in 4 dimensions). The user simply creates descriptor files (TS or GT format) that describe the desired output data sets. The interactive Fslicer program prompts the user for the required input and output descriptor names. Several output "slices" can be created simultaneously from a single input data set - performance is optimized for one-to-many slicing.

Fsort Usage: Fsort filename_template
Fsort is a support routine for sorting file versions. Fsort reorders the incorrect ordering of emacs-style version numbers assigned by the Unix "ls" utility. e.g., when sorting, ls will place filename.~19~ before filename.~2~. "Fsort filename*" will take care of this problem. Fsort may be used in Unix pipes.

Ftoc Usage: Ftoc
Ftoc enters the table of contents of the FERRET Users' Guide using the Unix "more" command. Within "more" the following are the most commonly used commands:

? - interactive help for "more"
q - exit (quit)
space - advance to next screen
return - advance to next line
b - back one screen
/string - locate the next occurrence of "string" (Note: the string is case sensitive)

8 HELP

8.1 Unix on-line help

On Unix systems interactive FERRET help is available from the command line. If multiple windows are not available on your system the ^Z key can be used to suspend the current FERRET session and access the help; the Unix "fg" command resumes the suspended session.

Several Unix commands provide assistance with rapidly locating information in the FERRET Users' Guide. The entire FERRET Users' Guide is available on-line as document $FER_DIR/doc/ferret_users_guide.txt. A printable version is also available in PostScript: $FER_DIR/doc/ferret_users_guide.ps.

These commands are available to access the FERRET Users' Guide:
Ftoc - browse the table of contents of the Users' Guide

Fapropos - locate words or character strings in the Users' Guide
Fhelp - enter and browse the Users' Guide
Fman - enter and browse the User's Guide as formatted hypertext (not yet implemented)

Normally Ftoc or Fapropos is used first to locate the desired information in the Users' Guide. Then Fhelp is used to enter the Users' Guide at the selected location.

8.2 Examples and demonstrations

As discussed earlier in this chapter (Getting Started, GO files), the demonstrations which come with the FERRET distribution are a source of help. See Chapter 1 section "Demonstration files" for a list of demonstrations, or look in $FER_DIR/examples; you may find something that addresses your problem.

8.3 Help from within FERRET

Typing "help" while running FERRET will give you information on using the Unix tool Fhelp to access the Users' Guide.

The FERRET command SHOW COMMANDS will list all FERRET commands; SHOW COMMAND "command" will display all qualifiers for the specified command.

Chapter 2: DATA SETS

1 OVERVIEW

FERRET accepts input data from both ASCII and binary files and recognizes two standardized, self-describing data formats - NetCDF, and TMAP. As of FERRET version 2.30, Network Common Data Format (NetCDF) is the suggested method of data storage.

SET DATA_SET or just SET DATA specifies a data set for access. ASCII and binary files can be read using SET DATA/EZ (also known as "FILE"). To unambiguously specify the format of a data set, include the extension .cdf or .des in its name, or use the qualifier /FORMAT=CDF.

To examine what each data set consists of (variables, grids etc.) after specifying them with SET DATA, use SHOW DATA. This command displays the variables in the data set and over what geographical and time ranges they are defined.

Here is an example of FERRET's output:

```
   yes? SET DATA coads_climatology
   yes? SHOW DATA
      currently SET data sets:
     1> /home/e1/tmap/fer_dsets/descr/coads_climatology.des  (default)
 name     title                             I         J         K         L
 SST      SEA SURFACE TEMPERATURE          1:180     1:90      1:1       1:12
 AIRT     AIR TEMPERATURE                  1:180     1:90      1:1       1:12
 SPEH     SPECIFIC HUMIDITY                1:180     1:90      1:1       1:12
 WSPD     WIND SPEED                       1:180     1:90      1:1       1:12
 UWND     ZONAL WIND                       1:180     1:90      1:1       1:12
 VWND     MERIDIONAL WIND                  1:180     1:90      1:1       1:12
 SLP      SEA LEVEL PRESSURE               1:180     1:90      1:1       1:12
```

If multiple data sets have been requested in a single FERRET session, the last requested will be the default data set. To specify other data sets, use the name of the data set or the number of the set as given by the SHOW DATA statement. For example:

```
   yes? LIST/D=2  temp
```

will list the data for the variable "temp" in data set number 2 as displayed by SHOW DATA/BRIEF, while

```
   yes? LIST temp[D=levitus_climatology] - temp[D=coads_climatology]
```

will list the differences between the variable "temp" in data set "levitus_climatology" and data set "coads_climatology".

2 NETCDF DATA

The Network Common Data Format (NetCDF) is an interface to a library of data access routines for storing and retrieving scientific data. NetCDF allows the creation of data sets which are self-describing and platform-independent. NetCDF was created under contract with the Division of Atmospheric Sciences of the National Scientific Foundation and is available from the Unidata Program Center in Boulder, Colorado (unidata.ucar.edu).

See Chapter 8 "Converting Data to NetCDF" for a complete description of how to create NetCDF data sets or how to convert existing data sets into NetCDF.

To output a variable in NetCDF, simply use:

```
yes?  LIST/FORMAT=CDF variable_name
```

LIST/FORMAT=CDF (alias SAVE) can also be used with abstract variables:

```
yes? SAVE/FILE=example.cdf/I=1:100 sin(I/100)
```

This will create a file named example.cdf.

The current region and data sets determine the variable names in the saved file and the range over which they are saved. Saved data can then be accessed as follows:

```
yes? USE example
```

If a filename is not specified, FERRET will generate one. (See command SET LIST/FILE in the Commands Reference section of this manual.) An example of converting TMAP-formatted data to NetCDF goes as follows:

```
yes? SET DATA coads_climatology
yes? SAVE/L=1 sst,airt,uwnd,vwnd
```

These commands will save sst, airt, uwnd, and vwnd at the first time step over their entire regions to a NetCDF file named by FERRET.

One advantage to using NetCDF is that users on a different system (i.e. VMS instead of Unix) with different software (i.e. with an analysis tool other than FERRET) can share data easily without substantial conversion work. NetCDF files are self-describing; with a simple command the size, shape and description of all variables, grids and axes can be seen.

3 TMAP-FORMATTED DATA

As of FERRET version 2.30, NetCDF is the suggested format for data storage (see Chapter 8 "Converting to NetCDF"). This section describing TMAP information is included only for users who already work with data in TMAP format.

To access TMAP-formatted data sets use

```
SET DATA_SET TMAP_set1, TMAP_set2, ...
```

TMAP_setn must be the name of a descriptor file for a data set that is in TMAP "GT" (grids-at-timesteps) or "TS" (time series) format. ("FERRET" format and "TMAP" format are synonyms.)

If the directory portion of the filename is omitted the environment variable FER_DESCR will be used to provide a list of directories to search. The order of directories in FER_DESCR determines the order of directory searches. If the extension is omitted a default of ".des" will be assumed (if the filename has more than one period, the extension must be given explicitly).

Descriptors

For every TMAP-formatted data set there is a descriptor file containing summary information about the contents of the data set. This includes variable names, units, grids, and coordinates. When the command SET DATA_SET is given to FERRET pointing to a GT-formatted or TS-formatted data set, it is the name of the descriptor file that must be specified. Both "GT" and "TS" formats are supported by FERRET version 2.20 and above.

Special note on TMAP internal data sets:
If the data set specified contains Philander/Seigel/MOM/Pacific model output, special information may be provided to FERRET to compute diagnostic variables. Normally, this information is pulled from the TMAP model runs data base ($FER_MODEL_RUNS on Unix systems), but special cases may be handled by specifying the descriptor NAMELIST parameter D_ADD_PARM. As of FERRET 2.20 D_ADD_PARM may contain:

```
'HEAT FLUX = DOUBLE RAMP' (or "SINGLE RAMP" or "PHILANDER/SIEGEL")
'MINIMUM WIND = 488'      (or 385 or any other value)
'AIR-SEA DELTA T = 1.0'   (or any other value or "LEVITUS" or "CAC")
'Am_FACTOR = 2.0          (or any other value)
```

or for air temperature climatology data sets, only:

```
'205_AIRT = LEV'          (or "CAC","ALV"...)
'NTS30 WIND STRESS'
```

4 BINARY DATA

Binary data words must all begin on 4-byte boundaries to be accessible by FERRET. To understand how to access binary data files with FERRET it is necessary to understand something of the structure of binary files. Binary files are of two general types - files containing record length information and files without imbedded record length information. Within each of these categories the records may in actuality be of uniform length or of variable length.

4.1 FORTRAN-structured binary files

Files containing record length information are created by FORTRAN programs using the ACCESS="SEQUENTIAL" (the FORTRAN default) mode of file creation and also by FERRET using LIST/FORMAT=unf. Suppose "rrrr" represents 4 bytes of record length information and "dddd" represents a 4-byte data value. Then FORTRAN-structured files are organized in one of the following two ways:

4.1.1 Records of uniform length

A FORTRAN-structured file with records of uniform length (3 single-precision floating point data values per record in this figure) look as follows:

```
rrrr dddd dddd dddd rrrr
rrrr dddd dddd dddd rrrr
...
```

FORTRAN code that creates a data file of this type might look something like this (sequential access is the default and need not be specified in the OPEN statement):

```
REAL VARI(10), VAR2(10), VAR3(10)
...
OPEN(UNIT=20,FORMAT="UNFORMATTED",ACCESS="SEQUENTIAL",FILE="MYFILE.DAT")
...
DO 10 I=1,10
      WRITE (20) VAR1(I), VAR2(I), VAR3(I)
10 CONTINUE
....
```

To access data from this file, use

```
yes? SET DATA/EZ/FORMAT=UNF/VAR=var1,var2,var3/COL=3  myfile.dat    or,
yes? FILE/FORMAT=UNF/VAR=var1,var2,var3/COLUMNS=3  myfile.dat
```

This is very similar to accessing ASCII data with the addition of the /FORMAT=unf qualifier. The /COLUMNS= qualifier tells FERRET the number of data values per record. Although optional in the above example, this qualifier is required if the number of data values per record is greater than the number of variables being read (examples follow in section "ASCII Data").

4.1.2 Records of non-uniform length

A FORTRAN-structured file with variable-length records may look as follows:

```
rrrr dddd dddd rrrr
rrrr dddd rrrr
rrrr dddd dddd dddd dddd rrrr
etc.
```

With care, it is possible to read a data file containing variable-length records which was created using the simplest unformatted FORTRAN OPEN statement and a single WRITE statement for each variable. Use /FORMAT=stream to read such files. Note that sequential access is the FORTRAN default and does not need to be specified in the OPEN statement:

```
REAL VAR1(1000), VAR2(500)
...
OPEN (UNIT=20, FORMAT="UNFORMATTED", FILE="MYFILE.DAT")
...
WRITE (20) VAR1
WRITE (20) VAR2
....
```

Use the qualifier /SKIP to skip past the record length information (/SKIP arguments are in units of words), and define a grid which will not read past the data values. The /COLUMNS= qualifier can be used when reading multiple variables to specify the number of words separating the start of each variable:

```
yes? DEFINE AXIS/X=1:500:1  xaxis
yes? DEFINE GRID/X=XAXIS  mygrid
yes? FILE/FORMAT=stream/SKIP=1003/GRID=mygrid/VAR=var2  myfile.dat
```

The argument 1003 is the sum of the 1000 data words in record 1, plus 2 words of record length information surrounding the data values in record 1 (variable var1), plus 1 word of record information preceding the data in record 2.

4.2 Unstructured binary files

Files without imbedded record length information are created by FORTRAN programs using ACCESS="DIRECT" in OPEN statements and by C programs. Suppose "dddd" represents a 4-byte data value. Then unstructured binary files are simply:

```
            dddd dddd dddd dddd dddd dddd ...
```

Such files are also referred to as "direct access binary" files. The structure of the records is *implied* by the program accessing the data. FORTRAN code which generates a direct access binary file might look as follows:

```
REAL*4 MYVAR(10,5)
...
C Use RECL=40 for machines that specify in bytes

OPEN(UNIT=20, FILE="myfile.dat", ACCESS="DIRECT", RECL=10)
...
DO 100 j = 1, 5
100   WRITE (20,REC=j) (MYVAR(i,j),i=1,10)
....
```

Use the following FERRET commands to read variable "myvar" from this file:

```
yes? DEFINE AXIS/X=1:10:1 x10
yes? DEFINE AXIS/Y=1:5:1 y5
yes? DEFINE GRID/X=x10/Y=y5 g10x5
yes? FILE/VAR=MYVAR/GRID=g10x5/FORMAT=stream  myfile.dat
```

5 ASCII DATA

To access ASCII data files sets use

```
yes? SET DATA/EZ   ASCII_file_name
```

or equivalently

```
yes? FILE   ASCII_file_name
```

The following are qualifiers to SET DATA/EZ or FILE:

Qualifier	Description
/VARIABLES	names the variables in the file
/TITLE	associates a title with the data set
/GRID	indicates multi-dimensional data and units
/COLUMNS	tells how many data values are in each record
/FORMAT	specifies the format of the file
/SKIP	skips initial records of the file
/ORDER	specifies order of axes (which varies fastest)

Use command SET VARIABLE to individually customize the variables.

5.1 Reading ASCII files

Below are several examples of reading ASCII data properly. (Uniform record length, FORTRAN-structured binary data are read similarly with the addition of the qualifier /FORMAT= "unf". See Chapter 2 section "Binary Data" for other binary types). First, we look briefly at the relationship between FERRET and standard matrix notation.

Linear algebra uses established conventions in matrix notation. In a matrix A(i,j), the first index denotes a (horizontal) row and the second denotes a (vertical) column.

A11 A12 A13 ... A1n
A21 A22 A23 ... A2n Matrix A(i,j)
...

Am1 Am2 Am3 ... Amn

X-Y graphs follow established conventions as well, which are that X is the horizontal axis (and in a geographical context, the longitude axis) and increases to the right, and Y is the vertical axis (latitude) and increases upward (FERRET provides the /DEPTH qualifier to explicitly designate axes where the vertical axis convention is reversed)..

In FERRET, the first index of a matrix, i, is associated with the first index of an (x,y) pair - x. Likewise, j corresponds to y. Element Am2, for example, corresponds graphically to x=m and y=2.

By default, FERRET stores data in the same manner as FORTRAN - the first index varies fastest. Use the qualifier /ORDER to alter this behavior. The following examples demonstrate how FERRET handles matrices.

Example 1 - 1 variable, 1 dimension

1a) Consider a data set containing the height of a plant at regular time intervals, listed in a single column:

```
2.3
3.1
4.5
5.6
. . .
```

To access, name, and plot this variable properly, use the commands

```
yes? FILE/VAR=height plant.dat
yes? PLOT height
```

1b) Now consider the same data, except listed in four columns:

```
2.3   3.1   4.5   5.6
5.7   5.9   6.1   7.2
. . .
```

Because there are more values per record (4) than variables (1), use:

```
yes? FILE/VAR=height/COLUMNS=4 plant4.dat
yes? PLOT height
```

Example 2 - 2 variables, 1 dimension

2a) Consider a data set containing the height of a plant and the amount of water given to the plant, measured at regular time intervals:

```
2.3 20.4
3.1 31.2
4.5 15.7
5.6 17.3
. . .
```

To read and plot this data use

```
yes? FILE/VAR="height,water" plant_wat.dat
yes? PLOT height,water
```

2b) The number of columns need be specified only if the number of columns exceeds the number of variables. If the data are in six columns

```
2.3 20.4   3.1 31.2   4.5 15.7
5.6 17.3 ...
```

use

```
yes? FILE/VAR="height,water"/COLUMNS=6 plant_wat6.dat
yes? PLOT height,water
```

Example 3 - 1 variable, 2 dimensions

3a) Consider a different situation: a greenhouse with three rows of four plants and a file with a single column of data representing the height of each plant at a single time (successive values represent plants in a row of the greenhouse):

```
3.1
2.6
5.4
4.6
3.5
6.1
. . .
```

If we want to produce a contour plot of height as a function of position in the greenhouse, axes will have to be defined:

```
yes? DEFINE AXIS/X=1:4:1 xplants
yes? DEFINE AXIS/Y=1:3:1 yplants
yes? DEFINE GRID/X=xplants/Y=yplants gplants
yes? FILE/VAR=height/GRID=gplants greenhouse_plants.dat
yes? CONTOUR height
```

When reading data the first index, x, varies fastest. Schematically, the data will be assigned as follows:

```
        x=1         x=2          x=3         x=4
y=1     3.1         2.6          5.4         4.6
y=2     3.5         6.1 . . .
y=3    . . .
```

3b) If the file in the above example has, instead, 4 values per record:

```
3.1   2.6   5.4   4.6
3.5   6.1  . . .
```

then add /COLUMNS=4 to the FILE command:

```
yes? FILE/VAR=height/COLUMNS=4/GRID=gplants greenhouse_plants.dat
```

Example 4 - 2 variables, 2 dimensions

Like Example 3, consider a greenhouse with three rows of four plants each and a data set with the height of each plant and the length of its longest leaf:

```
3.1     0.54
2.6     0.37
5.4     0.66
4.6     0.71
3.5     0.14
```

```
6.1     0.95
.        .
.        .
```

Again, axes and a grid must be defined:

```
yes? DEFINE AXIS/X=1:4:1 xht_leaf
yes? DEFINE AXIS/Y=1:3:1 Yht_leaf
yes? DEFINE GRID/X=xht_leaf/Y=yht_leaf ght_leaf
yes? FILE/VAR="height,leaf"/GRID=ght_leaf greenhouse_ht_lf.dat
yes? SHADE height
yes? CONTOUR/OVER leaf
```

The above commands create a color-shaded plot of height in the greenhouse, and overlay a contour plot of leaf length. Schematically, the data will be assigned as follows:

```
         x=1          x=2          x=3          x=4
       ht ,  lf     ht ,  lf
y=1    3.1, 0.54    2.6, 0.37    5.4, 0.66    4.6, 0.71
y=2    3.5, 0.14    6.1, 0.95 . . .
y=3    . . .
```

Example 5 - 2 variables, 3 dimensions (time series)

Consider the same greenhouse with height and leaf length data taken at twelve different times. The following commands will create a three dimensional grid and a plot of the height and leaf length versus time for a specific plant.

```
yes? DEFINE AXIS/X=1:4:1 xplnt_tm
yes? DEFINE AXIS/Y=1:3:1 yplnt_tm
yes? DEFINE AXIS/T=1:12:1 tplnt_tm
yes? DEFINE GRID/X=xplnt_tm/Y=yplnt_tm/T=tplnt_tm gplant2
yes? FILE/VAR="height,leaf"/GRID=gplant2 green_time.dat
yes? PLOT/X=3/Y=2 height, leaf
```

Example 6 - 1 variable, 3 dimensions, permuted order (vertical profile)

Consider a collection of oceanographic measurements made to a depth of 1000 meters. Suppose that the data file contains only a single variable, salt. Each record contains a vertical profile (11 values) of a particular x,y (long,lat) position. Supposing that successive records are successive longistudes, the data file would look as follows (assume the equivalencies are not in the file):

```
               z=0   z=10   z=20 . . .
x=30W,y=5S  35.89 35.90 35.93 35.97 36.02 36.05 35.96 35.40 35.13 34.89 34.72
x=29W,y=5S  35.89 35.91 35.94 35.97 36.01 36.04 35.94 35.39 35.13 34.90 34.72
               . . .
```

Use the qualifier /DEPTH= when defining the Z axis to indicate positive downward, and /ORDER when setting the data set to properly read in the permuted data:

```
yes? DEFINE AXIS/X=30W:25W:1/UNIT=degrees salx
yes? DEFINE AXIS/Y=5S:5N:1/UNIT=degrees saly
yes? DEFINE AXIS/Z=0:1000:100/UNIT=meters/DEPTH salz
yes? DEFINE GRID/X=salx/Y=saly/Z=salz salgrid
yes? FILE/ORDER=zxy/GRID=salgrid/VAR=sal/COL=11 sal.dat
```

Chapter 3: VARIABLES AND EXPRESSIONS

1 VARIABLES

Variables are of 3 kinds:

1)	file variables	(read from disk files)
2)	user-defined variables	(defined by the user with LET command)
3)	diagnostic variables	(for GFDL/MOM Pacific ocean model runs)

All 3 types may be accessed identically in all commands and expressions.

All variables, regardless of kind, possess the following associated information:

1) grid - the underlying coordinate structure
2) units
3) title
4) title modifier (additional explanation of variable)
5) flag value for missing data points

Use the commands SHOW DATA, SHOW VARIABLES and SHOW VARIABLES/DIAGNOSTIC to examine the available variables of each type, respectively.

The pseudo-variables I, J, K, L, X, Y, Z, T and others may be used to refer to the underlying grid locations and characteristics and to create abstract variables.

1.1 File variables

File variables are stored in disk files. Input data files can be ASCII, binary, NetCDF or TMAP-formatted (see Chapter 2, Data Sets). File variables are made available with the SET DATA command.

1.2 Diagnostic variables

Diagnostic variables are available only with GDFL/MOM/Philander model output. They are computed interactively by FERRET from state variables and appear as part of the data set.

1.3 Pseudo-variables

Pseudo-variables are variables whose values are coordinates or coordinate information from a grid. Valid pseudo-variables are

X - x axis coordinates
Y - y axis coordinates
Z - z axis coordinates
T - t axis coordinates

I - x axis subscripts
J - y axis subscripts
K - z axis subscripts
L - t axis subscripts

XBOX - size of x grid box
YBOX - size of y grid box
ZBOX - size of z grid box
TBOX - size of t grid box

A grid box is a concept needed for some transformations along an axis; it is the length along an axis that belongs to a single grid point and functions as a weighting factor during integrations and averaging transformations.

The pseudo-variables I,J,K and L are subscripts; that is, they are a coordinate system for referring to grid locations in which the points along an axis are regarded as integers from 1 to the number of points on the axis. This is clear if you look at one of the sample data sets:

```
yes? SET DATA levitus_climatology
yes? SHOW DATA/FULL
  1> /home/e1/tmap/fer_dsets/descr/levitus_climatology.des  (default)
     Levitus annual climatology (1x1 degree)
           diagnostic variables: NOT available
    name   title                       I         J         K         L
    TEMP   TEMPERATURE               1:360     1:180     1:20      1:1
      ... on grid GLEVITR1   X=20E:20E(380)  Y=90S:90N  Z=0m:5000m
    SALT   SALINITY                  1:360     1:180     1:20      1:1
      ... on grid GLEVITR1   X=20E:20E(380)  Y=90S:90N  Z=0m:5000m

   time-independent data file: levitus_climatology.001
```

We see that there are 20 points along the z-axis (1:20 under K), for example, and that the z-axis coordinate values range from 0 meters to 5000 meters. Pseudo-variables depend only on the underlying grid, and not on the variables (in this case, temperature and salt).

Examples: Pseudo-variables

1) `yes? LIST/I=1:10 I`
2) `yes? LET xflux = u * vbox[G=u]`

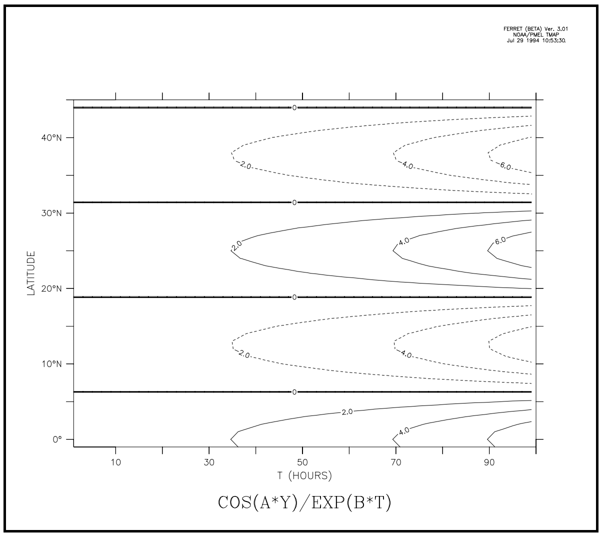
1.4 User-defined variables

New variables can be defined from existing variables and from abstract mathematical quantities (such as COS(latitude)) with command DEFINE VARIABLE (alias LET).

See command DEFINE VARIABLE and command LET in the Commands Reference.

Examples: User-defined variables

1) `yes? LET/TITLE="Surface Relief x1000 (meters)" r1000=rose/1000`
2) `yes? LET/TITLE="Temperature Deviation" tdev=temp - temp[Z=@ave]`


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Jul 29 1994 10:53:30.
LATITUDE
40°N
30°N
20°N
10°N
0°
0
2.0
4.0
6.0
10
30
50
70
90
T (HOURS)
COS(A*Y)/EXP(B*T)

Action commands may use any valid algebraic expression involving constants, operators (+,-,*,...), functions (SIN, MIN, INT,...), pseudo-variables (X, TBOX, ...) and other variables.

A variable name may optionally be followed by square brackets containing region, transformation, data set and regridding qualifiers. For example, "temp", "salt[D=2]", "u[G=temp"], "u[Z=0:200@AVE]".

The expressions may also contain a syntax of
IF condition THEN expression_1 ELSE expression_2

Examples: Expressions

i) `temp ^ 2`
temperature squared

ii) `temp - temp[Z=@AVE]`
for the range of Z in the current context, the temperature deviations from the vertical average

iii) `COS(Y)`
the cosine of the Y coordinate of the underlying grid (by default, the y-axis is implied by the other variables in the expression)

iv) `IF (vwnd GT vwnd[D=monthly_navy_winds]) THEN vwnd ELSE 0`
use the meridional velocity from the current data set wherever it exceeds the value in data set monthly_navy_winds, zero elsewhere.

2.1 Operators

Valid operators are

+	
-	
*	
/	
^	(exponentiate)
AND	
OR	
GT	
GE	
LT	
LE	
EQ	
NE	

2.2 Multi-dimensional expressions

Operators and functions (discussed in the next section, Functions) may combine variables of like dimensions or differing dimensions.

If the variables are of like dimension then the result of the combination is of the same dimensionality as inputs. For example, suppose there are two time series which have data on the same time axis; the result of a combination will be a time series on the same time axis.

If the variables are of unlike dimensionality, then the following rules apply:

1) To combine variables together in an expression they must be "conformable" along each axis.
2) Two variables are conformable along an axis if the number of points along the axis is the same, or if one of the variables has only a single point along the axis (or, equivalently, is normal to the axis).
3) When a variable of size 1 (a single point) is combined with a variable of larger size, the variable of size 1 is "promoted" by replicating its value to the size of the other variable.
4) If variables are the same size but have different coordinates, they are conformable, but FERRET will issue a message that the coordinates on the axis are ambiguous. The result of the combination inherits the coordinates of the FIRST variable encountered that has more than a single point on the axis.

Examples:

Assume a region J=50/K=1/L=1 for examples 1 and 2. Further assume that variables v1 and v2 share the same x-axis.

1)
```
yes? LET newv = v1[I=1:10] + v2[I=1:10]   !same dimension (10)
```

2)
```
yes? LET newv = v1[I=1:10] + v2[I=5]      !newv has length of v1 (10)
```

3) We want to compare the salt values during the first half of the year with the values for the second half. Salt_diff will be placed on the time coordinates of the first variable - L=1:6. FERRET will issue a warning about ambiguous coordinates.

```
yes? LET salt_diff = salt[L=1:6] - salt[L=7:12]
```

4) In this example the variable zero will be promoted along each axis.

```
yes? LET zero = 0 * (i+j)
yes? LIST/I=1:5/J=1:5  zero         !5X5 matrix of 0's
```

5) Here we calculate density; salt and temp are on the same grid. This expression is an XYZ volume of points (100x100x10) of density at 10 depths based on temperature and salinity values at the top layer (K=1).

```
yes? SET REGION/I=1:100/J=1:100
yes? LET dens = rho_un (temp[K=1], salt[K=1], Z[G=temp,K=1:10]
```

2.3 Missing value flags

Every FERRET variable has an associated missing value flag. For ASCII (and binary) files the missing value flag is set with SET VARIABLE/BAD=. For NetCDF files it is set with the attributes missing_value and/or _FillValue. For GT and TS data it is the D_BAD_FLAG variable in the descriptor file (*.des). Abstract variables are always assigned FERRET's default flag which is -1E34 (e.g. LIST/FORMAT=(E) 1/0). In all cases in which the missing value flag is unspecified, -1E34 is the default flag.

2.3.1 Missing value flag in expressions

If all of the component variables in an expression use the *same* bad data flag, the result inherits that bad data flag. If some of the component variables in an expression use differing bad value flags, the result inherits the bad value flag of the *first* variable in the expression.

2.3.2 Displaying the missing value flag

If the LIST command is used, missing values are, by default, displayed as "...." To examine the flag as a numerical value, use LIST/FORMAT=(E) (or some other suitable format).

2.4 Functions

Valid functions are

Name	#Args	Description
MAX	2	Compares two fields and selects the point by point maximum. `MAX( temp[K=1], temp[K=2] )` returns the maximum temperature comparing the first 2 z-axis levels
MIN	2	Compares two fields and selects the point by point minimum. `MIN( airt[L=10], airt[L=9] )` gives the minimum air temperature comparing two timesteps
INT	1	Truncates values to integers. `INT( salt )` returns the integer portion of variable "salt" for all values in the current region
ABS	1	Absolute value. `ABS( U )` takes the absolute value of U for all points within the current region
EXP	1	e^X - Exponential; argument is real. `EXP( X )` raises e to the power X for all points within the current region

LN	1	$\log_e X$ - Natural logarithm; argument is real. `LN( X )` takes the natural logarithm of X for all points within the current region
LOG	1	$\log_{10} X$ - Common logarithm; argument is real. `LOG( X )` takes the common logarithm of X for all points within the current region
SIN	1	Trigonometric sine; argument is in radians and is treated modulo 2*pi. `SIN( X )` computes the sine of X for all points within the current region
COS	1	Trigonometric cosine; argument is in radians and is treated modulo 2*pi. `COS( Y )` computes the cosine of Y for all points within the current region
TAN	1	Trigonometric tangent; argument is in radians and is treated modulo 2*pi. `TAN( theta )` computes the tangent of theta for all points within the current region
ASIN	1	Trigonometric arcsine (-pi/2,pi/2). The result will be flagged as missing if the absolute value of the argument is greater than 1; result is in radians. `ASIN( value )` computes the arcsine of "value" for all points within the current region
ACOS	1	Trigonometric arccosine (0,pi). The result will be flagged as missing of the absolute value of the argument greater than 1; result is in radians. `ACOS ( value )` computes the arccosine of "value" for all points within the current region
ATAN	1	Trigonometric arctangent (-pi/2,pi/2); result is in radians. `ATAN( value )` computes the arctangent of "value" for all points within the current region
ATAN2	2	2-argument trigonometric arctangent of Y/X (-pi,pi); discontinuous at Y=0. `ATAN2( X,Y )` computes the 2-argument arctangent of Y/X for all points within the current region

MOD	2	Modulo operation (arg1 - arg2*[arg1/arg2]). Returns the remainder when the first argument is divided by the second. `MOD( X,2 )` computes the remainder of X/2 for all points within the current region
MISSING	2	Replaces missing values in the first argument (multi-dimensional variable) with the second argument; the second argument may be any conformable variable. `MISSING( temp, -999 )` replaces missing values in temp with -999 `MISSING( sst, temp[D=coads_climatology] )` replaces missing sst values with temperature from the COADS climatology
IGNORE0	1	Replaces zeros in a variable with the missing value flag for that variable. `IGNORE0( salt )` replaces zeros in salt with the missing value flag
RANDU	1	Generates a grid of uniformly distributed [0,1] pseudo-random values. The first valid value in the field is used as the random number seed. Values that are flagged as bad remain flagged as bad in the random number field. `RANDU( temp[I=105:135,K=1:5] )` generates a field of uniformly distributed random values of the same size and shape as the field "temp[I=105:135,K=1:5]" using temp[I=105,k=1] as the pseudo-random number seed.
RANDN	1	Generates a grid of normally distributed pseudo-random values. As above, but normally distributed rather than uniformly distributed.
RHO_UN	3	Calculates rho (density kg/m^3) from salt (psu), temperature (deg C) and pressure (decibars) using the 1980 UNESCO International Equation of State (IES80). The routine uses the high pressure equation of state from Millero et al. 1980 and the one-atmosphere equation of state from Millero and Poisson 1981 as reported in Gill 1982. The notation follows Millero et al. 1980 and Millero and Poisson 1981. `RHO_UN( salt, temp, Z )`
THETA_FO	4	Calculates local potential pressure field at salt (psu), temperature (deg C), pressure (decibars) and specified reference pressure. This calculation uses Bryden 1973 polynomial for adiabatic lapse rate

and Runge-Kutta 4-th order integration algorithm. References:
Bryden,H., 1973, Deep-Sea Res.,20,401-408
Fofonoff,N,M, 1977, Deep-Sea Res.,24,489-491.
`THETA_FO( salt, temp, Z, 0 )`

2.5 Transformations

Transformations (e.g. averaging, integrating, etc.) may be specified along the axes of a variable. Some transformations (e.g. averaging) reduce a range of data to a point; others (e.g. differentiating) retain the range.

When transformations are specified along more than one axis of a single variable the order of execution is X axis first, then Y then Z then T.

Example syntax: `TEMP[Z=0:100@LOC:20]` (depth at which temp has value 20)

Valid transformations are

Transform	Default Argument	Description
@DIN		definite integral (weighted sum)
@IIN		indefinite integral (weighted running sum)
@AVE		average
@VAR		unweighted variance
@MIN		minimum
@MAX		maximum
@SHF	1 pt	shift
@SBX	3 pt	boxcar smoothed
@SBN	3 pt	binomial smoothed
@SHN	3 pt	Hanning smoothed
@SPZ	3 pt	Parzen smoothed
@SWL	3 pt	Welch smoothed
@DDC		centered derivative
@DDF		forward derivative
@DDB		backward derivative
@NGD		number of valid points
@NBD		number of bad (invalid) points flagged
@SUM		unweighted sum
@RSUM		running unweighted sum
@FAV	3 pt	fill missing values with average
@FLN:n	1 pt	fill missing values by linear interpolation
@FNR:n	1 pt	fill missing values with nearest point
@LOC	0	coordinate of ... (e.g. depth of 20 degrees)
@WEQ		"weighted equal" (integrating kernel)

The command SHOW TRANSFORM will produce a list of currently available transformations.

Examples: Transformations

`u[Z=0:100@AVE]` - average of u between 0 and 100 in Z
`sst[T=@SBX:10]` - box-car smooths sst with a 10 time point filter
`tau[L=1:25@DDC]` - centered time derivative of tau
`v[L=@IIN]` - indefinite (accumulated) integral of v
`qflux[X=@AVE,Y=@AVE]` - XY area-averaged qflux

2.5.1 General information about transformations

Transformations are normally computed axis by axis; if multiple axes have transformations specified simultaneously (e.g. `U[Z=@AVE,L=@SBX:10]`) the transformations will be applied sequentially in the order X then Y then Z then T. There are two exceptions to this: if @DIN is applied simultaneously to both the X and Y axes (in units of degrees of longitude and latitude, respectively) the calculation will be carried out on a per-unit-area basis (as a true double integral) instead of a per-unit-length basis, sequentially. This ensures that the COSINE(latitude) factors will be applied correctly. The same applies to @AVE simultaneously on X and Y.

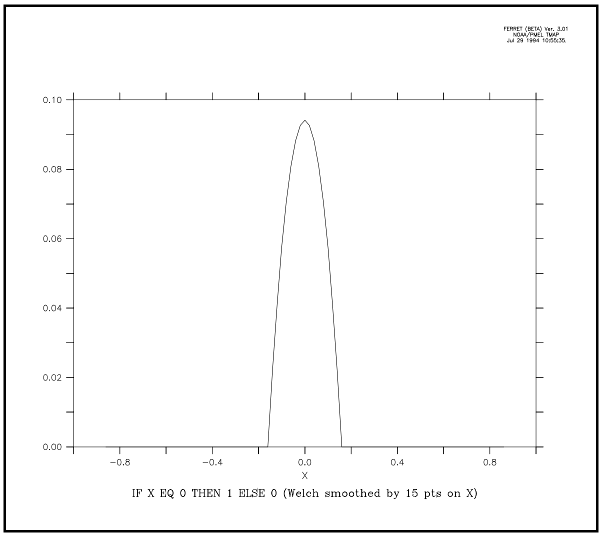
Data that are flagged as invalid are excluded from calculations.

When calculating integrals and derivatives (@IIN, @DIN, @DDC, @DDF, and @DDB) FERRET attempts to use standardized units for the grid coordinates. If the underlying axis is in a known unit of length FERRET converts grid box lengths to meters. If the underlying axis is in a known unit of time FERRET converts grid box lengths to seconds. If the underlying axis is degrees of longitude a factor of COSINE (latitude) is applied to the grid box lengths in meters.

If the underlying axis units are unknown FERRET uses those unknown units for the grid box lengths. (If FERRET does not recognize the units of an axis it displays a message to that effect when the DEFINE AXIS or SET DATA command defines the axis.) See command DEFINE AXIS/UNITS in the Commands Reference in this manual for a list of recognized units.

All integrations and averaging are accomplished by multiplying the width of each grid box by the value of the variable in that grid box - then summing and dividing as appropriate for the particular transformation.

If integration or averaging limits are given as world coordinates, the grid boxes at the edges of the region specified are weighted according to the fraction of grid box that actually lies within the specified region. If the transformation limits are given as subscripts, the full box size of each grid point along the axis is used - including the first and last subscript given. The region information that is listed with the output reflects this.


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Jul 29 1994 10:55:35
0.10
0.08
0.06
0.04
0.02
0.00
−0.8
−0.4
0.0
0.4
0.8
X
IF X EQ 0 THEN 1 ELSE 0 (Welch smoothed by 15 pts on X)

grid_box*variable product at each grid point. Grid points at the ends of the indicated range are weighted by the fraction of the grid box that falls within the integration interval.

If @DIN is specified simultaneously on both an X axis of longitude and a Y axis of latitude the calculation will be performed as a double integration rather than as sequential single integrations.

Example:

```
yes? CONTOUR/X=160E:160W/Y=5S:5N u[Z=0:50@DIN]
```

2.5.4 @IIN - indefinite integral

The transformation @IIN computes the indefinite integral - at each subscript of the result it is the value of the integral from the start value to the upper edge of that grid box. It is obtained as a running sum of the grid_box*variable product at each grid point. Grid points at the ends of the indicated range are weighted by the fraction of the grid box that falls within the integration interval.

Example:

```
yes? CONTOUR/X=160E:160W/Z=0 u[Y=5S:5N@IIN]
```

2.5.5 @AVE - average

The transformation @AVE computes the average weighted by grid box size - a single number representing the average of the variable between two endpoints.

If @AVE is specified simultaneously on both an X axis of longitude and a Y axis of latitude the calculation will be performed as a double integration rather than as sequential single integrations.

Example:

```
yes? CONTOUR/X=160E:160W/Y=5S:5N u[Z=0:50@AVE]
```

Note that the unweighted mean can be calculated using the @SUM and @NGD transformations.

2.5.6 @VAR - weighted variance

The transformation @VAR computes the weighted variance of the variable with respect to the indicated region (ref. *Numerical Recipes, The Art of Scientific Computing,* by William H. Press et al., 1986).

2.5.7 MIN - minimum

The transformation @MIN finds the minimum value of the variable within the specified axis range.

Example:
For fixed Z and Y

```
yes? PLOT/T="1-JAN-1982":"1-JAN-1983"    temp[X=160E:160W@MIN]
```

plots a time series of the minimum temperature found between longitudes 160 east and 160 west.

2.5.8 @MAX - maximum

The transformation @MAX finds the maximum value of the variable within the specified axis range. See also @MIN.

2.5.9 @SHF:n - shift

The transformation @SHF shifts the data up or down in subscript by the number of points given as the argument.

Examples:

```
u[L=@SHF:2]
```

associates the value of U[L=3] with the subscript L=1.

```
u[L=@SHF:1]-U
```

gives the forward difference of the variable U along the L axis.

2.5.10 @SBX:n - boxcar smoother

The transformation @SBX applies a boxcar window (running mean) to smooth the variable along the indicated axis. The width of the boxcar is the number of points given as an argument to the transformation. All points are weighted equally, regardless of the sizes of the grid boxes, making this transformation best suited to axes with equally spaced points. If the number of points specified is even, however, @SBX weights the end points of the boxcar smoother as 1/2.

Example:

```
yes? PLOT/X=160W/Y=0 u[L=1:120@SBX:5]
```

The transformation @SBX does not reduce the number of points along the axis; it replaces each of the original values with the average of its surrounding points. Regridding can be used to reduce the number of points.

2.5.11 @SBN:n - binomial smoother

The transformation @SBN applies a binomial window to smooth the variable along the indicated axis. The width of the smoother is the number of points given as an argument to the transformation. The weights are applied without regard to the widths of the grid boxes making this transformation best suited to axes with equally spaced points.

Example:

```
yes? PLOT/X=160W/Y=0/Z=0 u[L=1:120@SBN:15]
```

The transformation @SBN does not reduce the number of points along the axis; it replaces each of the original values with a weighted sum of its surrounding points. Regridding can be used to reduce the number of points. The argument specified with @SBN, the number of points in the smoothing window, must be an odd value; an even value would result in an effective shift of the data along its axis.

2.5.12 @SHN:n - Hanning smoother

Transformation @SHN applies a Hanning window to smooth the variable along the indicated axis (ref. *Numerical Recipes, The Art of Scientific Computing*, by William H. Press et al., 1986). In other respects it is identical in function to the @SBN transformation. Note that the Hanning window used by FERRET contains only non-zero weight values with the window width. Some interpretations of this window function include zero weights at the end points. Use an argument of N-2 to achieve this effect (e.g., @SBX:5 is equivalent to a 7 point Hanning window which has zeros as it first and last weights).

2.5.13 @SPZ:n - Parzen smoother

Transformation @SPZ applies a Parzen window to smooth the variable along the indicated axis (ref. *Numerical Recipes, The Art of Scientific Computing*, by William H. Press et al., 1986). In other respects it is identical in function to the @SBN transformation.

2.5.14 @SWL:n - Welch smoother

Transformation @SWL applies a Welch window to smooth the variable along the indicated axis (ref. *Numerical Recipes, The Art of Scientific Computing*, by William H. Press et al., 1986). In other respects it is identical in function to the @SBN transformation.

2.5.15 @DDC - centered derivative

The transformation @DDC computes the derivative with respect to the indicated axis using a centered differencing scheme. The units of the underlying axis are treated as they are with integrations. If the points of the axis are unequally spaced, note that the calculation used is still $(F_{i+1} - F_{i-1}) / (X_{i+1} - X_{i-1})$.

Example:

```
yes? PLOT/X=160W/Y=0/Z=0 u[L=1:120@DDC]
```

2.5.16 @DDF - forward derivative

The transformation @DDF computes the derivative with respect to the indicated axis. A forward differencing scheme is used. The units of the underlying axis are treated as they are with integrations.

Example:

```
yes? PLOT/X=160W/Y=0/Z=0 u[L=1:120@DDF]
```

2.5.17 @DDB - backward derivative

The transformation @DDF computes the derivative with respect to the indicated axis. A backward differencing scheme is used. The units of the underlying axis are treated as they are with integrations.

Example:

```
yes? PLOT/X=160W/Y=0/Z=0 u[L=1:120@DDB]
```

2.5.18 @NGD - number of good points

The transformation @NGD computes the number of good (valid) points of the variable with respect to the indicated axis. Use @NGD in combination with @SUM to determine the number of good points in a multi-dimensional region.

2.5.19 @NBD - number of bad points

The transformation @NBD computes the number of bad (invalid) points of the variable with respect to the indicated axis. Use @NBD in combination with @SUM to determine the number of bad points in a multi-dimensional region.

2.5.20 @SUM - unweighted sum

The transformation @SUM computes the unweighted sum (arithmetic sum) of the variable with respect to the indicated axis. This transformation is most appropriate for regions specified by subscript. If the region is specified in world coordinates, the edge points are *not* weighted - they are wholly included in or excluded from the calculation depending on the location of the grid points with respect to the specified limits.

2.5.21 @RSUM - running unweighted sum

The transformation @RSUM computes the running unweighted sum of the variable with respect to the indicated axis. @RSUM is to @IIN as @SUM is to @DIN. The treatment of edge points is identical to @SUM.

2.5.22 @FAV - averaging filler

The transformation @FAV fills holes (values flagged as invalid) in variables with the average value of the surrounding grid points along the indicated axis. The width of the averaging window is the number of points given as an argument to the transformation. All of the surrounding points are weighted equally, regardless of the sizes of the grid boxes, making this transformation best suited to axes with equally spaced points. If the number of points specified is even, however, @FAV weights the end points of the filling region by 1/2. If any of the surrounding points are invalid they are omitted from the calculation. If all of the surrounding points are invalid the hole is not filled.

Example:

```
yes? CONTOUR/X=160W:160E/Y=5S:0 u[X=@FAV:5]
```

2.5.23 @FLN:n - linear interpolation filler

The transformation @FLN:n fills holes in variables with a linear interpolation from the nearest non-missing surrounding point. n specifies the number of points beyond the edge of the indicated axis limits to include in the search for interpolants (default n = 1). Unlike @FAV, @FLN is sensitive to unevenly spaced points and computes its linear interpolation based on the world coordinate locations of grid points.

2.5.24 @FNR:n - nearest neighbor filler

The transformation @FNR:n is similar to @FLN:n, except that it replicates the nearest point to the missing value. In the case of points being equally spaced around the missing point, the mean value is used.

2.5.25 @LOC - location of

The transformation @LOC accepts an argument value - the default value is zero if no argument is specified. The transformation @LOC finds the single location at which the variable first assumes the value of the argument. The result is in units of the underlying axis. Linear interpolation is used to compute locations between grid points. If the variable does not assume the value of the argument within the specified region the @LOC transformation returns an invalid data flag.

For example, temp[Z=0:200@LOC:18] finds the location along the Z axis (often depth in meters) at which the variable "temp" (often temperature) first assumes the value 18, starting at Z=0 and searching to Z=200.

```
yes? CONTOUR/X=160E:160W/Y=10S:10N    temp[Z=0:200@LOC:18]
```

produces a map of the depth of the 18 degree isotherm. See also the General Information section in this chapter.

Note that the transformation @LOC can be used to locate non-constant values, too, as the following example illustrates:

Example: locating non-constant values
Determine the depth of maximum salinity.

```
yes? LET max_salt = salt[Z=@MAX]
yes? LET zero_at_max = salt - max_salt
yes? LET depth_of_max = zero_at_max[Z=@LOC:0]
```

2.5.26 @WEQ - weighted equal; integration kernel

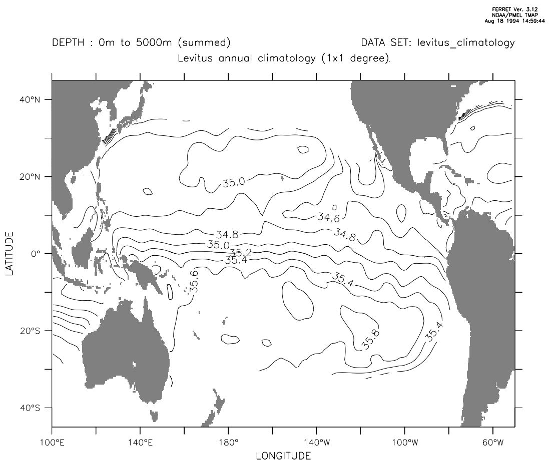
The @WEQ ("weighted equal") transformation is the subtlest and arguably the most powerful transformation within FERRET. It is a generalized version of @LOC; @LOC always determines the value of the axis coordinate (the variable X, Y, Z, or T) at the points where the gridded field has a particular value. More generally, @WEQ can be used to determine the value of *any* variable at those points.

Like @LOC, the transformation @WEQ finds the location along a given axis at which the variable is equal to the given (or default) argument. For example, V1[Z=@WEQ:5] finds the Z locations at which V1 equals "5". But whereas @LOC returns a single value (the linearly interpolated axis coordinate values at the locations of equality) @WEQ returns instead a field of the same size as the original variable. For those two grid points that immediately bracket the location of the argument, @WEQ returns interpolation coefficients. For all other points it returns missing value flags. If the value is found to lie identically on top of a grid point an interpolation coefficient of 1 is returned for that point alone.

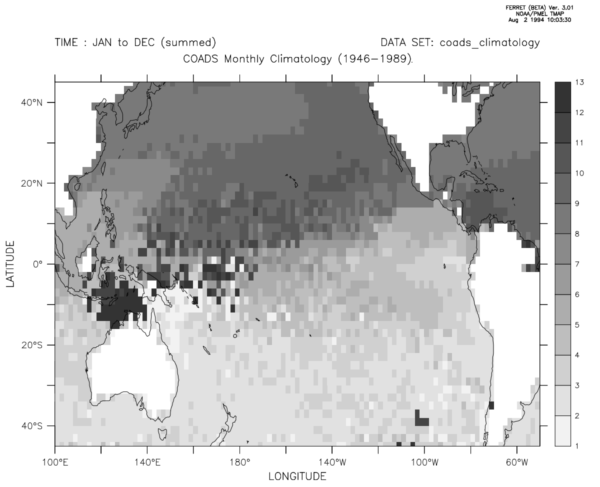
Example 1

```
yes? LET v1 = X/4
yes? LIST/X=1:6 v1, v1[X=@WEQ:1], v1[X=@WEQ:1.2]

 X     v1    @WEQ:1  @WEQ:1.2
 -   -----    -----   -----
 1:  0.250    ....      ....
 2:  0.500    ....      ....
 3:  0.750    ....      ....
 4:  1.000   1.000    0.2000
 5:  1.250    ....    0.8000
 6:  1.500    ....      ....
```



Salinity on the 20 degree isotherm



Month of warmest SST

Chapter 4: GRIDS AND REGIONS

1 OVERVIEW

Information describing a region in space/time, a data set, and a grid is collectively referred to as the "context". The current context may be examined with the commands SHOW DATA_SET, SHOW REGION and SHOW GRID. The context may be set explicitly with the commands SET DATA_SET, SET REGION and SET GRID.

The context may be modified for the duration of a single command with qualifiers to the command name (separated by slashes). The same qualifiers in square brackets may also modify single variables, changing the context only of that variable:

```
yes? PLOT/D=levitus_climatology  temp, salt

yes? CONTOUR  rose[D=etopo20]

yes? FILL/Z=0  temp[L=2] - temp[L=1]
```

2 GRIDS

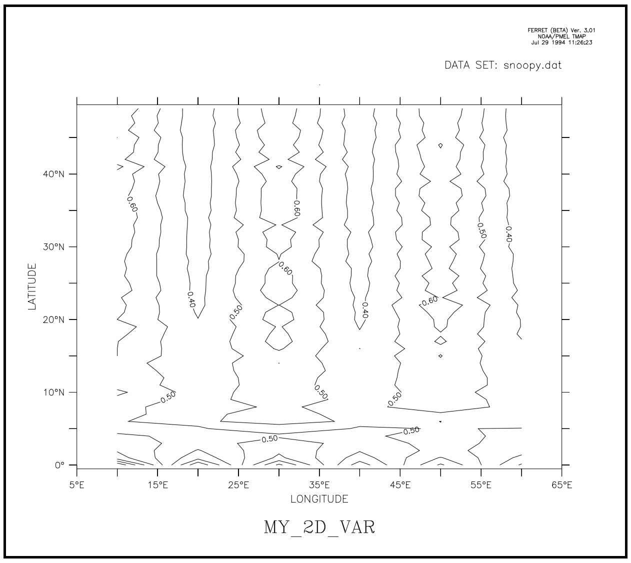
Every variable has an underlying grid which defines a coordinate space. All grids are in a sense 4 dimensional (X,Y,Z and T) but axes normal to the data are represented as "normal" (such as the Z axis of the surface wind stress).

Grids can be viewed, specified and created using SHOW GRID, SET GRID, DEFINE AXIS, and DEFINE GRID. These commands are all in the Commands Reference section of this manual. Data can be regridded by the G= modifier. (See Chapter 4 section "Regridding".)

2.1 Defining Grids

Axes and grids can be created by DEFINE AXIS and DEFINE GRID. NetCDF and TMAP-formatted data set variables have all of the necessary grid and axis definitions embedded in the data set files, but if you are reading data from an ASCII or binary file, you must tell FERRET about the underlying grid of your data.

If you are creating abstract expressions entirely from pseudo-variables, you may want to define a grid in order to define the coordinate space of your calculation. This will also help produce a nicely labelled plot. (See example in Chapter 3 section "Abstract Variables".)


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Jul 29 1994 11:26:23
DATA SET: snoopy.dat
LATITUDE
40°N
30°N
20°N
10°N
0°
5°E
15°E
25°E
35°E
45°E
55°E
65°E
LONGITUDE
0.60
0.40
0.50
0.60
0.60
0.50
0.50
0.40
0.60
0.50
0.50
0.40
0.50
0.50
MY_2D_VAR

```
yes? GO regridding_demo
```

Regridding is essential for algebraic operations that combine variables on incompatible grids. FERRET provides the commands DEFINE AXIS and DEFINE GRID to assist with the creation of arbitrary grids.

FERRET insists on consistent dimensionality during regridding operations. It is not permissible for a variable that is normal to a given axis to be directly regridded to a grid that has defined locations along that axis or visa versa.

Again, type "GO regridding_demo" at the FERRET prompt for some graphical examples.

Examples

1) Suppose the variables u and temp are on staggered X, Y, and Z axes but share the same T axis. Then the two variables can be multiplied together on the axes (grid) of the u variable as follows:

```
yes? CONTOUR u * temp[G=u]
```

This will regrid temp onto the u grid by multi-axis linear interpolation before performing the multiplication.

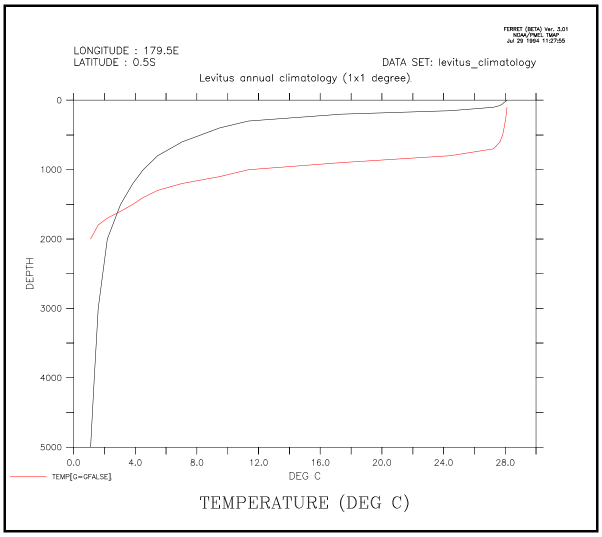
2) Two variables, v1 and v2, are defined on distinct 4-dimensional grids (X,Y,Z, and T axes). The T axes of the two grids are identical but the X, Y, and Z axes all differ between the two variables. (This is often the case in numerical model outputs.)

To obtain the variable v1 on its original Z (depth) locations but regridded in the XY plane to the grid locations of the variable v2, define a new grid (say, named "new_grid") that has the X and Y axes of v2 but the Z axis of v1.

```
yes? DEFINE GRID/LIKE=v2/Z=v1 new_grid        !define new grid
yes? LIST/X=160E:140W/Y=5S:5N v1[G=new_grid]  !request regridding
```

3) In this example we look at temperature data from two data sets. levitus_climatology, an annual climatology, has the variable "temp" on an XYZ grid which is 1x1 degree in XY, and coads_climatology, a monthly climatology, has the variable "sst" on an XYT grid which is 2x2 degrees in XY. Suppose we wish to look at the sea surface temperatures in January at the higher XY resolution of the Levitus data.

```
yes? SET DATA levitus_climatology
yes? SET DATA coads_climatology
yes? SET REGION/L=1/Z=0
yes? !get the name of the grid on which temp is defined
yes? SHOW GRID temp[D=levitus_climatology]  ! --> "glevitr1"
yes? DEFINE GRID/X=glevitr1/Y=glevitr1/Z=sst/L=sst glevitus_xy
yes? LIST/X=150E:155E/Y=0:5N sst[G=glevitus_xy]
```


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Jul 29 1994 11:27:55
LONGITUDE : 179.5E
LATITUDE : 0.5S
DATA SET: levitus_climatology
Levitus annual climatology (1x1 degree).
0
1000
2000
3000
4000
5000
DEPTH
0.0
4.0
8.0
12.0
16.0
20.0
24.0
28.0
DEG C
TEMP[G=GFALSE].
TEMPERATURE (DEG C)

If SET REGION is used, FERRET remembers the region as the default context for future commands, whereas a qualifier to a command name specifies the region for that command only, and a qualifier to a variable name specifies the region for that variable and command only.

Regions may be manipulated using DEFINE REGION, SET REGION, @ notation and CANCEL REGION. The Commands Reference section of this manual covers all of these topics.

Region information is normally specified in the following form:

`QUAL=val` or
`QUAL=lo_val:hi_val` or
`QUAL=val@transform` (as a variable qualifier only) or
`QUAL=lo_val:hi_val@transform` (as a variable qualifier only)

When the region for an axis is specified as a single value (instead of a range) FERRET, by default, selects the grid point of the grid box containing this value. The FERRET mode "interpolate" can control this behavior. See command SET MODE INTERPOLATE in Commands Reference.

Examples: Regions
Examples of valid region specifications.

1) Fully specify the region in an XY plane with the first vertical (Z) level and time 27739.

```
yes? SET REGION/X=140E:160W/Y=10S:20N/K=1/T=27739
```

2) Contour vertical heat advection within whatever region is the current default (previously set with SET REGION).

```
yes? CONTOUR qadz
```

3) Define, modify and set a named region and then modify with delta notation.

```
yes? DEFINE/REGION/Y=5S:5N YT !define region YT to be 5S:5N
yes? DEFINE REGION/DY=-1:+1 YT      !modify region YT to be 6S:6N
yes? SET REGION/@YT                 !set current region to YT
yes? SET REGION/DY=-1:+1            !modify current region to 7S:7N
```

4) List meridional currents calculated by averaging values between the surface and a depth of 50 m.

```
yes? LIST v[Z=0:50@AVE]
```

5) Equivalent to `v[Z=10] - v[Z=0:100@AVE]`, the anomaly at z=10 between v and the 0 to 100 meter depth average of v.

```
yes? LIST/Z=10 v - v[Z=0:100@AVE]
```

3.1 Latitude

Specify latitude or a latitude range with the qualifier Y or J. Specifications using J are integers between 1 and the number of points on the Y axis. Specifications using Y are in the units of the Y axis.

The units may be examined with SHOW GRID/Y. If the Y axis units are degrees of latitude then the Y positions may be specified as numbers followed by the letters "N" or "S".

Examples

```
yes? CONTOUR temp[Y=15S:10N]
yes? LIST/J=50 u
```

3.2 Longitude

Specify longitude or a longitude range with the qualifier X or I. Specifications using I are integers between 1 and the number of points on the X axis. Specifications using X are in the units of the X axis.

The units may be examined with SHOW GRID/X. If the units are degrees then X values may be given as numbers followed by "W" or "E" (e.g.160E,110.5W) or as values between 0 and 360 with Greenwich at 0 increasing eastward. Note: If the X axis is "modulo" then it is sometimes desirable to use X greater than 360.

Examples

```
yes? CONTOUR  temp[Y=160E:140W]
yes? LIST/I=100  u
yes? SHADE/X=100:460  temp    !360 degrees centered at 100W
```

See Chapter 4 section "Modulo Axes" for help with globe-encircling axes.

3.3 Depth

Specify depth or a depth range with the qualifier Z or K. Specifications using K are integers between 1 and the number of points on the Z axis. Specifications using Z are in the units of the Z axis.

The units may be examined with SHOW GRID/Z.

Examples

```
yes? CONTOUR temp[Z=0:100]
yes? LIST/K=3 u
```

3.4 Time

Specify time or a time range with the qualifier T or L. Specifications using L are integers between 1 and the number of points on the T axis. Specifications using T may refer to calendar dates or to the time step units in which the time axis of the data set is defined.

Calendar date/time values are entered in the format dd-mmm-yyyy:hh:mm:ss, for example 14-FEB-1988:12:30:00. At a minimum the string must contain day, month and year. If the string contains any colons it must be enclosed in quotation marks to differentiate from colons used to designate a range. If a time increment is specified with the repeat command given in calendar format (e.g. REPEAT/T="1-JAN-1982":"15-JAN-1982":6) it is interpreted as hours always. *Calendar dates in the years 0000 and 0001 are regarded as year-independent dates* (suitable for climatological data).

SHOW GRID/T can be used to display time step values. (Units may vary between data sets.) The commands SET MODE CALENDAR and CANCEL MODE CALENDAR can be used to view date strings or time steps, respectively.

Examples

```
yes? LIST/T="1-JAN-1982:13:50":"15-FEB-1982"   density
yes? CONTOUR temp[T=27740:30000]
yes? LIST/L=90 u
```

See Chapter 4 section "Modulo Axes" for help with climatological axes.

3.5 Delta

The notation q=lo:hi:delta (e.g. Y=20S:20N:5) specifies that the data in the requested range is regularly subsampled at interval "delta".

This notation is valid only for the REPEAT, SHOW GRID and DEFINE AXIS commands, and the qualifiers /XLIMITS and /YLIMITS used in action commands with graphical output.

3.6 @ notation

Regions may be named and referred to using the syntax "@name". Some commonly used regions are predefined. See commands SET REGION and DEFINE REGION in the Commands Reference section for further information.

If a region is specified using a combination of "@" notation and explicit axis limits the explicit axis limits will be evaluated after the "@" specification, possibly superseding the "@" limits.

Using the @ notation only sets/alters the axis limits specified in the named region. For example, suppose that region "XY" is defined for the X and Y axes, but not for the Z and T axes. Then

```
yes? SET REGION/@XY
```

modifies only X and Y limits. BUT,

```
yes? SET REGION XY
```

modifies all axes - X and Y to the limits specified by XY, and Z and T to unspecified (even if they were previously specified).

Examples

1) Contour the 25th time step of temperature data at depth 10 within region T, the "Tropical Pacific".

```
yes? CONTOUR/@T/Z=10/L=25 temp
```

2) Produce a contour plot over region W, the "Whole Pacific Ocean", in the XY plane (the variable to be contoured as well as the depth and time will be inferred from the current context).

```
yes? CONTOUR/@W var1
```

3) Set the default region to "T", the Tropical Pacific Ocean (latitude 23.5S to 23.5N).

```
yes? SET REGION/@T
```

4) Define a region and then supersede with an axis limit specification.

```
yes? DEFINE REGION/X=180:140W/Y=2S:2N/Z=5   BOX1
yes? SET REGION/@BOX1/Z=15                 !replace Z
```

Pre-defined regions

As a convenience in the analysis of TMAP's model outputs over the Tropical Pacific Ocean the following regions are pre-defined:

Name	Region	Latitude	Longitude
T	Tropical Pacific	23.5S:23.5N	130E:70W
N	Narrow Pacific	10.0S:10.0N	130E:70W
W	Whole Pacific	30.0S:50.0N	130E:70W

These may be redefined by the user for the duration of a FERRET session or until the definitions are canceled.

3.7 Modulo axes

Some axes are inherently "modulo" indicating that the axis wraps around - the first point immediately following the last.

To determine if an axis is modulo use SHOW AXIS or SHOW GRID. A letter "m" following the number of points in the axis indicates a modulo axis. The command SHOW GRID qualified by the appropriate axis limits can be used to examine any part of the axis -including points beyond the nominal length of the axis. The commands SET AXIS/MODULO and CANCEL AXIS/MODULO can be used to control this feature on an axis-by-axis basis.

Example

```
yes? SET DATA coads_climatology
yes? SHOW GRID/I=180:183 sst    !range request beyond last point
  GRID COADS1
  name       axis       # pts     start            end
  COADSX  LONGITUDE     180mr     21E              19E(379)
  [text omitted]
        I     X                  BOX_SIZ
      180>  19E(379)               2
      181>  21E(381)               2
      182>  23E(383)               2
      183>  25E(385)               2
```

The most common uses of modulo axes are:

1) As longitude axes for globe-encircling data sets. This allows any starting and any ending longitudes to be used, for example, X=140E:140E indicates the entire earth with data beginning and ending at 140E.

2) As time axes for climatological data. By this device the time axis appears to extend from 0 to infinity and the climatological data can be referred to at any point in time. This facilitates comparisons with data sets at fixed times.

As of FERRET version 3.1 a restriction exists on the use of modulo axes. For GT-formatted and NetCDF data sets only a single axis can be accessed modulo style at a time. Thus in a 12 month, globe-encircling climatology, it is valid to request X=180/L=1:36 or X=180:540/L=5, but *not* X=180:540/L=1:36.

Chapter 5: ANIMATIONS

1 OVERVIEW

A sequence of FERRET plots can be stored and then animated. Each plot is stored as one frame in a movie file. FERRET stores movie frames in Hierarchical Data Format (HDF), a format designed by the National Center for Supercomputing Applications (NCSA). A movie file can then be displayed as an animated sequence of frames with NCSA's xds - X Data Slice (not distributed with FERRET; see Chapter 4 section "Displaying a movie", for details).

2 CREATING A MOVIE

Creating a movie requires two steps:
1) designate an output file with SET MOVIE
2) generate a sequence of frames with REPEAT and FRAME

See commands SET MOVIE, CANCEL MOVIE, SHOW MOVIE, FRAME and REPEAT in the Commands Reference section of this manual.

Example: basic movie

```
yes? SET MOVIE/FILE=my_movie.mgm       !specify output file
yes? SET DATA coads_climatology        !specify data set
yes? SET REGION/@W                     !specify Pacific Ocean
yes? LET/TITLE="SST Anomaly" SST_ANOM = SST - SST[L=1:12@AVE]
yes? REPEAT/L=1:12 FILL/FRAME SST_ANOM       !filled contour of sea surface\
                                             temp  anomaly  captured  and\
                                             written to HDF file
yes? CANCEL MOVIE                      !close movie file (not required)
```

If you omit the SET MOVIE/FILE= command, the default name "ferret.mgm" will be assigned to the movie file.

REPEAT executes its argument (in the above example, FILL) successively for each timestep specified. REPEAT can have multiple arguments separated by semi-colons and enclosed in parentheses.

FRAME is a stand-alone command, but also a qualifier for the graphical output commands PLOT, CONTOUR, FILL (alias for CONTOUR/FILL), SHADE, VECTOR and WIRE.

The saved animation frames are exactly the size and shape of the window from which they are created. Thus a large window results in a larger, slower animation that demands more disk space and memory to play back. The SET WINDOW/SIZE= command is generally used to specify minimally acceptable frame size.

See Chapter 4 section "Advanced Moviemaking" for more examples.

3 DISPLAYING A MOVIE

Viewing a movie requires software which is not included with the FERRET distribution (although in some cases we have made the binary available in FERRET's anonymous ftp area). NCSA's X Data Slice reads HDF files and is available via anonymous ftp from NCSA. It requires about 1.7Mb of disk space. NCSA's ftp server is

ftp.ncsa.uiuc.edu login id is "anonymous", give your email address as the password

Consult the README files you will find there for instructions on obtaining X Data Slice. Other utilities from NCSA can also be used for animations.

4 ADVANCED MOVIE-MAKING

4.1 REPEAT command

The REPEAT command is quite flexible - it allows you to repeat a sequence of commands, not just a single command as in the basic example above. You can give the GO command as an argument to REPEAT. The following examples demonstrate these techniques.

Note: MODE VERIFY must be SET (this is the default state) for loop counting to work.

Example 1
Here we give multiple arguments to REPEAT; note the semi-colon separation and the parentheses. Note that FRAME, in this is example, is used as a stand-alone command.

```
yes? REPEAT/L=1:12 (FILL SST; GO fland; FRAME)
```

Example 2
In this example the user calls setup_movie.jnl (text included below), title.jnl, which creates a title frame, then repeats main_movie.jnl (text included below) for each time step desired. Finally, the user adds a frame of credits at the end of the movie. Each of the scripts would end with the FRAME command (except setup_movie). Using GO scripts as arguments to REPEAT allows you customize the plot with many commands before finally issuing FRAME, as the text of main_movie.jnl below demonstrates.

```
yes? ! make the movie
yes? GO setup_movie
yes? GO title
yes? REPEAT/L=1:12 GO main_movie
yes? GO credits

! setup_movie.jnl
SET WINDOW/SIZE=.45/ASPECT=0.7
SET MOVIE/file=my_movie.mgm
```

```
SET DATA coads_climatology
SET REGION/X=130E:75W/Y=8S:8N
SET MODE CALENDAR:months
GO bold
PPL SHAKEY ,,.15,.2
PPL AXLEN 8.8,4.8

! main_movie.jnl
FILL/SET_UP/LEVELS=(16,31,1) sst
PPL LABS; PPL TITLE
PPL FILL
LABEL 210,9.5,0,0,.22 @TRCOADS MONTHLY CLIMATOLOGY (1946-1989)
LABEL 210,-12,0,0,.22 @TRSEA SURFACE TEMPERATURE (DEG C)
LABEL 130,11,-1,0,.22 @TR'LAB4'
FRAME
```

Note: If you use the FILL command, we suggest that you use SHADE while customizing and fine-tuning your movie, then use FILL for the final run. SHADE is much faster.

4.2 Initializing the color table

If you create a movie with a title frame, or a first frame which otherwise uses different colors than the rest of the movie, you should be aware of an HDF peculiarity: all the colors which you plan to use in your movie must be in the first frame, or else color behavior will be unpredictable when you animate.

To "reserve" the colors you need, use overlapping full-window viewports. Make a representative plot in the title frame, then cover over it with either a black or white rectangle and finally write the title text. Here is a script which initializes the color table while creating a title frame.

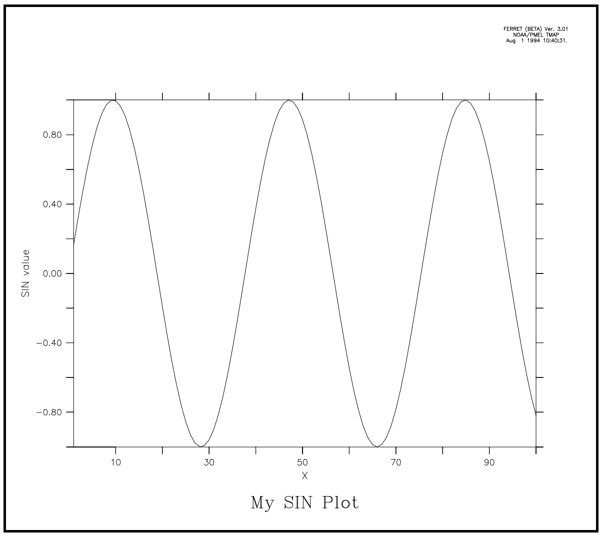
```
! define 3 identical full-frame viewports
DEFINE VIEW full1;  DEFINE VIEW full2;  DEFINE VIEW full3

! draw frame one of the movie in full color
SET VIEW full1
SET DATA coads_climatology
SHADE/LEVELS=(16,31,1)/L=1 sst                  ! dummy frame

! white-out over the picture
SET VIEW full2
GO setup_text
SHADE/PALETTE=white/NOLAB/NOKEY/i=1:2/j=1:2  (i+j)*0

!put on title frame labels (using [0,1] coordinate space)
SET VIEW full3
GO setup_text
PPL PLOT
LABEL .5,.7,0,0,.3 @TRMy Title
PPL ALINE 1,.2,.55,.8,.55
PPL ALINE 1,.2,.53,.8,.53
LABEL .5,.4,0,0,.2 @CRBy me

!capture the title frame and clean up
FRAME
GO cleanup_text
```


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 10:40:31.
SIN value
0.80
0.40
0.00
-0.40
-0.80
10
30
50
70
90
X
My SIN Plot

The examples throughout this chapter show how the /SET_UP qualifier on graphics commands can be used to delay rendering of a plot while the user modifies plot appearance with PPLUS commands.

Below is a list of PPLUS commands which are reset by FERRET:

PPLUS command	when reset by FERRET
XFOR, YFOR	reset for every plot
XLAB, YLAB	reset for every plot
XAXIS, YAXIS	reset for every plot
LABS	reset for every plot
ALINE	reset for every plot
TAXIS OFF	reset for every plot
TITLE	reset for every plot
TICS	reset for every plot (small tic size, only)
WINDOW ON	reset for every plot
PEN 1,n	reset for every plot
LIMITS	reset for every plot
ORIGIN	reset by SET WINDOW/ASPECT and SET VIEWPORT; Y origin may be shifted to accommodate many line style keys
AXLEN	modified by SET WINDOW/ASPECT and SET VIEWPORT
VIEWPORT	modified by WIRE/VIEW
LEV	modified by CONTOUR and SHADE unless /LEVELS_SAME given
VECSET	modified by VECTOR unless /LENGTH_SAME given
WINDOW	modified for "fresh" plots but not for overlay plots

2 GRAPHICAL OUTPUT

2.1 FERRET graphical output controls

FERRET command	Function
CONTOUR	produces a contour plot of a single field
FILL	alias for CONTOUR/FILL; produces color-filled contour plot
PLOT	produces a line or symbol plot of one or more arrays
SHADE	produces a shaded representation (rectangular cells)
VECTOR	produces a vector arrow plot
WIRE	produces a 3D wire frame plot
SET WINDOW	manipulates graphics windows
SET VIEWPORT	places graphics output into a sub-window (pane)

2.2 PPLUS graphical output commands

Whenever a plot is customized using /SET_UP to delay display, the plot will ultimately be rendered using a PPLUS graphical output command (not the FERRET counterpart). A customized contour or filled-contour plot is rendered with PPL CONTOUR, a wire frame plot with PPL VIEW and so on.

Command	Function
CONTOUR	makes a contour plot
PLOT	plots x-y pairs for all lines of data
PLOTUV	makes a stick plot of vector data
SHADE	makes a shaded representation
VIEW	makes a wire frame plot
VECTOR	makes a plot of a vector field

The graphical output command PLOTUV can be used to make stick plots easily, as the following time series example shows.

```
yes? SET DATA coads; SET REGION/X=180/Y=0/L=400:500
yes? PLOT/SET uwnd, vwnd
yes? PPL PLOTUV
```

3 AXES

By default, FERRET displays X- and Y-axes with tics and numeric labels at reasonable intervals and a label for each axis. Time axes are also automatically formatted and used as needed. These axis features can be modified or suppressed using the following FERRET direct controls and PPLUS commands.

3.1 FERRET axis controls

The following qualifiers are used with graphical output commands PLOT, VECTOR, SHADE, and CONTOUR to specify axis limits, tic spacing and possible axis reversal:

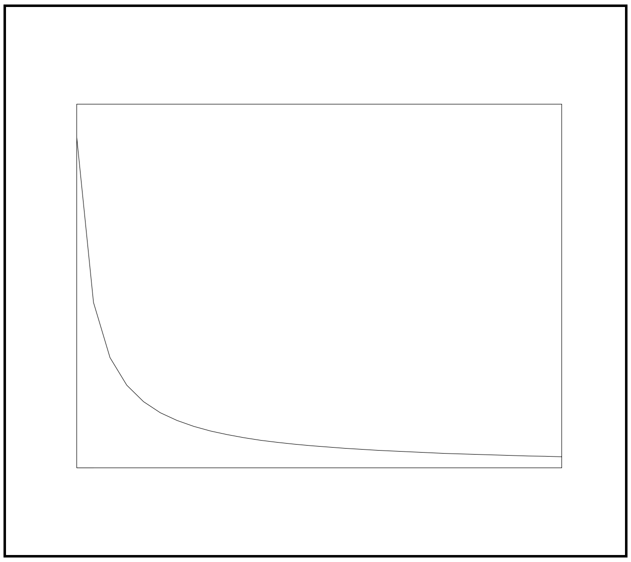
FERRET qualifers
/XLIMITS, /YLIMITS

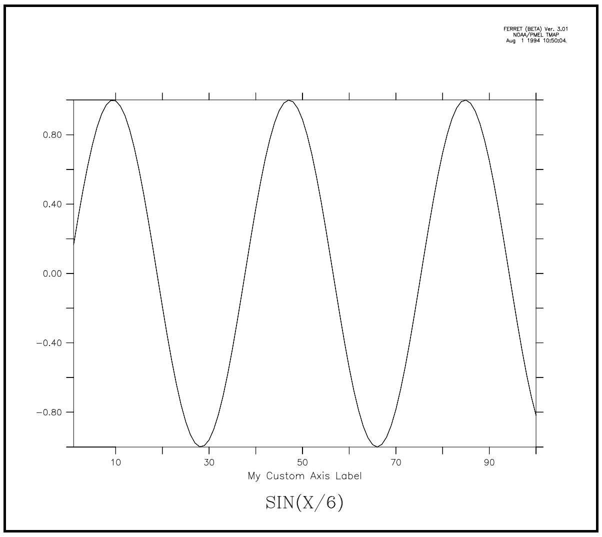
The /XLIMITS and /YLIMITS qualifiers use the syntax /XLIMITS=lo:hi:delta. Tic marks are placed every "delta" units, starting at "lo" and ending at "hi". *Every other* tic mark is labelled. "delta" may be negative, in which case the axis is reversed.

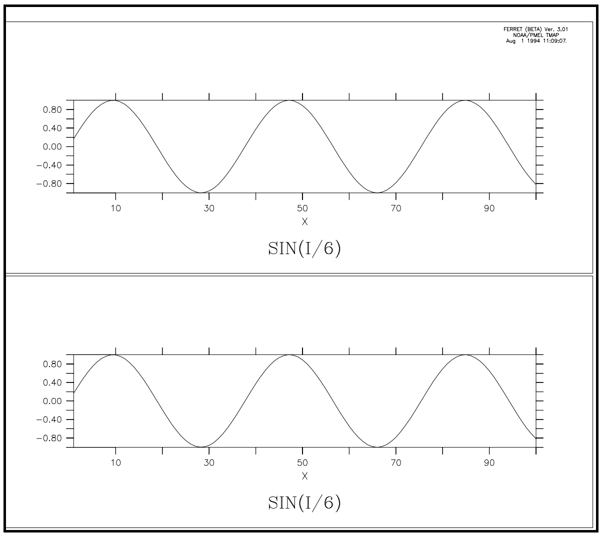
The following arguments to SET MODE and CANCEL MODE determine axis style (e.g. SET MODE CALENDAR:days) :

FERRET arguments
CALENDAR
LATIT_LABEL
LONG_LABEL

See the Commands Reference section of this manual for more information.




FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 10:50:04.
0.80
0.40
0.00
−0.40
−0.80
10
30
50
70
90
My Custom Axis Label
SIN(X/6)


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 11:09:07.
0.80
0.40
0.00
-0.40
-0.80
10
30
50
70
90
X
SIN(I/6)
0.80
0.40
0.00
-0.40
-0.80
10
30
50
70
90
X
SIN(I/6)

FERRET qualifiers
/NOLABELS
/TITLE=

and arguments to SET MODE and CANCEL MODE:

FERRET arguments
ASCII_FONT
CALENDAR
LATIT_LABEL
LONG_LABEL

4.4 PPLUS label commands

FERRET stores the text strings of the following labels in PPLUS symbols. The symbol names are:

symbol name	label
LABTIT	title label
LABX	X axis label
LABY	Y axis label
LABn	nth movable label

As stated above, PPLUS commands regarding movable labels are largely superceded by the FERRET command LABEL and 'GO unlabel n'.

Command	Function
LABS*	makes, removes or alters a movable label
HLABS	sets height of each movable label
RLABS	sets angle for each movable label
LABSET	sets character heights for labels
LLABS	sets start position for and draws a line to a movable label
TITLE*	sets and clears main plot label
XLAB*	sets label of X axis
YLAB*	sets label of Y axis

* issued by FERRET with every relevent plot

Example

This example customizes a plot using PPLUS label controls.

```
yes? PLOT/Z=20/I=1:100/SET_UP   z * sin(i/6)
yes? PPL LABS 4,48,0,0 @p2'lab4'
yes? PPL HLABS 4,.25
yes? PPL LABS/NOUSER 5,0,6.3,-1  *** Magnified SIN function ***
yes? PPL LABSET ,,,.35
yes? PPL PLOT
```

5 COLOR

FERRET and PPLUS use colors stored by index. Storage indices 0 and 1 are used as window background and foreground colors. Indices 1-6 are reserved for lines. As the user makes SHADE and FILL requests, each color is assigned to the next available storage index beginning at 7, and that assignment is automatically "protected" when viewports or color overlays are added.

If your SHADE and FILL commands request more colors than there are storage indices(256), you will be informed with an error message and the color behavior may become unpredictable. For example, if you have multiple viewports defined within a window you may run out of color storage indices. If you are using the same color palette(s) in each viewport, you can free up indices by cancelling the color protections with PPL SHASET RESET. See the examples later in this section for details on removing color protection. Currently, there is no way to ask PPLUS how many colors it is using in a plot.

The following discussion is divided into a treatment of text and line colors, and a discussion of shade and fill color.

5.1 Text and line colors[2]

Line and text colors are regulated by use of storage indices 1-6, each index associated with a default color. It is possible to change the six available line colors with the PPLUS enhancements command COLOR. (See *Plotplus Plus: Enhancements to Plotplus.*) When you create a plot with multiple data lines, FERRET automatically draws each line in a different color. By default, axes, labels and the first data line are all drawn in the same color. You can modify this behavior with the following FERRET and PPLUS commands.

5.1.1 FERRET color controls for lines

Plotted line colors can be set using

```
yes? PLOT/LINE=n
yes? VECTOR/PEN=n
yes? CONTOUR/PEN=n
```

where "n" is an integer between 1 and 18. Type "GO line_samples" in FERRET to see the default line style possibilities using a combination of /LINE= and /SYMBOL=. See command PLOT/LINE in the Commands Reference for more information.

---

[2] In the following discussion, "line color/thickness" is used as equivalent to "line style" for the sake of simplicity. However, if you are using a black and white printer, then the metafile translator will substitute a dash pattern for each line color. See *Plotplus Plus: Enhancements to Plotplus* to see monochrome line styles.

5.1.2 PPLUS text and line color commands

The PPLUS command PEN assigns a color and thickness index to a specified pen. The command takes the form:

```
yes? PPL PEN pen_#, color_thickness
```

where pen_# is the PPLUS pen number and color_thickness is a color and thickness index. PPLUS uses different pens for different tasks. By default, color_thickness index 1 is assigned to pen 0. The following chart may be helpful.

pen number	default color_thickness index	drawing task
0	1 (black or white)	axes and labels
1	1 (black or white)	first data line
2	2 (red)	second data line
3	3 (green)	third data line
4	4 (blue)	fourth data line
5	5 (cyan)	fifth data line
6	6 (magenta)	sixth data line

Note: Whether you plot several data lines simultaneously, or use the /OVERLAY qualifier on your FERRET commands, the color/thickness result will be the same. But the FERRET/PPLUS interaction is different. When FERRET plots multiple data lines simultaneously, PPLUS automatically cycles through pen numbers 1-6. However, if you are using /OVERLAY for additional data lines, FERRET controls the color_thickness assigned to pen 1 and PPLUS draws each overlay line with pen 1.

Pen numbers range from 0 to 6, and color_thickness indices range from 0 to 18. The values 1 to 18 follow the formula:

color_thickness = 6 * (thickness - 1) + color

where thickness ranges from 1 to 3 and color from 1 to 6. Type 'GO line_thickness' in FERRET to see actual colors and thicknesses.

The special color_thickness index 0 refers to the background color, which produces "invisible" lines that can be used as "white-out" for special purposes.

The following PPLUS commands use the color_thickness index.

Command	Function
@Cnnn	uses color_thickness index "nnn" when embedded in a label
PEN	sets color_thickness index for each data line (see chart above)
LEV	sets color_thickness index for contour plot lines

Examples

1) FERRET's default behavior - these two plots will look identical

```
yes? PLOT/i=1:10  1/i, 1/(i+3), 1/i + 1/(10-i)  !3 curves with 3 pens

yes? PLOT/i=1:10 1/i                   !first curve with pen 1
yes? PLOT/OVER/i=1:10 1/(i+3)          !overlay with pen 1 (next index)
yes? PLOT/OVER/i=1:10 1/i+1/(10-i)     !overlay with pen 1 (next index)
```

2) select different colors for pens 0 and 1

```
yes? PLOT/i=1:10/SET 1/i
yes? PPL PEN 1 4        !assign color_thickness 4 to pen 1 (plot curve)
yes? PPL PEN 0 3        !assign color_thickness 3 to pen 0 (axes & labels)
yes? PPL PLOT           !render the plot
yes? PPL PEN 0 1        !reset pen 0 to default color_thickness (not\
                        reset by FERRET as is pen 1)
```

3)better way to do above plot:

```
yes? PLOT/i=1:10/LINE=4/SET 1/i  !include line style with qualifer /LINE
yes? PPL PEN 0 3 ; PPL PLOT
yes? PPL PEN 0 1
```

5.2 Shade and fill colors

Colors specified with the PPLUS SHASET command or in spectrum files (also called palette files) containing pre-defined color palettes define points along an abstract path in RGB color space that runs from 0 to 100 percent. The spectrum file bluescale.spk, for example, contains these lines:

```
  0  0  0  95
100 95 95 95
```

The first number is the percentage distance along the path in color space, and the following numbers are the percents of red, green and blue, respectively. The actual colors used by SHADE or FILL are determined by dividing this abstract color scale into n equal increments, where n is the number of colors, and linearly interpolating between the red, green, and blue values from the neighboring SHASET percentage points.

5.2.1 FERRET shade and fill color controls

By default, FERRET will use the PPLUS spectrum file default.spk for shades and fills (normally default.spk is a Unix soft link to rnb.spk). FERRET comes with many color palettes. The UNIX command "Fenv" lists the environment variable $FER_PALETTE which is a list of paths to be searched for palette files (the palette file names all end in .spk). The UNIX command "Fpalette" allows you to find and examine these files (type "Fpalette -help" at the Unix prompt). You can easily create your own palette files with a text editor.

Use the FERRET qualifier /PALETTE= with FERRET graphical output commands CONTOUR/FILL and SHADE to specify a color palette. See Chapter 6 section "Custom Contouring" for details on the CONTOUR qualifier /LEV, which controls colors and dash patterns, as well as sets contour levels.

FERRET qualifiers	
/PALETTE=	(alias for PPL SHASET SPECTRUM=)
/LEV=	

PALETTE is also a stand-alone command alias; it sets a new default color palette.

Be aware that when you use /PALETTE= in conjunction with /SET_UP, the color spectrum you specify becomes the new default palette; to restore the default palette use command PALETTE with no argument.

5.2.2 PPLUS shade color commands

Command	Function
SHASET	sets colors used by SHADE

SHASET is an enhancement of PPLUS designed for FERRET. You can specify a color spectrum, save a spectrum, change an individual color in the spectrum, or remove the protection (PPL SHASET RESET) for colors already on the screen. See *Plotplus Plus: Enhancements to Plotplus* for more information.

If you need precise control over each individual RGB color on your plot, run 'GO exact_colors' which contains instructions on modifying individual colors in a palette using SHASET.

Examples

1)look at the relief of the earth's surface

```
yes? SET DATA etopo120
yes? SHADE rose              !FERRET's default behavior
yes? SHADE/PAL=land_sea rose  !emphasize land and sea with palette
```

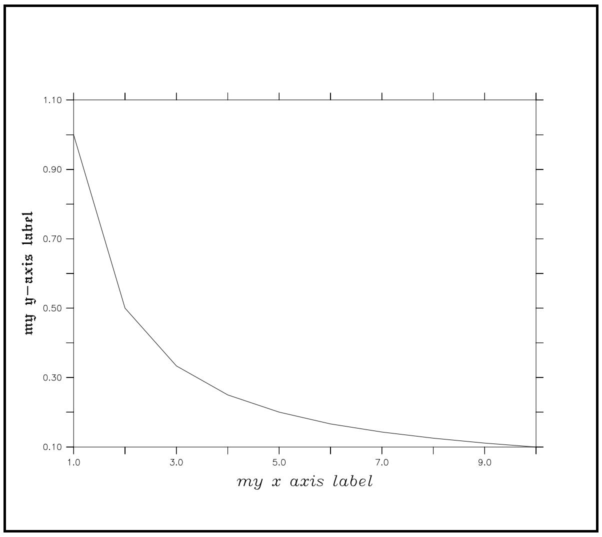
2)Perhaps you would like to compare two topography resolutions. To illustrate what happens when you use more colors than are available, request an excessively large number of levels:

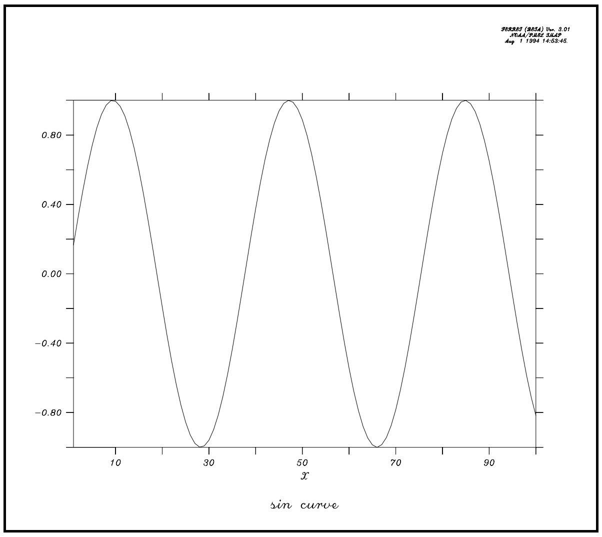
```
yes? SET DATA etopo120
yes? SET REGION/Y=-20:20
yes? SET VIEWPORT UPPER                  !upper half
yes? SHADE/LEV=(-8000,8000,100) rose     !160 colors, default palette
yes? SET VIEWPORT LOWER                  !lower half
yes? SET DATA etopo20                    !high resolution
yes? SHADE/LEV rose[d=etopo20]           !another 160 colors (320 > 256!)
yes? CANCEL VIEWPORT
```

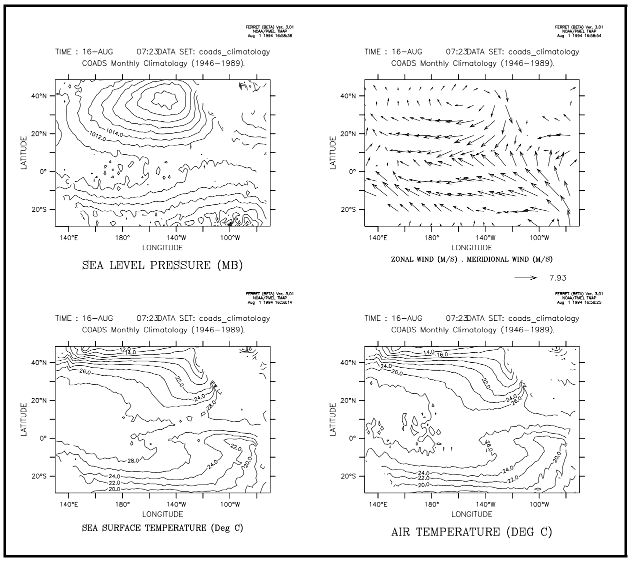
```
PPL+ error: You're attempting to use more colors than are available.
            Using SHASET RESET to re-use protected colors may help.
```

If you reuse the same palette, as in this example, you can issue PPL SHASET RESET after the first plot and plot the second picture without error:

```
yes? SET DATA etopo120
yes? SET REGION/Y=-20:20
yes? SET VIEWPORT UPPER
```


1.10
0.90
0.70
0.50
0.30
0.10
my y-axis label
1.0
3.0
5.0
7.0
9.0
my x axis label


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 14:53:45.
0.80
0.40
0.00
-0.40
-0.80
10
30
50
70
90
X
sin curve


FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 16:58:38
TIME : 16-AUG 07:23DATA SET: coads_climatology
COADS Monthly Climatology (1946-1989).
40°N
20°N
0°
20°S
LATITUDE
140°E
180°
140°W
100°W
LONGITUDE
1014.0
1012.0
SEA LEVEL PRESSURE (MB)
FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 16:58:54
TIME : 16-AUG 07:23DATA SET: coads_climatology
COADS Monthly Climatology (1946-1989).
40°N
20°N
0°
20°S
LATITUDE
140°E
180°
140°W
100°W
LONGITUDE
ZONAL WIND (M/S) , MERIDIONAL WIND (M/S)
7.93
FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 16:58:14
TIME : 16-AUG 07:23DATA SET: coads_climatology
COADS Monthly Climatology (1946-1989).
40°N
20°N
0°
20°S
LATITUDE
12.0
14.0
26.0
22.0
24.0
28.0
28.0
22.0
24.0
20.0
24.0
22.0
20.0
140°E
180°
140°W
100°W
LONGITUDE
SEA SURFACE TEMPERATURE (Deg C)
FERRET (BETA) Ver. 3.01
NOAA/PMEL TMAP
Aug 1 1994 16:58:25
TIME : 16-AUG 07:23DATA SET: coads_climatology
COADS Monthly Climatology (1946-1989).
40°N
20°N
0°
20°S
LATITUDE
14.0
16.0
24.0
26.0
22.0
24.0
26.0
26.0
20.0
24.0
22.0
24.0
22.0
20.0
140°E
180°
140°W
100°W
LONGITUDE
AIR TEMPERATURE (DEG C)

7.1.3 Advanced usage of viewports

For the purposes of defining viewports, a graphics window is considered to have length 1 and height 1. All viewport commands refer to positions relative to the current aspect ratio of the window. Thus,

```
yes? DEFINE VIEWPORT/XLIM=.5,1/YLIM=.5,1  V5
```

will locate the origin of viewport V5 at the middle of the output window regardless of the shape of the window.

The qualifiers /XLIMITS=x1,x2 and /YLIMITS=y1,y2 allow the user to specify a portion of the graphics window to be the defined viewport. The arguments must be values between [0,1] (NOT world coordinates). x1 and x2 indicate the portion of the entire length of the window to be defined as the viewport; y1 and y2 serve an analogous purpose for height.

The /TEXT=n qualfier allows the user control over the shrinkage or enlargement of text on the plot. A value of /TEXT=1 indicates that the text size should be the same as it is on the full screen output. If a value less than 1 is specified the text will shrink. If a value is not specified FERRET chooses a value appropriate to the viewport size. Acceptable values are 0 < n < inf. but only values up to about 2 yield useful results.

7.2 PPLUS layout commands

Command	Function
ORIGIN	sets distance of plot origin from lower left corner
BOX	controls drawing of a box around the plotting area
CROSS	controls drawing of lines through (0, 0) on graph
ROTATE	rotates plot by 90 degrees on screen and plotter
AXLEN	sets axis lengths
SHAKEY	locates the color key
VECKEY	locates the vector key
AXSET	includes/excludes particular axes
SIZE	sets the overall size of the graphics window

8 CONTOURING

8.1 FERRET contour controls

The following qualifiers to the FERRET command CONTOUR allow customization of a contour plot.

Qualfier	Function
/FILL	produces a color-filled contour plot (command FILL is an alias for CONTOUR/FILL)
/LEVELS	specifies contour levels, dash patterns, line thickness and color

/KEY	turns on display of color key for color-filled contour plots (default)
/NOKEY	turns off display of color key for color-filled plots
/LINE	adds contour lines to a color-filled plot (lines replace key)
/PALETTE=	specifies a color palette for color-filled contour plot
/PEN=	sets line style for contour lines (same arguments as PLOT/LINE=. See Chapter 6 section "Text and Line Colors".)

8.1.1 /LEVELS qualifier

The /LEVELS qualifier is a powerful and multi-functional tool. Without arguments, /LEVELS reuses the contour levels from the last CONTOUR or SHADE command. With arguments, the /LEVELS qualifier uses the same syntax as the PPLUS "stand-alone" command LEV (documentation for which is in *PLOT PLUS for FERRET Users Guide).*

The /LEVELS= qualifer takes the form `/LEVELS=levels_descriptor`

Every levels descriptor begins with the following syntax:
`/LEVELS=(lo, hi, delta)` or `/LEVELS=(lo, hi, delta, ndigits)`

Examples

```
/LEVELS=(10,50,5)
/LEVELS=(-20,20,2)
/LEVELS=(33.5,35.0,.025,3)
```

Refinements to the basic levels may be applied using the syntaxes below. If blanks are included, surround the entire levels descriptor in double quotation marks.

1) To request additional levels, simply append additional (lo, hi, delta) and/or (value) specifiers.
   Example: `/LEVELS="(-100,100,10) (100,1000,100) (2000)"`

2) To remove selected levels, append the specifier DEL(lo, hi, delta) or DEL(value).
   Example: `/LEVELS="(-100,100,10) DEL(10)"`

3) To specify the line type as dark (heavy line), append DARK(lo, hi, delta) or DARK(value). Similar syntax can be applied to LINE (solid, thin) or DASH.
   Example: `/LEVELS="(-100,100,10) DARK(100) DARK(-100)"`

4) To specify the color_thickness index of contour lines (see Chapter 6 section "Color" for a discussion of color_thickness indices), append PEN(lo, hi, delta, index).
   Example: `/LEVELS="(-100,100,10) PEN(-100,-10,10,2) PEN(10,100,10,4)"`

8.2 PPLUS contour commands

Command	Function
CONPRE	sets prefix for contour labels (usually a font, e.g. "@TR")
CONPST	sets suffix for contour labels (usually units, e.g. "cm")
CONSET	controls various aspects of contour labels and curves (see below)

CONSET is a modified version of the PPLUS command. Two new parameters have been added - "spline_tension" and "draftsman". "spline_tension" controls a spline fitting routine for contour lines, and is primarily used in conjunction with the narc parameter. The new parameter "draftsman" enables the user to specify horizontally oriented contour labels (draftsman style) or the default, labels oriented along contour lines. Arguments for CONSET are as follows:

```
CONSET hgt,nsig,narc,dashln,spacln,cay,nrng,dslab,spline_tension,draftsman
```

hgt = height of contour labels. default=.08 inches
nsig = no. of significant digits in contour labels. default=2
narc = number of line segments to use to connect contour points default=1
dashln = dash length of dashes mode default=.04 inches
spacln = space length of dashes mode default=.04 inches
cay This argument has no effect on gridded data. It is documented in *PLOT PLUS for FERRET Users Guide* and also in the discussion of objective analysis under command USER in the Commands Reference section of this manual.
nrng This argument has no effect on gridded data. It is documented in *PLOT PLUS for FERRET Users Guide* and also (as parameter "rng") in the discussion of objective analysis under command USER in the Commands Reference section of this manual.
dslab= nominal distance between labels on a contour line. default=5.0 inches.
spline_tension = a real value that affects the fit of the contour line. default=0.
This parameter is only applied if narc is greater than 1. Otherwise, straight lines are drawn between data points and no interpolated points are contoured. This value indicates the curviness desired.

abs(spline_tension) is nearly zero (e.g. .01). The resulting curve is approximately a cubic spline.
abs(spline_tension) is large (e. g. 10.). The resulting curve is nearly a polygonal line.
spline_tension = 0. The resulting curve is a cubic spline (the default algorithm in ppl).

A typical value for spline_tension is 1, and the typical useful range of values is .01 to 10.
draftsman = a real value that controls the label format. default = 0.

0. = original label style - labels oriented along contour arcs
> 0. = draftsman label style - labels oriented horizontally on the page
< 0. = reserved for future use

Examples

Run the demonstration on custom contouring for many examples of label styles, contour line styles (color, thickness dash pattern), and contour intervals - `yes? GO custom_contour`

1) Color-filled contour plot of sea surface temperature

```
yes? SET DATA coads_climatology
yes? SET REGION/@t/l=6        !specify tropical Pacific, month 6
yes? SET VIEWPORT upper
yes? FILL sst                 !filled contour plot
yes? SET VIEWPORT lower
yes? FILL/LINE sst            !make the plot with contour lines
```

2) Let's improve on the earlier example (5.2.2) of shaded bathymetry with blue palette

```
yes? SET DATA ETOPO60
yes? LET/TITLE="Surface relief x1000 (meters)" r1000 rose/1000
yes? FILL/PAL=ocean_blue/LINE/LEV=(-8,-1,1,-3)LINE(-8,-1,1,-3)/PEN=4 r1000
```

Here is a breakdown of the final command line:

FILL	color-filled contour plot (alias for CONTOUR/FILL)
PAL	specifies color palette for fill colors
LINE	specifies that contour lines be overlaid on the filled plot (in lieu of a key)
LEV	first arg specifies contour levels without numerical labels, next requests solid lines (dashed lines are the default for negative contour values)
PEN	assigns line style 4 (blue) to contour lines

Chapter 7: COMPUTING ENVIRONMENT

1 SETTING UP AN ACCOUNT

This discussion assumes that FERRET is already installed on your system. Installation documentation is available separately from node abyss.pmel.noaa.gov.

STEP 1
Execute interactively or add to your .login file the Unix C-shell command

```
% source /usr/local/ferret_paths
```

(Note: If this command doesn't work consult your system manager, who may have located ferret_paths in a different directory.)

The FERRET program requires access to several files and directories. These Unix paths are stored in environment variables defined by the file "ferret_paths". Your Unix account must be "made aware" of where the FERRET utilities are located. This is done by adding to the definition of your environment variable PATH the directory "$FER_DIR/bin". Unless your system manager has modified the typical setup this will occur automatically when you execute the above command.

STEP 2 (personal customizations - optional)
Execute the "cp" command below:

```
% cp $FER_DIR/bin/my_ferret_paths_template \
     $HOME/my_ferret_paths
```

Then use a text editor to customize my_ferret_paths. Instructions are inside the file.

Some of the FERRET environment variables identify files and directories that are integral to the FERRET program but others identify files that you may maintain - your data files, GO scripts and palette files, for example. (The environment variables that you may want to customize are discussed at the end of this section). To assist in customizing the FERRET environment variables the template file in the "cp" command, above, has been provided. The file is self-explanatory.

STEP 3
Execute the command below interactively or add it to your .login file.

```
% setenv DISPLAY node:0.0        e.g., % setenv DISPLAY anorak:0.0
```

This command sets the environment variable "DISPLAY" to point to the workstation console or X-terminal where you want FERRET graphical output displayed. In the example above, graphical output is directed to the screen of workstation "anorak".

2 FILES AND ENVIRONMENT VARIABLES USED BY FERRET

.ferret - the FERRET initialization file. This optional file holds a list of FERRET commands that will be executed immediately each time FERRET is started, permitting FERRET to be tailored to individual needs and styles. The file must be located in your $HOME (login) directory. A simple way to set up such a file is to enter FERRET, enter the commands that you want executed each time you enter FERRET, exit FERRET and rename the file "ferret.jnl" to ".ferret". Thereafter, all commands in ".ferret" will be executed automatically whenever you enter FERRET.

The following environment variables are defined in the file ferret_paths:

FER_DATA - a list of directories to be searched to locate data files. Usually this list includes ".", the current directory, and $FER_DSETS/data, a directory of sample data sets provided with FERRET. Your system manager may have set this variable to include other data areas as well. This is the list of directories searched to locate NetCDF files.

FER_DESCR - a list of directories to be searched to locate descriptor files. Descriptors are required by FERRET to access a data sets that are in FERRET's "GT" (grids at timesteps) or "TS" (time series) formats. Usually this list includes ".", the current directory, and $FER_DSETS/descr, a directory of sample descriptors provided with FERRET.

FER_GRIDS - a list of directories to be searched to locate grid definition files. Data sets will usually have a grid definition file associated with them so that the grids on which the data are defined may be known.

FER_DIR - top directory of the FERRET distribution on your system.

FER_DSETS - directory of sample data sets provided with the FERRET distribution.

FER_PALETTE - a list of directories to be searched to locate palette files. Usually this list includes "." and $FER_DIR/ppl.

FER_GO - a list of directories to be searched to locate GO scripts. This list usually includes ".", $FER_DIR/go, $FER_DIR/examples (demonstrations and tutorial), and $FER_DIR/contrib (user contributions demonstrating various applications; accuracy not guaranteed).

3 MEMORY USE

FERRET indicates memory problems by issuing the error message "insufficient memory." If memory is a problem running FERRET the following suggestions may help:

1) Make sure that the region is fully defined (i.e. check SHOW REGION and check the region qualifiers of your command). When the region along some axis is not specified FERRET defaults to the full span of the data along that axis and is unable to optimize memory usage. This is the most common cause of "insufficient memory" errors.

2) Use CANCEL MEMORY whenever you are sure that the data referenced thus far by FERRET will not be referenced again. This is particularly appropriate to batch procedures that use FERRET. This eliminates any memory fragmentation that may be left by previous commands.

3) Use CANCEL MODE SEGMENTS to minimize the memory usage by graphics (on a few X-window systems this may prevent windows from being restored after they are obscured).

4) When using DEFINE VARIABLE (alias LET) avoid embedding upper and lower axis bounds within the variable definition. FERRET cannot split up large calculations along axes when the limits are fixed in the definition. For example,

```
yes? LET V2=TEMP/10
yes? PLOT/K=1:10 V2
```

is preferable to

```
yes? LET V2=TEMP[K=1:10]/10
yes? PLOT V2
```

5) Try to group together calculations that are on smaller dimensioned objects. For example, the expression VAR[i=1:100, j=1:100]*2*PI will make less efficient use of cpu and memory than the expression VAR[i=1:100, j=1:100]*(2*PI). The former multiplies each of the 10000 points of VAR by 2 and then performs a second multiplication of the 10000 result points by PI. The latter computes the scalar 2*PI and uses it only once in multiplying the 10000 points of VAR.

4 HARD COPY AND METAFILE TRANSLATION

4.1 Hard copy

To obtain hard copy of plots produced by FERRET, follow these steps:

1) Within FERRET, enter the command

```
yes? SET MODE METAFILE
```

This tells FERRET to generate a graphic metafile (GKSM format) for each plot created thereafter. To stop making the metafiles type

```
yes? CANCEL MODE METAFILE
```

2) Produce each plot as you would normally. Each new plot on your screen generates an additional file named "metafile.plt.~n~" where "n" will be incremented for each metafile. Overlay commands do not produce additional metafiles. (The metafile name may be set by the SET MODE METAFILE command.)

3) After exiting from FERRET use the command Fprint.
   Note: If it is necessary to use Fprint without exiting FERRET, then issue the command `yes? PPL CLSPLT`. This will close the current metafile.

Fprint is a script which translates metafiles generated by FERRET. It uses the program 'gksm2ps' and is intended to simplify sending plots to printers, to an output file only, or to a workstation screen. As of FERRET version 3.1, output to the screen is not implemented.

For monochrome printers the metafile translator, gskm2ps, uses different line styles (dash-dot patterns) rather than colors for different lines. See Appendix I of *Plotplus Plus: Enhancements to Plotplus* for a complete list of line styles for monochrome devices.

The script translates metafiles to PostScript or Xwindow output. Your system manager should customize the script at your site to permit your specification of the actual printers you have as output devices. Fprint uses standard Unix command line syntax.

```
Fprint [-h] [-P printer || -o file_name || -X]
        [-p orient] [-# n] [-l line] [-R] metafile(s)
```

Options

-h	prints help on your terminal screen.
-P printer	Routes output to named printer.
-o file_name	Routes output to named disk postscript file.
-X	Routes output to your workstation screen. (This option is not implemented as of FERRET version 3.1.)
-p orient	The page orientation option determines whether the plot will be placed on the page in landscape format, with the horizontal side longer than the vertical, or portrait, with the vertical side longer. Valid option values are "landscape" and "portrait". The default behavior is to orient the plot to best fit the page.

-# n	Specifies number of copies (n).
-l line	This option lets you specify line styles. Valid options are "ps" and "cps". "ps" uses dot-dashed line types; "cps" uses colored lines. The default is "ps" for monochrome printers and "cps" for color printers.
-R	Turns off the default behavior of the metafile translator to append a date stamp to metafile names when they are sent to a printer or a disk file. The default action is intended to distinguish metafiles that have been printed out; this option keeps the metafile names unmodified.

Examples

```
% Fprint metafile.plt
```
renders "metafile.plt" on the default printer identified by the environment variable PRINTER.

```
% Fprint -P myprinter metafile.plt*
```
renders all versions of "metafile.plt" on printer myprinter.

```
% Fprint -o my_plot.ps metafile.plt.~1~
```
writes plot "metafile.plt.~1~" to a postscript file named "my_plot.ps".

4.2 Metafile translation

The command `gksm2ps' allows you to control the translation of the device-independent metafiles made by FERRET into device-specific output files. 'gksm2ps' was written by Larry Oolman at the University of Wyoming and modified at NOAA for use with FERRET. The `gksm2ps' command uses standard Unix command line syntax. See usage hints provided by the -h option.

```
gksm2ps [-h] [-p landscape||portrait] [-l ps||cps] [-d cps||phaser] \
        [-X || -o <ps_output_file>] [-R] [-a] [-g WxH+X+Y] file(s)
```

Options

-h	prints help message.
-p orient	The page orientation option determines whether the plot will be placed on the page in landscape format, with the horizontal side longer than the vertical, or portrait, with the vertical side longer. The default is to orient the plot to best fit the page.
-l line	This option permits specification of line styles in the hardcopy plot. Valid options are "ps" (the default) and "cps". "ps" renders lines as solid and

	dot-dashed and is suited for monochrome printers. "cps" renders lines in color.
-d devtype	The target device type of the translator. If the -d option is omitted and output is to a file gksm2ps will use devtype 'ps'. Valid devtype values: cps - color PostScript, phaser - Tektronix Phaser PX. The phaser is a PostScript printer, but it uses transfer sheets which reduce the usable page size.
-X	Sends the output to your Xwindow for preview (this option is not implemented as of FERRET version 3.1).
-o ofile	The output will be directed to the file ofile. Omit both this and the device type option when directing output to your workstation screen with -X. If neither -o nor -X is specified, gksm2ps creates a postscript file in the current directory called "gksm2ps_output.ps".
-a	Makes the plot the size of the original plot as specified in PPLUS inches (absolute size), rather than fitting the plot to the page (the default behavior).
-g WxH+X+Y	Sizes and fits the plot on the page according to the geometry specification WxH+X+Y, where W is the width, H the height, and X and Y the offset from the lower left corner. Units are points (72pts = 1 inch).
-R	Turns off the default behavior of the metafile translator to append a date stamp to metafile names when they are sent to a printer or a disk file. The default action is intended to distinguish metafiles that have been printed out; this option keeps the metafile names unmodified.

If the user does not specify an output option (-o or -X) gksm2ps translates the metafile and produces a PostScript file called gksm2ps_output.ps. After translation by gksm2ps, metafiles are renamed with a date stamp unless -R was specified. To get hard copy printed, the output PostScript file needs to be sent to the appropriate printer.

5 OUTPUT FILE NAMING

FERRET uses a file naming scheme to differentiate successive graphic metafiles and journal files. The scheme is styled after the gnu (Free Software Foundation) emacs editor. The scheme appends numbers to the end of the file name as in the following examples:

```
metafile.plt.~2~
```

```
metafile.plt.~12~
metafile.plt
```

The third example, `metafile.plt' with no suffix appended, is the most recent file. When the next successive file is created, this file will have the suffix ".~nnn~" appended to its name. 'nnn' is the current highest file suffix number plus one.

Two Unix tools are provided to assist with managing multiple file suffix numbers:

Fpurge - removes all but the current version of the named file (that is, all but the most recent).
Example: `% Fpurge ferret.jnl`

Fsort - sorts the versions of a file into increasing numerical order
Example: ``% Fprint `Fsort metafile.plt*` ``

See Chapter 1 section "Unix tools" for further information.

6 INPUT FILE NAMING

There are several FERRET commands that use filenames. These include:

```
GO filename
SET DATA filename
LIST/FILE=filename  (do not use relative versions (below) with LIST)
USER/FILE=filename
SET MODE META filename
SET MODE JOURNAL filename
SET MODE PPLLIST filename
```

The filename specified can be just the filename itself, or it can include the path to the file. For example:

```
GO ferret.jnl
```
or
```
GO "/home/disk1/jnl_files/far_side.jnl"
```

Note that if the path is specified as part of the filename, the entire name must be enclosed in quotation marks.

6.1 Relative version numbers

Under some circumstances (see the GO command) a special syntax called "relative version numbers" will apply. If a filename has a version value of zero or less its value is interpreted relative to the current highest version number.

For example, if the current directory contains the files

```
ferret.jnl   ferret.jnl.~1~   ferret.jnl.~2~   ...  ferret.jnl.~99~
```

then the filename ferret.jnl.~0~ refers to ferret.jnl and the filename ferret.jnl.~-1~ refers to ferret.jnl.~99~.

The syntax for relative version numbers is quite flexible. For example, if the desired file is ferret.jnl.~99~, both of the following are valid:

```
yes? GO ferret.jnl.~-1~      or
yes? GO ferret.jnl~-1
```

Chapter 8: CONVERTING TO NetCDF

1 OVERVIEW

The Network Common Data Format (NetCDF) is an interface to a library of data access routines for storing and retrieving scientific data. NetCDF allows the creation of data sets which are self-describing and network-transparent. NetCDF was created under contract with the Division of Atmospheric Sciences of the National Scientific Foundation and is available from the Unidata Program Center in Boulder, Colorado (on Internet: unidata.ucar.edu).

This chapter provides directions for creating NetCDF data files. In addition to the documentation provided here, refer to the NetCDF User's Guide, published by Unidata Program Center, for further guidance. A user who uses and creates NetCDF files within the FERRET environment needs no additional software.

2 SIMPLE CONVERSIONS USING FERRET

In straightforward conversion operations where ASCII or unformatted binary data files are already readable by FERRET, the conversion to direct access, self-describing NetCDF formatted data can be accomplished by FERRET itself. The following set of examples illustrates these procedures:

Example 1
Consider an ASCII file uv.data, with two variables, u and v, defined on a grid 360 by 180. The following set of commands will properly read in u and v and convert them to a NetCDF formatted data set:

```
yes? DEFINE AXIS/x=1:360:1 xaxis
yes? DEFINE AXIS/y=1:180:1 yaxis
yes? DEFINE GRID/x=xaxis/y=yaxis uv_grid
yes? FILE/GRID=uv_grid/BAD=-999/VAR="u,v"  uv.data
yes? SAVE/FILE=uv.cdf u,v
```

See command DEFINE AXIS in the Commands Reference. See Chapter 4 for setting up formatted latitude, longitude and time axes.

Example 2
Consider now two separate ASCII files, u.data and v.data, defined on a grid 360 by 180. The following set of commands will properly read in u and v and convert them to a single NetCDF formatted data set:

```
yes? DEF AXIS/x=1:360:1 xaxis
yes? DEF AXIS/y=1:180:1 yaxis
yes? DEF GRID/x=xaxis/y=yaxis uv_grid
yes? FILE/GRID=uv_grid/BAD=-999/VAR=u u.data
yes? FILE/GRID=uv_grid/BAD=-999/VAR=v v.data
```

```
yes? SAVE/FILE=uv2.cdf u[D=1]
yes? SAVE/APPEND/FILE=uv2.cdf v[D=2]
```

Example 3 - multiple time steps

Consider 12 ASCII files, uv.data1 to uv.data12, each defined on the same grid as above but each representing a successive time step. The following set of commands illustrates how to save these data into a single NetCDF data set (time series):

```
yes? DEF AXIS/x=1:360:1 xaxis
yes? DEF AXIS/y=1:180:1 yaxis
yes? DEF AXIS/t=1:1:1 taxis1
yes? DEF GRID/x=xaxis/y=yaxis/t=taxis1 uv_grid1
yes? FILE/GRID=uv_grid1/BAD=-999/VAR="u,v" uv.data1
yes? SAVE/FILE=uv1_12t.cdf u,v
yes? CANCEL DATA uv.data1
yes? DEF AXIS/t=2:2:1 taxis1
yes? FILE/GRID=uv_grid1/BAD=-999/VAR="u,v" uv.data2
yes? SAVE/APPEND/FILE=uv1_12t.cdf u,v
. . .
```

and so on, redefining the time axis to be 3:3:1, 4:4:1, ... each time a new file is set.

Example 4 - multiple slabs (FERRET version 4.0)

The procedure used in example 3, above, is possible because NetCDF files can be extended along the time axis. In order to append multiple *levels* (Z axis), the NetCDF file must first be created including all of its vertical levels (the levels initially are filled with a missing data flag).

Consider 5 ASCII files, uv.data1 to uv.data5, each defined on the same grid as above but each representing a successive vertical level. The following set of commands illustrates how to save these data into a single NetCDF data set:

```
yes? DEF AXIS/x=1:360:1  xaxis
yes? DEF AXIS/y=1:180:1  yaxis
yes? DEF AXIS/Z=0:100:25/DEPTH  zaxis
yes? DEF GRID/X=xaxis/Y=yaxis/Z=zaxis uv_grid
yes? DEF AXIS/Z=0:0:1  zaxis1
yes? DEF GRID/LIKE=uv_grid/Z=zaxis1  uv_grid1
yes? FILE/GRID=uv_grid1/BAD=-999/VAR="u,v"  uv.data1
yes? LET/TITLE="My U data"  u1 = u[G=uv_grid]
yes? LET/TITLE="My V data"  v1 = v[G=uv_grid]
yes? SAVE/FILE=uv1_5z.cdf/KLIMITS=1:5  u1, v1
yes? DEF AXIS/Z=25:25:1  zaxis1
yes? FILE/GRID=uv_grid1/BAD=-999/VAR="u,v"  uv.data2
yes? SAVE/FILE=uv1_5z.cdf/APPEND  u1,v1
. . .
```

The NetCDF utilities "ncdump" and "ncgen" can also be combined with a text editor to make final refinements to the NetCDF files created by SAVE. (These utilities are not provided with the FERRET distribution; they can be obtained from unidata.ucar.edu.) Here is a simple example that removes all "history" attributes from a NetCDF file using pipes and the Unix "grep" utility:

```
% ncdump old_file.cdf | grep -v history | ncgen -o new_file.cdf
```

3 WRITING A CONVERSION PROGRAM

There are three steps required to convert data to NetCDF if your data is not already readable by FERRET:

1. Create a CDL (the ASCII NetCDF Description Language) file that describes the axes, grids, and variables of the desired output data set.

2. Convert this CDL file into a NetCDF file with the ncgen utility.

3. Create a program that will read your particular data and insert them into the NetCDF file.

The file converting_to_netcdf.f which is located in the FERRET documentation directory ($FER_DIR/doc) contains a complete description and example of these 3 steps. The remainder of this section provides further details.

3.1 The CDL file

A CDL file consists of three sections: Dimensions, Variables, and Data. All of the following text in `Courier font` constitutes a real CDL file; it can be copied verbatim and used to generate a NetCDF file. The full text of this file is included with the FERRET distribution as $FER_DIR/doc/converting_to_netcdf.basic.

```
netcdf converting_to_netcdf.basic {
```

3.1.1 Dimensions

In this section, the sizes of the grid dimensions are specified. One of these dimensions can be of "unlimited" size (i.e. it can grow).

```
dimensions:

  lon   = 160 ;     //  longitude
  lat   = 100 ;     //  latitude
  depth = 27 ;      //  depth
  time  = unlimited ;
```

These are essentially parameter statements that assign certain numbers which will be used in the Variables section to define axes and variable dimensions. The "//" is the CDL comment syntax.

3.1.2 Variables

Variables, variable attributes, axes, axis attributes, and global attributes are specified.

```
variables:

   float temp(time, depth, lat, lon) ;
         temp: long_name = "TEMPERATURE" ;
         temp: units     = "deg. C" ;
         temp: _FillValue = 1E34 ;
   float salt(time, depth, lat, lon) ;
         salt: long_name = "(SALINITY(ppt) - 35) /1000" ;
         salt: units     = "frac. by wt. less .035" ;
         salt: _FillValue = -999. ;
```

The declaration "float" indicates that the variable is to be stored as single precision, floating point (32 bit IEEE representation). The declarations "long" (32 bit integer), "short" (16 bit integer), "byte" (8 bit integer) and "double" (64 bit IEEE floating point) are also supported by FERRET. Note that although these data types may result in smaller files, they will not affect FERRET's memory usage, as all variables are converted to "float" internally as they are read by FERRET.

The _FillValue attribute informs FERRET that any data value matching this value is a missing (invalid) data point. For example, an ocean data set may label land locations with a value such as 1E34. By identifying 1E34 as a fill value, FERRET knows to ignore points matching this value. The attribute "missing_value" is similar to "_FillValue" when reading data; but "_FillValue" also specifies a value to be inserted into unspecified regions during file creation. You may specify two distinct flags for invalid data in the same variable by using "_FillValue" and "missing_value" together.

FERRET variables may have from 1 to 4 dimensions. If any of the axes have the special interpretations of: 1) latitude, 2) longitude, 3) depth, or 4) time (date), then the relative order of those axes in the CDL variable declaration must be T, then Z, then Y, and then X, as above. Any of these special axes can be omitted and other axes (for example, an axis called "distance") may be inserted between them.

```
axis definitions:

   float lon(lon) ;
         lon: units = "degrees";
   float lat(lat) ;
         lat: units = "degrees";
   float depth(depth) ;
         depth: units = "meters";
   float time(time) ;
         time: time_origin = "1-JAN-1972";  date associated with t=0
         time: units = "seconds";
```

Axes are distinguished from other 1-dimensional NetCDF variables by their variable names being identical to their dimension names. Special axis interpretations are inferred by FERRET through a variety of "clues".

Date/time axes are inferred by units of "years", "months", days", "hours", "minutes", or "seconds", or by axis names "time", "date", or "t" (case-insensitive). Calendar date formatting requires the "time_origin" attribute as well as a valid time unit.

Vertical axes are identified by axis names containing the strings "depth", "elev", or "z", or by units of "millibars". Depth axes are positive downward by default. The attribute positive="down" can also be used to create a downward-pointing axis.

Latitude axes are inferred by units of "degrees" or "latitude" with axis names containing the strings "lat" or "y". Longitude axes are inferred by units of "degress" or "longitude" with axis names containing the strings "lon" or "x".

Global attributes, or attributes which apply to the entire data set, can be specified as well.

```
global attributes:
        :title = "NetCDF Example";
        :message =  "This message will be displayed when the CDF file is
                    opened by FERRET";
        :history =  "Documentation on the origins and evolution of this data
                    set or variable";
```

3.1.3 Data

In this section, values are assigned to grid coordinates and to the variables of the file. Below are 100 latitude coordinates entered (in degrees) into the variable axis "lat", 160 longitude coordinates in "lon", and 27 depth coordinates (in meters) in "depth". Longitude coordinates must be specified with 0 at Greenwich, continuous across the dateline, with positive eastward (modulo 360).

```
data:

lat=
-28.8360729218,-26.5299491882,-24.2880744934,-22.1501560211,-20.151357650,
-18.3207626343,-16.6801033020,-15.2428140640,-14.0134353638,-12.987424850,
-12.1513509750,-11.4834814072,-10.9547319412,-10.5299386978,-10.169393539,
-9.8333206177,-9.4999876022,-9.1666536331,-8.8333196640,-8.4999856949,
-8.1666526794,-7.8333187103,-7.4999847412,-7.1666512489,-6.8333182335,
-6.4999852180,-6.1666517258,-5.8333182335,-5.4999852180,-5.1666517258,
-4.8333187103,-4.4999852180,-4.1666517258,-3.8333187103,-3.4999852180,
-3.1666517258,-2.8333184719,-2.4999852180,-2.1666519642,-1.8333185911,
-1.4999852180,-1.1666518450,-0.8333183527,-0.4999849498,-0.1666515470,
0.1666818559,0.5000152588,0.8333486915,1.1666821241,1.5000154972,
1.8333489895,2.1666824818,2.5000159740,2.8333494663,3.1666829586,
3.5000162125,3.8333497047,4.1666831970,4.5000162125,4.8333497047,
5.1666831970,5.5000162125,5.8333497047,6.1666827202,6.5000162125,
6.8333497047,7.1666827202,7.5000166893,7.8333501816,8.1666841507,
8.5000181198,8.8333511353,9.1666851044,9.5000190735,9.8333530426,
10.1679363251,10.5137376785,10.8892869949,11.3138961792,11.8060989380,
12.3833675385,13.0618314743,13.8560228348,14.7786512375,15.8403968811,
17.0497493744,18.4128704071,19.9334945679,21.6128730774,23.4497566223,
25.4404067993,27.5786647797,29.8560409546,32.2618522644,34.7833900452,
37.4061241150,40.1139259338,42.8893203735,45.7137718201,48.5679702759;
lon=
```

```
130.5,131.5,132.5,133.5,134.5,135.5,136.5,137.5,138.5,139.5,140.5,141.5,14
2.5,143.5,144.5,145.5,146.5,147.5,148.5,149.5,150.5,151.5,152.5,153.5,154.
5,155.5,156.5,157.5,158.5,159.5,160.5,161.5,162.5,163.5,164.5,165.5,166.5,
167.5,168.5,169.5,170.5,171.5,172.5,173.5,174.5,175.5,176.5,177.5,178.5,17
9.5,180.5,181.5,182.5,183.5,184.5,185.5,186.5,187.5,188.5,189.5,190.5,191.
5,192.5,193.5,194.5,195.5,196.5,197.5,198.5,199.5,200.5,201.5,202.5,203.5,
204.5,205.5,206.5,207.5,208.5,209.5,210.5,211.5,212.5,213.5,214.5,215.5,21
6.5,217.5,218.5,219.5,220.5,221.5,222.5,223.5,224.5,225.5,226.5,227.5,228.
5,229.5,230.5,231.5,232.5,233.5,234.5,235.5,236.5,237.5,238.5,239.5,240.5,
241.5,242.5,243.5,244.5,245.5,246.5,247.5,248.5,249.5,250.5,251.5,252.5,25
3.5,254.5,255.5,256.5,257.5,258.5,259.5,260.5,261.5,262.5,263.5,264.5,265.
5,266.5,267.5,268.5,269.5,270.5,271.5,272.5,273.5,274.5,275.5,276.5,277.5,
278.5,279.5,280.5,281.5,282.5,283.5,284.5,285.5,286.5,287.5,288.5,289.5;
depth=
5.0,15.0,25.0,35.0,45.0,55.0,65.0,75.0,85.0,95.0,106.25,120.0,136.25,155.0
,177.5,205.0,240.0,288.5,362.5,483.5,680.0,979.5,1395.5,1916.0,2524.0,3174
.0,3824.0;  }
```

To use this CDL file type:

```
% ncgen -o my_data.cdf converting_to_netcdf.basic
```

This will will create a file called "my_data.cdf" to which data can be directed (see next section).

Another NetCDF command, "ncdump", can be used to generate a CDL file from an existing NetCDF file. The command

```
% ncdump -h my_data.cdf
```

will list the CDL representation of a preexisting my_data.cdf without the Data section, while

```
% ncdump my_data.cdf
```

will list the CDL file with the Data section. The command

```
% ncdump -c my_data.cdf
```

will create a CDL file in which only coordinate variables are included in the Data section. The listed output can be redirected to a file as in the command

```
% ncdump -c my_data.cdf > my_data.cdl
```

3.2 Directing data to a CDF file

The following is an example program which can be used on-line to convert existing data sets into NetCDF files. It also should provide guidance on sending data generated by numerical models directly to NetCDF files. This program assumes you have created the NetCDF file as described in the previous section. It is included in the distribution as $FER_DIR/doc/converting_to_netcdf.f.

```
     program converting_to_netcdf

c written by Dan Trueman
c updated 4/94 *sh*

c This program provides a model for converting a data set to NetCDF.
c The basic strategy used in this program is to open an existing NetCDF
c file, query the file for the ID's of the variables it contains, and
c then write the data to those variables.

c The output NetCDF file must be created **before** this program is run.
c The simplest way to do this is to cd to your scratch directory and
c    % cp $FER_DIR/doc/converting_to_netcdf.basic converting_to_netcdf.cdl
c and then edit converting_to_netcdf.cdl (an ASCII file) to describe YOUR
data
c set.  If your data set requires unequally spaced axes, climatological time
c axes, staggered grids, etc. then converting_to_netcdf.supplement may be
c a better starting point then the "basic" file used above.
c After you edit converting_to_netcdf.cdl then create the NetCDF file with
c the command
c    % ncgen -o converting_to_netcdf.cdf converting_to_netcdf.cdl

c Now we will read in **your** data (gridded oceanic temperature and
c salt in this example) and write it out into the NetCDF file
c converting_to_netcdf.cdf.  Note that the axis coordinates can be written
c out exactly the same methodology - including time step values (as below).
*************************************************************************
c An alternative to modifying this program is to use the command:

c     ncgen -f converting_to_netcdf.cdl

c This will create a large source code to which select lines can
c be added to write out your data.
*************************************************************************
c To compile and link converting_to_netcdf.f, use:

c     f77 -o converting_to_netcdf converting_to_netcdf.f -lnetcdf

*************************************************************************
c include file necessary for NetCDF

      include 'netcdf.inc'    ! may be found in $FER_DIR/fmt/cmn
*************************************************************************
c parameters relevant to the data being read in
c THESE NORMALLY MATCH THE DIMENSIONS IN THE CDL FILE
c (except nt which may be "unlimited")

      integer imt, jmt, km, nt, lnew, inlun
      parameter (imt=160, jmt=100, km=27, nt=5)

c imt is longitude, jmt latitude, km depth, and nt number of time steps
*************************************************************************
c variable declaration

      real temp(imt,jmt,km),salt(imt,jmt,km),time_step

      integer cdfid, rcode
c           ** cdfid = id number for the NetCDF file my_data.cdf
c     ** rcode = error id number

      integer tid, sid, timeaxid
c           ** tid = variable id number for temperature
c           ** sid = variable id number for salt
```

```
c           ** timeaxid = variable id for the time axis
      integer itime
c           ** itime = index for do loop
**************************************************************************
c dimension corner and step for defining size of gridded data

      integer corner(4)
      integer step(4)

c corner and step are used to define the size of the gridded data
c to be written out.  Since temp and salt are four dimensional arrays,
c corner and step must be four dimensions as well.  In each output
c to my_data.cdf within the do loop, the entire array of data (160 long.
c pts, 100 lat. pts., 27 depth pts.) will be written for one time step.
c Corner tells NetCDF where to start, and step indicates how many steps
c in each dimension to take.

      data corner/1, 1, 1, -1/      ! -1 is arbitrary; the time value
                                    ! of corner will be initialized
                                    ! within the do loop.

      data step/imt, jmt, km, 1/    ! NOT /1, km, jmt, imt/

c ***NOTE*** Since Fortran and C access data differently, the order of
c the variables in the Fortran code must be opposite that in the CDL
c file.  In Fortran, the first index varies fastest while in C, the
c last index varies fastest.
***************************************************************************
c initialize cdfid by using ncopn

      cdfid = ncopn('converting_to_netcdf.cdf', ncwrite, rcode)
      if (rcode.ne.ncnoerr) stop 'error with ncopn'

***************************************************************************
c get variable id's by using ncvid
c THE VARIABLE NAMES MUST MATCH THE CDL FILE (case sensitive)

      tid = ncvid(cdfid, 'temp', rcode)
      if (rcode.ne.ncnoerr) stop 'error with tid'
      sid = ncvid(cdfid, 'salt', rcode)
      if (rcode.ne.ncnoerr) stop 'error with sid'
      timeaxid = ncvid(cdfid, 'time', rcode)
      if (rcode.ne.ncnoerr) stop 'error with timeaxid'
***************************************************************************
c this is a good place to open your input data file
      ! OPEN (FILE=my_data.dat,STATUS='OLD')
***************************************************************************
c begin do loop.  Each step will read in one time step of data
c and then write it out to my_data.cdf.

      do 10 itime = 1, nt

      corner(4) = itime             ! initialize time step in corner
      time_step = float(itime)      ! or you may read this from your file

* insert your data reading routine here
!      CALL READ_MY_DATA(temp,salt)  ! you write this

       write (6,*) 'writing time step: ',itime, time_step    ! diagnostic
output

      call ncvpt(cdfid,tid,corner,step,temp(1,1,1),rcode)  ! write data to
      if (rcode.ne.ncnoerr) stop 'error with t-put'
```

```
      call ncvpt(cdfid,sid,corner,step,salt(1,1,1),rcode)  ! my_data.cdf with
      if (rcode.ne.ncnoerr) stop 'error with s-put'
      call ncvpt1(cdfid,timeaxid,itime,time_step,rcode)    ! ncvpt
      if (rcode.ne.ncnoerr) stop 'error with timax-put'

c ncvpt1 writes a single data point to the specified location within
c timeaxid. The itime argument in ncvpt1 specifies the location within
c time to write.
c float(itime) is used (rather than simply itime) so the type matches the
c type of time declared in the CDL file.

10    continue
*************************************************************************
c close my_data.cdf using ncclos
      call ncclos(cdfid, rcode)
      if (rcode.ne.ncnoerr) stop 'error with ncclos'
*************************************************************************
      stop
      end
```

3.3 Advanced NetCDF procedures

This section describes:

1. Setting up a CDL file capable of handling data on staggered grids.
2. Defining coordinate systems such that the data in the NetCDF file may be regarded as hyperslabs of larger coordinate spaces.
3. Defining boundaries between unevenly spaced axis coordinates (used in numerical integrations).
4. Setting up "modulo" axes such as climatological time and longitude.
5. Converting dates into numerical data appropriate for a NetCDF time axis.

The final section of this chapter contains the text of the CDL file for the example we use throughout this section.

In this sample data set, we will consider wind, salt, and velocity calculated using a staggered-grid, finite-difference technique. The wind data is limited to the surface layer of the ocean (i.e. normal to the depth axis). We will also consider the salt data to be limited to a narrow slab from 139E to 90W (I=10 to 140), 32.5N to 34.9N (J=80 to 82), and for all depth and time values.

3.3.1 Staggered grids

FERRET permits each variable of a NetCDF file to be defined on distinct axes and grids. Staggered grids are a straightforward application of this principle. Dimensions for each grid axis must be defined, the axes themselves must be defined (in Variables), and the coordinate values for each axis must be initialized (in Data). In the case of the example we use throughout this and the next section, there are two grids - a wind grid, and a velocity grid; slon, slat and sdepth are defined for the wind grid, and ulon, ulat, and wdepth for the velocity grid. The variables are then given dimensions to place them in their proper grids (i.e. wind(time, sdepth, slat, slon)).

3.3.2 Hyperslabs

There are a number of steps required to set up a NetCDF data set that represents a hyperslab of data from a larger grid definition (a parent grid).

1. Define a dimension named "grid_definition". This dimension should be set equal to 1.

2. Define parent grids in Variables with the argument "grid_definition".

   ```
   char wind_grid(grid_definition) ;
   char salt_grid(grid_definition) ;
   ```

3. Define the 4 axes of the parent grids using the "axes" attribute.

   ```
   wind_grid: axes = "slon slat normal time" ;
   salt_grid: axes = "slon slat sdepth time" ;
   ```

   The arguments are always a list of four axis names. Note that the order of arguments is opposite that in the variable declaration. The argument "normal" indicates that wind_grid is normal to the depth axis.

4. Define the variables which are hyperslabs of these grids with the proper dimensions.

   ```
         float wind(time, slat, slon) ;
         float salt(time, sdepth, slat80_82, slon10_140) ;
   ```

   where the dimension slat80_82 = 3 and slon10_140 = 131. Optionally, these axes may be defined themselves with the attribute "child_axis".

   ```
         float slat80_82(slat80_82) ;
         slat80_82: child_axis = " " ;
   ```

   These "child axes" need not be initialized in data, nor do edges need to be defined for them; FERRET will retrieve this information from the parent axis definitions. However, it is recommended that they be initialized to accommodate other software which may not recognize parent grids.

5. Use the "parent_grid" variable attribute to point to the parent grid.

   ```
         wind: parent_grid = "wind_grid"
         salt: parent_grid = "salt_grid"
   ```

6. Also as a variable attribute, define the index range of interest within the parent grid.

   ```
         wind: slab_min_index = 1s, 1s, 1s, 0s ;
   ```

```
wind: slab_max_index = 160s, 100s, 1s, 0s ;
salt: slab_min_index = 10s, 80s, 1s, 0s ;
salt: slab_max_index = 140s, 82s, 27s, 0s ;
```

The "s" after each integer indicates a "short" 16-bit integer rather than the default "long" 32-bit integer. If an axis dimension is designated as "unlimited" then the index bounds for this axis must be designated as "0s".

These attributes will effectively locate the wind and salt data within the parent grid.

3.3.3 Unevenly spaced coordinates

For coordinate axes with uneven spacing, the boundaries between each coordinate can be indicated by pointing to an additional axis that contains the locations of the boundaries. The dimension of this "edge" axis is necessarily one larger than the coordinate axis concerned. If edges are not explicitly defined for an unevenly spaced axis, the midpoint between coordinates is assumed by default.

1. Define a dimension one larger than the coordinate axis. For the sdepth axis, with 27 coordinates, define:

```
sdepth_edges = 28 ;
```

2. Define an axis called sdepth_edges.
3. Initialize this axis with the desired boundaries (in Data).
4. As an attribute of the main axis, point to edges list:

```
sdepth: edges = "sdepth_edges" ;
```

3.3.4 Evenly spaced coordinates (long axes)

If the coordinate axes are evenly spaced, the attribute "point spacing" should be used:

```
slat: point_spacing = "even" ;
```

When used, this attribute will improve memory use efficiency in FERRET. This is especially important for very large axes - 10,000 points or more.

3.3.5 "Modulo" axes

The "modulo" axis attribute indicates that the axis wraps around, the first point immediately following the last. The most common uses of modulo axes are:

1. longitude axes for globe-encircling data
2. time axes for climatological data

```
time: modulo = " " ;  // any arbitrary string is allowed
```

If the climatological data occurs in the years 0000 or 0001 then the year will be omitted from FERRET's output format.

3.3.6 Converting time word data to numerical data

To set up a time axis for data which is represented as dates (e.g. "1972 January 15 at 12:15") it is necessary to determine a numerical representation for each of the dates. FERRET can assist with this process, as the following example shows.

Suppose the data are 6-hourly observations from 1-JAN-1991 at 12:00 to 15-MAR-1991 at 18:00. These commands will assist in creating the necessary time axis for a NetCDF file:

```
yes? DEFINE AXIS/T="1-JAN-1991:12:00":"15-MAR-1991:18:00":6/UNITS=hours\
       my_time
yes? DEFINE GRID/T=my_time tgrid
yes? SET REGION/T="1-JAN-1991:12:00":"15-MAR-1991:18:00"
yes? LIST T[g=tgrid]                              !to see the time values
yes? SAVE/FILE=my_time.cdf T[g=tgrid]
```

The file my_time.cdf now contains a model of exactly the desired time axis. Use the Unix command

```
% ncdump my_time.cdf > my_time.cdl
```

to obtain the time axis definition as text that can be inserted into your CDL file.

3.4 Example CDL file

The following is an example CDL file utilizing many of the features described in the preceding section.

```
netcdf converting_to_netcdf_supplement {
//      CONVERTING DATA TO THE "NETWORK COMMON DATA FORM" (NetCDF):
//                          A SUPPLEMENT
//
// NOAA PMEL Thermal Modeling and Analysis Project (TMAP)
// Dan Trueman, Steve Hankin
// last revised: 1 Dec 1993:  slat80_82 and slon10_140 coordinates included
//
// I. INTRODUCTION
//
// This supplement to "Converting Data to the Network Common Data Form:
// an Introduction" describes:
//
//      1. How to set up a cdl file capable of handling data
//         on staggered grids.
//      2. How to define coordinate systems such that the data
//         in the NetCDF file may be regarded as hyperslabs of
//         larger coordinate spaces.
//      3. How to define variables of 1, 2, or 3 dimensions.
//      4. How to define boundaries between unevenly spaced axis
```

```
//          coordinates (used in numerical integrations).
//       5. How to set up climatological "modulo" time axes.
//       6. How to convert time word data into numerical data
//          appropriate for NetCDF.
//
// In this sample data set, we will consider wind, salt, and
// velocity calculated using a staggered-grid, finite-difference
// technique. The wind data is naturally limited to the surface
// layer of the ocean (i.e. normal to the  depth axis).  We will
// also consider the salt data to be limited to a narrow slab from
// 139E to 90W (I=10 to 140), 32.5N to 34.9N (J=80 to 82), and for
// all depth and time values.
//
// II. STAGGERED GRIDS
//
// Dealing with staggered grids is fairly straightforward.  Dimensions
// for each grid axis must be defined, the axes themselves must be
// defined (in Variables), and the coordinate values for each axis must
// be initialized (in Data).  In this case, there are two grids, a
// wind grid, and a velocity grid, so tlon, tlat and tdepth are
// defined for the wind grid, and ulon, ulat, and udepth for the velocity
// grid.  The variables are then given arguments to place them in their
// proper grids (i.e. wind(time, sdepth, slat, slon)).
//
// III. HYPERSLABS
//
// There are a number of steps required to set up a NetCDF data set that
// represents a hyperslab of data from a larger grid definition.
//
//       1. Define a dimension named "grid_definition".  This dimension
//          should be set equal to 1.
//       2. Define parent grids in Variables with the argument
//          "grid_definition".
//
//               char wind_grid(grid_definition) ;
//               char salt_grid(grid_definition) ;
//
//      3. Define the 4 axes of the parent grids using the "axes" attribute.
//
//                    wind_grid: axes = "slon slat normal time" ;
//                    salt_grid: axes = "slon slat sdepth time" ;
//
//          Note that the order of arguments is opposite that in the
//          variable declaration.  The argument "normal" indicates that
//          wind_grid is normal to the depth axis.
//
//       4. Define the variables which are hyperslabs of these grids with
//          the proper dimensions.
//
//               float wind(time, slat, slon) ;
//               float salt(time, sdepth, slat80_82, slon10_140) ;
//
//          where slat80_82 = 3 and slon10_140 = 131. The axis names are
///         arbitrary - chosen for readability. These axes (child axes)
//          must be defined with the attribute "child_axis" as follows:
//
//               float slat80_82(slat80_82) ;
//                     slat80_82: child_axis = " " ;
//
//          These "child axes" need not be initialized in Data, nor do their
//          edges need be defined; FERRET retrieves this information from
//          the parent axes.
//
```

```
//      5. Use the "parent_grid" variable attribute to point to the
//         parent grid.
//
//                   wind: parent_grid = "wind_grid"
//
//      6. Also as a variable attribute, define the index range of interest
//         within the parent grid.
//
//                   wind: slab_min_index = 1s, 1s, 1s, 0s ;
//                   wind: slab_max_index = 160s, 100s, 1s, 0s ;
//                   salt: slab_min_index = 10s, 80s, 1s, 0s ;
//                   salt: slab_max_index = 140s, 82s, 27s, 0s ;
//
//         The "s" after each integer indicates a "short" 16-bit integer
//         rather than the default "long" 32-bit integer.  If an axis
//         dimension is designated as "unlimited" then the index bounds
//         for this axis must be designated as "0s".
//
// These commands will effectively locate the wind and salt data within
// the full grid.
//
// IV. VARIABLES OF 1, 2, or 3 DIMENSIONS
//
// One, two, or three dimensional variables may be set up in one of
// two ways - either using the parent_grid and child_axis attributes
// as illustrated in the 3-dimensional variable "wind" from the hyperslab
// example, above, or by selecting axis names and units that provide
// FERRET with adequate hints to map this variable onto 4-dimensional
// space and time.  The following hints are recognized by FERRET:
//
//     Units of days, hours, minutes, etc. or an axis name of "TIME", "DATE"
//         implies a time axis.
//     Units of "degrees xxxx" where "xxxx" contains "lat" or "lon" implies
//         a latitude or longitude axis, respectively.
//      Units of "degrees" together with an axis name containing "LAT" or
//         "Y" implies a latitude axis else longitude is assumed.
//      Units of millibars, "layer" or "level" or an axis name containing
//         "Z" or "ELEV" implies a vertical axis.
//
// V. UNEVENLY SPACED COORDINATE BOUNDARIES
//
// For coordinate axes with uneven spacing, the boundaries between each
// coordinate can be indicated by pointing to an additional axis that
// contains the locations of the boundaries.  The dimension of this "edge"
// axis will necessarily be one larger than the coordinate axis concerned.
// If edges are not defined for an unevenly spaced axis, the midpoint
// between coordinates will be assumed by default.
//
//      1. Define a dimension one larger than the coordinate axis.  For
//         the sdepth axis, with 27 coordinates, define:
//
//               sdepth_edges = 28 ;
//
//      2. Define an axis called sdepth_edges.
//      3. Initialize this axis appropriately (in Data).
//      4. As a sdepth axis attribute, point to sdepth_edges:
//
//                   sdepth: edges = "sdepth_edges" ;
//
// If the coordinate axes are evenly spaced, the attribute "point spacing"
// should be used:
//
//                   slat: point_spacing = "even" ;
```

```
//
// When used, this attribute will improve memory use efficiency in FERRET.
//
// VI. CLIMATOLOGICAL "MODULO" AXES
//
// The "modulo" axis attribute indicates that the axis wraps around,
// the first point immediately following the last.  The most common
// uses of modulo axes are:
//
//      1. As longitude axes for globe-encircling data.
//      2. As time axes for climatological data.
//
//                  time: modulo = " " ; // any arbitrary string is allowed
//
// If the climatological data occurs in the years 0000 or 0001 then FERRET
// will omit the year from the output formatting.
//
// VII. CONVERTING TIME WORD DATA TO NUMERICAL DATA
//
// If the time data being converted to NetCDF format exists in string format
// (i.e. 1972 - JANUARY 15 2:15:00), rather than numerical format (i.e. 55123
// seconds) a number of TMAP routines are available to aid in the conversion
// process.  The steps required for conversion are as follows:
//
//      1. Break the time string into its 6 pieces.  If the data is of the
//         form dd-mmm-yyyy:hh:mm:dd, the TMAP routine "tm_break_date.f" can
//         be used.
//      2. Choose a time_origin before the beginning of the time data to
//         assure that all time values are positive.  i.e. if the data begins
//         at 15-JAN-1982:05:30:00, choose a time origin of
//         15-JAN-1981:00:00:00. This time_origin should then be an attribute
//         of the time axis variable in the CDL file.
//      3. Produce numerical time data by using "tm_sec_from_bc.f", which
//         calculates the number of seconds between 01-01-0000:00:00:00 and
//         the date specified.  Continuing the example from (2), the time value
//         for the first time step with respect to the time_origin could be
//         calculated as follows:
//
//         time(1) = tm_sec_from_bc(1982, 1, 15, 5, 30, 0) -
//                   tm_sec_from_bc(1981, 1, 15, 0, 0, 0)
//
//         or more generally
//
//       time(n)=tm_sec_from_bc(nyear,nmonth,nday,nhour,nminute,nsecond) -
//                  tm-sec_from_bc(oyear,omonth,oday,ohour,ominute,osecond)
//
//        where nyear is the year for the nth time step and oyear is the year
//         of time_origin.
//
// VII. EXAMPLE CDL FILE
dimensions:

// staggered grid dimension definitions:

        slon = 160 ;    // wind/salt longitude dimension
        ulon = 160 ;    // velocity longitude dimension
        slat = 100 ;    // wind/salt latitude dimension
        ulat = 100 ;    // velocity latitude dimension
        sdepth = 27 ;   // salt depth dimension
        wdepth = 27 ;   // velocity depth dimension

        slon10_140 = 131 ;      // for salt hyperslab
        slat80_82 = 3 ;         // for salt hyperslab
```

```
        time = unlimited ;

// grid_definition is the dimension name to be used for all grid definitions

        grid_definition = 1 ;

// edge dimension definitions:

        sdepth_edges = 28 ;
        wdepth_edges = 28 ;

variables:

  // variable definitions:

        float wind(time, slat, slon) ;  // 3-dimensional variable
              wind: parent_grid = "wind_grid" ;
              wind: slab_min_index = 1s, 1s, 1s, 0s ;
              wind: slab_max_index = 160s, 100s, 1s, 0s ;
              wind: long_name = "WIND" ;
              wind: units     = "deg. C" ;
              wind: _FillValue = 1E34f ;
        float salt(time, sdepth, slat80_82, slon10_140) ; // 4-dim. variable
              salt: parent_grid = "salt_grid" ;
              salt: slab_min_index = 10s, 80s, 1s, 0s ;
              salt: slab_max_index = 140s, 82s, 27s, 0s ;
              salt: long_name = "(SALINITY(ppt) - 35) /1000" ;
              salt: units     = "frac. by wt. less .035" ;
              salt: _FillValue = -999.f ;

        float u(time, sdepth, ulat, ulon) ;
              u: long_name    = "ZONAL VELOCITY" ;
              u: units        = "cm/sec" ;
              u: _FillValue = 1E34f ;
        float v(time, sdepth, ulat, ulon) ;
              v: long_name    = "MERIDIONAL VELOCITY" ;
              v: units        = "cm/sec" ;
              v: _FillValue = 1E34f ;
        float w(time, wdepth, slat, slon) ;
              w: long_name    = "VERTICAL VELOCITY" ;
              w: units        = "cm/sec" ;
              w: _FillValue = 1E34f ;

  // axis definitions:

        float slon(slon) ;
              slon: units = "degrees" ;
              slon: point_spacing = "even" ;
        float ulon(ulon) ;
              ulon: units = "degrees" ;
              ulon: point_spacing = "even" ;
        float slat(slat) ;
              slat: units = "degrees" ;
              slat: point_spacing = "even" ;
        float ulat(ulat) ;
              ulat: units = "degrees" ;
              ulat: point_spacing = "even" ;
        float sdepth(sdepth) ;
              sdepth: units = "meters" ;
              sdepth: positive = "down" ;
              sdepth: edges = "sdepth_edges" ;
        float wdepth(wdepth) ;
```

```
              wdepth: units = "meters" ;
              wdepth: positive = "down" ;
              wdepth: edges = "wdepth_edges" ;
        float time(time) ;
              time: modulo = " " ;
              time: time_origin = "15-JAN-1981:00:00:00" ;
              time: units = "seconds" ;

  // child grid definitions:

        float slon10_140(slon10_140) ;
              slon10_140: child_axis = " " ;

              slon10_140: units = "degrees" ;
        float slat80_82(slat80_82) ;
              slat80_82: child_axis = " " ;
              slat80_82: units = "degrees" ;

  // edge axis definitions:

        float sdepth_edges(sdepth_edges) ;
        float wdepth_edges(wdepth_edges) ;

  // parent grid definition:

        char wind_grid(grid_definition) ;
             wind_grid: axes = "slon slat normal time" ;
        char salt_grid(grid_definition) ;
             salt_grid: axes = "slon slat sdepth time" ;

  // global attributes:
        :title = "NetCDF Title" ;

data:

//  //  ignore this block  //
//This next data entry, for time, should be ignored. Time is initialized here
// only so that FERRET can read test.cdf (the file created by this cdl file)
// with no additional data inserted into it.
time=1000;
// // end of ignored block //

slat=
-28.8360729218,-26.5299491882,-24.2880744934,-22.1501560211,-20.1513576508,
-18.3207626343,-16.6801033020,-15.2428140640,-14.0134353638,-12.9874248505,
-12.1513509750,-11.4834814072,-10.9547319412,-10.5299386978,-10.1693935394,
-9.8333206177,-9.4999876022,-9.1666536331,-8.8333196640,-8.4999856949,
-8.1666526794,-7.8333187103,-7.4999847412,-7.1666512489,-6.8333182335,
-6.4999852180,-6.1666517258,-5.8333182335,-5.4999852180,-5.1666517258,
-4.8333187103,-4.4999852180,-4.1666517258,-3.8333187103,-3.4999852180,
-3.1666517258,-2.8333184719,-2.4999852180,-2.1666519642,-1.8333185911,
-1.4999852180,-1.1666518450,-0.8333183527,-0.4999849498,-0.1666515470,
0.1666818559,0.5000152588,0.8333486915,1.1666821241,1.5000154972,
1.8333489895,2.1666824818,2.5000159740,2.8333494663,3.1666829586,
3.5000162125,3.8333497047,4.1666831970,4.5000162125,4.8333497047,
5.1666831970,5.5000162125,5.8333497047,6.1666827202,6.5000162125,
6.8333497047,7.1666827202,7.5000166893,7.8333501816,8.1666841507,
8.5000181198,8.8333511353,9.1666851044,9.5000190735,9.8333530426,
10.1679363251,10.5137376785,10.8892869949,11.3138961792,11.8060989380,
12.3833675385,13.0618314743,13.8560228348,14.7786512375,15.8403968811,
17.0497493744,18.4128704071,19.9334945679,21.6128730774,23.4497566223,
25.4404067993,27.5786647797,29.8560409546,32.2618522644,34.7833900452,
37.4061241150,40.1139259338,42.8893203735,45.7137718201,48.5679702759;
```

```
ulat=
-27.6721439362,-25.3877544403,-23.1883945465,-21.1119174957,-19.1907978058,
-17.4507274628,-15.9094810486,-14.5761461258,-13.4507236481,-12.5241250992,
-11.7785758972,-11.1883859634,-10.7210769653,-10.3387994766,-9.9999876022,
-9.6666545868,-9.3333206177,-8.9999866486,-8.6666526794,-8.3333196640,
-7.9999856949,-7.6666517258,-7.3333182335,-6.9999847412,-6.6666512489,
-6.3333182335,-5.9999847412,-5.6666517258,-5.3333182335,-4.9999847412,
-4.6666517258,-4.3333182335,-3.9999849796,-3.6666517258,-3.3333184719,
-2.9999852180,-2.6666519642,-2.3333184719,-1.9999853373,-1.6666518450,
-1.3333184719,-0.9999850392,-0.6666516662,-0.3333182633,0.0000151545,
0.3333485723,0.6666819453,1.0000153780,1.3333487511,1.6666821241,
2.0000154972,2.3333489895,2.6666827202,3.0000162125,3.3333497047,
3.6666829586,4.0000162125,4.3333497047,4.6666827202,5.0000162125,
5.3333492279,5.6666827202,6.0000162125,6.3333492279,6.6666827202,
7.0000157356,7.3333497047,7.6666831970,8.0000171661,8.3333511353,
8.6666841507,9.0000181198,9.3333520889,9.6666860580,10.0000190735,
10.3358526230,10.6916217804,11.0869522095,11.5408391953,12.0713586807,
12.6953773499,13.4282865524,14.2837600708,15.2735414505,16.4072513580,
17.6922454834,19.1334934235,20.7334957123,22.4922523499,24.4072608948,
26.4735546112,28.6837768555,31.0283031464,33.4953994751,36.0713844299,
38.7408676147,41.4869842529,44.2916526794,47.1358833313,50.0000534058;
slon=
130.5,131.5,132.5,133.5,134.5,135.5,136.5,137.5,138.5,139.5,140.5,141.5,
142.5,143.5,144.5,145.5,146.5,147.5,148.5,149.5,150.5,151.5,152.5,153.5,
154.5,155.5,156.5,157.5,158.5,159.5,160.5,161.5,162.5,163.5,164.5,165.5,
166.5,167.5,168.5,169.5,170.5,171.5,172.5,173.5,174.5,175.5,176.5,177.5,
178.5,179.5,180.5,181.5,182.5,183.5,184.5,185.5,186.5,187.5,188.5,189.5,
190.5,191.5,192.5,193.5,194.5,195.5,196.5,197.5,198.5,199.5,200.5,201.5,
202.5,203.5,204.5,205.5,206.5,207.5,208.5,209.5,210.5,211.5,212.5,213.5,
214.5,215.5,216.5,217.5,218.5,219.5,220.5,221.5,222.5,223.5,224.5,225.5,
226.5,227.5,228.5,229.5,230.5,231.5,232.5,233.5,234.5,235.5,236.5,237.5,
238.5,239.5,240.5,241.5,242.5,243.5,244.5,245.5,246.5,247.5,248.5,249.5,
250.5,251.5,252.5,253.5,254.5,255.5,256.5,257.5,258.5,259.5,260.5,261.5,
262.5,263.5,264.5,265.5,266.5,267.5,268.5,269.5,270.5,271.5,272.5,273.5,
274.5,275.5,276.5,277.5,278.5,279.5,280.5,281.5,282.5,283.5,284.5,285.5,
286.5,287.5,288.5,289.5;
ulon=
131.0,132.0,133.0,134.0,135.0,136.0,137.0,138.0,139.0,140.0,141.0,142.0,
143.0,144.0,145.0,146.0,147.0,148.0,149.0,150.0,151.0,152.0,153.0,154.0,
155.0,156.0,157.0,158.0,159.0,160.0,161.0,162.0,163.0,164.0,165.0,166.0,
167.0,168.0,169.0,170.0,171.0,172.0,173.0,174.0,175.0,176.0,177.0,178.0,
179.0,180.0,181.0,182.0,183.0,184.0,185.0,186.0,187.0,188.0,189.0,190.0,
191.0,192.0,193.0,194.0,195.0,196.0,197.0,198.0,199.0,200.0,201.0,202.0,
203.0,204.0,205.0,206.0,207.0,208.0,209.0,210.0,211.0,212.0,213.0,214.0,
215.0,216.0,217.0,218.0,219.0,220.0,221.0,222.0,223.0,224.0,225.0,226.0,
227.0,228.0,229.0,230.0,231.0,232.0,233.0,234.0,235.0,236.0,237.0,238.0,
239.0,240.0,241.0,242.0,243.0,244.0,245.0,246.0,247.0,248.0,249.0,250.0,
251.0,252.0,253.0,254.0,255.0,256.0,257.0,258.0,259.0,260.0,261.0,262.0,
263.0,264.0,265.0,266.0,267.0,268.0,269.0,270.0,271.0,272.0,273.0,274.0,
275.0,276.0,277.0,278.0,279.0,280.0,281.0,282.0,283.0,284.0,285.0,286.0,
287.0,288.0,289.0,290.0;
sdepth=
5.0,15.0,25.0,35.0,45.0,55.0,65.0,75.0,85.0,95.0,106.25,120.0,136.25,155.0,
177.5,205.0,240.0,288.5,362.5,483.5,680.0,979.5,1395.5,1916.0,2524.0,3174.0,
3824.0;
sdepth_edges=
0.0,10.0,20.0,30.0,40.0,50.0,60.0,70.0,80.0,90.0,100.0,112.5,127.5,
145.0,165.0,190.0,220.0,260.0,317.0,408.0,559.0,801.0,1158.0,1633.0,2199.0,
2849.0,3499.0,4149.0;
wdepth=
10.0,20.0,30.0,40.0,50.0,60.0,70.0,80.0,90.0,100.0,112.5,127.5,145.0,165.0,
190.0,220.0,260.0,317.0,408.0,559.0,801.0,1158.0,1633.0,2199.0,2849.0,3499.0,
4149.0;
```

```
wdepth_edges=
5.0,15.0,25.0,35.0,45.0,55.0,65.0,75.0,85.0,94.375,105.625,119.375,135.625,
153.75,176.25,202.5,235.75,280.0,347.5,460.75,651.25,950.0,1372.75,1895.0,
2524.0,3174.0,3986.5,4311.0;
slon10_140=
    139.5, 140.5, 141.5, 142.5, 143.5, 144.5, 145.5, 146.5, 147.5,
    148.5, 149.5, 150.5, 151.5, 152.5, 153.5, 154.5, 155.5, 156.5, 157.5,
    158.5, 159.5, 160.5, 161.5, 162.5, 163.5, 164.5, 165.5, 166.5, 167.5,
    168.5, 169.5, 170.5, 171.5, 172.5, 173.5, 174.5, 175.5, 176.5, 177.5,
    178.5, 179.5, 180.5, 181.5, 182.5, 183.5, 184.5, 185.5, 186.5, 187.5,
    188.5, 189.5, 190.5, 191.5, 192.5, 193.5, 194.5, 195.5, 196.5, 197.5,
    198.5, 199.5, 200.5, 201.5, 202.5, 203.5, 204.5, 205.5, 206.5, 207.5,
    208.5, 209.5, 210.5, 211.5, 212.5, 213.5, 214.5, 215.5, 216.5, 217.5,
    218.5, 219.5, 220.5, 221.5, 222.5, 223.5, 224.5, 225.5, 226.5, 227.5,
    228.5, 229.5, 230.5, 231.5, 232.5, 233.5, 234.5, 235.5, 236.5, 237.5,
    238.5, 239.5, 240.5, 241.5, 242.5, 243.5, 244.5, 245.5, 246.5, 247.5,
    248.5, 249.5, 250.5, 251.5, 252.5, 253.5, 254.5, 255.5, 256.5, 257.5,
    258.5, 259.5, 260.5, 261.5, 262.5, 263.5, 264.5, 265.5, 266.5, 267.5,
    268.5, 269.5 ;
slat80_82=
    11.8060989379883, 12.3833675384522, 13.0618314743042 ;

}
```

Part II: COMMANDS REFERENCE

1 ALIAS

Alias for DEFINE ALIAS.

2 CANCEL

Cancels a program state or definition - generally paired with a SET or DEFINE command.

See commands SET and DEFINE for further information.

2.1 CANCEL ALIAS

Cancels a user-defined command alias.

```
yes? CANCEL ALIAS ALIAS_NAME
```

The command UNALIAS is an alias for CANCEL ALIAS.

2.2 CANCEL AXIS
/MODULO

Cancels the modulo nature of a user-defined axis (only valid with /MODULO qualifier).

```
yes? CANCEL AXIS/MODULO my_x_axis
```

Command qualifiers for CANCEL AXIS:

CANCEL AXIS/**MODULO**

2.3 CANCEL DATA_SET
/ALL /NOERROR

Removes the specified data set from the list of available sets.

```
yes? CANCEL DATA_SET dset1, dset2, ..., dsetn
```

where each dset may be the name or number of a data set; or

```
yes? CANCEL DATA/ALL
```

(See also SET DATA_SET and SHOW DATA SET.)

Command qualifiers for CANCEL DATA_SET:

CANCEL DATA/**ALL**
Eliminates all data sets from the list of accessible data sets.

CANCEL DATA/**NOERROR**
Suppresses the error message otherwise generated when a data set is canceled which was never set. Useful in GO scripts for closing data sets that *may* have been opened in previous usage of the script.

2.4 CANCEL EXPRESSION

Un-specifies the current context expression. FERRET's "action" commands can be issued without an argument (e.g. `yes? PLOT`), in which case FERRET uses the current context expression. This expression is either the argument of the most recent action command, or an expression set explicitly with SET EXPRESSION.

```
yes? CANCEL EXPRESSION
```

The qualifier /ALL can be used with this command, but it exists for compatibility purposes only and has no effect.

2.5 CANCEL LIST

/ALL /APPEND /FILE /FORMAT /HEADING /PRECISION

Toggles the effects of the SET LIST command. See command SET LIST.

```
yes? CANCEL LIST[/qualifiers]
```

Command qualifiers for: CANCEL LIST

CANCEL LIST/**ALL**
Restores all aspects of the LIST command to their default behavior.

CANCEL LIST/**APPEND**
Resets the listed output to NOT append to existing file.

CANCEL LIST/**FILE**
Resets the listed output to automatic file naming.

CANCEL LIST/**FORMAT**
Resets the listed output to its default formatting.

CANCEL LIST/**HEAD**
Instructs listed output to omit the descriptive data header.

CANCEL LIST/**PRECISION**
Resets the precision of listed data to 4 significant digits.

2.6 CANCEL MEMORY

/ALL /PERMANENT /TEMPORARY

Clears data currently cached in memory.

```
yes? CANCEL MEMORY[/qualifier]
```

Use this command to save memory space - by clearing data as soon as it is no longer needed virtual memory requirements can be reduced. This is especially useful for efficient batch processing. Default is CANCEL MEMORY/TEMPORARY.

Example:

To produce an animation using minimal virtual memory try:

```
yes? REPEAT/T=lo:hi:delta GO min_mem_movie
```

Where the file min_mem_movie.jnl contains

```
CONTOUR/FRAME  temp[Z=0]        ! contour plot
CANCEL MEMORY/ALL               ! clear memory for next time step
```

Command qualifiers for CANCEL MEMORY:

CANCEL MEMORY/**ALL**
Clears all variables stored in memory.

CANCEL MEMORY/**PERMANENT**
Clears all "permanent" variables stored in memory (i.e. variables loaded into memory with LOAD/PERMANENT).

CANCEL MEMORY/**TEMPORARY** (default)
Clears all non-permanent variables stored in memory.

2.7 CANCEL MODE

Sets the state of a mode to "canceled".

```
   yes? CANCEL MODE mode_name
```

(See command SET MODE for descriptions of modes.)

2.8 CANCEL MOVIE

This command is unnecessary in FERRET version 3.1. It restores the default movie file name (ferret.mgm) but is not needed to conclude capturing graphics to a movie file.

```
yes? CANCEL MOVIE
```

The qualifier /ALL can be used with this command, but it exists for compatibility purposes only and has no effect.

2.9 CANCEL REGION

/I/J/K/L /X/Y/Z/T /ALL

Cancels part or all of the current or named region.

```
   yes? CANCEL REGION[/qualifier] [region_name]
```

Examples:

```
   yes? CANCEL REGION            !clear the current region
   yes? CANCEL REGION/T          !eliminate T from the current context
   yes? CANCEL REGION reg1       !clear the region named "reg1"
```

Command qualifiers for CANCEL REGION:

CANCEL REGION**/I /J /K /L /X /Y /Z /T**
Eliminates I, J, K, L, X, Y, Z or T axis information from current context or named region.

CANCEL REGION/**ALL**
Eliminates ALL stored region information (rarely used).

2.10 CANCEL VARIABLE

/ALL

Deletes a user-defined variable definition.

```
   yes? CANCEL VARIABLE[/qualifier]  [var_name]
```

Command qualifiers for CANCEL VARIABLE:

CANCEL VARIABLE/**ALL**
Deletes all user-defined variable definitions.

2.11 CANCEL VIEWPORT

Cancels a defined viewport or cancels use of viewports.

```
yes? CANCEL VIEWPORT view_name      !un-define view_name
yes? CANCEL VIEWPORT                !return to full window output
```

2.12 CANCEL WINDOW

/ALL

Removes graphics window(s) from the screen.

```
yes? CANCEL WINDOW  n   !or

yes? CANCEL WINDOW/ALL
```

Command qualifiers for CANCEL WINDOW:

CANCEL WINDOW/**ALL**
Removes all graphics windows.

3 CONTOUR

/I/J/K/L /X/Y/Z/T /D /FILL /FRAME /KEY /LEVELS /LINE /NOKEY
/NOLABEL /OVERLAY /PALETTE /PEN /SET_UP /TITLE /TRANSPOSE
/XLIMITS /YLIMITS

Produces a contour plot.

```
yes? CONTOUR[/qualifiers]   [expression]
```

Example:

```
yes? CONTOUR var1       !produce a contour plot of the variable var1
```

Parameters
Expressions may be any valid expression. See Chapter 3 section "Expressions" for a definition of valid expressions. The expression will be inferred from the current context if omitted from the command line.

Command qualifiers for CONTOUR:

CONTOUR/**I=/J=/K=/L=/X=/Y=/Z=/T=**
Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when evaluating the expression being plotted.

CONTOUR/**D=**
Specifies the default data set to use when evaluating the expression being contoured.

CONTOUR/**FILL** (alias FILL)
Creates a color filled contour image.

CONTOUR/**FRAME**
Causes the graphic image produced by the command to be captured as an animation frame in the file specified by SET MOVIE.

CONTOUR/**KEY**
Displays a color key for the palette used in a color-filled contour plot. Only valid in conjunction with /FILL (default with CONTOUR/FILL or alias FILL).

CONTOUR/**LEVELS**
Specifies the contour levels or how the levels will be determined. If the /LEVELS qualifier is omitted FERRET automatically selects reasonable contour levels.

The /LEVELS qualifier can be used in two different ways:

1) To reuse the levels from the last CONTOUR or SHADE plot use /LEVELS with no argument:

```
yes? CONTOUR/LEVELS expr
```

2) To specify the contour levels manually use a combination of arguments with the following syntaxes:

(val)
val = value of a contour level

(lo,hi,delta,ndig)
lo = lowest contour level
hi = highest contour level
delta = interval between levels
ndig = number of digits after the decimal in contour level labels (optional)
-1 integer labels
-3 no labels

```
yes? CONT/LEV=(lo,hi,delta)(val1)(val2)(lo2,hi2,delta2)...  expr
```

After specifying contour levels, the user can specify line types for various levels with a combination of arguments using the following syntax:

TYPE(lo,hi,delta,index) or TYPE(val)

where "TYPE" may be any of the following and applies to the specified levels:

DARK	draws thicker lines
DASH	draws dashed lines (default for negative contour levels)
DEL	omits specified levels
LINE	draws solid lines of regular thickness (default for positive levels)
PEN	draws lines with specified color_thickness index

Only type PEN requires specification of "index"; it may be omitted otherwise. If blanks are imbedded within the list of arguments you must enclose the entire list within quotation marks.

Example:

```
yes? CONT/LEV=(-50,50,10)(-25)(25)DARK(0)PEN(30,40,10,6)  sst
```

See Chapter 6 section "Contouring" for examples and more documentation on /LEVELS and color_thickness indices, and also the demonstration "custom_contour_demo.jnl".

CONTOUR/**LINE**
Overlays contour lines on a color-filled plot. Valid only with /FILL (or as a qualifier to alias FILL). When /LINE is specified the color key, by default, is omitted. Use FILL/LINE/KEY to obtain both contour lines and a color key.

CONTOUR/**NOKEY**
Turns off display of a color key for the palette used in a color-filled contour plot. Only valid in conjunction with /FILL (or with alias FILL).

CONTOUR/**NOLABELS**
Suppresses all plot labels except axis labels.

CONTOUR/**OVERLAY**
Causes the indicated expression to be overlaid on the existing plot.

Note (CONTOUR/OVERLAY with time axes):
A restriction in PPLUS requires that if time is an axis of the contour plot, the overlaid variable must share the same time axis encoding as the base plot variable. If this condition is not met, you may find that the overlaid contour fails to be drawn. The solution is to use the FERRET regridding capability to regrid the base plot variable and the overlaid plot variable onto the same time axis.

CONTOUR/**PALETTE=**
Specifies a color palette (otherwise, a default rainbow palette is used). Valid only with /FILL (or as a qualifier to alias FILL). Try the Unix command `% Fpalette '*'` to see available palettes. The file suffix *.spk is not necessary when specifying a palette. See command PALETTE for more information.

Example:

```
yes? CONTOUR/FILL/PALETTE=land_sea  world_relief
```

CONTOUR/**PEN=**

Sets line style for contour lines (same arguments as PLOT/LINE=). Argument can be an integer between 1 and 18; run `GO line_samples` to see the styles for color devices.

Example:

```
yes? CONTOUR/PEN=2  sst
```

CONTOUR/**SET_UP**

Performs all the internal preparations required by program FERRET for contouring but does not actually render output. The command PPL can then be used to make changes to the plot prior to producing output with the PPL CONTOUR command. This permits plot customizations that are not possible with FERRET command qualifiers. See Chapter 6 section "Contouring".

CONTOUR/**TITLE=**

Allows user to specify a plot title (enclosed in quotation marks). Without this qualifier FERRET selects a title based on information about the expression. To include font change and color_thickness specifications (e.g., @TI@C002) in the title string, it is necessary either to CANCEL MODE ASCII or to include a leading ESC (escape) character.

CONTOUR/**TRANSPOSE**

Causes the horizontal and vertical axes to be interchanged. By default the X axis of the data is drawn horizontally on the plot and the Y and Z axes of the data are drawn vertically. For Y-Z plots the Z data axis is vertical by default.

CONTOUR/**XLIMITS=**

Specifies axis range and tic interval for the X axis. Without this qualifier, FERRET selects reasonable values.

```
yes? CONTOUR/XLIMITS=lo_val:hi_val[:increment]  [expression]
```

The optional "increment" parameter determines tic mark spacing on the axis. If the increment is negative, the axis will be reversed.

CONTOUR/**YLIMITS=**

Specifies the axis range and tic interval for the Y axis. See /XLIMITS.

4 DEFINE

Defines a new alias, region, grid, axis, variable or viewport.

4.1 DEFINE ALIAS

Defines an alias for a command. "ALIAS" is an alias for DEFINE ALIAS.

```
yes? DEFINE ALIAS NAME COMMAND
```

Example:

```
yes? DEFINE ALIAS SDF SHOW DATA/FULL
```

4.2 DEFINE AXIS

/X/Y/Z/T /DEPTH /FILE /FROMDATA /MODULO
/NAME /NPOINTS /T0 /UNIT

Defines an axis (axis name up to 16 characters).

```
yes? DEFINE AXIS[/qualifiers]  axis_name_or_expr
```

Example:

```
yes? DEFINE AXIS/X=140E:140W:.2   AX140
```

Command qualifiers for DEFINE AXIS:
DEFINE AXIS/**X=/Y=/Z=/T=**
Specifies the limits and point spacing of an axis.

```
yes? DEFINE AXIS/X=lo:hi:delta   axis_name
```

The limits may be in longitude, latitude or date format (for X, Y or T axis, respectively) or may be simple numbers. No units are assumed unless units are given explicitly with the /UNITS qualifier.

Use /UNITS=degrees to obtain latitude or longitude axes. The X or Y qualifier determines which orientation "degrees" refers to.

For T axis, the limits may be dates (dd-mmm-yyyy:hh:mm:ss) or may be time steps. The delta increment is regarded as hours unless the /UNITS qualifier specifies otherwise.

If the time limits are given as dates then this axis produces date-formatted output (unless CANCEL MODE CALENDAR is issued). If the time limits are given as time steps then all instances of this axis are labelled with time step values in the units specified with the /UNITS qualifier.

Examples (evenly-spaced axes):

```
yes? DEFINE AXIS/X=140E:140W:.2 ax140

yes? DEFINE AXIS/Y=15S:25N:.5 axynew
```

```
yes? DEFINE AXIS/Z=0:5000:20/UNITS=CM/DEPTH axzcm
```

```
yes? DEFINE AXIS/T="7-NOV-1953":"23-AUG-1988:11:00":24  axtlife
```

```
yes? DEFINE AXIS/T=25:125:5/UNITS=minutes   axt5min
```

DEFINE AXIS/**DEPTH**
Specifies the Z axis to be a depth, positive downward, axis. A depth axis is indicated by a "(-)" following its title in a SHOW GRID or SHOW AXIS command. Depth axes are notated by "UD" (up-down) in the grid definition file, while normal vertical axes (such as an elevation axis in meteorology) are notated by "DU" (down-up).

Example:

```
yes? DEFINE AXIS/Z=0:5000:20/DEPTH/UNITS=CM AXZDCM
```

DEFINE AXIS/**FILE=**
Reads a gridfile for grid and axis definitions. The gridfile specified should be in the standard TMAP gridfile format. There are several documents in $FER_DIR/doc regarding gridfiles and TMAP format (e.g. "about_grid_files.txt").

```
yes? DEFINE AXIS/FILE=grid_file.grd
```

DEFINE AXIS/**FROM_DATA**
Used only in conjunction with /NAME to define an axis from any expression that FERRET can evaluate.

```
yes? DEFINE AXIS/FROM_DATA/NAME=axis_name   expr
```

(This is a mechanism to convert dependent variables into independent axis data.)

Example (unevenly-spaced axis):

```
yes? DEFINE AXIS/FROM_DATA/X/NAME=my_xaxis pos[D=2]^0.5
```

defines each coordinate of the axis "my_xaxis" as the square root of variable "pos" from data set 2.

DEFINE AXIS/**MODULO**
Specifies that the axis being defined be treated as modulo; that is, the first point will wrap around and follow the last point (e.g., a longitude axis).

DEFINE AXIS/**NAME=**
Used only in conjunction with /FROM_DATA to specify the name of the axis to be defined.

```
yes? DEFINE AXIS/FROM_DATA/NAME=axis_name   expr
```

DEFINE AXIS/**NPOINTS=**
Specifies the number of coordinate points on the axis being defined.

```
yes? DEFINE AXIS/Z=lo:hi/NPOINTS=n   ax_name
```

This qualifier can be used instead of specifying Z=lo:hi:delta.

DEFINE AXIS/**T0=**
Specifies the date and time associated with the time step value 0.0

Example:

```
DEFINE AXIS/T="1-NOV-1980":"15-AUG-1988":72/T0="1-JAN-1800"  TNEW
```

Note: The /T0 qualifier is optional; the underlying time step values are transparent to FERRET users for most purposes. The default value is 15-JAN-1901.

DEFINE AXIS/**UNITS=**
Specifies the units of the axis being defined.

Example:

```
yes? DEFINE AXIS/Z=0:2000:100/UNITS=CM   ZCM
```

Any string (up to 10 characters) is acceptable as a units string, but only the following units are recognized and used when computing axis transformations:

cm (or centimeter)
km (or kilometer)
m (or meter)
deg (or lat or lon)
ft (or feet or foot)
in
mile
mm (or millimeter)
mb (or millibar)
level
layer
sec
min
hour
day
mon
yr (or year)

4.3 DEFINE GRID

/X/Y/Z/T /FILE /LIKE

Defines a grid (name may be up to 16 characters).

```
yes? DEFINE GRID[/qualifiers]   grid_name
```

Example:

```
yes? DEFINE GRID/LIKE=temp/T=my_t_axis   my_grid
```

Command qualifiers for DEFINE GRID:

DEFINE GRID/**X=/Y=/Z=/T=**
Specifies what particular axis is to be the X, Y Z or T axis for this grid.

```
yes? DEFINE GRID/X=axname  grid_name
```

The name axname may be the name of an axis, the name of a grid which uses the axis desired or the name of a variable for which the defining grid uses the axis desired.

For example,

```
yes? DEFINE GRID/X=U   gx
```

will create a grid named gx which is one-dimensional - normal to Y, Z and T.

Note: Many axes possess an orientation implicit in their units, especially latitude, longitude and time axes. The effects of using an axis in an inappropriate orientation, such as /X=time_axis, are unpredictable.

DEFINE GRID/**FILE=**
Reads a gridfile for GRID and AXIS definitions. The gridfile specified should be in the standard TMAP gridfile format. There are several documents in $FER_DIR/doc regarding gridfiles and TMAP format (e.g. about_grid_files.txt).

Example:

```
yes? DEFINE GRID/FILE=new_grids.grd
```

DEFINE GRID/**LIKE=**
Specifies a particular grid (by name or by reference to a variable defined on that grid) to use as a template to create a new grid.

```
yes? DEFINE GRID/LIKE=grid_or_variable_name   grid_name
```

All axes of the grid being created will be identical to the axes of the "LIKE=" grid except those explicitly changed with the /X, /Y, /Z or /T qualifiers.

Example:

```
yes? DEFINE GRID/LIKE=temp[D=2]/Z=ZAX  gnew     !temp from data set 2
```

Examples: DEFINE GRID

1)
```
yes? DEFINE AXIS/T="1-JAN-1980":"31-DEC-1983":24 axday
yes? DEFINE GRID/LIKE=temp/T=axday gday
```
Define grid gday to be like the defining grid for temp but with a 4 year, daily interval time axis.

2) `yes? DEFINE GRID/LIKE=temp[D=ba022]/T=sst[D=nmc] gnmc3d`
Define grid gnmc3d like temp from data set ba022 but with the same time axis as sst from data set nmc.

3)
```
yes? DEFINE AXIS/X=140E:140W:.2 xnew
yes? DEFINE AXIS/Y=5S:5N:.2 ynew
yes? DEFINE AXIS/T="15-FEB-1982":"15-FEB-1984":48 tnew
yes? DEFINE GRID/X=xnew/Y=ynew/T=tnew gnew
```
Define grid gnew from new axes. The grid, gnew, will be normal (perpendicular) to Z.

4.4 DEFINE REGION

/I/J/K/L /X/Y/Z/T /DI/DJ/DK/DL /DX/DY/DZ/DT /DEFAULT

Defines or redefines a named region_name (first 4 characters are recognized).

```
yes? DEFINE REGION[/qualifiers] region_name
```

If the qualifier /DEFAULT is not given only those axes explicitly named will be stored. If the qualifier /DEFAULT is given all axes will be stored.

Command qualifiers for DEFINE REGION:

DEFINE REGION/**I=/J=/K=/L=/X=/Y=/Z=/T=**
Specifies region limits (=lo:hi or =val).

DEFINE REGION/**DI=/DJ=/DK=/DL=/DX=/DY=/DZ=/DT=**
Specifies a change in region relative to the current settings (=lo:hi or =val). See examples below.

DEFINE REGION/**DEFAULT**
Saves all axes and transformations, not just those explicitly given. Commonly, a GO script begins with "DEFINE REGION/DEFAULT save" and ends with "SET REGION save".

Examples: DEFINE REGION
1) `yes? DEFINE REGION/DEFAULT  save`
Stores the current default region under the name "save". The region may be restored at a later time by the command `yes? SET REGION save.`

2) `yes? DEFINE REGION/X   xonly`
Stores the current default X axis limits, only, as region xonly.

3) `yes? DEFINE REGION/DX=-5   xonly`
Stores the current default X axis limits minus 5 as region xonly.

4) `yes? DEFINE REGION/Y=20S:20N/Z   yanz`
Stores the given limits from the Y axis and the default Z axis limits.

5) `yes? DEFINE REGION/DEFAULT/L=5   l5`
Stores the current default region with the modification that L, the time step, is stored as 5.

6) `yes? DEFINE REGION/DL=-1:+1  lp2`
Stores an L region beginning 1 time step earlier and ending 1 time step later than the current default region as region lp2.

4.5 DEFINE VARIABLE

/QUIET /TITLE /UNITS

Allows the user to define a variable from a valid algebraic expression. Note: LET is an alias for DEFINE VARIABLE.

```
yes? DEFINE VARIABLE[/qualifiers]  name=expression
```

Example:

```
yes? LET SPEED = U^2 + V^2
```

Parameters

The expression may be any valid expression. See Chapter 3 section "Expressions" for a definition of valid expressions.

The name specified with DEFINE VARIABLE can be 1 to 24 characters in length - letters, digits, $ and _, beginning with a letter. Pseudo-variable, operator and function names are reserved and cannot be used (I, J, EQ, SIN,...). See Chapter 3 for recognized pseudo-variables, operators and functions.

If the name defined is the same as a variable name in a data set, the user-defined variable is used instead of the file variable. (Look for LET/D=d_set to control this behavior in future FERRET versions.)

To enter expressions in Reverse Polish ordering see SET MODE POLISH.

Examples:

1) `yes? DEFINE VARIABLE sum = a+b`
or equivalently

```
yes? LET sum = a+b
```

2) `yes? DEFINE VARIABLE/TITLE="velocity"/UNIT="m/sec" pos[T=@DDC]*0.01`
Defines velocity in m/sec from position, pos, in cm.

Command qualifiers for DEFINE VARIABLE:

DEFINE VARIABLE/**QUIET**
Suppresses message which, by default, tells you when you are redefining an existing variable. This qualifier is useful in command files.

DEFINE VARIABLE/**TITLE=**
Specifies a title (in quotation marks) for the user-defined variable. This title will be used to label plots and listings. If no title is specified the text of the expression will be used as the title. (See also SET VARIABLE/TITLE.)

DEFINE VARIABLE/**UNITS=**
Specifies the units (in quotation marks) of the variable being defined. (See command SET VARIABLE/UNITS.)

4.6 DEFINE VIEWPORT

/CLIP /ORIGIN /SIZE /TEXT /XLIMITS /YLIMITS

Defines a new viewport (a sub-rectangle of the graphics window).

```
yes? DEFINE VIEWPORT[/qualifiers]   view_name
```

Issuing the command SET VIEWPORT is best thought of as entering "viewport mode." While in viewport mode all previously drawn viewports remain on the screen until explicitly cleared with either SET WINDOW/CLEAR or CANCEL VIEWPORT. If multiple plots are drawn in a single viewport without the use of /OVERLAY the current plot will erase and replace the previous one; the graphics in other viewports will be affected only if the viewports overlap. If viewports overlap the most recently drawn graphics will always lie on top, possibly obscurring what is underneath. By default, the state of "viewport mode" is canceled.

Example:

```
yes? DEFINE VIEWPORT/XLIMITS=0,.5/YLIMITS=0,.5   LL
```

Defines a viewport that will place graphical output into the lower left quarter of the screen, and names the viewport "LL".

Command qualifiers for DEFINE VIEWPORT.

DEFINE VIEWPORT/**CLIP=**
This qualifier is obsolete; see XLIMITS= and /YLIMITS=. Specifies the location of the upper right corner of the viewport.

DEFINE VIEWPORT/**ORIGIN=**
This qualifier is obsolete; see /XLIMITS= and /YLIMITS=. Specifies the location of the lower left corner of the viewport.

DEFINE VIEWPORT/**SIZE=**
This qualifier is obsolete; see /XLIMITS and /YLIMITS. Specifies the scaling factor to use relative to the size of the full window.

DEFINE VIEWPORT/**TEXT=**
Controls shrinkage (or expansion) of text.

```
yes? DEFINE VIEWPORT/TEXT=n  view_name
```

Text appearance may become unacceptable due to viewport size and aspect specifications. A value of 1 produces text of the same size as in the full window; 0 < n < 1 shrinks the text; n > 1 enlarges text. Sensible values go up to about 2. Without the qualifier /TEXT, FERRET selects a text size that is appropriate to the size of the viewport.

DEFINE VIEWPORT/**XLIMITS=/YLIMITS=**
Specifies the portion of the full window to be used.

```
yes? DEFINE VIEWPORT/XLIMITS=x1,x2/YLIMITS=y1,y2   view_name
```

The values of the limits must be in the range [0,1]; they refer to the portion of the window (of height and length 1) which defines the viewport. Together, /XLIMITS and /YLIMITS replace the CLIP, ORIGIN and SIZE qualifiers in older FERRET versions.

5 EXIT

/COMMAND_FILE

When issued interactively this command terminates program FERRET.

When executed within a command file this command terminates the execution of the command file and return control to the level in FERRET that executed the file (the user or another command file).

Command qualifiers for EXIT:

EXIT/**COMMAND_FILE**
When executed from within a command file EXIT/COMMAND_FILE forces an immediate exit from FERRET rather than returning to the user or another command file.

6 FILE

The FILE command is an alias for SET DATA/EZ. All qualifiers and restrictions are identical to SET DATA/EZ

Example:

```
   yes? FILE/VARIABLES="u,v" velocities.dat
```

is equivalent to

```
   yes? SET DATA/EZ/VARIABLES="u,v" velocities.dat
```

7 FILL

Alias for CONTOUR/FILL (color-filled contour plot). All qualifiers and restrictions are identical to CONTOUR/FILL.

Example:

```
   yes? FILL/PAL=land_sea  etopo60
```

is equivalent to

```
   yes? CONTOUR/FILL/PAL=land_sea  etopo60
```

8 FRAME

Saves the current graphics display image as a frame in the movie file initialized with the command SET MOVIE. FRAME is also a qualifier for the "action" commands PLOT, CONTOUR, SHADE, VECTOR and WIRE.

```
   yes? CONTOUR my_var
   yes? FRAME
```

9 GO

Executes a list of commands stored in a file.

```
   yes? GO file_name
```

If no filename extension is specified a default of .jnl will be assumed. If the full path is specified then the filename must be enclosed in double quotation marks.

The GO command can pass arguments to the script (tool) it executes. See Chapter 1 section "Writing GO Tools" for more information. Arguments to the GO command may be separated by blanks or commas. To specify multiple words as a single argument, enclose them in quotation marks. To specify an argument that is deliberately omitted, use " " or two consecutive commas.

The response of FERRET to errors encountered during execution of the command file is determined by mode IGNORE_ERRORS. (See command SET MODE.)

The echoing of command file lines is controlled by mode VERIFY.

The GO command understands a special syntax called "relative version numbers". If a filename is specified for the GO command which has a version value of zero or less its value is interpreted as relative to the current highest version number. See Chapter 7 section "Relative version numbers" for a discussion of relative version numbers of files.

Note: The command SET MODE IGNORE_ERRORS is useful when rerunning past sessions which may have errors.

10 HELP

On Unix systems interactive FERRET help is available from the command line with the commands Fapropos, Fhelp and Ftoc. If multiple windows are not available on your system the ^Z key can be used to suspend the current FERRET session and access the help; the Unix command "fg" will then restore the suspended session.

See Chapter 1 section "Unix on-line help" for more information.

11 LABEL

/NOUSER

Places a label on the current plot; alias for PPL %LABEL. %LABEL is one of PPLUS's primitive plot commands. It places a label on the plot immediately after being issued (rather than deferring placement). PPLUS does not assign numbers to labels created with LABEL, so they cannot be manipulated as movable labels.

```
yes? LABEL xpos, ypos, center, angle, size   text
```

xpos, ypos	position in user units (world coordinates)
center	-1 left justification
	0 centered
	1 right justification
angle	angle in degrees, 0 degrees at 3 o'clock
size	size of text in inches

See Chapter 6 section "Labels" for examples.

Command qualifiers for LABEL:

LABEL/**NOUSER**
Locates labels in inches instead of user units (xpos and ypos are specified in inches rather than in world coordinates).

12 LET

The LET command is an alias for DEFINE VARIABLE. All qualifiers and restrictions are identical to DEFINE VARIABLE.

Example:

```
yes? LET A = B
```

is equivalent to

```
yes? DEFINE VARIABLE A = B
```

13 LIST

/I/J/K/L /ILIMITS /JLIMITS /KLIMITS /LLIMITS /XLIMITS /YLIMITS /ZLIMITS /TLIMITS /X/Y/Z/T /D /APPEND /FILE /FORMAT /HEADING /NOHEAD /ORDER /RIGID /SINGLE

Produces a listing of the indicated data.

```
LIST[/qualifiers]   [expression_1 , expression_2 , ...]
```

Example:

```
yes? LIST/Z=10 u , v , u^2 + v^2
```

Lists the 3 quantities specified using the current default data set and region (at depth 10).

Parameters
Expressions may be any valid expression. See Chapter 3 section "Expressions" for a definition of valid expressions. If multiple variables or expressions are specified they may be listed together in columns or in sequence depending on the /SINGLY qualifier. The expression(s) will be inferred from the current context if omitted from the command line.

If multiple expressions are given on the command line and /SINGLY is not specified, then the expressions must be conformable. See Chapter 3 section "Multi-dimensional expressions" for a definition of conformable expressions. Degenerate or single point axis limits will be promoted up (values repeated) as needed.

Example:

```
yes? LIST/I=1:3/J=1:2  i+j, i
```

Command qualifiers for LIST:

LIST/**I= /J= /K= /L=/X= /Y= /Z= /T=**

Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when evaluating the expression(s) being listed.

LIST/**ILIMITS=/JLIMITS=/KLIMITS=/LLIMITS=**

Specifies the size of the desired NetCDF output file independently from the actual data being saved. By specifying axis limits in excess of the saved expression's limits it is possible to /APPEND data later. (See Chapter 8 section "Simple Conversions Using FERRET" ex. 4).

LIST/**XLIMITS=/YLIMITS=/ZLIMITS=/TLIMITS=**

Specifies the size of the desired NetCDF output file independently from the actual data being saved. By specifying axis limits in excess of the saved expression's limits it is possible to /APPEND data later. (See Chapter 8 section "Simple Conversions Using FERRET" ex. 4).

LIST/**D=**

Specifies the default data set to be used when evaluating the expression(s) being listed.

LIST/**APPEND**

Use this qualifier together with the /FILE qualifier to indicate that the listed data should be appended to a pre-existing file. If no file exists by the name indicated a new file is created. This qualifier is not applicable to /FORMAT=GT. When used with /FORMAT=CDF it permits any data in the file to be over-written, new variables to be added to the file, and appending of new indices along the T axis of the variables in the file. This qualifier overrides the command CANCEL LIST/APPEND.

LIST/**FILE** [=file_name]

Names a file to receive the listed data. If /FILE is specified with no name then the default name is used from the SET LIST/FILE command.

Example:

```
yes? LIST/FILE=my_file.dat sst[D=coads_climatology]
```

See command SET LIST for further information on automatic filename generation.

LIST/**FORMAT=**

Specifies an output format (=format_choice) for the data to be listed.

```
yes? SET LIST/FORMAT=format_choice
```

or

```
yes? SET LIST/FORMAT
```

(use format set by SET LIST/FORMAT)

Format choices:

FORTRAN format	produces ASCII output
"UNFORMATTED"	produces unformatted (binary) output

"CDF"	produces NetCDF format output
"GT"	produces TMAP GT format

This command has the same function as SET LIST/FORMAT except that it does not affect future LIST commands. See command SET LIST/FORMAT for detailed documentation.

Notes for LIST/FORMAT:

1) All output values, regardless of the /FORMAT designation, will be of type single precision floating point. For FORTRAN output formats this means all numerical field specifiers must be "F", "E", or "G".

2) For FORTRAN-formatted and UNFORMATTED (binary) output, the contents of a single output "record" are determined by the /ORDER qualifier. For example, each record will be a line of Y values for LIST/ORDER=YX. If /ORDER is omitted, the records will be the first output axis of greater than unity length taken in the order X, Y, Z, then T. FORTRAN-formatted output records may be further split by the usual rules of FORTRAN output formatting.

3) FORTRAN formats must be enclosed in parentheses. If blanks are included in the format it must be enclosed in quotation marks. Output strings are permitted in the format.

   Example:
   ```
   yes? LIST/FORMAT=("The temperature is:", F6.3)  sst[X=180, Y=0]
   ```

4) The default listing style includes labels for the rows and columns of the output. When a FORTRAN format is specified, these labels are omitted.

5) On Unix systems the /FORMAT=UNFORMATTED specifier produces FORTRAN-style variable-length records. On most implementations this means that a 4 byte field containing the record length begins and ends each record of data.

6) The command alias SAVE is provided for the commonly used LIST/FORMAT=CDF. NetCDF outputs are self-documenting, including grid definitions. The output files can be used as input with the command USE - alias for SET DATA/FORMAT=CDF. See command SAVE for further notes about NetCDF files.

LIST/**HEAD**
Precedes listing with a heading describing data set, variable and region. This qualifier overrides the CANCEL LIST/HEAD command. It is applicable only to the default output format and to FORTRAN formats.

LIST/**NOHEAD**
Does not precede listing with a heading describing data set, variable and region. This qualifier overrides the SET LIST/HEAD command.

LIST/**ORDER=**
Specifies the order (ORDER=permutation) in which axes are to be laid out in the listing.

Examples:
```
  yes? LIST/ORDER=XY sst        !X varies fastest
  yes? LIST/ORDER=YX sst        !Y varies fastest
```

The "permutation" string may be any permutation of the letters X,Y,Z and T. /ORDER is applicable only to /FORMAT=unf and FORTRAN formats.

LIST/**RIGID**
Valid only with /FORMAT=CDF. Indicates that FERRET should not create a NetCDF "record" axis as the time axis for any of the variables listed with this command. Time axes are, instead, of fixed length and the /APPEND qualifier is not usable to extend the listing.

LIST/**SINGLY**
When the /SINGLY qualifier is specified the entire listing of each expression including (optional) heading and all data is completed before proceeding to the next expression.

By default the expressions are not listed singly - each line contains one value of each expression. The qualifier has no effect if only a single expression is specified. If the /FILE qualifier is specified to use automatic filename generation and /APPEND is not specified, then each expression is listed to a separate file.

14 LOAD
/I/J/K/L /X/Y/Z/T /D /NAME /PERMANENT /TEMPORARY

Loads a variable or expression into memory.

```
  yes? LOAD[/qualifiers]   [expression_1 , expression_2 , ...]
```

Loading may speed execution of later commands which will require the loaded data. Often it is helpful to LOAD a large region of data encompassing several small regions in which the analysis will be pursued.

Load interacts with the current context exactly as other "action" commands CONTOUR, PLOT, SHADE, VECTOR, LIST, etc. do.

Parameters
Expressions may be any valid expression. See Chapter 3 section "Expressions" for a definition of valid expressions. If multiple variables or expressions are specified they are treated in sequence. The expression(s) will be inferred from the current context if omitted from the command line.

Command qualifiers for LOAD:

LOAD/**I=/J=/K=/L=/X=/Y=/Z=/T=**
Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when evaluating the expression(s) being loaded.

LOAD/**D=**
Specifies the default data set to be used when evaluating the expression(s) being loaded.

LOAD/**NAME**
Obsolete. Provided for compatibility with much older FERRET versions.

LOAD/**PERMANENT**
Data loaded with LOAD/PERMANENT are kept in memory until a LOAD/TEMPORARY command is given that refers to the same data. See command LOAD/TEMPORARY. Note that this command may cause memory fragmentation. It should generally be given immediately following CANCEL MEMORY and preferably is used only to load file variables (as opposed to expressions).

LOAD/**TEMPORARY** (default)
Data loaded with LOAD or LOAD/TEMPORARY is brought into memory but may be unloaded based on a priority scheme of least recent use when memory space is required.

15 MESSAGE

/CONTINUE /QUIET

Displays a message at the terminal.

```
yes? MESSAGE text
```

By default a carriage return is required from the keyboard for program execution to continue (used to halt the execution of a command file).

Command qualifiers for MESSAGE:

MESSAGE/**CONTINUE**
Continues program execution following the display of the message text without waiting for a carriage return from the operator.

MESSAGE/**QUIET**
Waits for a carriage return from the operator but does not supply a prompt for it.

16 PALETTE

Alias for PPL SHASET SPECTRUM=. Specifies or restores the default color .

```
yes? PALETTE pal_name
```

The argument is the name of a palette file. Many palettes are included in the FERRET distribution. Try the Unix command "Fpalette '*'" to see a list of available palette files.

Some of the palettes are designed for particular needs. "centered.spk", for example, emphasizes the contrast between positive and negative shade levels. "land_sea.spk" uses blue tones for negative values and browns and greens for positive values, making it suitable for topography displays.

Palette files end in the file suffix .spk, but the suffix is not necessary when specifying a palette. Use `GO try_palette pal_name` to display a palette. The GO files "exact_color.jnl" and "squeeze_colors.jnl" can be used to modify palettes. You can also create new palette files with a text editor. See Chapter 6 section "Shade and fill colors" for the format of a palette file.

PALETTE with no argument restores the default palette. When you use the qualifier /PALETTE= in conjunction with /SET_UP, PPLUS makes the specified color spectrum the new default palette, and all subsequent shaded or color-filled plots will use that palette as the default. To restore the previous palette to the default, use PALETTE with no argument after your customization.

17 PLOT

/I/J/K/L /X/Y/Z/T /D /FRAME /LINE /NOLABEL /OVERLAY
/SET_UP /SYMBOL /TITLE /TRANPOSE /VS
/XLIMITS /YLIMITS

Produces a line plot.

```
yes? PLOT[/qualifiers]   [expression_1 , expression_2 , ...]
```

The indicated expression(s) must represent a line (not a plane) of data (PLOT/VS is an exception). Unless the /VS qualifier is used, the independent variable is the underlying coordinate axis for this line of data.

Example:

```
yes? PLOT/l=1:100 sst
```

produces a time series plot of the first 100 points of sst.

Parameters
The argument(s) for PLOT specify the variable or expression to be plotted.

When the /VS qualifier is used the indicated expressions may have any geometry in 4D space but they must match in the total number of points in each expression. The points are associated in the order of their underlying axes. When the /VS qualifier is not used the indicated expression(s) must describe a line (not a plane) of data.

The expression(s) are inferred from the current context if omitted from the command line - i.e. if no expression is given then the argument most recently given is used, or the default expression may be explicitly set with SET EXPRESSION.

Command qualifiers for PLOT:

PLOT/**I=/J=/K=/L=/X=/Y=/Z=/T=**
Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when evaluating the expression(s) being plotted.

PLOT/**D=**
Specifies the default data set to be used when evaluating the expression(s) being plotted.

PLOT/**FRAME**
Causes the graphic image produced to be captured as an animation frame and written to the movie file specified by SET MOVIE.

PLOT/**LINE[=]**
The /LINE qualifier without =n causes the PLOT command to connect the plotted points with a line regardless of the state of the /SYMBOLS qualifier.

/LINE=n specifies the line style. "n" is an integer between 1 and 18. Line style "1" is always a solid line in the foreground color (black or white). Other line styles are device dependent (colors or dash patterns). For color devices, n=1-6 draws single-thickness lines each a different color. n=7-12 draws double-thick lines in the same color order, and n=13-18 draws triple-thick lines. See Chapter 6 section "Text and line colors" for a chart of the default colors.

PLOT/**NOLABELS**
Suppresses all plot labels except axis labels.

PLOT/**OVERLAY**
Causes the indicated field(s) to be overlaid on the existing plot. This qualifier can also be used to overlay lines or symbols on 2D plots (SHADE, CONTOUR, or VECTOR) provided the axis scalings are appropriate.

PLOT/**SET_UP**
Performs all the internal preparations required by program FERRET for plotting but does not actually render the plot. The command PPL can then be used to make changes to the plot

prior to producing output with the PPL PLOT command. This makes possible certain customizations that are not possible with FERRET command qualifiers. See Chapter 6.

PLOT/**SYMBOL[=]**
The /SYMBOL qualifier causes the PLOT command to mark each plotted point with a symbol. If the /LINE qualifier is given too the symbols are also connected with a line; if /LINE is omitted no connecting line is drawn.

Optionally, the symbol number may be explicitly specified as an integer value between 1 and 88. The integer refers to the PPLUS plot marker numbers (e.g. 1 for x, 3 for +, etc.). Type "GO show_symbols" and "GO show_88_syms" at the FERRET prompt to see available symbols and their reference numbers. The symbols are also documented on page 1 of the document $FER_DIR/doc/pplus_fonts.ps.

PLOT/**TITLE=**
Allows user to specify a plot title (enclosed in quotation marks). Without this qualifier FERRET selects a title based on information about the expression(s). To include font change and color_thickness specifications (e.g., @TI@C002) in the title string, it is necessary either to CANCEL MODE ASCII or to include a leading ESC (escape) character.

PLOT/**TRANSPOSE**
Causes the horizontal and vertical axes to be interchanged. By default the X axis is drawn horizontally on the plot and the Y and Z axes are drawn vertically. For Y-Z plots the Z data axis is vertical by default.

PLOT/**VS**
Specifies that the first expression given in the command line is to be used as the independent axis.

Example:

```
   yes? PLOT/Y=20S:20N/X=180/T=27740:27741/Z=100/VS  temp , salt
```

Produces a plot of salinity (vertical axis) against temperature (horizontal axis) along the indicated range of latitudes and times. The plot will be labeled "salt"; the vertical (dependent) variable is the one that determines the key. The qualifier /TRANSPOSE can be used in conjunction with /VS to further manipulate the labeling and axis orientation.

PLOT/VS is often used in conjunction with /SYMBOL to produce scatter plots.

PLOT/**XLIMITS=**
Specifies axis range and tic interval for the X axis. Without this qualifier FERRET selects a reasonable range.

```
   yes? PLOT/XLIMITS=lo:hi:[increment]  [expression(s)]
```

The optional "increment" parameter determines tic mark spacing on the axis. If the increment is negative, the axis is reversed.

PLOT/**YLIMITS=**
Specifies the axis range and tic interval for the Y axis. See /XLIMITS.

18 PPLUS

/RESET

Invokes PPLUS ("PLOT PLUS" written by Don Denbo) to execute a command or commands.

```
yes? PPLUS              !(also PPL); invokes PPLUS interactively
      or
yes? PPL pplus_command  !executes a single PPLUS command
      or
yes? PPL/RESET          !restores PPLUS to start-up defaults
```

Example:

```
yes? PPL CROSS 1  !reference line through zero
```

Executes the PPLUS command "CROSS" and immediately returns control to FERRET.

When PPLUS is invoked interactively the prompt is "`PPL>`" instead of the usual "`yes?`". The EXIT command given at the "`PPL>`" prompt returns control to FERRET.

See Chapter 6 for more information on FERRET/PPLUS interactions. A complete list of PPLUS commands is in *PLOT PLUS for FERRET Users Guide*.

Command Qualifiers for PPLUS:

PPLUS/**RESET**
Restores PPLUS to start-up settings.

19 QUIT

Alias for EXIT; also just Q.

20 REPEAT

/I/J/K/L /X/Y/Z/T

Repeats a command or group of commands over a range of values along an axis.

```
yes? REPEAT/q=lo:hi[:increment] COMMAND
```

The units of lo,hi and increment are the units of the underlying grid axis if the qualifier is X,Y,Z or T. The qualifiers I,J,K or L advance the repeat loop by incrementing the indicated index (the default index increment is 1). Use SHOW GRID to examine the axis units (if the units are not displayed try CANCEL MODE LATITUDE, LONGITUDE or CALENDAR as appropriate). To run the loop from the highest value decreasing towards the lowest value, specify increment to be less than zero. Any command or group of commands that can be specified at the command line can also be given as an argument to REPEAT. If MODE VERIFY is SET, the current loop index is displayed at the console as REPEAT executes.

Examples:

1) `yes? REPEAT/L=1:240 CONTOUR/Y=30S:50N/X=130E:70W/LEV/FRAME sst`

Produces a 240 frame movie of sea surface temperature.

2) `yes? REPEAT/Z=300:0:-30 GO compz`

Executes the command file compz.jnl at Z=300, Z=270, ..., Z=0.

3) `yes? REPEAT/L=1:250:5 (GO set_up; CONTOUR sst; FRAME)`

Repeats three commands - execution of a GO script, CONTOUR and FRAME - for each timestep specified.

Command qualifiers for REPEAT:

REPEAT**/I=/J=/K=/L=/X=/Y=/Z=/T=**

Repeats the requested command(s) for the specified range of axis subscripts (I,J,K, or L) or axis coordinates (X,Y,Z, or T). Note that when T axis limits are specified as dates, the units of increment are hours.

21 SAVE

The SAVE command is an alias for LIST/FORMAT=CDF. All qualifiers and restrictions are identical to LIST/FORMAT=CDF.

Example:

```
yes? SAVE temp, salt
```

is identical to

```
yes? LIST/FORMAT=CDF temp, salt
```

Notes:

1) Gaps in NetCDF outputs are filled with the missing value flag of the variable being written. (See Chapter 3 section "Missing value flags".) In the example below, if "temp" and "salt" share the same time axis then the L=2:4 values of salt will be so filled.

```
yes? SAVE/FILE=test.cdf   temp[L=1:5], salt[L=1], salt[L=5]
```

2) Transformations which compress an axis to a point produce results which FERRET regards as time-independent. thus, this 12 month average:

```
yes? SAVE/FILE=annual.cdf  sst[L=1:12@AVE]
```

creates a NetCDF file *with no time axis*. It would not be possible to append the average of the next 12 months as the next time step of this file. However, a time location can be inherited from another variable. In this example, we inherit the time axis of "timestamp" in order to create a time axis in the NetCDF file.

```
yes? DEFINE AXIS/T="1-JUL-1980":"1-JUL-1985"/UNIT=year  tannual
yes? DEFINE GRID/T=tannual  gannual
yes? LET  timestamp = T[G=gannual] * 0          !always 0
yes? LET  sst_ave = sst[L=1:12@AVE] + timestamp
yes? SAVE/FILE=annual.cdf  sst_ave[L=1]
yes? LET  sst_ave = sst[L=13:24@AVE] + timestamp
yes? SAVE/FILE=annual.cdf/APPEND  sst_ave[L=2]
.
.
. etc.
```

3) Background documentation about the definition and data set of origin for a variable are saved in the "history" attribute of a variable when it is first saved in the NetCDF file. If the definition of the variable is then changed, and more values are inserted into the file using SAVE/APPEND, the modified definition will NOT be documented in the output file. If the new definition changes the defining grid for the variable the results will be unpredictable.

22 SET

Sets features of the operating environment for program FERRET.

Generally, features may be toggled on and off with SET and CANCEL. Features affected by SET may be examined with SHOW (see also CANCEL and SHOW).

22.1 SET AXIS

Indicates that an axis is to be treated as a modulo axis (the first point "wraps" and follows the last point, as in a longitude axis). Valid only with /MODULO.

```
yes? SET AXIS/MODULO  x_ax
```

22.2 SET DATA_SET

/FORMAT /RESTORE /SAVE /EZ

SET DATA/EZ /COLUMNS /FORMAT /GRID /SKIP /TITLE /VARIABLE

Specifies ASCII, binary, NetCDF, GT or TS-formatted data set(s) to be analyzed.

1) ASCII or binary:

```
yes? SET DATA/EZ[/qualifiers] data_set1, data_set2, ...
```

or equivalently, with alias FILE:

```
yes? FILE[/qualifiers] data_set1, data_set2, ...
```

2) NetCDF:

```
yes? SET DATA/FORMAT=cdf  NetCDF_file
```

or equivalently, with alias USE

```
yes? USE NetCDF_file
```

3) GT or TS-formatted:

```
yes? SET DATA data_set1, data_set2, ...
```

In the case of GT or TS-formatted files, an extension of .des is assumed.
A previously SET data set can be SET by its reference number, as shown by SHOW DATA, rather than by name.

If a Unix filename includes a path (with slashes) then the full path plus name must be enclosed in double quotation marks.

Note: Maximum simultaneous data sets: 60 (as of FERRET ver. 3.1). Use CANCEL DATA if the limit is reached.

Command qualifiers for SET DATA_SET:

SET DATA/**FORMAT=**

Specifies the format of the data set(s) being SET. Allowable values for "file_format" are "cdf", "free", "unformatted", "stream" or a FORTRAN format in quotation marks and parentheses.

```
yes? SET DATA/FORMAT=file_format [data_set_name_or_number]
```

Valid arguments for /FORMAT=

1) free (default for SET DATA/EZ)

   To use the format "free" a file must consist entirely of numerical data separated by commas, blanks or tabs.

2) cdf

   If SET DATA/FORMAT=cdf (alias USE) is used, the data file must be in CDF format. The default filename extension is ".cdf".

Example:

```
yes? SET DATA/FORMAT=CDF  my_netcdf
```

or equivalently,

```
yes? USE  my_netcdf
```

3) unformatted

   To use the format "unformatted" the data must be floating point, binary, FORTRAN-style records with all of the desired data beginning on 4-byte boundaries. This option expects 4 bytes of record length information at the beginning and again at the end of each record. The "-" designator (see /VARIABLES) can be used to skip over unwanted 4-byte quantities (variables) in each record. See Chapter 2 section "Binary data".

4) FORTRAN format string

   FORTRAN format specifications should be surrounded by parentheses and enclosed in quotation marks.
   Example:

```
yes? SET DATA/EZ/FORMAT="(5X,F12.0)" my_data_set
```

or equivalently,

```
yes? FILE/FORMAT="(5X,F12.0)" my_data_set
```

5) stream   (FERRET version 3.1)

   /FORMAT=stream is used to indicate that a file contains either unstructured binary output (typical of C program output) or fixed-length records suitable for direct access (all records of equal length, no record length information embedded in the file). With caution it is also possible to read FORTRAN variable-length record output. This sort of file is typically created by "quick and dirty" FORTRAN code which uses the simplest FORTRAN OPEN statement and outputs entire variables with a single WRITE statement.

   This format specifier allows you to access any contiguous stretch of 4-byte values from the file. The /SKIP=n qualifier specifies how many values should be skipped at the file start. The /GRID=name qualifier specifies the grid onto which the data should be read and therefore the number of values to be read from the file (the number of points in the grid). *Note that an attempt to read more data than the file contains, or to read record length information, will result in a fatal FORTRAN error on UNIX systems and will crash the FERRET program.*

   For multiple variables, use the /COLUMNS=n specifier to specify how many 4-byte values separate each variable in the file. Each variable is assumed to represent a contiguous stream of values within the file and all variables are assumed to possess the same number of points. (A "poor man's" method is to create multiple Unix soft links pointing to the same file and multiple SET DATA/EZ commands to specify one variable from each link name.)

   See Chapter 2 section "Binary data" for further discussion and examples of binary types.

SET DATA/**RESTORE**

Restores the current default data set number that was saved with SET DATA/SAVE.

This is useful in creating GO files that perform their function and then restore FERRET to its previous state.

SET DATA/**SAVE**
Saves the current default data set number so it can be restored with SET DATA/RESTORE.

This is useful in creating GO files that perform their function and then restore FERRET to its previous state.

SET DATA/**EZ**
Accesses data from an ASCII or unformatted file that is not in a standardized format (TMAP or NetCDF). The command FILE is an alias for SET DATA/EZ.

```
yes? SET DATA/EZ[/qualifiers]   ASCII_or_binary_file
```

or, equivalently,

```
yes? FILE[/qualifiers]  ASCII_or_binary_file
```

Example:

```
yes? FILE/VARIABLE=my_var my_data.dat
```

See Chapter 2 section "ASCII data" for more information and examples.

Command qualifiers for SET DATA_SET/EZ:

SET DATA/EZ/**COLUMNS=n**
Specifies the number of columns in the EZ data file.

By default the number of columns is assumed to be equal to the number of variables (including "-"'s) specified by the /VARIABLES qualifier.

SET DATA/EZ/**GRID=**
Specifies the defining grid for the data in the EZ data set. The argument can be the name of a grid or the name of a variable that is already defined on the desired grid.

Example:

```
 yes? SET DATA/EZ/GRID=sst[D=coads] snoopy
```

This is the mechanism by which the shape of the data (1D along T axis, 2D in the XY plane, etc.) is specified. By default FERRET uses grid EZ, a line of up to 20480 points oriented along the X axis.

SET DATA/EZ/**ORDER=**   (FERRET version 3.11)
Specifies the order (ORDER=permutation) in which axes are to be read.

Examples:

```
yes? FILE/ORDER=XY sst        !X varies fastest
yes? LIST/ORDER=YX sst        !Y varies fastest
```

The "permutation" string may be any permutation of the letters X,Y,Z and T.

SET DATA/EZ/**SKIP=n**
Specifies the number of records to skip at the start of an EZ data set before beginning to read the data. By default, no records are skipped.

SET DATA/EZ/**TITLE=**
Associates a title with the data set.

```
yes? SET DATA/EZ/TITLE="title string"  file_name
```

This title appears on plotted outputs at the top of the plot.

SET DATA/EZ/**VARIABLES=**
Names the variables of interest in the file. Default is v1.

```
yes? FILE/VARIABLES="var1,var2,..."  file_name
```

FERRET always assumes that successive values in the data file represent successive variables. For example, if there are three variables in a file, the first value represents the first variable, the second represents the second variable, the third the third variable, and the fourth returns to representing the first variable.

Variable names may be 1 to 24 characters (letters, digits, $ and _) beginning with a letter. To indicate a column is not of interest use "-" for its name.

Example: (the third column of data will be ignored)

```
yes? SET DATA/EZ/VARIABLES="temp,salt,-,u,v" ocean_file.dat
```

22.3 SET EXPRESSION

Specifies the default context expression. When FERRET's "action" commands (PLOT, CONTOUR, SHADE, VECTOR, WIRE, etc.) are issued with no argument, the default context expression is used. This is the expression last used as argument to an action command, or it may be set explicitly with SET EXPRESSION. See Chapter 3 section "Expressions" for a full list of action commands.

```
yes? SET EXPRESSION expr1 , expr2 , ...
```

Examples:

1) `yes? SET EXPRESSION temp`

Sets the current expression to "temp".

2) `yes? SET EXPRESSION u , v , u^2 + v^2`

Set the current expressions to "u , v , u^2 + v^2"

22.4 SET GRID

/RESTORE /SAVE

Specifies the default grid for abstract expressions. Type "GO wire_frame" at the FERRET prompt for an example of usage.

```
yes? SET GRID[/qualifier] [grid_or_variable_name]
```

Examples:

```
yes? SET GRID sst[D=coads]

yes? SET GRID     ! use grid from last data accessed
```

See Chapter 4 "Grids and Regions".

Command qualifiers for SET GRID:

SET GRID/**RESTORE**

Restores the current default grid last saved by SET GRID/SAVE. Useful together with SET GRID/SAVE to create GO files that restore the state of FERRET when they conclude.

SET GRID/**SAVE**

Saves the current default grid to be restored later. Useful together with SET GRID/RESTORE to create GO files that restore the state of FERRET when they conclude.

22.5 SET LIST

/APPEND /FILE /FORMAT /HEADING /PRECISON

Uses SET LIST to specify the default characteristics of listed output.

```
yes? SET LIST/qualifiers
```

The state of the list command may be examined with SHOW LIST. See command CANCEL LIST and LIST.

Command qualifiers for SET LIST:

SET LIST/**APPEND**
Specifies that by default the listed output is to be appended to a pre-existing file. Cancel this state with CANCEL LIST/APPEND.

SET LIST/**FILE=**
Specifies a default file for the output of the LIST command

```
yes? SET LIST/FILE=filename
```

The filename specified in this way is a default only. It will be used by the command

`yes? LIST/FILE variable` but will be ignored in

```
yes? LIST/FILE=snoopy.dat variable
```

FERRET generates a filename based on the data set, variable and region if the filename specified is "AUTO". The resulting name is often quite long but may be shortened by following "AUTO" with a minus sign and the name(s) of the axes to exclude from the filename.
Note: the region information is not used in automatic NetCDF output filenames.
Examples:

```
yes? SET LIST/FILE=AUTO
yes? LIST/L=500/X=140W:110W/Y=2S:2N/FILE sst[D=coads]
```

Sends data to file WcoadsSST.X140W110WY2S2NL500.

```
yes? SET LIST/FILE=AUTO-XY
yes? LIST/L=500/X=140W:110W/Y=2S:2N/FILE sst[D=coads]
```

Sends data to file WcoadsSST.L500.

SET LIST/**FORMAT=**
Specifies an output format for the LIST command. (When a FORTRAN format is specified the row and column headings are omitted from the output.)

```
yes? SET LIST/FORMAT=option
yes? SET LIST/FORMAT    !reactivate previous format
```

Options:

FORTRAN format	produces ASCII output
"UNFORMATTED"	produces unformatted (binary) output
"CDF"	produces NetCDF output
"GT"	produces TMAP GT format

Examples:

1) `yes? SET LIST/FORMAT=(1X,12F6.1)`

Specifies a FORTRAN format (without row or column headings).

2) `yes? SET LIST/FORMAT=UNFORMATTED`

Specifies binary output. (FORTRAN variable record length record structure.)

Notes:

1) When using GT format all variables named in a single LIST command will be put into a single GT-formatted timestep.
2) Very limited error checking will be done on FORTRAN formats.
3) FORTRAN formats are reused as necessary to output full record.
4) Latitude axes are listed south to north when /FORMAT is specified.

SET LIST/**HEAD**

Specifies that ASCII output is to be preceded by a heading that documents data set, variable and region. Cancel the heading with CANCEL LIST/HEAD.

SET LIST/**PRECISION**

Specifies the data precision (number of significant digits) of the output listings. This qualifier has no effect when /FORMAT= is specified.

```
yes? SET LIST/PRECISION=#_of_digits
```

22.6 SET MODE

/LAST

Specifies special operating modes or states for program FERRET.

```
yes? SET MODE[/LAST] mode_name[:argument]
```

MODE	DESCRIPTION	DEFAULT STATE
ASCII_FONT	imposes PPLUS ASCII font types on plot labels	set
CALENDAR	uses date strings for T axis (vs. time step values)	set
DEPTH_LABEL	uses "DEPTH" as Z axis label	set
DESPERATE	attempts calculations too large for memory	canceled
DIAGNOSTIC	turns on internal program diagnostic output	canceled
GUI	unsupported; used in GUI development	
IGNORE_ERROR	continues command file after errors	canceled
INTERPOLATE	automatically interpolates data between planes	canceled
JOURNAL	records keyboard commands in a journal file	set
LATIT_LABEL	uses "N" "S" notation for labelling latitudes	set
LONG_LABEL	uses "E" "W" notation for labelling longitudes	set
METAFILE	captures graphics in GKS metafiles	canceled
POLISH	interprets expressions in Reverse Polish order	canceled
PPLLIST	listed output from PPLUS is directed to the named file	canceled
REFRESH	refreshes graphics on systems lacking "backing store"	canceled
SEGMENT	utilizes GKS segment storage	set
STUPID	does not cache data in memory (diagnostic)	canceled
VERIFY	displays each command file line as it is executed	set
WAIT	waits for carriage return after each plot	canceled

Command qualifiers for SET MODE:

SET MODE/**LAST**
Resets mode to its last state.

```
yes? SET MODE/LAST mode_name
```

Example: (a command file that will not alter FERRET modes)

```
yes? SET MODE IGNORE_ERRORS      ! 1st line of command file
         .
         . ... code which may encounter errors
         .
yes? SET MODE/LAST IGNORE_ERRORS ! last line of command file
```

22.6.1 SET MODE ASCII_FONT

The SET MODE ASCII_FONT command causes program FERRET to precede plot labels with the PPLUS font descriptor "@AS" (ASCII SIMPLEX font). This assures that special characters (e.g. underscores) are faithfully reproduced. For special plots it may be desirable to use other fonts (e.g., to obtain subscripts). CANCEL MODE ASCII_FONT is for these cases.

default state: set

22.6.2 SET MODE CALENDAR

SET MODE CALENDAR causes program FERRET to output times in date/time format (instead of time axis time step values). This affects both plotted and listed output.

This mode accepts an optional argument specifying the degree of precision for the output date. If the argument is omitted the precision is unchanged from its last value.

default state: set (argument: minutes)

Arguments
SET MODE CALENDAR accepts the following arguments:

Argument	Equivalent precision	
SECONDS	-6	
MINUTES	-5	(default)
HOURS	-4	
DAYS	-3	
MONTHS	-2	
YEARS	-1	

The argument is uniquely identified by the first two characters.

Example:

```
yes? SET MODE CALENDAR:DAYS
```

Causes times to be displayed in the format dd-mmm-yyyy.

When CALENDAR mode is canceled the "equivalent" in the table above determines the precision of the time steps displayed exactly as in SET MODE LONGITUDE.

22.6.3 SET MODE DEPTH_LABEL

SET MODE DEPTH_LABEL causes FERRET to label Z coordinate information in the units of the Z axis. This affects both plotted and listed output. This mode accepts an optional argument specifying the degree of precision for the output. If the argument is omitted the precision is unchanged from its last value.

```
yes? SET MODE DEPTH:argument
```

default state: set (argument: -4)

Arguments
See SET MODE LONG for a detailed description of precision control.

22.6.4 SET MODE DESPERATE

FERRET checks the size of the component data required for a calculation in advance of performing the calculation. If the size of the component data exceeds the value of the MODE DESPERATE argument FERRET attempts to perform the calculation in pieces.

For example, the calculation "LIST/I=1/J=1 U[K=1:100,L=1:1000@AVE]" requires 100*1000=100,000 points of component data although the result is only a line of 100 points on the K axis. If 100,000 exceeds the current value of the MODE DESPERATE argument FERRET splits this calculation into smaller sized chunks along the K axis, say, K=1:50 in the first chunk and K=51:100 in the second.

FERRET is also sensitive to the performance penalties associated with reading data from the disk. Splitting the calculation along axis of the stored data records can require the data to be read many times in order to complete the calculation. FERRET attempts to split calculations along efficient axes, and will split along the axis of stored data only in desperation, if MODE DESPERATE is SET.

Example:

```
yes? SET MODE DESPERATE:5000
```

default state: canceled (default argument: 80000)

Note: Use MODE DIAGNOSTIC to see when splitting is occuring.

Arguments
Use SHOW MEMORY/FREE to get a notion of the total memory available. The product of "total memory blocks" times "memory block size" is the total words of internal storage - to be compared to the argument of SET MODE DESPERATE.

By default the argument is set at one tenth of available memory. It may be raised or lowered depending on the number and size of simultaneous components needed for calculations. The upper bound for the argument is the total number of words of internal storage. The lower bound is "memory block size".

22.6.5 SET MODE DIAGNOSTIC

SET MODE DIAGNOSTIC causes FERRET to display diagnostic information in real-time about its internal functioning. It is intended to help FERRET developers diagnose performance problems by displaying what the FERRET memory management subsystem is doing. The message "strip gathering on xxx axis" indicates that FERRET has broken up a calculation into smaller pieces. Subsequent "strip" and "gathering" messages indicate that sub-regions of the calculations are being processed and brought together.

default state: canceled

22.6.6 SET MODE IGNORE_ERROR

SET MODE IGNORE_ERROR causes FERRET to continue execution of a command file despite errors encountered. (See command GO.)

default state: canceled

22.6.7 SET MODE INTERPOLATE

SET MODE INTERPOLATE affects the interpretation of world coordinate specifiers (/X,/Y,/Z and /T) in cases where the position is normal to the plane in which the data is being examined. When this mode is SET and a world coordinate is specified which does not lie exactly on a grid point, FERRET automatically interpolates from the surrounding grid point values. When this mode is canceled, the same world coordinate specification is shifted to the grid point of the grid box that contained it before computations were made (see examples).

default state: canceled

Example:
If the grid underlying the variable temp has points defined at Z=5 and at Z=15 (with the grid box boundary at Z=10) and data is requested at Z=12 then

```
yes? SET MODE INTERPOLATE
yes? LIST/T=18249/X=130W:125W/Y=0:3N/Z=12 temp
```

lists temperature data in the X-Y plane obtained by interpolating between the Z=5 and Z=15 planes. Whereas,

```
yes? CANCEL MODE INTERPOLATE
yes? LIST/T=18249/X=130W:125W/Y=0:3N/Z=12 temp
```

lists the data at Z=15. The output documentation always reflects the true location used.

22.6.8 SET MODE JOURNAL

SET MODE JOURNAL causes FERRET to record all commands issued in a journal file. Output echoed to this file may be turned on and off via mode JOURNAL at any time.

default state: set

The optional argument to MODE JOURNAL specifies the name of the output journal file -with no argument, the default name "ferret.jnl" is used. Journal files for successive FERRET sessions are handled by version number. See Chapter 7 section "Output file naming".

22.6.9 SET MODE LATIT_LABEL

SET MODE LATIT_LABEL causes FERRET to output latitude coordinate information in degrees N/S format (instead of the internal latitude coordinate). This affects both plotted and listed output.

This mode accepts an optional argument specifying the degree of precision for the output. If the argument is omitted the precision is unchanged from its last value.

Example:

```
yes? SET MODE LAT:2
```

default state: set (argument: 1)

Arguments
See command SET MODE LONG for a detailed description of precision control.

22.6.10 SET MODE LONG_LABEL

SET MODE LONG_LABEL causes FERRET to output longitude coordinate information in degrees E/W format (instead of the internal longitude coordinate). This will affect both plotted and listed output.

This mode accepts an optional argument specifying the degree of precision for the output. If the argument is omitted the precision will be unchanged from its last value.

Example:

```
yes? SET MODE LONG:2
```

default state: set (argument: 1)

Arguments
The argument of SET MODE LONG is an integer specifying the precision. If the argument is positive or zero it specifies the maximum number of decimal places to display. If the argument is negative it specifies the maximum number of significant digits to display.

Examples:

Suppose the longitude to be displayed is 165.23W. Then

```
yes? SET MODE LONG:1    will produce 165.2W
yes? SET MODE LONG:-3   will produce 165W
```

When LONG mode is canceled the argument still determines the output precision.

22.6.11 SET MODE METAFILE

SET MODE METAFILE causes FERRET to capture all graphics in metafiles. These metafiles can later be routed to various devices to obtain hard copy output.

The optional argument to MODE METAFILE specifies the name of the output metafile -with no argument, the default name "metafile.plt" is used. Multiple output files (i.e. successive plots) are handled by version number. See Chapter 7 section "Output file naming".

See Chapter 7 section "Hard copy" for details on generating hard copy.

Example:

```
yes? SET MODE METAFILE:june_sst.plt
```

default state: canceled (default argument when set: "metafile.plt")

22.6.12 SET MODE POLISH

The SET MODE POLISH command causes program FERRET to expect algebraic expressions to be entered in Reverse Polish order.

This mode exists only to assist with compatibility with earlier versions of FERRET. It has no efficiency advantages.

default state: canceled

22.6.13 SET MODE PPLLIST

Directs listed output from PPLUS commands (e.g. PPL LIST LABS) to the specified file. This mode is useful for creating scripts that customize plots. The user can specify the name of the output file by giving it as an argument, otherwise file name "ppllist.out" is assigned.

Example:

```
yes? SET MODE PPLLIST:plot_symbols.txt
yes? PPL LISTSYM
yes? SPAWN grep "WIDTH" plot_symbols.txt
```

default state: canceled

22.6.14 SET MODE REFRESH

The SET MODE REFRESH command causes FERRET to update windows following "occlusion" events on X-servers that lack a backing store (SGI workstations have been a case in point).

default state: canceled (except on SGI systems)

22.6.15 SET MODE SEGMENTS

SET MODE SEGMENTS causes FERRET to utilize GKS segments ("GKS" is the Graphical Kernel System - an international graphics standard). On some systems MODE SEGMENTS may be necessary to update windows following "occlusion" events or to resize window with the mouse.

Segments, however, make heavy demands on the systems virtual memory. If FERRET crashes during graphics output due to insufficient virtual memory try CANCEL MODE SEGMENTS.

default state: set

22.6.16 SET MODE STUPID

SET MODE STUPID causes FERRET to forget data cached in memory. The result is that all requests for variables are read from disk rather than located in memory and reused from a previous read. The program will be significantly slower as a result. (This command is included for diagnostic purposes.)

default state: canceled

22.6.17 SET MODE VERIFY

SET MODE VERIFY causes commands from a command file ("GO file") to be displayed on the screen as they are executed. Note that if MODE VERIFY is canceled, loop counting in the REPEAT command is turned off.

default state: SET

Note: Many GO files begin with CANCEL MODE VERIFY to inhibit output and end with SET MODE/LAST VERIFY to restore the previous state. Only if an error or interrupt occurs during the execution of such a command file will the state of MODE VERIFY be affected.

22.6.18 SET MODE WAIT

SET MODE WAIT causes FERRET to wait for a keyboard keystroke from the user after each plotted output is completed. This is useful on graphics terminals that do not have a separate graphics plane; on these terminals SET MODE WAIT prevents the graphical output from being wiped off the screen until the user is ready to proceed.

default state: canceled

22.7 SET MOVIE

/COMPRESS /FILE /LASER /START

Designates a file (specified or default) for storing graphical images as movie frames (in HDF Raster-8 format). See Chapter 5 for further explanation.

```
yes? SET MOVIE[/qualifiers]
```

Command qualifiers for SET MOVIE:

SET MOVIE**/COMPRESS=**

Turns on or off compression of HDF frames using run length compression.

```
yes? SET MOVIE/COMPRESS=OFF
```

The allowed arguments are "on" and "off" - CANCEL MOVIE does not affect this qualifier.

default state: on

SET MOVIE**/FILE**

Specify an output file to receive movie frames.

```
yes?  SET MOVIE/FILE=filename       !specify a new filename
      or
yes?  SET MOVIE/FILE    !reactivate  a  previously  specified  filename\
                        after CANCEL MOVIE
```

The default movie filename extension is ".mgm"
The default movie filename is "ferret.mgm"

SET MOVIE/**LASER**
Output to Panasonic OMDR. Valid only on older VAX/VMS systems.

SET MOVIE/**START**
Only valid for use on older VAX/VMS systems with the Panasonic Optical Memory Disk Recorder (OMDR). Only valid with /LASER qualifier.

22.8 SET REGION

/I/J/K/L /X/Y/Z/T /DI/DJ/DK/DL /DX/DY/DZ/DT

Specifies the default space-time region for the evaluation of expressions.

```
yes? SET REGION[/qualifiers] [ reg_name]
```

See Chapter 4 section "Regions" for further information.

Examples:

1) `yes? SET REGION/X=140E`
Sets X axis position in the default context.

2) `yes? SET REGION/@N       !N specifies X and Y but not Z or T`
Sets only X and Y in the default context (since X and Y are defined in region N but Z and T are not).

3) `yes? SET REGION N`
Sets ALL AXES in the default region to be exactly the same as region N. Since Z and T are undefined in region N they will be set undefined in the default context.

4) `yes? SET REGION/@N/Z=50:250`
Sets X and Y in the default region to be exactly the same as region N and then sets Z to the range 50 to 250.

5) `yes? SET REGION/DZ=-5`
Set the region along the Z axis to be 5 units less than its current value.

6) `yes? SET REGION/DJ=-10:10`
Increases the current vertical axis range by 10 units on either end of the axis.

Command qualifiers for SET REGION:

SET REGION/**I=/J=/K=/L=/X=/Y=/Z=/T=**
Sets region bounds for specified axis subscript (I,J,K or L) or axis coordinates (X,Y,Z, or T). See examples above.

SET REGION/**DI=/DJ=/DK=/DL=/DX=/DY=/DZ=/DT=**
Modifies current region information by the specified increment of an axis subscript (I,J,K or L) or axis coordinate (X,Y,Z or T). See examples above. Syntax: /D*=val, or /D*=lo:hi.

22.9 SET VARIABLE

/BAD /GRID /TITLE /UNIT

Modifies attributes of a variable defined by DEFINE VARIABLE or SET DATA/EZ. This command permits variables within a single EZ data set to be defined on different grids.

```
yes? SET VARIABLE/qualifiers variable_name
```

Parameters
The variable name can be a simple name or a name qualified by a data set.

Example:

```
yes? SET VAR/UNITS="CM" WIDTH[D=snoopy]
```

Command qualifiers for SET VARIABLE:

SET VARIABLE/**BAD=**
Designates value to be used as the missing data flag. The qualifier is not applicable to variables defined with DEFINE VARIABLE.

SET VARIABLE/**GRID=**
Sets the defining grid for a variable in an EZ data set.

Example:

```
yes? SET VARIABLE/GRID=my_grid width[D=snoopy]
```

This is the mechanism by which the shape of the data (1D along T axis, 2D in the XY plane, etc.) is specified. By default FERRET will use grid EZ, a line of up to 20480 points oriented along the X axis. The qualifier is not applicable to variables defined with DEFINE VARIABLE.

SET VARIABLE/**TITLE=**
Associates a title with the variable. This title appears on plotted outputs and listings.

```
yes? SET VARIABLE/TITLE="title string" var_name
```

SET VARIABLE/**UNITS=**
Associates units with the variable. The units appear on plotted outputs and listings.

```
yes? SET VARIABLE/UNITS="units string" var_name
```

22.10 SET VIEWPORT

Sets the rectangular region within the output window where output will be drawn.

```
yes? SET VIEWPORT view_name
```

Issuing the command SET VIEWPORT is best thought of as entering "viewport mode." While in viewport mode all previously drawn viewports remain on the screen until explicitly cleared with either SET WINDOW/CLEAR or CANCEL VIEWPORT. If multiple plots are drawn in a single viewport without the use of /OVERLAY the current plot will erase and replace the previous one; the graphics in other viewports will be affected only if the viewports overlap. If viewports overlap the most recently drawn graphics will always lie on top, possibly obscurring what is underneath. By default, the state of "viewport mode" is canceled.

Pre-defined viewports exist for dividing the window into four quadrants and for dividing the window in half horizontally and vertically. See Chapter 6 section "Pre-defined viewports" for a list.

22.11 SET WINDOW

/ASPECT /CLEAR /LOCATION /NEW /SIZE

Creates, resizes, reshapes or moves graphics output windows.

```
yes? SET WINDOW[/qualifiers]  [window_number]
```

Note: Multiple windows may be simultaneously viewable but only a single window receives output at any time.

See commands SHOW WINDOW and CANCEL WINDOW for additional information.

Examples:

1) `yes? SET WINDOW/NEW`
Creates a new output window and sends subsequent graphics to it.

2) `yes? SET WINDOW 3`
Sends subsequent graphics to window 3.

3) `yes? SET WINDOW/SIZE=.5`
Resizes current window to 1/2 of full.

4) `yes? SET WINDOW/ASPECT=.5`
Reshapes current window with Y/X equal to 1:2.

5) `yes? SET WINDOW/LOCATION=0,.5`
Puts the lower left corner of the current window at the left border of the display and half way up it.

Command qualifiers for SET WINDOW:

SET WINDOW/**ASPECT**
Sets the aspect ratio of the output window and hard copy.

Examples:
1) `yes? SET WINDOW/ASPECT=y_over_x  n`
Sets the overall aspect ratio of window n.

2) `yes? SET WINDOW/ASPECT=y_over_x`
Sets the overall aspect ratio of the current window.

3) `yes? SET WINDOW/ASPECT=y_over_x:AXIS`
Sets the axis length aspect ratio of the current window.

The total size (area) of the output window is not changed.
The default value for the overall window ratio is y/x = 8.8/10.2 ~ 0.86.
The default value for the axis length ratio is y/x = 6/8 = 0.75.
Use PPLUS/RESET or SET WINDOW/ASPECT=.75:AXIS to restore defaults.
The aspect ratio specified is a default for future SET WINDOW commands
The origin (lower left) is restored to its default values: 1.2, 1.4

When using "SET WINDOW n" to return to a previous window that differs from the current window in aspect ratio, it is necessary to re-specify its aspect ratio with /ASPECT, otherwise PPLUS will not be properly reset.

SET WINDOW/**CLEAR**
Clears the image(s) in the current or specified window. Useful with viewports.

SET WINDOW/**LOCATION**
Sets the location for the lower left corner of named (or current) window. The coordinates x and y must be values between 0 and 1 and refer to distances from the lower left corner of the display screen (total length and width of which are each 1).

```
yes? SET WINDOW/LOCATION=x,y [window_number]
```

SET WINDOW/**NEW**
Causes future graphical output to be directed to a new window. The window will be created at the next graphics output.

```
yes? SET WINDOW/NEW
```

SET WINDOW/**SIZE**
Resizes a window to r times the size of the standard window. If the window number is omitted the command will resize the currently active window. (The default window size is 0.7.)

```
yes? SET WINDOW/SIZE=r [window_number]
```

23 SHADE

/I/J/K/L /X/Y/Z/T /D /FRAME /KEY /LEVELS /LINE /NOKEY /NOLABEL
/OVERLAY /PALETTE /SET_UP /TITLE /TRANPOSE /XLIMITS /YLIMITS

Produces a shaded (rectangular raster) plot of a 2-D field. By default a color key is drawn and contour lines are not drawn.

```
SHADE[/qualifiers]  expression
```

Parameters
The expression may be any valid expression. See Chapter 3 section "Expressions" for a definition of valid expressions. The expression will be inferred from the current context if omitted from the command line. Multiple expressions are not permitted in a single SHADE command.

Command qualifiers for SHADE:

SHADE/**I=/J=/K=/L=/X=/Y=/Z=/T=**
Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when evaluating the expression being plotted.

SHADE/**D=**
Specifies the default data set to be used when evaluating the expression being plotted.

SHADE/**FRAME**
Causes the graphic image produced by the command to be captured as an animation frame in the file specified by SET MOVIE.

SHADE/**KEY**
Displays a color key for the palette used in the shaded plot. By default a key is drawn unless the /LINE or /NOKEY qualifier is specified.

SHADE/**LEVELS**

Specifies the SHADE levels or how the levels will be determined. If the /LEVELS qualifier is omitted FERRET automatically selects reasonable SHADE levels.

The /LEVELS qualifier can be used in two different ways:

1) To reuse the levels from the last CONTOUR or SHADE plot use /LEVELS with no argument:

```
yes? SHADE/LEVELS expr
```

2) To specify the shade levels manually use a combination of arguments with the following syntaxes:

(val)

val = value of a shade level (useful for achieving unevenly spaced levels)

(lo,hi,delta,ndig)

lo = lowest shade level
hi = highest shade level
delta = interval between levels
ndig = number of digits after the decimal (only valid with SHADE/LINE)
-1 integer labels
-3 no labels

```
yes? SHADE/LEV=(lo,hi,delta)(val1)(val2)(lo2,hi2,del2)...  expr
```

If you are using SHADE/LINE, which overlays contour lines on the shaded plot, after specifying shade levels you can specify line types for various levels with a combination of arguments using the following syntax:

TYPE(lo,hi,delta,index) or TYPE(val)

where "TYPE" may be any of the following and applies to the specified levels:

DARK	draws thicker lines
DASH	draws dashed lines (default for negative shade levels)
LINE	draws solid lines of regular thickness (default for positive levels)
PEN	draws lines with specified color_thickness index

Only type PEN requires specification of "index"; it may be omitted otherwise. If blanks are imbedded within the list of arguments you must enclose the entire list within quotation marks.

For example,

```
yes? SHADE/LINE/LEVELS=(10,30,5)DASH(10) sst
```

produces SHADE levels at 10-15, 15-20, 20-25, and 25-30, overlays contour lines and draws the line at level 10 in a dashed pattern.

SHADE/**LINE**

Overlays contour lines on a shaded plot. When /LINE is specified the color key is omitted unless specifically requested via /KEY.

SHADE/**NOKEY**

Suppresses the drawing of a color key for the palette used in the plot.

SHADE/**NOLABELS**

Suppresses all plot labels except axis labels.

SHADE/**OVERLAY**

Causes the indicated shaded plot to be overlaid on the existing plot.

Note (SHADE/OVERLAY with time axes):

A restriction in PPLUS requires that if time is an axis of the shaded plot, the overlaid variable must share the same time axis encoding as the base plot variable. If this condition is not met, you may find that the overlaid shaded plot fails to be drawn. The solution is to use the FERRET regridding capability to regrid the base plot variable and the overlaid plot variable onto the same time axis.

SHADE/**PALETTE=**

Specifies a color palette (otherwise, a default rainbow palette is used). Try the Unix command `% Fpalette '*'` to see available palettes. The file suffix *.spk is not necessary when specifying a palette. See command PALETTE for more information.

```
yes? SHADE/PALETTE=land_sea  rose
```

SHADE/**SET_UP**

Performs all the internal preparations required by program FERRET for a shaded plot but does not actually render output. The command PPL can then be used to make changes to the plot prior to producing output with the PPL SHADE command. This permits plot customizations that are not possible with FERRET command qualifiers. See Chapter 6.

SHADE/**TITLE=**

Allows user to specify a plot title (enclosed in quotation marks). Without this qualifier FERRET selects a title based on information about the expression(s). To include font change and color_thickness specifications (e.g., @TI@C002) in the title string, it is necessary either to CANCEL MODE ASCII or to include a leading ESC (escape) character. See Chapter 6 section "Fonts".

```
yes? SHADE/TITLE="title string"  expression
```

SHADE/**TRANSPOSE**

Causes the horizontal and vertical axes to be interchanged. By default the X axis is drawn horizontally on the plot and the Y and Z axes are drawn vertically. For Y-Z plots the Z data axis is vertical.

SHADE/**XLIMITS=**
Specifies the X axis range and tic interval (otherwise, FERRET selects reasonable values).

```
yes? SHADE/XLIMITS=lo:hi:increment
```

The optional "increment" parameter determines tic mark spacing on the axis. If the increment is negative, the axis will be reversed.

SHADE/**YLIMITS=**
Specifies the Y axis range and tic interval. See /XLIMITS.

24 SHOW
/ALL

Displays program states and stored values.

Command qualifiers for SHOW:

SHOW/**ALL**
Executes all SHOW options. This command gives a complete description of the current state, including information about region, grids, axes, variables and the state of various modes (default or set with SET MODE).

```
yes? SHOW/ALL
```

24.1 SHOW ALIAS

Lists all command aliases and the full command names for which they stand, or, with an argument, shows a specified command alias.

```
yes? SHOW ALIAS  [alias_name]
```

24.2 SHOW AXIS
/ALL

Shows a basic description of the named axis.

```
SHOW AXIS[/ALL]   [axis_name]
```

A typical output appears below. The columns are:

name	name of axis (used also in DEFINE AXIS and DEFINE GRID)
axis	the orientation of the axis; "(-)" on a depth axis indicates increasing downward
# pts	number of points on axis; "r" or "i" for regular or irregular spacing, "m" if the axis is "modulo" (repeating)
start	position of first point on the axis
end	position of last point on the axis

```
yes? SHOW AXIS/ALL
name       axis              # pts   start                end
PSXT      LONGITUDE          160 r   130.5E               70.5W
PSYT      LATITUDE           100 i   28.836S              48.568N
PSZT      DEPTH(-)            27 i   5m                   3824m
TIME      TIME                25 r   17-AUG-1982 12:00    10-JAN-1983
NMCX      LONGITUDE          180 r   20E                  18E
NMCY      LATITUDE            91 r   90S                  90N
TIME1     TIME                12mr   16-JAN-1901 05:00    16-DEC-1901
```

Command qualifiers for SHOW AXIS:

SHOW/**ALL**
Show a brief summary of all axes defined.

```
yes? SHOW AXIS/ALL
```

24.3 SHOW COMMANDS

Displays commands, subcommands and qualifiers recognized by program FERRET. This command does *not* display aliases; use SHOW ALIAS.

```
SHOW COMMAND    [command_name or partial_command]
```

Note: This is the most reliable way to view command qualifiers. The output of this command will be current even when this manual is out of date.

Examples:

```
yes? SHOW COMMAND S      ! show all commands beginning with "S"
yes? SHOW COMMAND        ! show all commands
yes? SHOW COMMAND PLOT   ! shows command PLOT and all its qualifiers
```

24.4 SHOW DATA_SET
/ALL /BRIEF /FILES /FULL /VARIABLE

Shows information about the data sets which have been SET and indicates the current default data set. By default the variables and their subscript ranges are also listed.

```
yes? SHOW DATA[/qualifiers]   [set_name_or_number1,set2,...]
```

If no data set name or number is specified then all SET data sets are shown.

Command qualifiers for SHOW DATA_SET:

SHOW DATA/**ALL**
This qualifier has no effect on this command; it exists for compatibility reasons.

SHOW DATA/**BRIEF**
Shows only the names of the data sets; does not describe the data contained in them.

SHOW DATA/**FILES**
Displays the names of the data files for this data set and the ranges of time steps contained in each. Output is formatted as date strings or as time step values depending on the state of MODE CALENDAR.

SHOW DATA/**FULL**
Equivalent to /VARIABLES and /FILES used together.

SHOW DATA/**VARIABLES**
In addition to the information given by the SHOW DATA command with no qualifiers, this query also provides the grid name and world coordinate limits for each variable in the data set.

Example: SHOW DATA
SHOW DATA produces a listing similar to the one below. The output begins with the descriptor file name (for TMAP-formatted data) and data set title. The columns I, J, K and L give the subscript limits for each variable with respect to its defining grid (use SHOW DATA/FULL and SHOW GRID variable_name for more information).

```
yes? SET DATA levitus_climatology
yes? SHOW DATA
     currently SET data sets:
    1> /home/e1/tmap/fer_dsets/descr/levitus_climatology.des  (default)
      name     title                I         J         K         L
      TEMP     TEMPERATURE         1:360     1:180     1:20      1:1
      SALT     SALINITY            1:360     1:180     1:20      1:1
```

24.5 SHOW EXPRESSION

Shows the current expression(s) implied or set with SET EXPRESSION. If not explicitly set with this command, the default current context expression is the argument of the most recent "action" command (PLOT, SHADE, CONTOUR, VECTOR, WIRE, etc.) See Chapter 3 section "Expressions" for an explanation and list of action commands.

```
yes? SHOW EXPRESSION
```

24.6 SHOW GRID

/I/J/K/L /X/Y/Z/T /ALL

Shows the name and axis limits of a grid.

```
yes? SHOW GRID[/qualifiers]   [var_or_grid1 var_or_grid2 ...]
```

Example:
(See command SHOW AXIS for an explanation of the columns.)

```
yes? SET DATA levitus_climatology
yes? SHOW GRID salt
    GRID GLEVITR1
 name       axis           # pts   start                end
 XAXLEVITR LONGITUDE       360mr   20.5E                19.5E(379.5)
 YAXLEVITR LATITUDE        180 r   89.5S                89.5N
 ZAXLEVITR DEPTH(-)         20 i   0m                   5000m
```

Parameters
The parameter(s) may be the name of one or more grid(s) or variable(s). If no parameter is given SHOW GRID displays the grid of the last variable accessed. This is the only mechanism to display the grid of an algebraic expression.

Note: To apply SHOW GRID to an algebraic expression it is necessary for FERRET to have evaluated the expression in a previous command. The command LOAD is useful for this purpose in some circumstances.

Command qualifiers for SHOW GRID:

SHOW GRID**/I=/J=/K=/L=/X=/Y=/Z=/T=**
Displays the coordinates and grid box sizes for the specified axis. Optionally, low and high limits and a delta value may be specified to restrict the range of values displayed.

```
yes? SHOW GRID/X[=lo:hi:delta] [variable_or_grid]
```

Example:

```
yes? SHOW GRID/L=1:12:3  sst[coads_climatology]
```

SHOW GRID**/ALL**
Shows the names only of all grids defined.

```
yes? SHOW GRID/ALL
```

24.7 SHOW LIST

Shows the current states of the LIST command.

```
yes? SHOW LIST
```

The qualifier /ALL may be used with this command but exists merely for compatibility reasons and has no effect.

24.8 SHOW MEMORY

/ALL/FREE/PERMANENT/TEMPORARY

Shows the variables that are currently cached in memory.

```
yes? SHOW MEMORY[/qualifiers]
```

Command qualifiers for SHOW MEMORY:

SHOW MEMORY/**ALL**
Shows all variables currently in memory - permanent and temporary.

SHOW MEMORY/**FREE**
Shows bulk memory and memory table space that remains unused.

Bulk memory is organized into "blocks". One block is the smallest unit that any variable stored in memory may allocate. The total number of variables that may be stored in memory cannot exceed the size of the memory table. The "largest free region" gives an indication of memory fragmentation. A typical SHOW MEMORY/FREE output looks as below:

```
total memory table slots: 150
total memory blocks: 500
memory block size:1600

number of free memory blocks: 439
largest free region: 439
number of free regions:   1
free memory table slots: 149
```

SHOW MEMORY/**PERMANENT**
Lists the variables stored in memory and cataloged as permanent. These variables will not be deleted even when memory space is needed. They become cataloged in memory as permanent when the LOAD/PERMANENT command is used.

SHOW MEMORY/**TEMPORARY**
Lists the variables stored in memory and cataloged as temporary (they may be deleted when memory capacity is needed).

24.9 SHOW MODE

Shows the names, states and arguments of the FERRET SET MODE command.

```
SHOW MODE [partial_mode_name1,name2,...]
```

Example:

```
yes? SHOW MODE VERIFY,META
```

The qualifier /ALL may be used with this command but exists merely for compatibility reasons and has no effect.

24.10 SHOW MOVIE

Shows the current state of SET MOVIE. This state affects FRAME and graphics commands specified with the /FRAME qualifier.

```
yes? SHOW MOVIE
```

The qualifier /ALL can be used with this command, but it exists for compatibility purposes only and has no effect.

24.10.1 SHOW QUERIES

Queries are a vehicle for communication between FERRET and a stand-alone interface program. They are not supported for general use.

24.11 SHOW REGION

Shows the current default region or the named region.

```
yes? SHOW REGION[/ALL]  [region_name]
```

The region displayed is formatted appropriately for the axes of the last data accessed. For example, suppose the region along the Y axis was specified as Y=5S:5N. Then if the Y axis of the last data accessed is in units of degrees-latitude the Y location is shown as Y=5S:5N but if the Y axis of the last data accessed is "ABSTRACT" then the Y location is shown as Y=-5:5.

24.12 SHOW TRANSFORM

Shows the available transformations.

```
yes? SHOW TRANSFORM
```

Note: This is the most reliable way to view transformations. The output of this command will be current even when this manual is out of date.

The qualifier /ALL may be used with this command but exists merely for compatibility reasons and has no effect.

24.13 SHOW VARIABLES

/ALL /DIAGNOSTIC /USER

Lists diagnostic or user-defined variables.

```
SHOW VARIABLES[/qualifier]   [partial_name]
```

Examples:

```
yes? SHOW VARIABLES      !all user-defined variables
yes? SHOW VAR/DIAG Q     !all diagnostic vars beginning with Q
```

Command qualifiers for SHOW VARIABLES:

SHOW VARIABLES**/ALL**
Lists both diagnostic variables (available for the COX/PHILANDER model) and user-defined variables.

SHOW VARIABLES**/DIAGNOSTIC**
Lists "diagnostic" variables available for the COX/PHILANDER model.

Note that if a component variable required to compute a diagnostic quantity is not available on disk then that diagnostic variable will not, in fact, be available. An error message indicating that the data is not available is printed at execution time if such a variable is requested.

SHOW VARIABLES**/USER**
Lists expressions which have been defined by the user as new variables. This is the default behavior of SHOW VARIABLES with no qualifier.

24.14 SHOW VIEWPORT

Shows one or more of the currently defined viewports. Omitting an argument gives information on all viewports.

```
yes? SHOW VIEWPORT [view_name1,view_name2,...]
```

The qualifier /ALL may be used with this command but exists merely for compatibility reasons and has no effect.

24.15 SHOW WINDOWS

Lists open window numbers and indicates which is the active one.

```
yes? SHOW WINDOWS
```

The qualifier /ALL may be used with this command but exists merely for compatibility reasons and has no effect.

25 SPAWN

Executes a command line (Unix shell) command from within FERRET.

```
yes? SPAWN unix_shell_command
```

Example:

```
yes? SPAWN rm -f file.dat
```

Also, "SPAWN shell_name" allows the user to fork into an interactive shell. For example:

```
yes? SPAWN csh
```

enters the user into a c-shell. Use EXIT to return to FERRET.

26 STATISTICS

/I/J/K/L X/Y/Z/T /D

Computes summary statistics about the data specified.

```
yes? STATISTICS[/qualifiers]    expression_1 , expression_2 , ...
```

As of FERRET version 3.1 the statistics include:

total number of data values in the region specified
number of data values flagged as bad data
minimum value
maximum value
mean value (arithmetic mean - not weighted by grid spacing)
standard deviation (also not weighted by grid spacing)

All values are reported to 5 significant digits.

STATISTICS interacts with the current context exactly as the commands CONTOUR, PLOT and LIST do.

Parameters
Expressions may be anything described under Expressions. If multiple variables or expressions are specified they are treated in sequence. The expression(s) are inferred from the current context if omitted from the command line.

Command qualifiers for STATISTICS:

STATISTICS**/I=/J=/K=/L=/X=/Y=/Z=/T=**
Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when computing statistics about the expression(s).

STATISTICS**/D=**
Specifies the default data set to be used when computing statistics about the expression(s).

27 UNALIAS

Alias for CANCEL ALIAS.

28 USE

The USE command is an alias for SET DATA/FORMAT=cdf.

All qualifiers and restrictions are identical to SET DATA/FORMAT=cdf. If no filename extension is given, ".cdf" is assumed.

Example:

```
yes? USE test
```

is equivalent to

```
yes? SET DATA/FORMAT=cdf test
```

29 USER

Executes a user-written extension to the FERRET program.

```
yes? USER[/COMMAND=]    expression_1 , expression_2, ...
```

The USER command is a means of incorporating custom changes in FERRET. It is currently supported only by special request to the FERRET developers (ferret@pmel.noaa.gov). Two special features are currently accessible through the USER command - objective analysis and scattered sampling of grids. These commands will eventually be replaced by more thoroughly integrated features with the same functionality.

We recommend the user access objective analysis via the script objective.jnl. The scattered sampling feature is used in the polar plotting GO tools (try "GO polar_demo" at the FERRET prompt).

29.1 Objective analysis

The command selection "objective" grids the (X,Y,value) triples onto a grid of specified resolution using objective analysis techniques exactly as available in PLOTPLUS.

```
yes? USER/COMMAND="objective"/OPT1=/FILE=/FORMAT=  xpts, ypts, vals
```

/OPT1=xlo:xhi:xdel,ylo:yhi:ydel,[cay],[rng]

OPT1 specifies a grid that is xlo:xhi:xdel in X and ylo:yhi:ydel in Y. The parameters "cay" and "rng" are interpreted as in the PPLUS CONSET command ("rng" is called "nrng" in CONSET) :

cay = is the interpolation scheme. default=5.0

cay=0.0 Laplacian interpolation is used. The resulting surface tends to have rather sharp peaks and dips at the data points. There is no chance of spurious peaks appearing.

cay > 0. As CAY is increased, spline interpolation predominates over the Laplacian, and the surface passes through the data points more smoothly. The possibility of spurious peaks increases with CAY.

cay= infinity. This is pure spline interpolation. An over-relaxation process in used to perform the interpolation.

rng = Grid points farther than RNG away from the nearest data point will be set to "undefined" (1.0E35). default=5

/FILE=filename

Specifies the name of the output file, to which the gridded data is directed.

/FORMAT=format

The format may be a FORTRAN format (in parentheses). If specified, each FORTRAN WRITE will output nx points (one row of the output grid). If unspecified, the format will be binary floating point written one value per (FORTRAN) variable-length record.

See objective_analysis_demo.jnl which uses objective.jnl (type "GO objective_analysis_demo" at the FERRET prompt).

29.2 Scattered sampling

The command selection "sample" uses multi-linear interpolation to sample a gridded field of data at a list of scattered coordinates. This procedure is useful in statistics, in experimental design studies, and in advective tracer animations. The variable "four_dee_field" is a 1 to 4 dimensional gridded field of values. The variables xpts, ypts, zpts, and tpts are the ordered-tuples of points at which the field should be sampled. To indicate that a particular axis is not to be involved in the sampling process, pass a missing value flag ("1/0") for that axis. For example,

```
yes? USER/COMMAND="sample"  temp[d=levitus_climatology,K=1], \
          xpts, ypts, 1/0, 1/0
```

will sample (X,Y) points from the K=1 field of Levitus climatological temperatures. Multi-linear interpolation is used to interpolate between grid points.

```
yes? USER/COMM="sample"/OPT1=/OPT2=/FILE=/FORMAT= \
                  four_dee_field, xpts, ypts, zpts, tpts
```

/OPT1=coaching

The parameter "coaching" may contain these substrings:

"hole" or "fill"	To perform the linear interpolation all of the grid points surrounding the requested sample, tuple must contain valid data. If "fill" is specified then FERRET attempts to fill any missing surrounding points with an average of their neighbors. If "hole" (the default) then the result is "missing" whenever a neighboring point is missing.
"standard_bad"	indicating that the standard FERRET bad value flag of -1E34 is used for missing data points (default is the missing data flag from "four_dee_field".

/OPT2=coordinates_or_indices

The parameter "coordinates_or_indices" determines if the input sampling points are to be interpreted as world coordinates or as subscripts. If it is "c" (default) the input points are world coordinates. If "i", they are indices. Indices may be fractional locations (e.g. 2.5 is midway between 2 and 3). By specifying a 4-character "coordinates_or_indices" each axis may be separately specified. (e.g. /OPT2="ccci" will interpret T axis positions, only, as indices)

/FILE=filename

Specifies the name of an output file to which sampled data will be written.

/FORMAT=format

The format may contain a FORTRAN format in parentheses or "unf" (default) to indicate unformatted, binary, floating point output. A special, condensed (high performance) format option is possible by specifying "unf:nnn". In this case only the sampled values are written (no coordinates or codes) and nnn specifies the number of values to be written per record (e.g. "unf:100").

By default each output record consists of:

```
value  xcoord  ycoord  zcoord  tcoord  code
```

where "code" is

0	fully successful
1	successful via hole filling
-1...-4	failed due to beyond input data limits on axis -code
-9	failed due to hole in input data field
-99	failed due to hole in input coordinate

See polar_demo.jnl which uses convert_to_polar_2d.jnl which uses USER/COMMAND=SAMPLE.

30 VECTOR

/I/J/K/L /X/Y/Z/T /D /ASPECT /FRAME /LENGTH /NOLABEL /OVERLAY /PEN /SET_UP /TITLE /TRANPOSE /XLIMITS /XSKIP /YLIMITS /YSKIP

Produces a vector arrow plot.

```
VECTOR[/qualifiers] x_expr,y_expr
```

Parameters

x_expr, y_expr

Algebraic expressions (or simple variables) specifying the x components and y components of the vector arrows. The expression pair will be inferred from the current context if omitted from the command line.

Command qualifiers for VECTOR:

VECTOR**/I=/J=/K=/L=/X=/Y=/Z=/T=**

Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when evaluating the expression being plotted.

VECTOR**/D=**

Specifies the default data set to be used when evaluating the expression pair being plotted.

VECTOR**/ASPECT**

Adjusts the direction of the vectors to compensate for differing axis scaling.

```
yes? VECTOR/ASPECT[=aspect_ratio]  x_expr, y_expr...
```

The size of vectors is unchanged - only the direction is modified. The aspect ratio is (Y-scale/X-scale). Under normal circumstances no aspect ratio is specified by the user - FERRET will compute the correct ratio. If one of the plot axes is degrees of latitude, then the aspect ratio correction will be adjusted as a function of COS(LATITUDE) on the plot.

For example, in a typical oceanographic XZ plane plot the vertical (Z) axis is in tens of meters while the horizontal (X) axis is in hundreds of kilometers. This means the vertical scale is greatly magnified in comparison to the horizontal. The /ASPECT qualifier correspondingly magnifies the vertical component of the vector relative to the horizontal while preserving the length of the vector. The magnification factor is displayed on the plot.

VECTOR/**FRAME**
Causes the graphic image produced to be captured as an animation frame in the file specified by SET MOVIE.

VECTOR/**LENGTH=**
Controls the size of vectors.

```
   yes? VECTOR/LENGTH[=value_of_standard]
```

If the /LENGTH qualifier is omitted FERRET automatically selects reasonable vector lengths. To reuse the vector length from the last VECTOR plot use VECTOR/LENGTH.

To specify the vector lengths manually use the value_of_standard argument. This associates the value "val" with the standard vector length, normally 1/2 inch. Note that the PPLUS command VECSET can be used to modify the length of the standard vector. This is also the length that is displayed in the vector key.

Example:

```
   yes? VECTOR/LENGTH=100 U,V
```

Creates a vector arrow plot of velocities with 1/2 inch vectors for speeds of 100.

VECTOR/**NOLABELS**
Suppresses all plot labels except axis labels.

VECTOR/**OVERLAY**
Causes the indicated vector field to be overlaid on the existing plot.

VECTOR/**PEN=**
Specifies the line style for the vectors. /PEN= takes the same arguments the /LINE= qualifier for command PLOT. See command PLOT/LINE=. 'n' ranges from 1 to 18.

```
   yes? VECTOR/PEN=n   x_expr, y_expr
```

VECTOR/**SET_UP**

Performs all the internal preparations required by program FERRET for vector plots but does not actually render output. The command PPL can then be used to make changes to the plot prior to producing output with the PPL VECTOR command. This permits plot customizations that are not possible with FERRET command qualifiers. See Chapter 6.

VECTOR**/TITLE=**
Allows user to specify a plot title (enclosed in quotation marks). Without this qualifier FERRET selects a title based on information about x_expr and y_expr. To include font change and color_thickness specifications (e.g., @TI@C002) in the title string, it is necessary either to CANCEL MODE ASCII or to include a leading ESC (escape) character. See Chapter 6 section "Fonts".

```
yes? VECTOR/TITLE="title_string"  x_expr, y_expr
```

VECTOR**/TRANSPOSE**
Causes the horizontal and vertical axes to be interchanged. By default the X axis is always drawn horizontal and the Y and Z axes are drawn vertical. For Y-Z plots the Z data axis is vertical.

VECTOR**/XLIMITS=**
Specifies X axis limits and tic interval. Without this qualifier, FERRET selects reasonable values.

```
yes? VECTOR/XLIMITS=lo:hi:increment   x_expr, y_expr
```

The optional "increment" parameter determines tic mark spacing on the axis. If the increment is negative, the axis will be reversed.

VECTOR**/XSKIP=/YSKIP=**
Draws every nth vector along the requested axis beginning with the first vector requested.

```
yes? VECTOR/XSKIP=nx/YSKIP=ny  u,v
```

By default, FERRET "thins" vectors to achieve a clear plot. These qualifiers allow control over thinning.

VECTOR**/YLIMITS=**
Specifies Y axis limits and tic interval. See /XLIMITS=.

31 WIRE

/I/J/K/L /X/Y/Z/T /D /FRAME /NOLABEL /OVERLAY
/SET_UP /TITLE /TRANPOSE /VIEWPOINT /ZLIMITS /ZSCALE

Produces a wire frame representation of a two dimensional field.

```
yes? WIRE[/qualifiers] expression
```

Parameters
The expression may be anything described in Chapter 3 section "Expressions". The expression will be inferred from the current context if omitted from the command line. Multiple expressions are not permitted in a single WIRE command. The indicated region should denote a plane (2D) of data.

Command qualifiers for WIRE:

WIRE**/I=/J=/K=/L=/X=/Y=/Z=/T=**
Specifies value or range of axis subscripts (I,J,K or L), or axis coordinates (X,Y,Z or T) to be used when evaluating the expression being plotted.

Example:
The following commands will create a wire frame representation of a simple mathematical function in 2 dimensions.

```
yes? SET REGION/I=1:80/J=1:80
yes? WIRE/VIEWPOINT=-4,-10,2 exp(-1*(((I-40)/20)^2 + ((J-40)/20)^2))
```

WIRE**/D=**
Specifies the default data set to be used when evaluating the expression being plotted.

WIRE**/FRAME**
Causes the graphic image produced to be captured as an animation frame in the file specified by SET MOVIE.

WIRE**/NOLABEL**
Suppresses all plot labels except axis labels.

WIRE**/OVERLAY**
Causes the indicated wire frame plot to be overlaid on the existing plot.

WIRE**/SET_UP**
Performs all the internal preparations required by program FERRET for wire frame graphics but does not actually render output. The command PPL can then be used to make changes to the plot prior to producing output with the PPL WIRE command. This permits plot customizations that are not possible with FERRET command qualifiers. See Chapter 6.

WIRE**/TITLE=**
Allows user to specify a plot title (enclosed in quotation marks). Without this qualifier FERRET selects a title based on information about the expression. To include font change and color_thickness specifications (e.g., @TI@C002) in the title string, it is necessary either to CANCEL MODE ASCII or to include a leading ESC (escape) character. See Chapter 6 section "Fonts".

WIRE/**TRANSPOSE**
Causes the X and Y axes to be interchanged.

WIRE/**VIEWPOINT=**
Specifies a viewpoint for viewing the wire frame.

```
yes? WIRE/VIEWPOINT=x,y,z   expression
```

The x,y values are specified as coordinates on the X and Y axes (though they may exceed the axis limits). The z value is in units of the requested variable.

WIRE/**ZLIMITS=**
Specifies limits of Z axis for wire frame.

```
yes? WIRE/ZLIMITS=zmin,zmax,delta   expression
```

The values given are in units of the requested variable. (The string given as an argument to /ZLIMITS= is passed unmodified to the PPLUS command WIRE as the zmin and zmax parameters.)

WIRE/**ZSCALE=**
Controls Z axis scaling of the 3-D plot.

```
yes? WIRE/ZSCALE=s  expression
```

The default value is equivalent to (ymax-ymin)/(zmax-zmin) (i.e. the aspect ratio of the Z axis to the Y axis). This qualifier is identical to the PPLUS VIEW command parameter of the same name.

GLOSSARY

ABSTRACT EXPRESSION (or VARIABLE)

An expression which contains no dependencies on any disk-resident data is referred to as "abstract". For example, SIN(x), where x is a pseudo-variable.

AXIS

A line along one of the dimensions of a grid. The line is divided into n points, or more precisely, n grid boxes where each grid box is a length along the axis. Adjacent grid boxes must touch (no gaps along the axis) but need not be uniform in size (points may be unequally spaced). Axes may be oriented (e.g. latitude, depth, ...) or simply abstract values.

CONTEXT

The information needed to obtain values for a variable: the location in space and time (points or ranges), the name of the data set (if a file variable) and an optional grid.

DATA SET

A collection of variables in one or more disk files that may be specified with a single SET DATA command.

DESCRIPTOR

A file containing background data about a GT or TS-formatted data set: variable names, coordinates, units and pointers to the data files. Descriptor file names normally end with ".DES".

DIAGNOSTIC VARIABLE

A quantity internal to the models at the Thermal Modeling and Analysis Project at NOAA that represents a physically significant value (such as heat diffusion). These values are computed interactively from the state variables (TEMP, SALT, U, V, PSI) by FERRET in such a way that they appear to be a part of the data set.

EXPRESSION

Any valid combination of operators, functions, transformations, variables and pseudo-variables is an expression. For example, "ABS(U)", "TEMP/(-0.03^Z)" or "COS(TEMP[Y=0:40N@LOC:15])".

EZ DATA SET

Any disk data file that is readable by FERRET but is not in GT, TS or NetCDF format.

FILE VARIABLE

A variable made available with the SET DATA command. File variables are data in disk files suitable for plotting, listing, using in user-variable definitions, etc.

GKS

The "Graphical Kernel System" - a graphics programming interface that facilitates the development of device-independent graphics code.

GO FILE or **GO SCRIPT**

A file of FERRET commands intended to be executed as a single command with the GO command.

GRID

A group of 1 to 4 axes defining a coordinate space. A grid can associate the axes as "outer products" creating a rectangular array of points. Grids may be defined with the DEFINE GRID command or from inside data sets.

GRID BOX

A length along an axis assumed to belong to a single grid point. It is represented by a box "middle", a box upper and a box lower limit. The "middle" need not actually be at the center of the box but the upper limit of box m must always be the lower limit of box m+1. (This concept is needed for integration of variables along an axis.)

GRID FILE

A file containing the definition of grids and axes - part of the GT and TS formats.

GT FORMAT

"grids at time steps" format. A direct access format using a separate descriptor file for descriptive metadata.

METAFILE

A representation of graphics stored in a computer file. Such a file can be processed by an interpreter program (such as Fprint) and sent to a graphics output device.

MODULO AXIS

An axis where the first point of the axis logically follows the last. Examples of this are degrees of longitude or dates in a climatological year.

NETCDF

Network Common Data Format is an interface to a library of data access routines for storing and retrieving scientific data. NetCDF allows the creation of data sets which are self-describing and network transparent. As of FERRET version 2.30, NetCDF is the suggested method of data storage.

OPERATOR

A function that is syntactically expressed in-line instead of as a name followed by arguments. The FERRET operators are +, -, *, /, ^, AND, OR, EQ, NE, LT, LE, GT and GE.

PSEUDO-VARIABLE

A special variable whose values are coordinates or coordinate information about a grid. X, I and XBOX are the pseudo-variables for the X axis - similarly for the other axes.

QUALIFIER

Commands and variable names may require auxiliary information supplied by qualifiers. In the command "SHOW DATA/FULL," "/FULL" is a qualifier. In the variable "SST[Y=20N]," "Y=20N" is a qualifier.

REGION

The location in space and time (or other axis units) at which a variable is to be evaluated. The locations may be points or ranges. For example, T="1-JAN-1982",Y=12S:12N describes a region in latitude and time.

REGRID

The process of converting the values of a variable from one grid to another. By default this is done through multi-linear interpolation along all axes from the old grid to the new. Other methods are also supported.

SUBSCRIPT

A coordinate system for referring to grid locations in which the points along an axis are regarded as integers from 1 to the number of points on the axis. The qualifiers I, J, K and L are provided to specify locations by subscript.

TRANSFORMATION

An operation performed on a variable along a particular axis and specified via the syntax "@trn". Some transformations, such as averaging (e.g. U[Z=@AVE]), reduce the range of the variable along the axis to a single point. Others, such as taking a derivative (e.g. V[T=@DDC]) do not.

TMAP-FORMAT

Special formats created by the Thermal Modeling and Analysis Project (TMAP). These formats use descriptor files to store information about the variables, units, titles and grids for the data. Separate formats allow optimized access as time series (TS format) or as geographical regions (GT format). As of FERRET version 2.30, NetCDF is the suggested method of data storage.

TS FORMAT

"time step" format. A direct access format using a separate descriptor file for descriptive metadata.

USER-DEFINED VARIABLE

A variable created with DEFINE VARIABLE (alias LET).

VARIABLE
Value defined on a grid.

VARIABLE NAME
The name by which a variable will be indicated in commands and expressions. Names begin with letters and may include letters, digits, dollar signs and underscores.

VARIABLE TITLE
A title string used to label plots and listed outputs of a variable.

VIEWPORT
A graphical display region which may be any subrectangle of a window. Graphical commands (PLOT, CONTOUR, etc.) take complete control of a viewport, clearing it as needed. A window may contain several viewports - possibly overlapping. Viewports are defined with DEFINE VIEWPORT and controlled with SET and CANCEL VIEWPORT.

WINDOW
A rectangular graphical display region. On a graphics terminal the terminal screen is the one and only window available. On a graphics workstation there may be many output windows.

WORLD COORDINATE
A coordinate system for referring to grid locations in which the points along an axis are regarded as continuous values in some particular units (e.g. meters of depth, degrees of latitude). The qualifiers X,Y,Z and T are provided to specify locations by world coordinate.