NOAA Data Report ERL PMEL-38

FERRET

A Computer Visualization and Analysis Tool for Gridded Data

S. Hankin
J. Davison
K. O'Brien
D. E. Harrison

Pacific Marine Environmental Laboratory
Seattle, Washington
January 1992

noaa NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION / Environmental Research Laboratories

NOAA Data Report ERL PMEL-38

FERRET

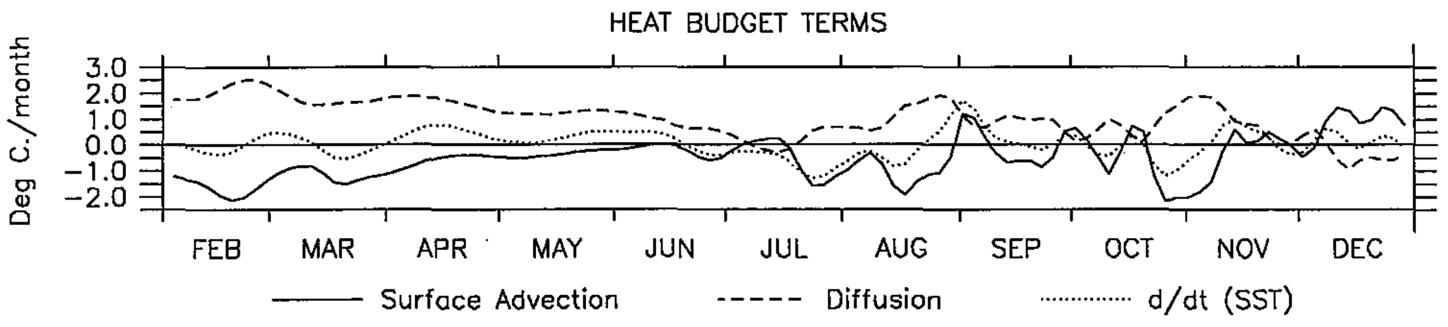
A Computer Visualization and Analysis Tool for Gridded Data

S. Hankin
Pacific Marine Environmental Laboratory

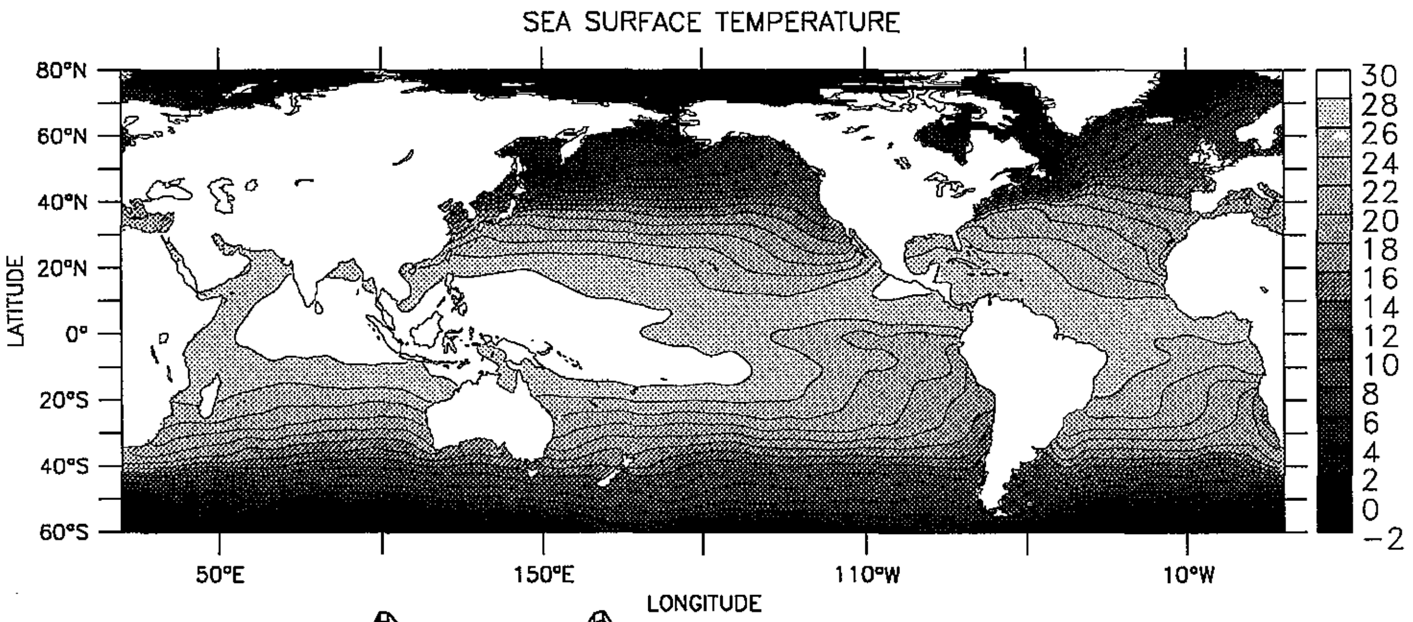
J. Davison
K. O'Brien
Joint Institute for the Study of Atmosphere and Ocean
University of Washington
Seattle, Washington

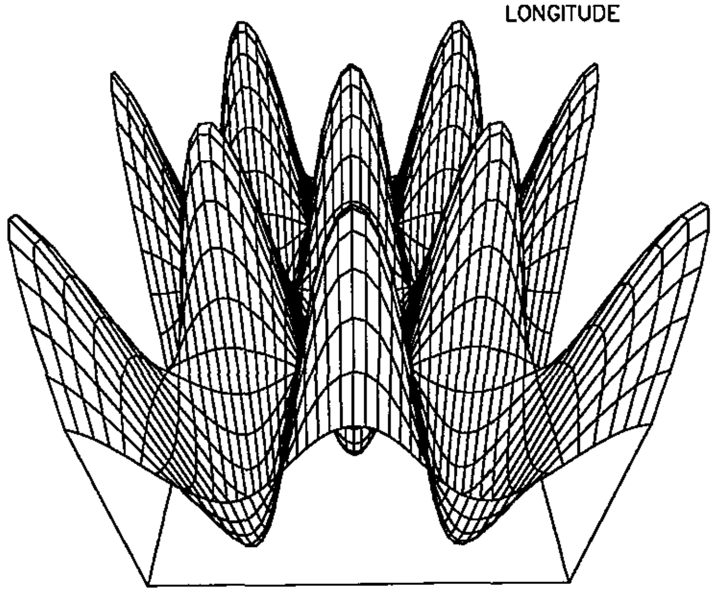
D. E. Harrison
Pacific Marine Environmental Laboratory

Pacific Marine Environmental Laboratory
Seattle, Washington
January 1992

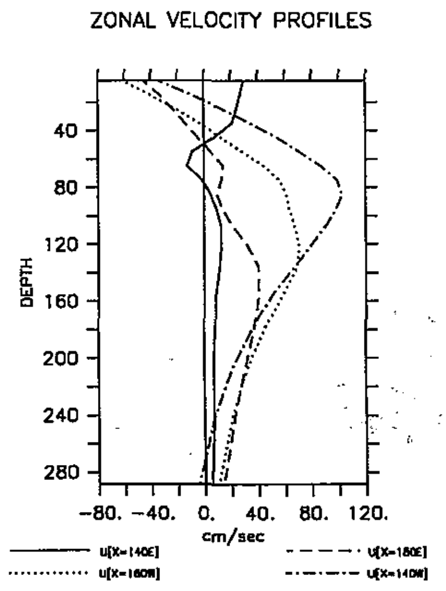


FERRET: An Analysis Tool for Gridded Data





COS(x)*SIN(x)



About the Frontispiece

The frontispiece opposite was produced by FERRET; several plot labels were removed with FERRET to reduce clutter on the page. The plot labelled “Heat Budget Terms” is based on output from the GFDL Ocean model forced with the “EDIT 2” wind data set produced by D.E. Harrison. Working from raw state variables output by the model FERRET was used to compute the heat budget terms and average the results over the domain 165°W to 155°W longitude and 1°S to 1°N latitude. The plot labelled “Zonal Velocity Profiles” contains profiles at the equator from the same data set. The plot labelled “Sea Surface Temperature” is from the surface level of the annual Climatological Atlas of the World Oceans by Sydney Levitus. The fish net plot of “COS(x)*SIN(x)” was created using a user-defined abstract variable within FERRET—no external data was used for this plot.

NOTICE

Contribution No. 1249 from NOAA/Pacific Marine Environmental Laboratory

For sale by the National Technical Information Service, 5285 Port Royal Road
Springfield, VA 22161

CONTENTS

FERRET
A Computer Visualization and Analysis Tool for Gridded Data

Steve Hankin[1], Jerry Davison[2], Kevin O'Brien[2], and D.E. Harrison[1]

1. INTRODUCTION

Program FERRET is an interactive computer visualization and analysis environment designed to meet the needs of physical scientists analyzing large and complex gridded data sets. FERRET was originally conceived and written to analyze the numerical ocean model data sets of the Thermal Modeling and Analysis Project (TMAP) at NOAA's Pacific Marine Environmental Laboratory in Seattle, Washington.

Data sets created by super-computers running large ocean circulation models are typically sequences of 3-dimensional "snapshots" of the oceans as they evolve in time (hereafter referred to as "4-dimensional"). It is not unusual for such data sets to exceed 2000 megabytes in size, often containing mixed 3-dimensional and 4-dimensional variables defined on staggered grids. Large data products and scattered observational data in the form of time series, ship data, and CTD casts are also essential in model validation and experimental design work.

FERRET was developed in the belief that a modern graphical workstation environment has sufficient power to interactively probe the underlying processes in even these multi-gigabyte data sets. With such a capability a scientist can analyze the synthetic world of the model much as he/she would wish to analyze the real world if such a complete set of observational data existed.

Many excellent software packages have been developed recently for scientific visualization. The features that make FERRET distinct among these packages are

- **flexibility**

  FERRET provides a collection of basic transformations (averaging, integrating, smoothing, arithmetic and trigonometric operators, etc.) and a familiar, mathematics-like language for interactively defining new variables from older ones.
- **integration of analytical power into the graphical environment**

  FERRET's graphical and analytical tools can be applied as readily to variables defined by the user as they can to the "raw" data. FERRET's graphical tools include animations, color shaded plots, contour plots, line and scatter plots, 3-dimensional fish nets, multiple windows and overlays.

[1] NOAA/Pacific Marine Environmental Laboratory, 7600 Sand Point Way N.E., Seattle, WA 98115-0070

[2] Joint Institute for the Study of Atmosphere and Ocean, University of Washington, Seattle, WA 98195

- **symmetrical processing in four dimensions**
  FERRET treats all four axes of the space-time coordinate system as equivalent. No special commands are required and no limitations are imposed when working with time series data.
- **"intelligent" connection of FERRET to its data base**
  FERRET automatically provides full labelling and titling of all plots with information pulled from the data in a self-describing format. More than a labor saver, this feature is a quality assurance on FERRET's outputs.
- **ability to analyze very large data sets**
  FERRET has built-in memory and disk management capabilities to handle data sets too large to fit within the limitations of on-line storage. FERRET renders these machine limitations invisible to the user for most applications, giving the user the impression of a greatly enlarged virtual storage environment.

FERRET has evolved to its present mature state over a 7-year period with contributions by several programmers representing approximately 10 man-years of effort. FERRET is written in standard FORTRAN 77—approximately 35,000 lines of code—with graphics based on the ISO GKS standard. FERRET supports a number of terminals, windowing systems and hard-copy devices. It is currently running on VAX/VMS and DEC Ultrix computers with a port to SUN Unix underway.

2. SUMMARY OF FERRET CAPABILITIES

Graphical Outputs

(All graphical styles except 3-D fish nets are automatically labelled with complete, unambiguous labels)

- line plots (multiple lines generate a line style key)
- scatter plots
- contour plots
- vector arrow plots
- color shaded plots
- 3-dimensional fish net plots
- user-specifiable plot aspect ratio
- multiple plots per page (viewports)
- any number of overlaid plots
- detailed customization of all aspects of graphics

Output graphical devices

Note: FERRET is based on the ISO GKS device-independent graphical standard. GKSM metafiles and are used to provide hard copy. Output device support is ultimately determined by the GKS implementation.

- PostScript
- Color PostScript
- Encapsulated PostScript
- X-windows
- HPGL
- Tektronix (4014 and 4107)
- Sixel
- Regis
- Computer Graphics Metafile (ISO CGM)
- GKSM

Output formats

- free-format ACSII
- user-specified formatted ASCII (FORTRAN format descriptors)
- binary floating point
- FERRET GT format (direct access, "grids at timesteps")
- FERRET TS format (direct access, "time series")
- PMEL EPIC format
- netCDF (under development as of 12/91)

Input formats

- free-formatted ASCII (blank, comma, or tab separated fields)
- formatted ASCII (FORTRAN format descriptors)
- binary floating point
- FERRET GT format
- FERRET TS
- netCDF (under development as of 12/91)

Memory management

- Calculations too large for physical memory will be broken into smaller pieces in a manner optimized for efficient reading of data from disk.

Mathematical Expressions

Logic Structures: IF ... THEN ... ELSE ...

Operators: + – * / ^

Functions (evaluated at each grid point):

minimum, maximum, integer truncation, absolute value, exponential, natural and common logarithm, trigonometric functions, inverse trigonometric functions, modulo, missing value substitution, uniform and normally distributed random number

Transformations (applied to an axis of a variable):

definite and indefinite integral, average, location of value, minimum, maximum, shift, smoothers (boxcar, binomial, Hanning, Parzen, and Welch), fill-with-average for missing data, forward, backward, and centered derivatives, interpolation

Animations

Limited animation capability is available using FERRET GKSM metafiles and the utility "mtta" provided with FERRET.

3. FERRET PROGRAM CONCEPTS

Plottable variables

A "plottable variable" is any variable that may be displayed graphically, listed to a file, or used in the calculation of other variables. There are no non-plottable FERRET variables but the term "plottable variable" avoids ambiguities.

Plottable variables are of 4 types

FILE VARIABLES	- read from disk files
USER-DEFINED VARIABLES	- defined by the LET command
PSEUDO-VARIABLES	- grid information used as variables ("I", "J", "K", "L", "X", "Y", "Z", "T",...)
DIAGNOSTIC VARIABLES	- variables internal to the GFDL OCEAN model

As much as possible FERRET makes all types of variables indistinguishable.

All plottable variables are defined on grids

The grid on which a plottable variable is defined tells how to locate the variable in space and time. In cases where the variables are abstract in nature—not associated with geographical location or time—FERRET will associate those variables with grids that are abstract, too. Whereas a geographical grid will associate the Nth position along an axis with a location (say, 20 degrees north latitude) an abstract grid will simply associate the Nth position with the number N. It is simple to instruct FERRET to regrid plottable variables to other grids than the one on which they are defined.

Grids

All FERRET grids are 4-dimensional—composed of 4 axes, each describing locations along one dimension. Grids of 3, 2, 1, and 0 dimensions are regarded as special cases of the full 4 dimensions where one or more axes have been indicated as "NORMAL." In most cases the axes have the obvious interpretation of three space coordinates and time but other interpretations are supported, as well.

FERRET preserves symmetry among the axes; the same syntax of regions and transformations applies to all axes. Calendar dates, east-west longitudes and north-south latitudes are merely convenient ways to format numerical values along axes that have special interpretations to people—not to FERRET. (The only exception to this is that if the Y axis has units of "latitude" FERRET will insert the cosine of the latitudes as weighting factors where required in some calculations.)

Axes and grids may be defined interactively using the DEFINE AXIS and DEFINE GRID commands. They may also be defined by "grid files" (ASCII files which normally have the filename extensions, ".grd").

Contexts

All references to plottable variables must have a complete context: a region or point in space and time and the name of a data set. This is the information needed by FERRET to make sense of references to plottable variables. A command like "PLOT U" is meaningful only when FERRET knows what data set to access and where in 4-dimensional space it is supposed to seek the plottable variable, U.

Subscripts and world coordinate positions may be mixed in the context

Subscripts are specified using the syntax I=..., J=..., K=..., L=... for axes 1 through 4, respectively. For example, "I=3" specifies a value of 3 for the subscript on the first axis and "K=5:10" specifies a range of subscripts 5 through 10 for the third axis. World coordinates are similarly specified by X=..., Y=..., Z=..., T=.... Special formats are provided for X=longitude, (e.g., X=160W), Y=latitude, (e.g., Y=23.5S) and T=calendar date (e.g., T="7-NOV-1989").

The data set may be given by name or number

The commands SET DATA and CANCEL DATA and the D= context descriptor all accept the name of the data set or its number. The data sets are numbered by the order in which they are specified using SET DATA. FERRET will display this order in response to the SHOW DATA command.

The context may be specified or modified in 3 places

1) **The program context**

   Using the commands SET REGION and SET DATA you can describe the context to be used for interpreting plottable variables and expressions. You can look at the program context with SHOW REGION and SHOW DATA. (The command SET DATA is used both to initialize new data sets and to designate a previously initialized data set as the current default. When SET DATA initializes a new data set it automatically becomes the current default data set.)

   Examples:

   ```
   SET REGION/Y=23.5N/T="1-JAN-1981"
   SET DATA COADS
   ```

2) **The command context**

   Using the command qualifiers I, J, K, L, X, Y, Z, T, and D commands like PLOT, CONTOUR, SHADE, LIST, and VECTOR can specify context information. Command context information on any axis or on the data set will replace any program context information on the same axis or the data set for the duration of that command.

   Examples:

   ```
   PLOT/Z=200      TEMP
   SHADE/D=COADS      SST
   ```

3) **The variable context**

   Using square brackets following the variable name an individual plottable variable can be modified with additional context information. Variable context information will replace any program or command context information on the same axis or the data set for the modified variable, only.

   Examples:

   ```
   CONTOUR  U[D=COADS] - U[D=EDIT2]
   PLOT/Y=5N  SST - SST[Y=5S]
   ```

Transformations

Transformations are mathematical calculations that must be are applied along a specified axis or axes. Transformations may be specified only in the variable context—applied to the space or time region qualifiers of that plottable variable, only.

Examples:

```
PLOT U[Z=0:100@AVE]   - the variable U averaged between Z=0 and 100
LIST U[L=1:100@SBX:5]  - a smoother of width 5 points along the L axis
```

See the FERRET Users' Guide for a list of available transformations.

4. FERRET DATA CONCEPTS

Variables --> Grids --> Axes

All "plottable variables" (see FERRET Program Concepts in this technical memorandum) in FERRET are defined on grids. The grids are composed of axes which in turn are composed of points. From a data analysis perspective a sequence of these points may be regarded as the independent data where the dependent data values are the plottable variables, themselves.

FERRET grid structures are designed to:

- provide a symmetrical treatment of all axes;
- preserve grid structure relationships between variables (for example, models using staggered grids may have velocity staggered from temperature yet these variables may share the same time stepping structure and possibly the same vertical layering);
- separate the notion of a "plottable variable" from the specifics of the grid coordinates on which it is defined, ultimately allowing software to make intelligent decisions regarding the regridding of data; and
- provide a uniform set of structures describing the range of geometries of interest. (Note that not all of the possible grid geometries have been realized yet in FERRET as of version 2.2).

Definitions:

Axis - a structure consisting of a title, units and a sequence of values to be used as coordinates.

- The spacing of points may be REGULAR, IRREGULAR, or DISORDERED.
- Axes have titles (for labelling outputs).
- Axes have units (for labelling and for interconvertibility).
- Each axis has a unique name.

Grid - a structure consisting of 4 axes, an XY plane rotation angle and an inner/outer product association for each axis

- Each grid has a unique name.
- Grids point to axes by name.
- Dimensions which are orthogonal to the space of interest are designated "NORMAL" (for example the vertical axis of a grid used for horizontal wind stress).
- Several grids can share the same (named) axis.
- Grids can represent simple rectangular outer products.
- Grids can represent simple tuples (2- ,3- , and 4-dimensional).

- Grids can represent complex (but often occurring) combinations of tuples and rectangular structures. For example an XY scatter of current meters will frequently be deployed at the same depths and synchronized on the same time axes. This structure can be precisely represented.
- XY plane rotation permits ship tracks to be represented as lines.

Variable - a structure consisting of a grid, data values defined on that grid, a variable name, a title and units

- Variables point to grids by name.
- Multiple variables may share the same grid.
- Regridding is easily represented since variables (as "objects") are separated from grids.
- Titles and units are essential to generate automatic labelling.
- Unit conversions may be automated.

5. IMPLEMENTATION

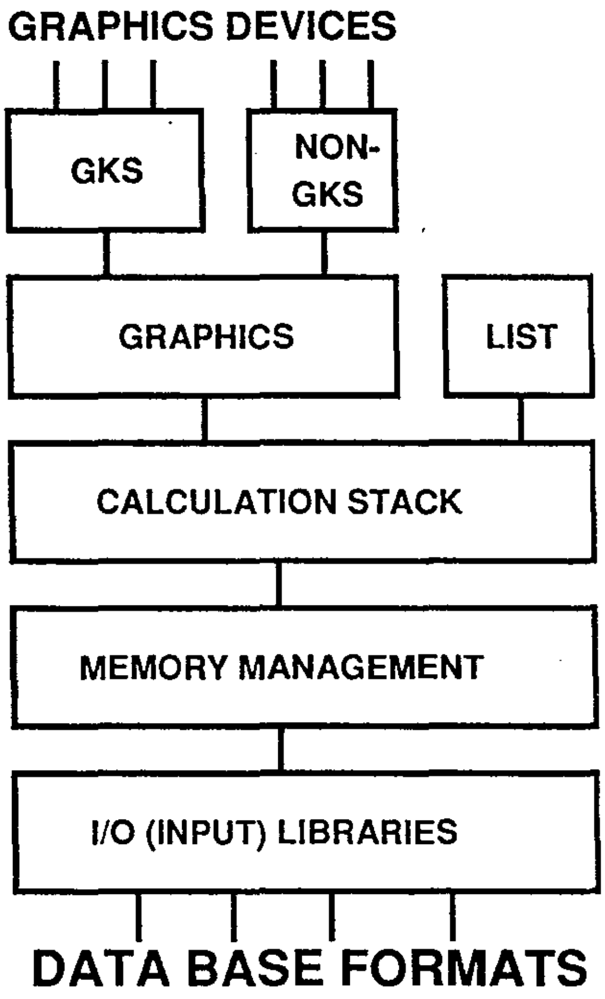
FERRET is written almost entirely in transportable FORTRAN 77; the only language extensions that have been employed are the use of variable and subroutine names in excess of six characters. A small amount of C language code has been incorporated to interface well with Unix file systems.

FERRET is highly modular in design. It consists of approximately 500 separate routines few of which exceed two pages in length. As Figure 1 shows, FERRET is broken down into distinct functional layers. The lowest layer is the I/O interface. The I/O libraries support sequential ASCII and binary floating point files as well as the direct access "GT" (grids at time steps) and "TS" (time series) formats. The libraries are designed to accommodate additional formats in the future; support for the netCDF libraries is currently under development at this level.

The memory management layer consists of low level routines that create an optimized virtual memory environment internal to FERRET. FERRET manages memory in blocks which it allocates and frees as variables are accessed for graphics and calculation. FERRET manages this memory on a "least recently used" basis. Users experience this as a performance enhancement: fields that have recently been examined will be instantly accessible as they will still remain in memory.

FERRET also manages memory to permit calculations of theoretically limitless size. FERRET anticipates the number of data values that will be required for a given calculation and if this number exceeds a threshold (controlled by the settable mode "desperate") FERRET determines how to divide the calculation into smaller pieces. This division is made in a way that will optimize disk data access times.

IMPLEMENTATION



- **VAX/VMS**
- **Unix**
- **FORTRAN 77**
- **PLOT+** (D. Denbo - custom graphics)
- **GKS** (device independence)

Figure 1. FERRET implementation.

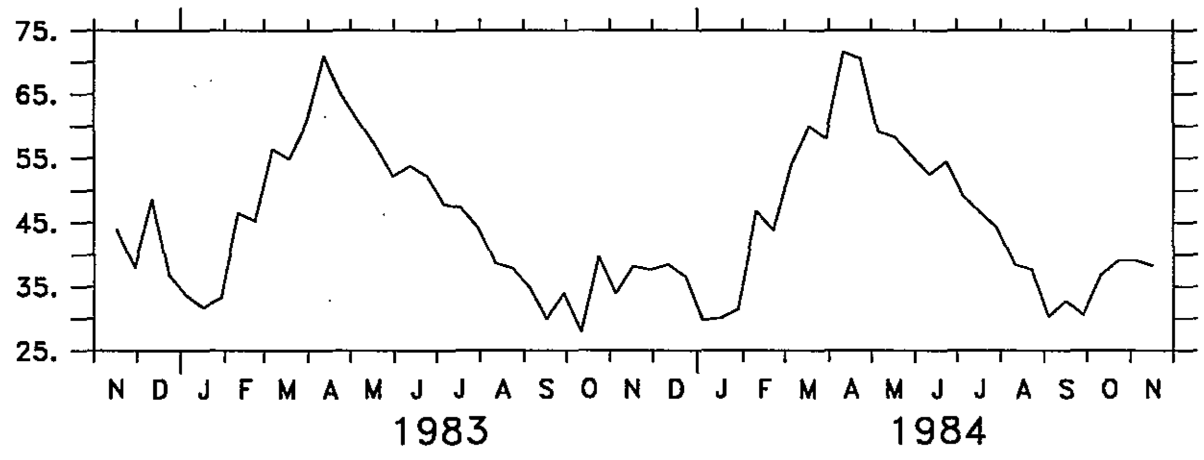
The optimization used in memory management is most easily understood by example. Suppose a user has requested FERRET to calculate the average temperature of a 3D volume of ocean over a range of time. Although the result is a single value the component data required is a 4-dimensional field which may be very large. If the size of this field exceeds the mode desperate threshold then FERRET will split the calculation into pieces. In determining the optimal way to do this it will consider the manner in which the component data are stored on the disk. If the data are in "GT" format then each time step of data is stored as a contiguous sequence of logical disk blocks resulting in disk seek delays when accessing data from separate time steps. In this case FERRET would divide the calculation into a sequence of 3-dimensional spacial averages, postponing the time-axis average until the final step. If the component data required for each 3-dimensional calculation still exceeded the mode desperate threshold these averaging operations would in turn be broken into smaller calculations using a similar optimization logic.

Above the memory management stack layer is the calculation stack. This layer breaks the complex mathematical expressions specified by the user into sequences of binary and unary operations that can be performed on a stack. The calculation stack makes requests of the memory management layer as it requires memory for intermediate results.

Above the calculation stack are the routines that direct output—formatting the data listings and determining the layout of graphics. These routines make requests of the calculation stack to obtain the variables to be displayed. These routines generate graphical commands. The user's commands directly address this layer of FERRET.

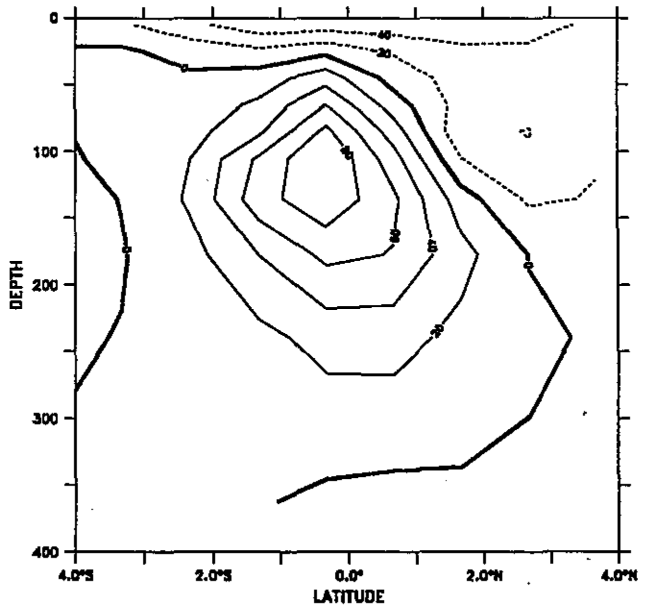
Graphical commands, in turn, pass through two more layers before they produces visible output. The graphical commands generated by FERRET are ASCII formatted commands to the PLOT+ program (written by Dr. D. Denbo) which is embedded inside of FERRET. (These commands may be captured in a file, modified, and played back as a command script—one method of obtaining customized graphical output.) The PLOT+ program may generate device dependent graphics instructions or calls to the ISO GKS device-independent graphics interface according to the state of PLOT+ "pltype" command. FERRET typically uses PLOT+ in GKS mode. GKS then creates device-dependent graphics instructions and device-independent graphical metafiles that can be rendered as hard-copy at a later time.

Appendix A. FERRET: A Worked Problem

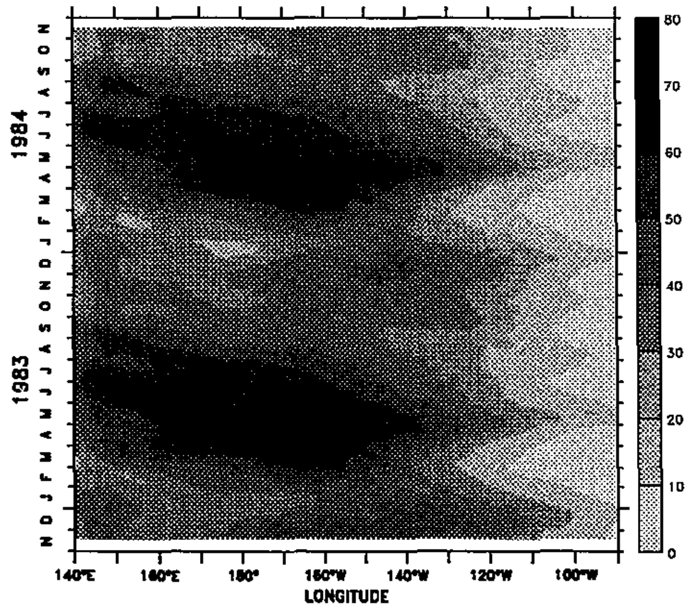


Transport of Undercurrent (Sverdrups)

FERRET
Version 2.2
A Worked Problem



Zonal Undercurrent



Transport

The following pages contain an example analysis using **FERRET.**

Example:

Compute the Volume transport of the equatorial counter-current from a gridded field of U (Zonal Velocity)

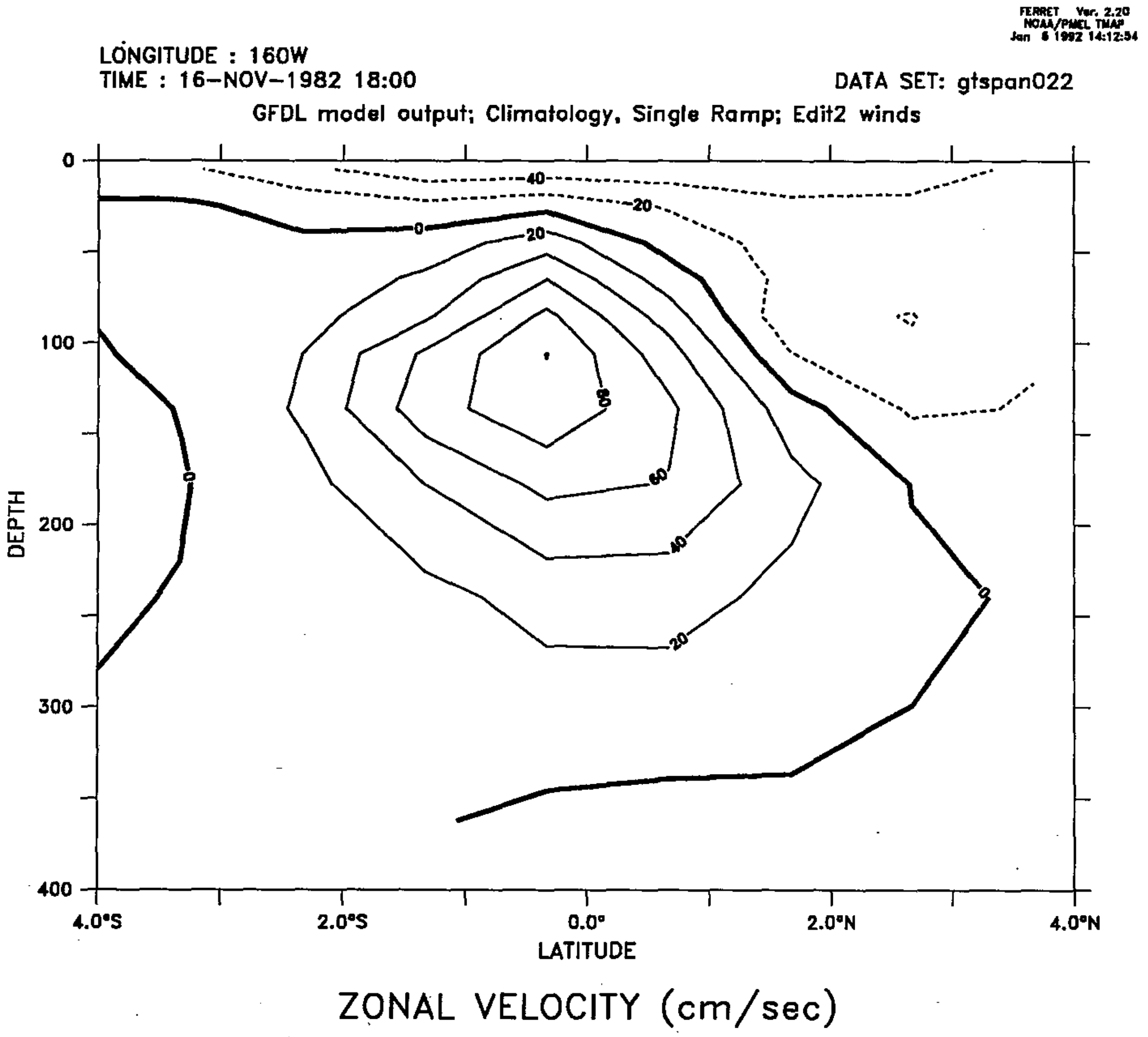
Step 1:

Designate the region of interest:

- Date: **16 November, 1982**
- Longitude: **160°W**
- Latitude: **range of 4°S to 4°N**
- Depth: **range of 0 to 400 meters**

Then enter the command:

"CONTOUR U"

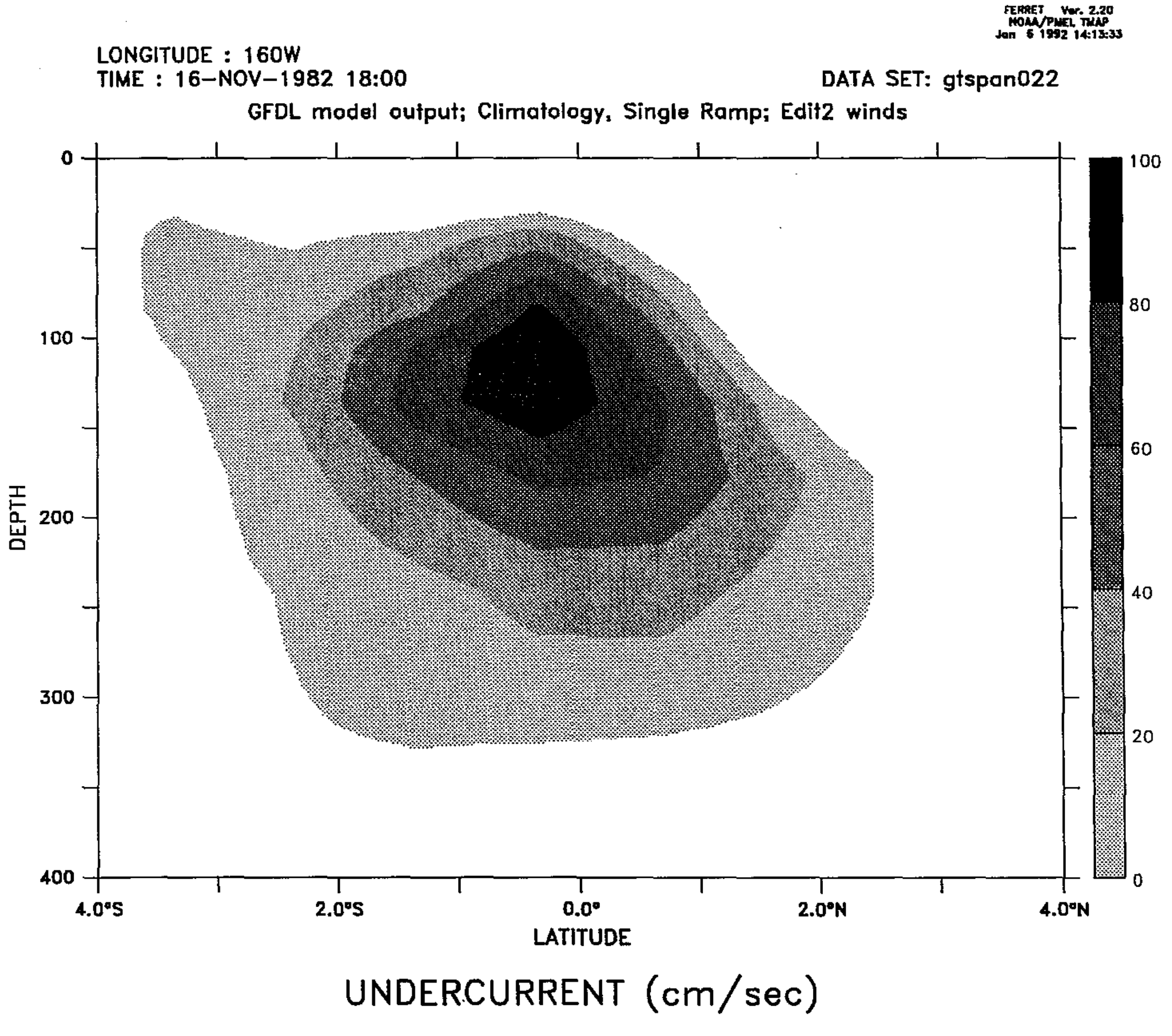


Step 2:

Define "und_cur" to be U only where U > 0
(eastward – other points omitted)

Then enter the command:

"SHADE und_cur"



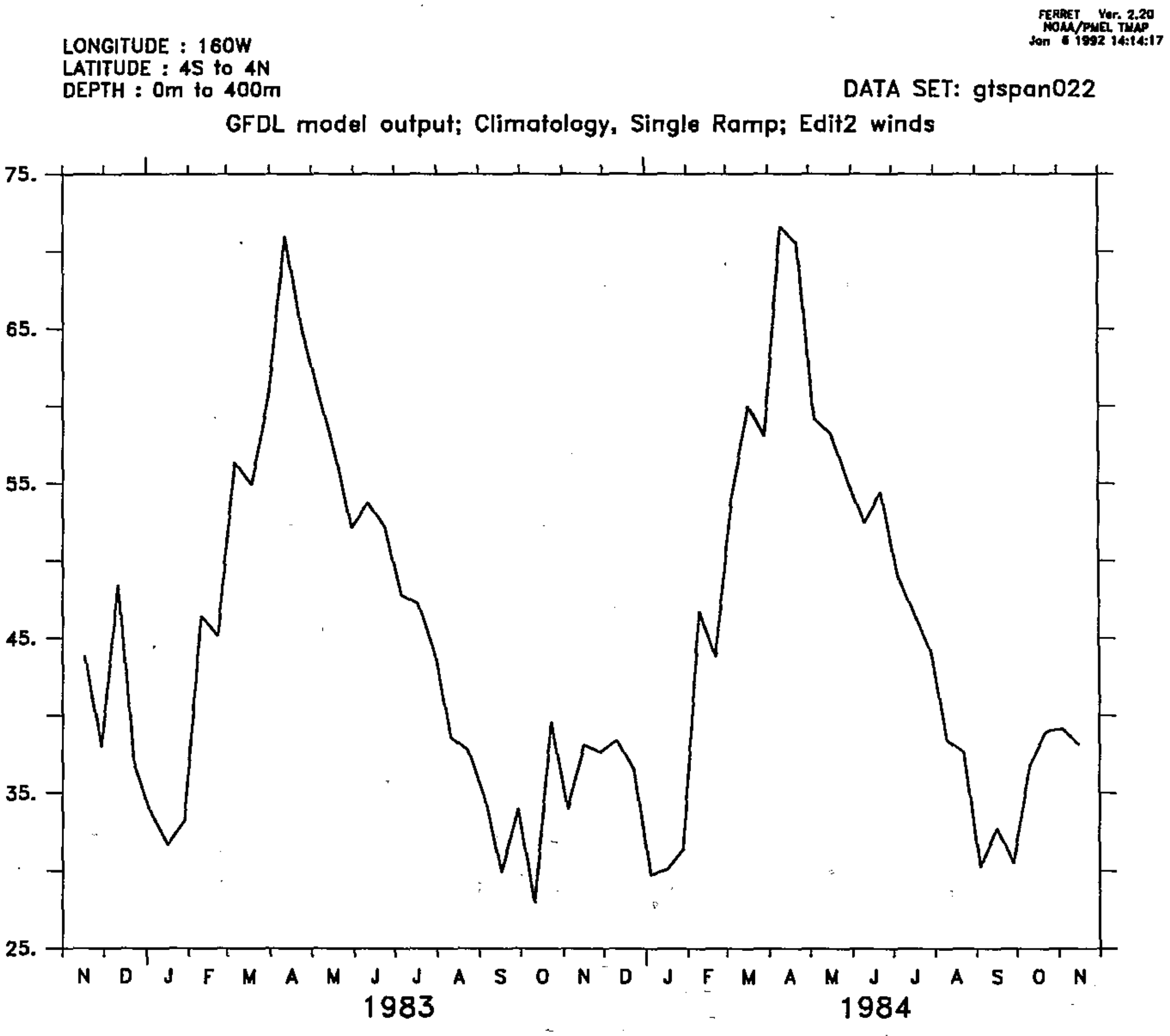
Step 3:

Define "und_trans" to be $\iint$(und_cur) dy dz

Designate
    Date: **16 November 1982 to 15 November 1984**

Then enter the command:

"PLOT und_trans"



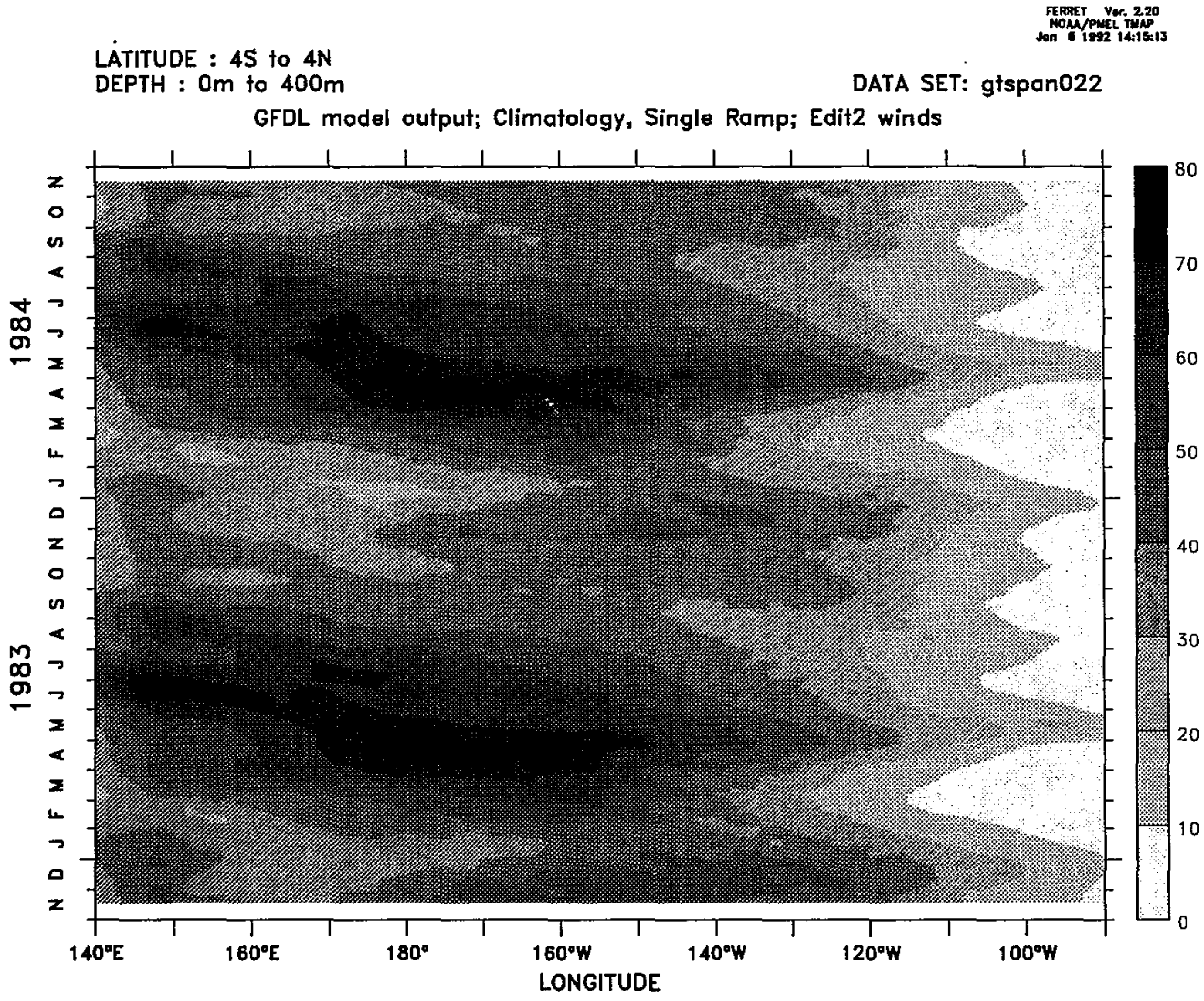
TRANSPORT OF UNDERCURRENT (Sverdrups)

Step 4:

Designate
  Longitude: **140°E to 90°W**

Then enter the command:

"SHADE und_trans"



EVOLUTION OF UNDERCURRENT TRANSPORT (Sverdrups)

Appendix B. FERRET User's Guide

FERRET

USERS GUIDE

Version 2.20

NOAA/PMEL/TMAP

Steve Hankin
November 1991

CONTENTS

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. Axes may be defined with the DEFINE AXIS command or in grid files.

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 if the data is to be regridded.

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 TMAP data set: variable names, coordinates, units and pointers to the data files. Descriptor file names normally end with ".DES". (See appendices)

DIAGNOSTIC VARIABLE

A quantity internal to the TMAP model runs 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 TMAP-format.

FERRET FORMAT

A synonym for "TMAP FORMAT."

FILE VARIABLE

A variable made available with the SET DATA command. File variables are data on 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

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 or as "inner products" creating ordered pairs, 3-tuples or 4-tuples. Grids may be defined with the DEFINE GRID command or from grid files.

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 TMAP-format. (See appendices.)

METAFILE

A representation of graphics stored in a computer file. Such a file can be processed by an interpreter program (such as MOM or GMOM) 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.

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 in qualifiers. In the command "SHOW DATA/FULL" the "/FULL" is a qualifier. In the variable "SST[Y=20N]" the "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 (see also GRIDS).

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 "@ttt". 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 have been created by the Thermal Modeling and Analysis Project (TMAP) to ease the handling of very large data sets. These formats use descriptor files to store information about the variables, units, titles and grids for the data so FERRET plots and listings will be fully documented and labelled. The formats are compact (binary) and efficient (direct access) and allow large data sets to be only partly resident on-line. Separate formats allow optimized access as time series (TS format) or as geographical regions (GT format).

USER-DEFINED VARIABLE

A variable created with the DEFINE VARIABLE (or LET) command.

VARIABLE

Values defined on a grid.

VARIABLE NAME

The 1 to 8 character 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; by default each window will have only a single viewport filling the entire window. 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.

Chapter 1: OVERVIEW

1. INTRODUCTION

Program FERRET is an interactive program useful for examining and analyzing well-ordered data such as time series or the output of numerical models. Program FERRET is especially designed to handle very large data sets - too large to fit on-line. Program FERRET allows the user to probe the data set by slicing along various planes and axes, interactively examining raw and transformed data.

Program FERRET permits the user to work with variables that are explicitly stored on disk, with abstract mathematical functions, with mathematical transformations and combinations of disk variables and, for GFDL/Philander/Siegel model users, with "diagnostic" variables computed from the on-line data.

Output from FERRET may be listings of data on the computer screen or in computer files, graphics on the computer screen or graphical "metafiles" which may later be rendered as hard copy graphics. FERRET will provide fully-documented and usable graphical output with single commands: line plots, vector arrow plots, contours and shaded contours. Fully customized graphics are also available without leaving program FERRET by using commands to program PPLUS. PPLUS is imbedded in program FERRET.

Further information describing general features of program FERRET is available under the topics:

Getting Started	get acquainted with FERRET
Syntax	syntax of statements used to give commands to FERRET
Data sets	specifying the data set(s) to access
Regions	specifying the space/time region
Variables	what variables are normally available
Expressions	creating mathematical expressions involving variables
Grids	how to examine and manipulate grids underlying data
Graphics	options available and techniques for advanced graphics
Movies	generating animations
Tools	pre-defined command files to perform special functions

2. GETTING STARTED

2.1 Concepts

Words in bold below, are defined in the glossary section of this help manual.

In program FERRET all **variables** are regarded as defined on **grids**. The grids tell FERRET how to locate the data in space and time (or whatever the underlying units of the grid **axes** are). A collection of variables stored on disk is a **data set**.

To access a variable FERRET must know its name, data set and the **region** of its grid that is desired. Regions may be specified as **subscripts** or in **world coordinates**. Data sets, after they have been pointed to with the SET DATA command, may be referred to by data set number or name.

Using the LET command new variables may be created "from thin air" as **abstract expressions** or created from other known variables as arbitrary **expressions** combining them. These new variables may be specified for all graphical and computational commands. When multiple component variables are used in a single expression they must be on compatible grids. If a component variable is on a different grid than the one desired it may be **regridded** simply by naming the desired grid. (see also GRIDS).

The user need never explicitly to tell FERRET to read data. From start to finish the sequence of operations needed to obtain results from FERRET is simply:

1) specify the data set
2) specify the region
3) request the output

For example,

```
yes? SET DATA MY_DATA
yes? SET REGION/Z=0/T="1-JAN-1982"/X=160E:160W/Y=20S:20N
yes? VECTOR TAUX,TAUY
```

2.2 Sample sessions

A typical short session using program FERRET:

```
$ FERRET                                     ! initiate FERRET
yes? SET DATA SADLER_HINDCAST                ! specify data set
yes? SHOW DATA/FULL                          ! what's in it ?
yes? SET REGION/X=180W/Y=5S:5N/Z=0:100/L=1   ! geographical region
yes? LIST  U                                 ! look at the numbers
yes? CONTOUR U                               ! make a contour plot
yes? EXIT
```

2.2.1 Reading an ASCII data file

Many examples of accessing ASCII data are available under Data Sets

The simplest access, a single variable with one value per record in the data file would typically look like

```
$ FERRET
yes? FILE my_file.dat
yes? SET VARIABLE/TITLE="Widths of computer cabinets" V1
yes? PLOT V1
yes? EXIT
```

2.2.2 Using viewports

Many examples and detailed information are available under Graphics Viewports, especially under "Advanced usage"

An application drawing 4 plots in 4 quadrants of a window would typically look something like

```
$ FERRET
yes? SET DATA SADLER_HINDCAST
yes? SET REGION/X=180W/Y=5S:5N/Z=0:100/L=1
yes? SET VIEWPORT LL                          ! lower left
yes? CONTOUR QFLX
yes? SET VIEWPORT LR                          ! lower right
yes? CONTOUR TEMP
yes? SET VIEWPORT UL                          ! upper left
yes? VECTOR U,V
yes? SET VIEWPORT UR                          ! upper right
yes? VECTOR TAUX,TAUY
yes? EXIT
```

2.2.3 Using abstract variables

(see glossary for a definition of "abstract variable")

Several examples and detailed information are available under Variables Abstract variables

A user wishing quickly to examine the function SIN(X) on the interval [0,3.14] might use

```
$ FERRET
yes? PLOT/I=1:100 SIN(3.14*I/100)
yes? EXIT
```

2.2.4 Using transformations

(see glossary for a definition of "transformation")

Detailed information about the use of transformations is available under Expressions Transformations.

In a typical access a user may wish to look at ocean temperatures averaged from 0 to 100 meters of depth using a data set which has detailed resolution in depth

```
$ FERRET
yes? SET DATA OCEAN_MODEL
yes? SET REGION/Y=5n/X=160W/T="1-JAN-1982":"31-DEC-1982"
yes? PLOT TEMP[Z=0:100@AVE]
yes? EXIT
```

2.2.5 Using algebraic expressions

See detailed information under Expressions

An actual example from a TMAP climatological data set:
The data set contains raw temperature observations. The data set has many holes and is very noisy. We wish to look at the field of d/dT(SST) in units of degrees per 730 hour "model month".

```
$ FERRET
yes? SET DATA GTMNCLED2             - monthly "EDIT 2" climatology
yes? SET REGION/@T                  - tropics
yes? LET/TITLE="HOLE-FILLED SST"        FILL = SST[X=@FAV:10]
yes? LET/TITLE="FILLED,SMOOTHED SST"    SMTH = FILL[X=@SBX:10]
yes? LET C = 60*60*730
yes? LET/TITLE="FILLED,SMOOTHED DTDT"   DTDT = SMTH[L=@DDF] * C
yes? CONTOUR/L=1 DTDT
yes? EXIT
```

2.2.6 Finding the 20 degree isotherm

Isotherms can be located with the "@LOC" transform, which returns axis location where the value of the argument of @LOC first occurs. Thus, "TEMP[Z=0:200@LOC:20]" will locate the first occurrence of the value 20 along the Z axis of TEMP, scanning all the data between 0 and 200 meters.

A typical session examining mid-Pacific ocean data:

```
$ FERRET
yes? SET DATA OCEAN_MODEL
yes? SET REGION/Y=20s:20n/X=140E:140W/T="15-MAR-1985"
yes? CONTOUR TEMP[Z=0:200@LOC:20]
yes? EXIT
```

3. Common commands

A quick reference to the most commonly used FERRET commands:

SET DATA	name the data set to be analyzed
SHOW DATA	produce a summary of data sets (also /FULL)
SHOW GRID	examine the coordinates of variable
SET REGION	set the region to be analyzed
LIST	produce a listing of data
PLOT	produce an X-Y plot
CONTOUR	produce a contour plot
VECTOR	produce a vector arrow plot
SHADE	produce a shaded-area plot
STATISTICS	produce summary statistics about a variable
LET	create a new variable
USER	execute a user-defined command

Help is available for all of these topics.

4. Syntax

Program FERRET commands conform to the following template:

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

where

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

notes...

i) Items in square brackets are optional;

ii) 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;

iii) Command names, subcommand names, and qualifiers require at most 4 characters.
(e.g. `yes? CANCEL LIST/PRECISION` is equivalent to
`yes? CANC LIST/PREC`);

iv) Some qualifiers take an argument following an equals sign
(e.g. `yes? LIST/Y=10S:10N`).

5. Data Sets

To specify the data set use

```
yes? SET DATA_SET set_name1,set_name2,...
```

for example

```
yes? SET DATA GTBA018
```

To examine the data sets after specifying them with SET DATA use

```
yes? SHOW DATA_SET
```

ASCII files can be read using SET DATA/EZ (also known as "FILE").

To refer to a data set other than the current default use the "D=dset" or "D=number" syntax where "dset" is the name of the data set and "number" is the number given by SHOW DATA_SET. For example,

```
yes? LIST/D=2  U
   or
yes? LIST U[D=GTSA014] - U[D=GTSA013]
```

5.1 TMAP-formatted data

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" or "TS" format. ("FERRET" format and "TMAP" format are synonyms.)

On VMS systems, if the directory or extension is omitted from the descriptor filename, the defaults will be used. The VMS directory name default is the logical TMAP_SETS. The VMS filename extension default is ".DES".

On Unix systems, if the directory portion of the path 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.

5.1.1 Descriptors

For every TMAP-formatted data set there is a descriptor file containing summary information about the contents of the data set. When the command SET DATA_SET is given to FERRET it is the name of the descriptor file that must be specified. Both "GT" (grids-at-timesteps) and "TS" (time series) formats are supported by FERRET version 2.20. (See Chapter 3)

If the data set specified contains TMAP Philander/Seigel/Mom model output, special information is needed by FERRET to compute diagnostic variables. Normally, this information is pulled from the TMAP 205 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'
```

5.2 Binary data

To access binary (unformatted) data sets use

```
SET DATA/EZ ASCII_or_binary_file_name
```
or equivalently
```
FILE ASCII_or_binary_file_name
```

The procedure to access binary data is identical to the procedure for ASCII data except that the qualifier

```
/FORMAT=UNFORMATTED
```

is used.

Note: Binary data words must all begin on 4-byte boundaries to be accessible by FERRET

See also SET DATA ASCII data

5.3 ASCII data

To access ASCII data files sets use

```
SET DATA/EZ   ASCII_or_binary_file_name
```
or equivalently
```
FILE    ASCII_or_binary_file_name
```

Use /VARIABLES to name the variables in the file
Use /TITLE to associate a title with the data set
Use /GRID for multi-dimensional data and for units
Use /COLUMNS to tell how many columns are in the file
Use /FORMAT to specify the format of the file
Use /SKIP to skip initial records of the file

See SET VARIABLE to individually customize the variables

Several examples are presented for how to access ASCII data. "1 dimensional" refers to a line of data (e.g. a time series).

1) 1 dimensional data, single variable (simplest case)
2) 1 dimensional data, multiple variables
3) 2 dimensional data, single variable
4) 2 dimensional data, multiple variables
5) 3 and 4 dimensional data

5.4 Reading ASCII files

Example 1

1 dimensional data, single variable (simplest case)

Case 1:
Fragment of the data file, SNOOPY.DAT, with variable WIDTH

```
     2.3
     3.1
     4.5
     .
     .

     yes? FILE/VAR=WIDTH SNOOPY.DAT
     yes? PLOT WIDTH
```

Case 2: (multiple columns of the same variable)
Fragment of the data file, SNOOPY.DAT, with variable WIDTH

```
     2.3  3.1  4.5  6.2
     5.1  4.2  8.2  7.9
     .
     .

     yes? FILE/VAR=WIDTH/COLUMNS=4 SNOOPY.DAT
     yes? PLOT WIDTH
```

Example 2

1 dimensional data, multiple variables

Case 1:
Fragment of the data file, SNOOPY.DAT - variables WIDTH and HT (headings NOT included in the file)

```
     WIDTH  HT
      2.3  20.4
      3.1  31.2
      4.5  15.7
      .     .
      .     .

     yes? FILE/VAR="WIDTH,HT" SNOOPY.DAT
     yes? PLOT WIDTH,HT
```

Case 2:
Fragment of the data file, SNOOPY.DAT - variables WIDTH and HT with variables repeated in groups (headings NOT included in file)

```
     WIDTH  HT        WIDTH  HT       WIDTH  HT
      2.3  20.4        3.1  31.2       4.5  15.7
      1.5  12.2        4.1  25.3       3.9  17.7
      .     .                          .
      .     .

     yes? FILE/VAR="WIDTH,HT"/COLUMNS=6 SNOOPY.DAT
     yes? PLOT WIDTH,HT
```

Example 3

2 dimensional data, single variable

Assume the data are on a 4 (column) by n (row) grid. Need first to define a grid conforming to the data.

Fragment of the data file, SNOOPY.DAT - variable WIDTH

```
     2.3  3.1   4.5  3.9
     3.3  7.7   5.6  9.2
      .    .
      .    .

     yes? DEFINE AXIS/X=1:4:1 XSNOOPY
     yes? DEFINE AXIS/Y=1:30:1 YSNOOPY
     yes? DEFINE GRID/X=XSNOOPY/Y=YSNOOPY   GSNOOPY
     yes? FILE/VAR=WIDTH/GRID=GSNOOPY/COLUMNS=4   SNOOPY.DAT
     yes? CONTOUR WIDTH
```

Example 4

2 dimensional data, multiple variables

Assume the data are on a 3 (column) by n (row) grid. Need first to define a grid conforming to the data.

Fragment of the data file, SNOOPY.DAT - variables WIDTH,HT (headings NOT included in the file)

```
     WIDTH  HT
      2.3  20.4
      3.1  31.2
      4.5  15.7
      5.6  17.3
      4.7  12.9
      5.4  21.3
       .     .
       .     .

     yes? DEFINE AXIS/X=1:3:1 XSNOOPY
     yes? DEFINE AXIS/Y=1:30:1 YSNOOPY
     yes? DEFINE GRID/X=XSNOOPY/Y=YSNOOPY GSNOOPY
     yes? FILE/VAR="WIDTH,HT"/GRID=GSNOOPY SNOOPY.DAT
     yes? SHADE WIDTH
     yes? CONTOUR/OVERLAY HT
```

Example 5

3 and 4 dimensional data

(example for 3D data, one variable)

Assume the data are on a 3 (column) by 2 (row) by n (plane) grid. Need first to define a grid conforming to the data.

Note that FERRET version 2.00 insists on a particular sequencing of data. The data must be sequenced such that successive lines of X data form planes of XY, blocks of XYZ and, hyper-blocks of XYZT. Any of the axes may be omitted but FERRET requires the axes that are present to maintain the order X-Y-Z-T.

Fragment of the data file, SNOOPY.DAT - variable WIDTH

```
     2.3  2.4
     3.1  3.2
     4.5  1.7
     5.6  5.3
     2.3  2.4
     3.1  3.2
     4.5  4.7
     5.6  3.3
     .     .
     .     .

    yes? DEFINE AXIS/X=1:3:1 XSNOOPY
    yes? DEFINE AXIS/Y=1:2:1 YSNOOPY
    yes? DEFINE AXIS/Z=1:12:1 ZSNOOPY
    yes? DEFINE GRID/X=XSNOOPY/Y=YSNOOPY/Z=ZSNOOPY   GSNOOPY
    yes? FILE/VAR=WIDTH/GRID=GSNOOPY/COLUMNS=2   SNOOPY.DAT
    yes? CONTOUR WIDTH[Z=1:2@AVE]
```

6. Regions

The region in space and time where expressions are evaluated may be specified in 3 different ways:

i) via the command SET REGION;
ii) as qualifiers to the command name;
iii) as qualifiers to variable names.

If SET REGION is used FERRET will remember the region as the default context for future commands.

Regions may be defined, modified and restored using DEFINE REGION, SET REGION and region @ notation. For more information refer to these topics.

Region information is normally specified in the following form:

```
QUAL=lo_val:hi_val@transform (e.g.Y=20S:10N@AVE)
```

See also SET MODE INTERPOLATE.

Examples: Regions

Examples of valid region specifiers

i) `yes? SET REGION/X=140E:160W/Y=10S:20N/K=1/T=27739`
   - fully specify the region in an XY plane

ii) `yes? CONTOUR QADZ`
   - contour vertical heat advection within whatever region is the current default

iii) `yes? CONTOUR/@T/Z=10/L=25 TEMP`
   - contour the first time step of temperature data at Z=10 deep within region T, the Tropical Pacific

iv) `yes? SET REGION/DY=-5:+5`
- widen the current default region by 5 degrees north and south

v) `yes? LIST/I=80:120:10/J=30:60:15 U`
- list zonal velocity at every 10th point in longitude and every 15th point in latitude over the indicated range of subscripts. Depth and time must be inferred from the current default context.

vi) `yes? LIST V[Z=0:50@AVE]`
- list meridional currents calculated by averaging values between the surface and a depth of 50 m.

vii) `yes? LIST/Z=10 V - V[Z=0:100@AVE]`
- modify the default region by specifying a depth of 10 m. Then list values of meridional current with the mean value between the surface and 100m removed.

6.1 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".

For example:
```
    yes? CONTOUR TEMP[Y=15S:10N]
    yes? LIST/J=50 U
```

6.2 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.

For example:
```
    yes? CONTOUR TEMP[Y=160E:140W]
    yes? LIST/I=100 U
```

See also Regions Modulo for help with globe-encircling axes.

6.3 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.

For example:
```
     yes? CONTOUR TEMP[Z=0:100]
     yes? LIST/K=3 U
```

6.4 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 time 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 will be interpreted as hours always. CALENDAR DATES IN THE YEARS 0000 AND 0001 WILL BE REGARDED AS YEAR-INDEPENDENT DATES (suitable for climatological data).

Time-step values cover the range shown by SHOW GRID/T. (Units may vary between data sets.)

For example:
```
     yes? LIST/T="1-JAN-1982:13:50":"15-FEB-1982"   DENSITY
     yes? CONTOUR TEMP[T=27740:30000]
     yes? LIST/L=90 U
```

See also Regions Modulo for help with climatological axes.

6.5 Delta

The notation q=lo:hi:delta (e.g. Y=20S:20N:5) can be used to specify that the data in the specified range is to be regularly subsampled at interval delta.

In version 2.20 of FERRET this notation is valid only for the REPEAT and SHOW GRID commands.

6.6 @ notation

Regions may be named and referred to using the syntax "@name". Some regions commonly used by TMAP are predefined. See SET REGION and DEFINE REGION for further information.

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

For example:

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

will set the current default region to "BOX1" and then replace Z in the current default with a depth of 15 meters.

Examples: @ notation

```
yes? CONTOUR/@W
```
- 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 FERRET's current context.)

```
yes? SET REGION/@T
```
- set the default region to "T", the Tropical Pacific Ocean (latitude 23.5S to 23.5N)

6.6.1 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 definitions can be deleted with CANCEL REGION or redefined with DEFINE REGION.

6.7 Modulo

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 before and after the nominal length of the axis.

The most common uses of modulo axes are:

i) 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.

ii) 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.

7. Variables

Variables are of 3 kinds:

1) file variables	(read from disk files)
2) user-defined variables	(user defined with LET command)
3) diagnostic variables	(for TMAP model runs)

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

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

In addition, the pseudo-variables I, J, K, L, X, Y, Z, T and others may be used to refer to the underlying grid locations and to create abstract variables. See also EXPRESSIONS PSEUDO.

7.1 Abstract variables

FERRET can be used to manipulate abstract mathematical quantities such as SIN(x) or EXP(-k*t).

This is done by creating expressions exclusively from pseudo-variables.

Examples: Abstract variables

Contour the function
```
     COS(a*Y)/EXP(b*T)  where a=0.25 and b=-0.02
```
over the range
```
     Y=1:45 (degrees) and  T=1:100 (hours)
```
with a resolution of

1 degree on the Y axis and 2 hours on the T axis.

Quick and dirty solution:
```
     yes? CONTOUR/Y=1:45/T=1:50 COS(0.25*Y)/EXP(-.02*T)
```

Nicer: (plot will be documented with correct units and titles)
```
     yes? DEFINE AXIS/Y=-1:45:1/UNIT=DEGREES yax
     yes? DEFINE AXIS/T=1:100:2/UNIT=HOURS tax
     yes? DEFINE GRID/T=tax/Y=yax my_grid
     yes? SET GRID my_grid
     yes? LET a=0.25
     yes? LET b=-0.01
     yes? CONTOUR/Y=-1:45/T=1:100 COS(a*Y)/EXP(b*T)
```

7.2 User defined variables

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

See also DEFINE VARIABLE and LET

8. Contexts

Information describing a region in space/time, a data set and a grid is collectively referred to as the "context". Program FERRET maintains a default for each of these quantities in the "current context" (sometimes referred to as the "default 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 within a single command with

o the /D=data_set qualifier
o the /X,/Y,/Z,/T,/I,/J,/K and /L qualifiers

The same qualifiers may also modify single variables changing the context only of that variable.

Examples: Contexts

i) Set up the current region
```
yes? SET REGION/Z=0/Y=0/T="1-JAN-1982":"31-DEC-1983"
```

ii) Modify the current region for a single command, only
```
yes? PLOT/Y=10N U
```

iii) Modify the current region for a single variable, only
```
yes? PLOT U-U[Z=0:100@AVE]
```

iv) Set the current context and contour a field:
```
yes? SET DATA GTAA011
yes? SET REGION/Y=10S:10N/X=130E:70W/Z=50/L=25
yes? CONTOUR SALT
```

9. Expressions

A command that requires data may use any valid algebraic expression involving constants, operators, functions, pseudo-variables (e.g. X,TBOX,...), and other variables. The variables may be transformed along one or more axes. The expressions may optionally be in Reverse Polish ordering (See also SET MODE POLISH).

A variable name may optionally be followed by square brackets containing region, transformations, 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 vertical temperature deviation

iii) `COS(Y)`
- the cosine of the Y coordinate of the underlying grid

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

9.1 Operators

Valid operators are

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

9.2 Functions

Valid functions are

NAME	#ARGS	
MAX	2	- computes point by point maximum of two fields
MIN	2	- computes point by point minimum of two fields
INT	1	- truncates values to integers
ABS	1	- absolute value
EXP	1	- exponential
LN	1	- natural logarithm
LOG	1	- common logarithm
SIN	1	- trigonometric sine
COS	1	- trigonometric cosine
TAN	1	- trigonometric tangent
ASIN	1	- trigonometric arcsine (-PI/2,PI/2)
ACOS	1	- trigonometric arccosine (0,PI)
ATAN	1	- trigonometric arctangent (-PI/2,PI/2)
ATAN2	2	- 2-argument arctangent: (discontinuous at -PI)
MOD	2	- modulo operation (arg1 - arg2*[arg1/arg2])
MISSING	2	- replaces missing values in arg1 with arg2
IGNORE0	1	- replaces zeros with missing values
RANDU	1	- generate a grid of uniform [0,1] random values
RANDN	1	- generate a grid of normal random values

Examples: Functions

i) ABS(U)
- takes the absolute value of U for all points within the current region

ii) `MAX( TEMP[K=1], TEMP[K=2] )`
- takes the maximum temperature comparing the first 2 depth levels

iii) `MISSING( TEMP, -999 )`
- replaces missing value flags in TEMP with -999

iv) `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]". The first valid value in the field is used as the random number seed. Values that are flagged as bad will remain flagged as bad in the random number field.

9.3 Pseudo-variables

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

Pseudo-variables depend only on the underlying grid. The grid will be determined by the hierarchy
1) explicit grid or variable name (e.g. Z[G=TEMP])
2) implicit grid by association (e.g. TEMP/Z)
3) the grid specified by SET GRID (e.g. I+J)

Examples: Pseudo-variables

i) yes? `LIST/I=1:10 I`

ii) yes? `PLOT/I=1:100 SIN(X)`

iii) yes? `CONTOUR/Y=20S:20N/Z=0:20 COS(Y[G=U])*EXP(-.2*Z[G=U])`

9.4 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 along more than one axis are specified on a single variable or expression the order of execution is X axis, first, then Y then Z then T.

Example syntax: `TEMP[Z=0:100@LOC:20]` (depth of 20 degrees)

Valid transformations are

	DEFAULT	
TRANSFORM	ARGUMENT	DESCRIPTION
@DIN		definite integral
@IIN		indefinite integral
@AVE		average
@LOC	0	coordinate of (e.g. depth of 20 degrees)
@MIN		minimum
@MAX		maximum
@SHF	1 pt	shift
@SBX	3 pt	boxcar smoothed
@SBN	3 pt	binomial smoothed
@SHN	3 pt	Hanning wmoothed
@SPZ	3 pt	Parzen smoothed
@SWL	3 pt	Welch smoothed
@FAV	3 pt	fill missing values w/ ave
@DDC		centered derivative
@DDF		forward derivative
@DDB		backward derivative
@ITP *		interpolated (see SET MODE INTERPOLATE)
@AAV *		XY area averaged with COS(lat) factors
@AIN *		XY area integrated with COS(lat)

* for internal use by FERRET, only

Examples: Transformations

The following are valid 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 point filter
`TAU[L=1:25@DDC]`	- centered derivative of TAU over 25 points
`V[L=@IIN]`	- indefinite (accumulated) integral of V
`QFLUX[X=@AVE,Y=@AVE]`	- XY area averaged QFLUX

9.4.1 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 is flagged as invalid will be excluded from calculations.

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

If the underlying axis units are unknown FERRET will use those unknown units for the grid box lengths. (If FERRET does not recognize the units of an axis it will display a message to that effect when the DEFINE AXIS or SET DATA command defines the axis.)

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 will be 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 will be used - including the first and last subscript given. The region information that is listed with the output will reflect this.

9.4.2 General Information About Smoothing Transformations

FERRET provides several transformations useful for smoothing variables (removing high frequency variability). These transformations replace each value on the grid to which they are applied with a weighted average of the surrounding data values along the axis specified. For example, the expression U[T=@SPZ:3] replaces the value at each (i,j,k,l) grid point of the variable U with the weighted average

```
U at t = 0.25*(U at t-1) + 0.5*(U at t) + 0.25*(U at t+1)
```

The various choices of smoothing transformations (@SBX, @SBN, @SPZ, @SHN, @SWL) represent different shapes of weighting functions or "windows" with which the original variable is convolved. New window functions can be obtained by nesting the simple ones provided. For example, using the definitions

```
LET UBOX = U[L=@SBX:15]
LET UTAPER = UBOX[L=@SHN:7]
```

will produce a 21 point window whose shape is a boxcar (constant weight) with COSINE (Hanning) tapers at each end.

FERRET may be used to directly examine the shape of any smoothing window. Mathematically, the shape of the smoothing window can be recovered as a variable by convolving it with a delta function. In the example below we examine (PLOT) the shape of a 15 point Welch window.

```
GO UNIT_SQUARE                    ! define X axis as [-1,1] bo 0.2
SET REGION/X=-1:1
LET DELTA = IF X EQ 0 THEN 1 ELSE 0
PLOT DELTA[I=@SWL:15]             ! convolve DELTA with Welch window
```

9.4.3 Transformation @DIN

The transformation @DIN computes the definite integral - a single value that is the integral between two points along an axis (compare with @IIN). It is obtained as the 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.

If @DIN is specified simultaneously on both an X (longitude) and a Y (latitude) axis 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]`

See also the General Information section under transformations.

9.4.4 Transformation @IIN

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]`

See also the General Information section under transformations.

9.4.5 Transformation @AVE

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

If @AVE is specified simultaneously on both an X (longitude) and a Y (latitude) axis 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]`

See also the General Information section under transformations.

9.4.6 Transformation @LOC

The transformation @LOC requires 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 - 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]
```

will find 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 checking to Z=200. Thus,

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

will produce a map of the depth of the 18 degree isotherm.

See also the General Information section under transformations.

9.4.7 Transformation @MIN

The transformation @MIN finds the minimum value of the variable within the specified axis range. Thus, for fixed Z and Y

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

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

9.4.8 Transformation @MAX

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

See also @MIN.

9.4.9 Transformation @SHF

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

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

will associate the value of U[L=3] with the subscript L=1. and the expression

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

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

9.4.10 Transformation @SBX

The transformation @SBX applies a boxcar window 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 an equal number of points, each the average of its surrounding points. Regridding can be used to reduce the number of points.

9.4.11 Transformation @SBN

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 an equal number of points, each 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.

9.4.12 Transformation @SHN

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 will be equivalent to a 7 point Hanning window which has zeros as it first and last weights).

9.4.13 Transformation @SPZ

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.

9.4.14 Transformation @SWL

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.

9.4.15 Transformation @FAV

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]`

9.4.16 Transformation @DDC

The transformation @DDC computes the derivative with respect to the indicated axis. A centered differencing scheme is used. The units of the underlying axis are treated as they are with integrations. (see General information on transformations)

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

9.4.17 Transformation @DDF

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. (see General information on transformations)

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

9.4.18 Transformation @DDB

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. (see General information on transformations)

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

10. Grids

Every variable has an underlying grid (to define coordinates). 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)

The grid for a variable can be examined via

```
   yes? SHOW GRID variable_name_or_grid_name
```

See also SHOW GRID.

Data can be regridded by the G= modifier. (see Grids Regridding)

Axes and grids can be created by DEFINE AXIS and DEFINE GRID. See also DEFINE.

10.1 Regridding

Syntax:

`VAR[G=name]` for linear interpolation to new grid

or

`VAR[G=name:X@trn;Y@trn;...]` for special transformations (version 2.1, 2.2 only)

where

VAR	is the name of the variable (e.g. TEMP, U, TAU, ...)
name	is either the name of a variable (e.g.TEMP[G=U]) or the name of a grid (e.g.TEMP[G=GU01])
trn	is the code for a special transformation (e.g. @AVE)

Regridding is essential for algebraic operations that combine variables on incompatible grids. For example, if U and TEMP are on staggered grids

```
yes? CONTOUR U * TEMP[G=U]
```

will regrid TEMP onto the U grid before performing the multiplication.

FERRET provides the commands DEFINE AXIS and DEFINE GRID to assist with the creation of arbitrary grids. These commands provide detailed control over regridding. For example, suppose that two variables, V1 and V2, are defined on distinct 4-dimensional grids (X,Y,Z, and T axes). Suppose further that the T axes of the two grids are identical but that 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 simply define a new grid (say, named "NEW_GRID") that has the X and Y axes of V2 but is otherwise like the grid of variable V1.

```
yes? DEFINE GRID/LIKE=V1/X=V2/Y=V2 NEW_GRID
```

Then request the regridding using the syntax V1[G=NEW_GRID]. For example

```
yes? LIST/X=160E:140W/Y=5S:5N V1[G=NEW_GRID].
```

(Note that LIST/I=1:10 V1[G=NEW_GRID] will list V1 at the first 10 X points of the grid NEW_GRID.)

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. For example, suppose the variable WIND is the wind speed at the ocean surface and the variable CURR is the magnitude of ocean currents with grid points at several depths. Then the expressions

```
CURR[G=WIND]   or   WIND[G=CURR]
```

will generate the error message

"ERROR regridding: grids have incompatible Z axes"

The command DEFINE GRID can be used to make the regridding unambiguous and acceptable to FERRET. To obtain WIND regridded to the X,Y and T locations of CURR define a new grid (say, named "G2") using

```
yes? DEFINE GRID/LIKE=CURR/Z=WIND    G2
```

and refer to WIND[G=G2]. For example, in this command we list the surface currents and winds at the same latitude/longitude grid points

```
yes? LIST/I=1:10/J=1:10 WIND[G=G2],CURR[K=1]
```

10.1.1. Regridding Transformations

As of FERRET 2.20 the only special transformation supported is area averaging - computing the area-weighted average of all points on the source grid that lie partly or completely within each grid box of the destination grid. The AVE transformation must be specified on both the X and Y axes to achieve this result.

Example:
Let grid G1 possess points spaced regularly at 1 degree intervals in both X and Y.
Let grid G10 possess points spaced regularly at 10 degree intervals in both X and Y.
Let variable TEMP be defined on G1.
Then
```
TEMP[G=G10:X@AVE;Y@AVE]
```
will produce 10x10 degree box averages of TEMP.

Future versions of FERRET will support regridding by other methods.

11. Graphics

A general overview of graphic output options from program FERRET:

Command:	
CONTOUR	- produce a contour plot of one or more fields
PLOT	- produce a line plot of one or more arrays
VECTOR	- produce a vector arrow plot
SHADE	- produce a shaded representation
PPLUS	- execute PPLUS command(s) directly
SET WINDOW	- manipulate VAXstation graphics windows
SET VIEWPORT	- place graphics output into a sub-window

11.1 Plot_labels

By default the plot title will be in ASCII COMPLEX font and all other labels will be in ASCII SIMPLEX font. See SET MODE ASCII_FONT for further information.

The text strings of all the labels will also be stored in PPLUS symbols. The symbol names are

LABTIT	- title label string
LABX	- X axis label
LABY	- Y axis label
LABn	- nth movable label

These symbols together with the PPLUS "Special Functions", $EDIT, $EXTRACT,$LENGTH and $LOCATE, can be used to produce customized FERRET plot labels.

The /NOLABELS qualifier can be used with PLOT,CONTOUR,SHADE or VECTOR to inhibit labels.

The formats of labels describing the position of a plot (e.g. depth and time of an XY plane plot) may be adjusted with the SET MODE command using arguments CALENDAR, LATIT_LABEL, LONG_LABEL, and DEPTH_LABEL

11.2 Viewports

A viewport is a sub-rectangle of a full window. Viewports can be used to put multiple plots onto a single window.

A number of the most commonly desired viewports are pre-defined.

The command SET VIEWPORT will direct future graphics to a viewport.

The commands DEFINE VIEWPORT and CANCEL VIEWPORT can be used to create custom viewports.

The command SHOW VIEWPORT can be used to examine the viewports

Examples: Graphics Viewports

Put TEMPERATURE, U, V and W contour plots into the four quadrants of a single window:

```
yes? SET VIEWPORT LL
yes? CONTOUR TEMP
yes? SET VIEWPORT LR
yes? CONTOUR U
yes? SET VIEWPORT UL
yes? CONTOUR V
yes? SET VIEWPORT UR
yes? CONTOUR W
```

11.2.1 Pre-defined viewports

NAME	RESULT
FULL	output to full screen (default)
LL	output to lower left quadrant of window
LR	output to lower right quadrant of window
UR	output to upper right quadrant of window
UL	output to upper left quadrant of window
RIGHT	output to right half of window
LEFT	output to left half of window
UPPER	output to upper half of window
LOWER	output to lower half of window

11.2.2 Advanced usage of viewports

All viewport commands refer to positions relative to the current aspect ratio of the window. Thus,

```
yes? DEFINE VIEWPORT/ORIGIN=0.5,0.5  V5
```

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

The PPLUS commands SIZE and AXLEN are not modified by FERRET except when the commands SET WINDOW/ASPECT and PPLUS/RESET are used. Thus a user is free to reshape and resize the plotting surface at will. FERRET will locate the plot labels around the axes such that the overall plot exceeds the x axis length by 2.2 (PPLUS inches) and exceeds the y axis length by 2.8.

To create a plot with multiple viewports follow these steps:

1) Determine the overall size of the desired window (or page, if hard copy). Use the command `yes? PPLUS SIZE xlen,ylen` to set these dimensions.
   Note: If the dimensions specified exceed the size of the output device (the screen on a workstation or the paper size on a hardcopy device) the plot will be reduced to the maximum size that will fit.
2) Determine the size and location of each viewport within the overall window. Use the command `yes? DEFINE VIEWPORT` ... to define these regions - the /ORIGIN qualifier to locate the lower left corner and the /CLIP qualifier to locate the upper right corner.
3) Determine the axis lengths to use within each viewport. FERRET's default labels require the viewport size to exceed the axis lengths by at least 2.2 PPLUS inches in X and by at least 2.8 in Y. To customize the labels use the PPLUS commands LABS and ORIGIN.
4) Produce graphics in each viewport in turn by directing output to the viewport with the command `yes? SET VIEWPORT` ... .

Note: FERRET can assist in determining axis lengths and sizes if the desired aspect ratio of the figure is known. Use the command `yes? SET WINDOW/ASPECT` ... specifying the desired aspect ratio. Then use the command `yes? PPLUS LIST`

PLOT to find the overall size for the figure (look at the values "WIDTH" and "HEIGHT"). Use `yes? PPLUS LIST XAXIS` to find the x axis length and similarly for the y axis.

See examples for further information.

Examples: Advanced usage of viewports

i) Plot U and V in a single window occupying the lower and upper halves of the window, respectively. Use an X axis length of 8, as usual, but a Y axis length of 4 instead of the default 6. Allow the default labelling margins of 2.2 inches for the X axis and 2.8 inches for each plot along the Y axis.

```
yes? PPL AXLEN 8,4
yes? PPL SIZE 10.2,13.6
yes? SET VIEWPORT LOWER
yes? PLOT U
yes? SET VIEWPORT UPPER
yes? PLOT V
```

ii) Plot U and V and W in a single window as three plots stacked vertically. Use an aspect ratio which doubles the X axis length relative to the default. (The results of the PPL LIST PLOT command below are WIDTH=13.51 HEIGHT=7.04 .)

```
yes? SET WINDOW/ASPECT=0.375:AXIS                    ! .5 * (6/8)
yes? PPL LIST PLOT
yes? PPL SIZE 13.51,21.12
yes? DEFINE VIEWPORT/ORIGIN=0,0/CLIP=1.0,0.33 T1
yes? SET VIEWPORT T1
yes? PLOT U
yes? DEFINE VIEWPORT/ORIGIN=0,0.33/CLIP=1.0,0.67 T2
yes? SET VIEWPORT T2
yes? PLOT V
yes? DEFINE VIEWPORT/ORIGIN=0,0.67/CLIP=1.0,1.0 T3
yes? SET VIEWPORT T3
yes? PLOT W
```

12. Memory use

If memory is a problem running FERRET the following suggestions may help:

1) 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.

2) Use CANCEL MODE SEGMENTS to minimize the memory usage by graphics (X windows will then not be restored after they are obscured as of Feb., 1990 DECwindows)

3) When using DEFINE VARIABLE (LET) avoid embedding upper and lower axis bounds within the variable definition. FERRET cannot split up large calculations along axes with fixed limits. 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
```

4) 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 be less efficient 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.

Chapter 2: COMMANDS

13. CANCEL

Cancel a program state - generally paired with a SET command.

Refer to SET command for further information.

13.1 CANCEL DATA_SET

Remove the specified data set from the list of available sets

Format:
```
  yes? CANCEL DATA_SET dset1, dset2, ..., dsetn
     where each dset may be the name or number of a data set
        or
  yes? CANCEL DATA_SET/ALL
```

(See also SET DATA_SET and SHOW DATA SET)

Command qualifiers for: CANCEL DATA_SET

/ALL
CANCEL DATA/ALL

Eliminate ALL data sets.

13.2 CANCEL EXPRESSION

Un-specify the current expression.

Format:
```
  yes? CANCEL EXPRESSION -- un-specify current context expression
```

13.3 CANCEL LIST

CANCEL LIST[/qualifiers]

Toggle the effects of the SET LIST command.

See detailed documentation under SET LIST

Command qualifiers for: CANCEL LIST

/ALL
CANCEL LIST/ALL

Restore all aspects of the LIST command to their default behavior.

/PRECISION
CANCEL LIST/PRECISION

Reset the precision of listed data to 4 significant digits

/FORMAT
CANCEL LIST/FORMAT

Reset the listed output to its default formatting

/FILE
CANCEL LIST/FILE

Reset the listed output to automatic file naming

/APPEND
CANCEL LIST/APPEND

Reset the listed output to NOT append to existing file

/HEAD
CANCEL LIST/HEAD

Instruct listed output to omit the descriptive data header.

13.4 CANCEL MEMORY

Erase data currently stored in memory.

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

Use this command to save memory space - by erasing unneeded data as soon as it is no longer needed virtual memory requirements can be reduced. This is especially useful for efficient batch processing.

Example: CANCEL MEMORY

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/SET_UP var1 , var2 ! set up basic contour plot
PPL ...                    ! customize plot
PPL CONTOUR                ! draw it
FRAME                      ! save animation frame
CANCEL MEMORY/ALL          ! clear memory for next time step
EXIT
```

Command qualifiers for: CANCEL MEMORY

/TEMPORARY
CANCEL MEMORY/TEMPORARY (default)

Erase all non-permanent variables stored in memory.

/PERMANENT
CANCEL MEMORY/PERMANENT

Erase all "permanent" variables stored in memory.
(i.e. variables loaded into memory with LOAD/PERMANENT)

/ALL
CANCEL MEMORY/ALL

Erase all variables stored in memory.

13.5 CANCEL MODE

Set the state of a mode to "cancelled".

Format:
```
yes? CANCEL MODE mode_name
```

(See also SET MODE)

13.6 CANCEL MOVIE

Terminate the capture of graphic images into a movie file.

Format:
yes? CANCEL MOVIE

13.7 CANCEL REGION

Un-define part or all of the current or named region

Format:
`CANCEL REGION[/qualifier] [region_name]`

Examples:
`yes? CANCEL REGION/T` - eliminate T from the current context

`yes? CANCEL REGION reg1` - eliminate the region named "reg1"

Command qualifiers for: CANCEL REGION

/ALL
CANCEL REGION/ALL

Eliminate ALL stored region information

/X
CANCEL REGION/X [region_name]

Eliminate X axis information from current context or named region

/Y /Z /T /I /J /K /L
... see CANCEL REGION/X

13.8 CANCEL VARIABLE

Format:
`CANCEL VARIABLE[/qualifier]`

Delete a user-defined variable definition

Command qualifiers for: CANCEL VARIABLE

/ALL
CANCEL VARIABLE/ALL

Delete all user-defined variable definitions

13.9 CANCEL VIEWPORT

Un-define a viewport or cancel use of viewports.

Format:
`yes? CANCEL VIEWPORT view_name` -- un-define view_name

`yes? CANCEL VIEWPORT` -- equivalent to SET VIEWPORT FULL

13.10 CANCEL WINDOW

Erase a window(s) from the VAXstation screen

Format:
```
yes? CANCEL WINDOW   n

yes? CANCEL WINDOW/ALL
```

Command qualifiers for: CANCEL WINDOW

/ALL
CANCEL WINDOWS/ALL

Erase all graphics windows

14. CONTOUR

Produce a contour plot.

Format:
```
CONTOUR[/qualifiers]   expression
```

Example:
`yes? CONTOUR U`
 - produce a contour plot of the variable U

Parameters

Expressions may be anything described under the Expressions section in this help manual.

The expression will be inferred from the current context if omitted from the command line.

Command qualifiers for: CONTOUR

/SET_UP
CONTOUR/SET_UP

CONTOUR/SET_UP performs all the internal preparations required by program FERRET for contouring but does not actually produce output. The command PPL can then be used to make changes to the plot prior to producing output with the PPLUS CONTOUR command. This makes possible custom colors, labels, contour levels, etc.

/TRANSPOSE
CONTOUR/TRANSPOSE

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

/OVERLAY
CONTOUR/OVERLAY

Causes the indicated expression to be overlaid on the existing plot.

/FRAME
CONTOUR/FRAME

Causes the graphic image produced by the command to be captured as an animation frame on the device specified by SET MOVIE

/NOLABELS
CONTOUR/NOLABELS

Inhibits all plot labels.

/LEVELS
CONTOUR/LEVELS

Specify the contour levels or how the levels will be determined.

If the /LEVELS qualifier is omitted FERRET will automatically select reasonable contour levels.

To reuse the levels from the last CONTOUR or SHADE plot use CONTOUR/LEVELS.

To specify the contour levels manually use the syntax
```
   CONTOUR/LEVELS=(lo,hi,delta).
```
For example, we might contour the sea surface temperature using:
```
   yes? CONTOUR/LEVELS=(10,30,5) SST
```
to produce contour levels at 10, 15, 20, 25, and 30.

The arguments on the right hand side of the equal sign can also be used to create unequally spaced contour lines and to control line color, thickness and style using the DARK, PEN, DASH, DEL, and LINE specifiers. For detailed documentation on using these refer to the "LEV" command in the PPLUS manual. If blanks are imbedded within the list of specifiers it will be necessary to enclose the entire list within quotation marks.

/I /J /K /L
CONTOUR/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
CONTOUR/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/D
/D=data-set

See Context and Data_set

15. DEFINE

Define a new region, grid or variable

15.1 DEFINE AXIS

Define an axis (axis name up to 8 characters).

Format:
```
     yes? DEFINE AXIS[/qualifiers]  axis_name
```

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

Command qualifiers for: DEFINE AXIS

/FILE
DEFINE AXIS/FILE=filename

See also DEFINE GRID/FILE - command is identical

/X
DEFINE AXIS/X=lo:hi:delta   axis_name

Specify the limits and point spacing of an X axis to define. The limits may be in longitude format or may be simple numbers. The limits and delta will be regarded as degrees of longitude unless the /UNITS qualifier specifies otherwise.

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

/Y
DEFINE AXIS/Y=lo:hi:delta   axis_name

Specify the limits and point spacing of a Y axis to define. The limits may be in latitude format or may be simple numbers. The limits and delta will be regarded as degrees of latitude unless the /UNITS qualifier specifies otherwise.

Example:
```
     yes? DEFINE AXIS/Y=15S:25N:.5 AXYNEW
```

/Z
DEFINE AXIS/Z=lo:hi:delta   axis_name

Specify the limits and point spacing of a Z axis to define. The limits and delta will be regarded as meters unless the /UNITS qualifier specifies otherwise.

Example:
```
     yes? DEFINE AXIS/Z=0:5000:20/UNITS=CM AXZCM
```

/T
DEFINE AXIS/T=lo:hi:delta   axis_name

Specify the limits and point spacing of an T axis to define. The limits may be dates (dd-mmm-yyyy:hh:mm:ss) or may be time steps. The delta increment will be regarded as hours unless the /UNITS qualifier specifies otherwise.

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

Examples:
```
     yes? DEFINE AXIS/T="7-NOV-1953":"23-AUG-1988:11:00":24  AXTLIFE
     yes? DEFINE AXIS/T=25:125:5/UNITS=minutes   AXT5MIN
```

/UNITS
DEFINE AXIS/UNITS=unit_name/...   axis_name

Specify the units of the axis being defined.

Example:
```
     yes? DEFINE AXIS/Z=0:2000:100/UNITS=CM   ZCM
```

/T0
DEFINE AXIS/T0=start_date/...   axis_name

Specify 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.

/NAME
DEFINE AXIS/FROM_DATA/NAME=axis_name expr

Used only in conjunction with /FROM_DATA to specify the name of the axis to be defined.

/FROM_DATA
DEFINE AXIS/FROM_DATA/NAME=axis_name expr

Used only in conjunction with /NAME to define an axis from any expression that FERRET can evaluate.

(This is a mechanism to convert dependent variables into independent axis data.)
Example:
```
    yes? DEFINE AXIS/FROM_DATA/X/NAME=my_xaxis POS[D=2]^0.5
```

15.2 DEFINE GRID

Define a grid (name may be up to 8 characters)

Format:
```
    yes? DEFINE GRID[/qualifiers]   grid_name
```

Example:
```
    yes? DEFINE GRID/LIKE=TEMP/T=my_t_axis   my_grid
```

Command qualifiers for: DEFINE GRID

/FILE
DEFINE GRID/FILE=filename

Read a grid-file for GRID and AXIS definitions. The grid-file specified should be in the standard TMAP grid-file format. (For examples check files with .GRD extension in directory DISK2:[TMAP.SETS] on the PMEL VAXcluster.)

Example:
```
    yes? DEFINE GRID/FILE=NEW_GRIDS.GRD
```

/LIKE
DEFINE GRID/LIKE=grid_or_variable_name/... grid_name

Specify a particular grid to use as a template to create a new grid. 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
```

/X
DEFINE GRID/X=xaxname/...   grid_name

Specify what particular axis is to be the X axis for this grid.

The name xaxname may be the name of an X axis, the name of a grid which uses the X axis desired or the name of a variable for which the defining grid uses the X axis desired.

For example,
```
     yes? DEFINE GRID/LIKE=TEMP/X=U   GTU
```

will create a grid named GTU which is like the defining grid from the variable TEMP but with the X axis replaced by the X axis from the defining grid of variable U.

/Y /Z /T
... see DEFINE GRID/X

Examples: DEFINE GRID

i)
```
     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

ii)
```
     yes? DEFINE GRID/LIKE=TEMP[D=BA022]/T=SST[D=NMC] GNMC3D
```
- Define grid GNMC3D like TEMP from set BA022 but with the same time axis as set NMC

iii)
```
     yes? DEFINE AXIS/X=140E:140W:.2 XNEW
     yes? DEFINE AXIS/Y=5S:5N:.2 YNEW
     yes? DEFINE AXIS/Z=0:100:5 ZNEW
     yes? DEFINE AXIS/T="15-FEB-1982":"15-FEB-1984":48 TNEW
     yes? DEFINE GRID/X=XNEW/Y=YNEW/Z=ZNEW/T=TNEW GNEW
```
- define grid GNEW from 4 new axes

15.3 DEFINE REGION

Define or redefine a named region_name (up to 4 characters).

Format:
```
     yes? DEFINE REGION[/qualifiers] region_name
```

If the /DEFAULT qualifier is not given only those axes explicitly named will be stored.
If the /DEFAULT qualifier is given all axes will be stored.

Command qualifiers for: DEFINE REGION

/DEFAULT
DEFINE REGION/DEFAULT

Save all axes and transformations, not just those explicitly given.

/I /J /K /L
DEFINE REGION/I=I-limits (similar for J, K, and L)

/X /Y /Z /T
DEFINE REGION/X=X-limits (similar for Y, Z, and T)

/DI /DJ /DK /DL
DEFINE REGION/DI=delta-I-limits (similar for DJ, DK, and DL)

/DX /DY /DZ /DT
DEFINE REGION/DX=delta-X-limits (similar for DY, DZ, and DT)

Examples: DEFINE REGION

i) `yes? DEFINE REGION  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

ii) `yes? DEFINE REGION/X   XONLY`
- stores the current default X axis limits, only, as region XONLY

iii) `yes? DEFINE REGION/DX=-5   XONLY`
- stores the current default X axis limits minus 5 as region XONLY

iv) `yes? DEFINE REGION/Y=20S:20N/Z   YANZ`
- stores the given limits from the Y axis and the default Z axis limits.

v) `yes? DEFINE REGION/DEFAULT/L=5   L5`
- stores the current default region modified by the specification that L, the time step, is stored as 5.

vi) 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.

15.4 DEFINE VARIABLE

Define a user-defined variable from a valid algebraic expression.

Format:
```
     DEFINE VARIABLE[/qualifiers] name = expression
```

Note: "DEFINE VARIABLE" may be abbreviated as "LET"
Example:
```
          LET SPEED = U^2 + V^2
```

Parameters

The expression specified with DEFINE VARIABLE can be any valid expression as described in the Expressions section of this manual.

The name specified with DEFINE VARIABLE can be 1 to 8 characters in length - letters, digits, $ and _ beginning with a letter. Pseudo-variable names are reserved and cannot be used (I,J,...).

It is not advisable to use names that are the same as other file variables.

To enter expressions in Reverse Polish ordering see SET MODE POLISH

Examples: DEFINE VARIABLE

DEFINE VARIABLE

a simple example:
```
     yes? DEFINE VARIABLE sum = a+b
          or equivalently
     yes? LET sum = a+b
```

define velocity in m/sec from position, POS, in cm
```
     yes? DEFINE VARIABLE/TITLE="velocity"/UNIT="m/sec" POS[T=@DDC]*0.01
```

Command qualifiers for: DEFINE VARIABLE

/TITLE
DEFINE VARIABLE/TITLE="string" name = expression

Specify 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 "expression" will be used as the title.

15.5 DEFINE VIEWPORT

Define a new viewport.

Format:
```
     yes? DEFINE VIEWPORT[/qualifiers]   view_name
```

Any qualifiers not explicitly given will default to the values of the FULL viewport.

Examples: DEFINE VIEWPORT

i) `yes? DEFINE VIEWPORT/SIZE=.25/ORIGIN=0,0/CLIP=0.5,0.5 LL`
   - Define a viewport that will place a full screen image into the lower left quarter of the screen and name it "LL"

ii) `yes? DEFINE VIEWPORT/SIZE=.25/CLIP=0.5,0.5 LL`
   - equivalent to i) since 0,0 is the default origin

Command qualifiers for: DEFINE VIEWPORT

/SIZE
DEFINE VIEW_PORT/SIZE=s   view_name

Specify the scaling factor to use relative to the size of the full window. For example, a size factor of .25 is used in the pre-defined viewport LL ("lower left") to fit an entire plot into one fourth of the window without clipping.

If this qualifier is omitted the size factor, s, will be assumed to have the value 1.

Example:
```
     yes? DEFINE VIEW_PORT/SIZE=0.5  HALF
```

/ORIGIN
DEFINE VIEW_PORT/ORIGIN=x,y view_name

Specify the location of the lower left corner of the viewport. The values x,y must be in the range [0,1]. They refer to fractions of the full size of the current window.

If this qualifier is omitted x,y will be assumed to have the values 0,0.

Example:
```
     yes? DEFINE VIEW_PORT/ORIGIN=0.5,0.5  CNTR
```

/CLIP
DEFINE VIEW_PORT/CLIP=x,y   view_name

Specify the location of the upper right corner of the viewport. The values x,y must be in the range [0,1]. They refer to fractions of the full size of the current window.

If this qualifier is omitted x,y will be assumed to have the values 1,1.

Example:
```
    yes? DEFINE VIEW_PORT/CLIP=0.5,0.5  VNEW
```

16. EXIT

When given interactively this command will terminate program FERRET execution and return to control to the next higher VMS level (generally DCL).

When executed within a command file this command will terminate the execution of the command file and return control to the level in FERRET that executed this file (the user or another command file).

A quick interactive EXIT is also achieved with ^Z^Z (double control Z)

Command qualifiers for: EXIT

/COMMAND_FILE
EXIT/COMMAND_FILE

When executed from within a command file EXIT/COMMAND_FILE will force an immediate exit from program FERRET rather than returning to the user or another command file.

17. FILE

The FILE command is an abbreviation for SET DATA/EZ.

All qualifiers and restrictions are identical to SET DATA/EZ

Example:
```
    yes? FILE/VARIABLES="U,V" VELOCITIES.DAT
         is identical in function to
    yes? SET DATA/EZ/VARIABLES="U,V" VELOCITIES.DAT
```

18. FRAME

Save the current graphics display image as a frame in the movie initialized with the command SET MOVIE.

Note: The FRAME command is not implemented in version 2.2 an Ultrix computers. The UNIX tool mtta can be used to animate standard FERRET metafiles.

Format:

```
     yes? FRAME
```

19. GO

Execute a list of commands stored in command file file_name. The commands in the file should be exactly as if given interactively.

Format:

```
     yes? GO file_name
```

If no filename extension is specified a default of .JNL will be assumed. If a directory (with slashes) is specified then the filename must be enclosed in double quotation marks.

The response of program FERRET to errors encountered during execution of the command file is determined by mode IGNORE_ERRORS (see 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. This can be very handy to rerun the exact sequence of commands given at a previous FERRET session.

For example,

If FERRET is running and the current directory contains the current version and numerous past versions of ferret.jnl,

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

then the command

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

is equivalent to

```
     yes? GO ferret.jnl.~99~
```

and will rerun the commands from the last FERRET session.

Note: The command SET MODE IGNORE_ERRORS is also useful when rerunning past sessions which may have errors.

20. HELP

On Unix systems:

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 will then resume the suspended session.

For further information see Chapter 3 of this manual describing On-Line Help on Unix systems.

On VMS systems:

Specific information on all FERRET commands is available via

```
     yes? HELP command_name
```

where command_name is the command with which you would like help.

Help is also available when error messages occur in FERRET. Immediately after the error occurs type

```
     yes? HELP ERROR
```

Program FERRET will attempt to explain the cause of the error.

21. LET

The LET command is an abbreviation for DEFINE VARIABLE.

All qualifiers and restrictions are identical to DEFINE VARIABLE

Example:

```
     yes? LET A = B
          is identical in function to
     yes? DEFINE VARIABLE A = B
```

22. LIST

Produce a listing of the indicated data

Format:

```
     LIST[/qualifiers]   expression_1 , expression_2 , ...
```

Note that "delta" values are not supported in FERRET v2.00

Further information under SET LIST

Example: LIST

i)
```
yes? LIST/Z=10 u , v , u^2 + v^2
```
- list the 3 quantities specified using the current default context and region

Parameters

Expressions may be anything described under Expressions.

If multiple variables or expressions are specified they may be listed together or in sequence depending on the /SINGLY qualifier.

The expression(s) will be inferred from the current context if omitted from the command line.

Command qualifiers for: LIST

/FILE
LIST/FILE[=file_name]

Name a file to receive the listed data. If /FILE is specified with no name then the default name used will be from the SET LIST/FILE command.

Example:
```
    yes? LIST/FILE=MY_FILE.DAT SALINITY[D=TOGA]
```

Further information on automatic filename generation under SET LIST.

/APPEND
LIST/APPEND

Use this qualifier together with the /FILE qualifier to indicate that the listed data should be appended to a previously existing file. If no file exists by the name indicated a new file will be created.

This qualifier will over-ride the command of CANCEL LIST/APPEND.

/HEAD
LIST/HEAD

Precede listing with a heading describing data set, variable and region. This qualifier will over-ride the CANCEL LIST/HEAD command.

/NOHEAD
LIST/NOHEAD

Do not precede listing with a heading. This qualifier will over-ride the SET LIST/HEAD command.

/ORDER
LIST/ORDER=permutation

Specify the order in which axes are to be laid out on the listing.

Example:
```
     yes? LIST/ORDER=XY SST   - lists X across the page, Y down

     yes? LIST/ORDER=YX SST   - lists Y across the page, X down
```

The "permutation" string may be any permutation of the letters X,Y,Z and T.

/SINGLY
LIST/SINGLY expression_1, expression_2, ...
When the /SINGLY qualifier is specified the entire listing of each expression including (optional) heading and all data will be completed before proceeding to the next expression.

By default the expressions will not be listed singly - each line will contain one value each expression.

The qualifier has no effect if only a single expression is specified.

If the /FILE qualifier is specified without /APPEND then each expression will be listed to a separate file.

/FORMAT
LIST/FORMAT=format_choice

Specify an output format for the data to be LISTed. (When a FORTRAN format is specified the row and column headings are omitted from the output.)

This command has the same function as SET LIST/FORMAT except that it does not effect future LIST commands.

See detailed documentation under SET LIST/FORMAT.

/EPKEY
LIST/FILE/FORMAT=EPIC/EPKEY=key#
Specify an EPIC variable key number (for EPIC output, only)
(Not functional on Unix systems)

/I /J /K /L
LIST/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
LIST/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/D
/D=data-set

See Context and Data_set

23. LOAD

Load a variable or expression into memory.

Format:
```
    LOAD[/qualifiers]   expression_1 , expression_2 , ...
```

Loading speeds 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 the commands CONTOUR, PLOT, SHADE, VECTOR and LIST do. See also Context.

Parameters

Expressions may be anything described under Expressions.

If multiple variables or expressions are specified they will be treated in sequence.

The expression(s) will be inferred from the current context if omitted from the command line.

Command qualifiers for: LOAD

/PERMANENT
LOAD/PERMANENT

Data loaded with LOAD/PERMANENT will be kept in memory until a LOAD/TEMPORARY command is given that refers to the same data. See LOAD/TEMPORARY

/TEMPORARY
LOAD/TEMPORARY (default)

Data loaded with LOAD or LOAD/TEMPORARY will be brought into memory but may be deleted based on a priority scheme of least recent use when memory space is required.

/I /J /K /L
LOAD/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
LOAD/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/D
/D=data-set

See Context and Data_set

24. MESSAGE

Display a message at the terminal

Format:
```
     yes? MESSAGE text
```

By default a carriage return will be required from the terminal for program execution to continue (used to halt the execution of a command file).

Command qualifiers for: MESSAGE

/CONTINUE
MESSAGE/CONTINUE

Continue program execution following the display of the message text without waiting for a carriage return from the operator.

/QUIET
MESSAGE/QUIET

Wait for a carriage return as input from the operator but do not supply a prompt for it.

25. PLOT

Produce a line plot.

Format:
```
     PLOT[/qualifiers]   expression_1 , expression_2 , ...
```

The indicated expression(s) must represent a line (vs a plane) of data. Unless the /VS qualifier is used the independent variable will be the underlying coordinate axis for this line of data.

Example:
```
     yes? PLOT/l=1:100 SST
```
- produce a time series plot of the first 100 points of SST

Parameters

Specifies 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 will be associated in the order of their underlying axes.

When the /VS qualifier is not used the indicated expression(s) must describe a line (vs. say, a plane) of data.

The expression(s) will be inferred from the current context if omitted from the command line.

(See Variables and Expressions)

Command qualifiers for: PLOT

/SET_UP
PLOT/SET_UP
Performs all the internal preparations required by program FERRET for plotting but does not actually produce output. The command PPL can then be used to make changes to the plot prior to producing output with the PPLUS PLOT command. This makes possible custom colors, labels, contour levels, etc.

/VS
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
```
- produce a plot of salinity against temperature along the indicated range of latitudes and times.

/LINE
PLOT/LINE[=line_style]

The /LINE qualifier causes the PLOT command to connect the plotted points with a line regardless of the state of the /SYMBOLS command

Optionally, the line style may be specified as an integer 1...6. Line style "1" is always a solid line. Other line styles are device dependent (colors or dash patterns).

/SYMBOL
PLOT/SYMBOL[=symbol_number]

The /SYMBOL qualifier causes the PLOT command to mark each plotted point with a symbol. If the /LINE qualifier is given, too, the symbols will also be connected with a line; if /LINE is omitted no connecting line will be 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.)

/TRANSPOSE
PLOT/TRANSPOSE

Causes the default horizontal and vertical axes to be interchanged.

/OVERLAY
PLOT/OVERLAY

Causes the indicated field(s) to be overlaid on the existing plot.

/FRAME
PLOT/FRAME

Causes the graphic image produced by the command to be captured as an animation frame on the device specified by SET MOVIE

/NOLABELS
PLOT/NOLABELS

Produce a plot lacking all informative labels.

/I /J /K /L
PLOT/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
PLOT/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/D
/D=data-set

See Context and Data_set

26. PPLUS

Invoke PPLUS (written by Don Denbo) to execute a command or commands

Format:

PPLUS	- invoke PPLUS interactively
or	
PPLUS pplus_command	- execute a single PPLUS command
or	
PPLUS/RESET	- restore PPLUS to start-up defaults

Example:

```
yes? PPLUS LEV () (1,100,5)
```
- execute the PPLUS LEV command and immediately return control to program FERRET.

When PPLUS has been invoked interactively it will be made apparent by a prompt of "PPL>" instead of the usual "yes?". The EXIT command given at the "PPL>" prompt will return control to program FERRET.

More information on FERRET/PPLUS interactions is available under Graphics.

27. REPEAT

Repeat a command over a range of values along an axis.

Format:

```
yes? REPEAT/q=lo:hi:delta command
```

The units of lo,hi and delta will be the units of the underlying grid axis if the qualifier is X,Y,Z or T. The units will be subscripts if qualifier is I,J,K or L. Use SHOW GRID to examine the axis units (if the units are not displayed try CANCEL MODE LATITUDE,LONGITUDE or CALENDAR as appropriate).

Example:

i)
```
yes? REPEAT/L=1:240 CONTOUR/Y=30S:50N/X=130E:70W/K=1/FRAME TEMP
```
- produce a 240 frame movie of sea surface temperature.

ii)
```
yes? REPEAT/Z=300:0:-30 GO COMPZ
```
- execute command file COMPZ.JNL at Z=300, Z=270, ..., Z=0.

Command qualifiers for: REPEAT

/I /J /K /L
REPEAT/*=subscripts-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
REPEAT/*=coordinates-on-*-axis where "*" is X,Y,Z or T

28. SET

Set features of the operating environment for program FERRET

Generally, features may be toggled on and off with SET and CANCEL. Feature affected by SET may be examined with SHOW.

(see further help under CANCEL and SHOW)

28.1 SET DATA_SET

Specify the data set(s) to be analyzed use

4 FORMATS:
```
    SET DATA_SET data_set1, data_set2, ...
         or
    SET DATA/EZ[/qualifiers]   ASCII_or_binary_file_name
         or
    SET DATA n
         or
    SET DATA[/SAVE or /RESTORE]
```

In the first format an extension of .des is assumed.
In the third format "n" must be a previously SET data set as shown by SHOW DATA.

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: 30 (FERRET ver. 2.00) Use CANCEL DATA set if limit is reached

Command qualifiers for: SET DATA_SET

/SAVE
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 original state.

/RESTORE
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 original state.

/EZ
SET DATA/EZ

Access data from an ASCII or unformatted file that is not in a TMAP format

Format:
```
    SET DATA/EZ[/qualifiers]   ASCII_or_binary_file_name
```

Note:
"SET DATA/EZ" may be abbreviated as "FILE"
for example
```
        yes? FILE/VARIABLE=my_var my_data.dat
```

See also Data Sets ASCII_data.

Command qualifiers for: SET DATA_SET/EZ

/VARIABLES
SET DATA/EZ/VARIABLES="var1,var2,..."

Names the variables of interest in the file.

Names may be 1 to 8 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
```

default: "V1"

/TITLE
SET DATA/EZ/TITLE="title string" [data_set_name_or_number]

Associates a title with the data set.

This title will appear on plotted outputs at the top of the plot.

default: none

/FORMAT
SET DATA/EZ/FORMAT=file_format [data_set_name_or_number]

Allowable values for "file_format" are FREE, UNFORMATTED or a FORTRAN format in quotation marks and parentheses.

To use the format FREE a file must consist entirely of numerical data separated by commas, blanks or tabs.

To use the format UNFORMATTED all data must be binary, floating point with all of the desired data beginning on 4-byte boundaries. The "-" designator (see /VARIABLES) can be used to skip over unwanted 4-byte quantities in each record.

Examples:
```
    yes? SET DATA/FORMAT="(5X,4F12.0)" MY_DATA_SET

    yes? SET DATA/FORMAT=UNFORMATTED 3
```

default: FREE

/GRID
SET DATA/EZ/GRID=grid_or_variable [data_set_name_or_number]

Specify the defining grid for the data in the EZ data set. "grid_or_variable" 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=TEMP[D=gt4d011] SNOOPY
```

This is the mechanism by which the shape of the data (1D along T axis, 2D in the XY plane, etc.) is specified is specified.

By default FERRET will use grid EZ, a line of up to 2048 points oriented along the X axis.

default: grid EZ

/SKIP
SET DATA/EZ/SKIP=number_of_records [data_set_name_or_number]

Specify the number of records to skip at the start of an EZ data set before beginning to read the data.

default: 0

/COLUMNS
SET DATA/EZ/COLUMNS=number_of_columns [data_set_name_or_number]

Specify the number of columns in the EZ data file.

By default the number of columns will be assumed to be equal to the number of variables (including "-"'s) specified by the /VARIABLES qualifier.

default: number of variables

28.2 SET EXPRESSION

Specify the default context expression

Format:

```
    yes? SET EXPRESSION expr1 , expr2 , ...
```

Examples: SET EXPRESSION

i) `yes? SET EXPRESSION temp`
   - set the current expression to "TEMP"

ii) `yes? SET EXPRESSION u , v , u^2 + v^2`
   - set the current expression to "U , V , U^2 + V^2"

28.3 SET GRID

Use SET GRID to specify the default grid for abstract data

Format:

```
    SET GRID[/qualifier] [grid_or_variable_name]
```

(See GLOSSARY for clarification of terms)

Examples:

```
    yes? SET GRID TEMP[D=GT4D011]

    yes? SET GRID(use grid from last data accessed)

    yes? SET GRID/RESTORE
```

See also Variables Abstract_variables

Command qualifiers for: SET GRID

/SAVE
SET GRID/SAVE

Save the current default grid to be restored later

/RESTORE
SET GRID/RESTORE

Restore the current default grid that last was saved by SET GRID/SAVE

28.4 SET LIST

Use SET LIST to specify the default characteristics of listed output.

Format:
```
     yes? SET LIST/qualifiers
```

The state of the list command may be examined with SHOW LIST. See also CANCEL LIST.

Command qualifiers for: SET LIST

/PRECISION
SET LIST/PRECISION

To specify the data precision of the output listings use

```
     SET LIST/PRECISION=#_of_digits
```

/FORMAT
SET LIST/FORMAT

Specify an output format for the LIST command. (When a FORTRAN format is specified the row and column headings are omitted from the output.)

Format:
```
     yes? SET LIST/FORMAT=option
          or
     yes? SET LIST/FORMAT      (reactivate previous format)
```

Options:

FORTRAN format	- produces ASCII output
"UNFORMATTED"	- produces unformatted (binary) output
"EPIC"	- produces PMEL EPIC-format output
"GT"	- produces TMAP GT format

Note: EPIC format is not implemented in version 2.2 on Ultrix computers. Use a VMS system for EPIC format.

Examples:

i) `yes? SET LIST/FORMAT=(1X,12F6.1)`
   - specify a FORTRAN format (without row or column headings)

ii) `yes? SET LIST/FORMAT=UNFORMATTED`
   - specify binary output

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 will be repeated if necessary to output full record.
4) Latitude axes will be listed south to north when /FORMAT is specified.

FORTRAN format example:

```
yes? SET LIST/FORMAT=(1X,5F6.2)
yes? LIST/i=100:107/j=46:47 TEMP
 23.05 23.12 23.31 23.67 23.77
 23.43 22.87 22.47
 23.02 23.03 23.09 23.32 23.38
 23.05 22.64 22.42
```

/FILE
SET LIST/FILE

Specify a default file for the output of the LIST command

Format:

```
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 replaced by the name "SNOOPY.DAT" in

```
yes? LIST/FILE=SNOOPY.DAT variable
```

FERRET will generate 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.

Examples:

```
yes? SET LIST/FILE=AUTO
yes? LIST/Z=0/L=1/X=140W:110W/Y=2S:2N/FILE TEMP[D=GTAA014]
```

- sends data to file WGTAA014TEMP.X140W111WY2S2NZ5ML0001

```
yes? SET LIST/FILE=AUTO-XY
yes? LIST/Z=0/L=1/X=140W:110W/Y=2S:2N/FILE TEMP[D=GTAA014]
```

- sends data to file WGTAA014TEMP.Z5ML0001

/APPEND
SET LIST/APPEND

To specify that by default the listed output is to be appended to a pre-existing file. Cancel this state with CANCEL LIST/APPEND.

/HEAD
SET LIST/HEAD

To specify that ASCII output is to be preceded by a heading that documents data set, variable and region. Cancel the heading with CANCEL LIST/HEAD.

28.5 SET MODE

Specify special operating modes or states for program FERRET.

Format:
SET MODE[/LAST] mode_name[:argument]

MODE	EXPLANATION
DIAGNOSTIC	turn on internal program diagnostic output
VERIFY	display each command file line as it is executed
INTERPOLATE	automatically interpolate data between planes
IGNORE_ERROR	continue command file after errors
STUPID	do not use data from memory (diagnostic)
JOURNAL	record commands given in a journal file
LONG_LABEL	use "E" "W" notation for labelling longitudes
LATIT_LABEL	use "N" "S" notation for labelling latitudes
DEPTH_LABEL	use "DEPTH" as Z axis label
CALENDAR	use date strings for T axis (vs. time step values)
ASCII_FONT	impose PPLUS ASCII font types on plot labels
SEGMENT	utilize GKS segment storage
WAIT	wait for carriage return after each plot
REJEC	flag calculations with bad data as invalid
DESPERATE	attempt calculations possibly too large for memory
POLISH	interpret expressions in Reverse Polish order
GKS_DEVICE	perform graphical output through GKS
REMOTE_X	create graphical windows on a remote node
METAFILE	capture all graphics in GKS metafiles

Command qualifiers for: SET MODE

/LAST
SET MODE/LAST

Reset mode to its last state.

Format:
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
```

28.5.1 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) will be faithfully reproduced. For special plots it may be desirable to use other fonts - especially via the PPLUS DFLTFNT command. CANCEL MODE ASCII_FONT is for these cases.

default state: SET

28.5.2 MODE CALENDAR

The SET MODE CALENDAR command causes program FERRET to output times in date/time format (instead of time axis time step values). This will effect 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 will be unchanged from its last value.

default state: SET (argument: minutes)

Arguments

SET MODE CALENDAR command will accept the following arguments:

ARGUMENT	EQUIVALENT
SECONDS	-6
MINUTES	-5
HOURS	-4
DAYS	-3
MONTHS	-2
YEARS	-1

The argument will be 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 CANCELled the "equivalent" in the table above will determine the precision of the time steps displayed exactly as in SET MODE LONGITUDE.

28.5.3 MODE DEPTH_LABEL

The SET MODE DEPTH_LABEL command causes program FERRET to label Z coordinate information in the units of the Z axis. This will effect 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 DEPTH:argument
```

default state: SET (argument: -4)

Arguments

See SET MODE LONG for a detailed description

28.5.4 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 will attempt 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 will split 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 will do this only in desperation, if MODE DESPERATE is SET.

Example:

```
yes? SET MODE DESPERATE:5000
```

default state: CANCELLED (default argument: 80000)

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 words of internal storage. The lower bound is "memory block size".

28.5.5 MODE DIAGNOSTIC

The SET MODE DIAGNOSTIC command causes program FERRET to display diagnostic information in real-time about its internal functioning. (It is intended to assist with diagnosing performance problems.)

default state: CANCEL

28.5.6 MODE GKS_DEVICE

The SET MODE GKS_DEVICE command causes program FERRET to produce graphics through the Graphical Kernel System interface.

Advantages:

1) direct access to the workstation display on a VAXstation (without going through a terminal emulator)
2) metafiles that can capture color SHADE output
3) device independent metafiles that translate colored lines into dash patterns on black and white devices
4) ability to control line thickness (device dependent)
5) ability to generate displays on remote nodes

Disadvantages:

1) not available on all systems
2) some performance penalty on dumb graphics terminals

Example:

```
yes? SET MODE GKS_DEVICE:TEK4014
```

default state: determined by environment
default argument: "DEFAULT"

Arguments

Valid argument values:

DEFAULT	see text below
VSII	VAXstation II or GPX
PS	PostScript
LN03P	LN03 Plus (Tektronix 4014 emulation)
TEK4107	Tektronix 4107
TEK4014	Tektronix 4014

The argument is, by default, "DEFAULT". On a VMS system this means that the GKS workstation type of 0 is used - the value of the logical GKS$WSTYPE then determines the workstation type.

The importance of the workstation type is in determining whether FERRET will use colors or dash patterns, fill areas or hatch patterns, etc. See Jerry Davison's PLOTPLUS Enhancements manual for details.

28.5.7 MODE IGNORE_ERROR

The SET MODE IGNORE_ERROR command causes program FERRET to continue execution of a command file (see GO command) despite errors encountered.

default state: CANCEL

28.5.8 MODE INTERPOLATE

The SET MODE INTERPOLATE command 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 program FERRET will automatically interpolate from the surrounding grid point values. When this mode is CANCELled the same world coordinate specification will be shifted to the center of the grid box that contains it before computations were made (see examples).

default state: CANCEL

Examples: SET MODE INTERPOLATE

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
```

will list 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
```

will list the data at Z=15.

28.5.9 MODE JOURNAL

The SET MODE JOURNAL command causes program FERRET to record all commands given in the journal file. Output echoed to this file may be turned on and off via mode JOURNAL at any time.

default state: SET

28.5.10 MODE LATIT_LABEL

The SET MODE LATIT_LABEL command causes program FERRET to output latitude coordinate information in degrees N/S format (instead of the internal longitude coordinate). This will effect 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 LAT:2
```

default state: SET (argument: 1)

Arguments

See SET MODE LONG for a detailed description

28.5.11 MODE LONG_LABEL

The SET MODE LONG_LABEL command causes program FERRET to output longitude coordinate information in degrees E/W format (instead of the internal longitude coordinate). This will effect 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 command 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 minimum number of significant digits to display.

Example:

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 CANCELled the argument will still determine the output precision.

28.5.12 MODE METAFILE

The SET MODE METAFILE command causes program FERRET to capture all graphics in metafiles. These metafiles can later be routed to various devices to obtain hard copy output.

The argument to MODE METAFILE determines the preferred device type of the hard copy output, although it does not necessarily prevent the user from selecting a different device. It effects the number of line styles and fill patterns that will be used.

Hard copy is ultimately obtained with the GMOM command outside of FERRET. This command is documented in Jerry Davison's PLOTPLUS Enhancements manual. Normal use of it is:

```
$ GMOM PS2 METAFILE.PLT;*
```

Example:

```
yes? SET MODE METAFILE:TEK4014
```

default state: CANCELLED
default argument: PS (PostScript)

Arguments

The argument is, by default, "PS" (PostScript).

Valid metafile type:

VSII	VAXstation II or GPX
PS	PostScript
LN03P	LN03 Plus (Tektronix 4014 emulation)
TEK4107	Tektronix 4107
TEK4014	Tektronix 4014

28.5.13 MODE POLISH

The SET MODE POLISH command causes program FERRET to expect 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.

Example:
```
     yes? SET MODE POLISH
```
default state: CANCELLED

28.5.14 MODE REJECT

The SET MODE REJECT command causes program FERRET to reject and flag as invalid any calculation that involves invalid input data. When REJECT mode is cancelled FERRET will attempt to complete the calculation by ignoring the contribution of the invalid data.

* * * NOT YET IMPLEMENTED IN FERRET VERSION 2.00 * * *

default state: CANCEL

28.5.15 MODE REMOTE_X

The SET MODE REMOTE_X command causes program FERRET to create all future graphics output windows on the named remote node. The user must have privileges to create X windows on the remote node.

Format:
```
     yes? SET MODE REMOTE_X:node_name
```

Example:
```
     yes? SET MODE REMOTE_X:HOBBES
```
default state: CANCELLED

28.5.16 MODE SEGMENTS

The SET MODE SEGMENTS command causes program FERRET to utilize GKS segments ("GKS" is the Graphical Kernel System - an international graphics standard). Some advantages of this are that it permits graphics windows to be resized without blanking the window, permits the clearing of individual viewports within a window and in future versions of FERRET it will permit "picking" of graphic objects by the mouse.

Segments, however, make heavy demands on the systems virtual memory - sometimes exceeding the quotas allowed by an account. If FERRET crashes during graphics output due to insufficient virtual memory try CANCEL MODE SEGMENTS

default state: SET

note: MODE SEGMENTS is relevant only if MODE GKS is SET

28.5.17 MODE STUPID

The SET MODE STUPID command causes program FERRET to forget data stored in memory. The result is that all requests for variables will be read from disk rather that located in memory and reused from a previous read. The program will be significantly slower as a result.

default state: CANCEL

(This command is included for diagnostic purposes.)

28.5.18 MODE VERIFY

The SET MODE VERIFY command causes commands from a command file to be displayed at the terminal as they are executed.

default state: CANCEL

28.5.19 MODE WAIT

The SET MODE WAIT command causes program FERRET to wait for a carriage return 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: CANCEL

28.6 SET MOVIE

Note: This command is not implemented in version 2.2 on Ultrix.
- Use mode metafile and animation procedures described in Chapter 3.

Specify program states relevant in the making of animations

Format:
```
    yes? SET MOVIE[/qualifiers]
```

(see also FRAME)

Command qualifiers for: SET MOVIE

/FILE
Specify an output file to receive movie frames.

Format:

yes?	SET MOVIE/FILE=filename	to specify a new filename
	or	
yes?	SET MOVIE/FILE	to reactivate a previously specified filename after CANCEL MOVIE

The default movie filename extension is .MGM
The default movie filename is FERRET.MGM

/APPEND
Append subsequent movie frames to an existing movie file.

(The function of this command on the Panasonic laser disk has not been determined as of FERRET version 2.00)

Format:
```
    yes? SET MOVIE/APPEND
```

/LASER
SET MOVIE/LASER

Output to Panasonic laser disk recorder.

28.7 SET REGION

Specify the default space-time region for output commands

Format:
```
    yes? SET REGION[/qualifiers]
```

Refer to Regions and Context for further information.

Examples: SET REGION

i) `yes? SET REGION/X=140E`
- sets X in the default context

ii) `yes? SET REGION/@N`
- sets X and Y, only in the default context (since X and Y are defined in region N but Z and T are not).
  (See REGION @ notation)

iii) `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.

iv) `yes? SET REGION/@N/Z=50`
- sets ALL AXES in the default region to be exactly the same as region N and then sets Z to 50.

v) `yes? SET REGION/DZ=-5`
- set the region along the Z axis 5 units less than its current value

Command qualifiers for: SET REGION

/I /J /K /L
SET REGION/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
SET REGION/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/DI /DJ /DK /DL
SET REGION/D*=delta-subscript(s)-on-*-axis where "*" is I,J,K or L

see examples

/DX /DY /DZ /DT
SET REGION/D*=delta-coordinate(s)-on-*-axis where "*" is X,Y,Z or T

see examples

28.8 SET VARIABLE

Modify aspects 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.

Format:
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/UNIT=CM WIDTH[D=SNOOPY]
```

Command qualifiers for: SET VARIABLE

/TITLE
SET VARIABLE/TITLE="title string" variable_name

Associates a title with the variable.

This title will appear on plotted outputs and listings.

Default: none

/UNITS
SET VARIABLE/UNITS="units string" variable_name

Associates units with the variable.

The units will appear on plotted outputs and listings.

Default: none

/GRID
SET VARIABLE/GRID=grid_or_variable_name variable_name

Set the defining grid for a variable in an EZ data set.

Example:
```
     yes? SET VARIABLE/GRID=my_grid WIDTH[G=SNOOPY]
```

This is the mechanism by which the shape of the data (1D along T axis, 2D in the XY plane, etc.) is specified is specified.

By default FERRET will use grid EZ, a line of up to 2048 points oriented along the X axis.

default: grid EZ

28.9 SET VIEWPORT

Set the region within the output window where output will be drawn.

Format:
```
     yes? SET VIEWPORT view_name
```

Pre-defined viewports exist for dividing the window into four quadrants and for dividing the window in half horizontally and vertically.

Custom viewports may be defined with yes? DEFINE VIEWPORT
Viewports may be displayed with yes? SHOW VIEWPORT
See further help under Graphics Viewports

Pre-defined

See Graphics Viewports

28.10 SET WINDOW

Create, resize, reshape or move graphics output windows.

Format:
```
     yes? SET WINDOW[/qualifiers]    [window_number]
```

Note: Multiple windows may be simultaneously viewable but only a single window is receiving output at any time.

For further information see SHOW WINDOW and CANCEL WINDOW

Examples : SET WINDOW

i) `yes? SET WINDOW/NEW`
   - send next graphics to a new window

ii) `yes? SET WINDOW 3`
   - send next graphics to window 3

iii) `yes? SET WINDOW/SIZE=.5`
   - resize current window to 1/2 of full

iv) `yes? SET WINDOW/ASPECT=.5`
   - reshape current window with Y/X equal to 1:2 for the window

v) `yes? SET WINDOW/LOCATION=0,.5`
   - put 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

/NEW
SET WINDOW/NEW causes future graphical output to be directed to a new window. The window will be created at the next graphics output.

Format:
```
     yes? SET WINDOW/NEW
```

/SIZE
Resize a window to x times the size of the standard window. If the window number is omitted the command will resize the currently active window.

Format:
```
     yes? SET WINDOW/SIZE=x [window_number]
```

Note: A bug in VAX GKS may cause this command to relocate the window if the /LOCATE qualifier is not given explicitly.

/LOCATION
Set 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 fractional distances from the lower left corner of the display screen.

Format:
```
     yes? SET WINDOW/LOCATION=x,y [window_number]
```

Note: A bug in VAX GKS causes this command to take effect only when the size of the named window is changed.

/ASPECT
Set the aspect ratio of plots.

Formats:
```
     yes? SET WINDOW/ASPECT=y_over_x     n
       -  set the overall aspect ratio of window n

     yes? SET WINDOW/ASPECT=y_over_x
       -  set the overall aspect ratio of the current window

     yes? SET WINDOW/ASPECT=y_over_x:AXIS
       -  set the axis length aspect ratio of the current window
```

o The total size (area) of the output window is not changed.
o The default value for the overall window ratio is y/x = 8.8/10.2 .
o The default value for the axis length ratio is y/x = 6/8 .
o Use PPLUS/RESET or SET WINDOW/ASPECT=.75:AXIS to restore defaults.
o The aspect ratio specified is a default for future SET WINDOW commands
o The origin (lower left) is restored to its default values: 1.2, 1.4

29. SHADE

Produce a shaded plot of a 2-D field. By default a color key will be drawn and contour lines will not be drawn.

Format:
```
     SHADE[/qualifiers]     expression
```

Warning:
If multiple complex SHADE outputs are to be viewed on-screen simultaneously (using SET WINDOW or SET VIEWPORT) the process may crash from insufficient memory. Use CANCEL MODE SEGMENTS to avoid this problem.

Parameters

The expression may be anything described under 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

/SET_UP
SHADE/SET_UP

Performs all the internal preparations required by program FERRET for shading but does not actually produce output. The command PPL can then be used to make changes to the plot prior to producing output with the command PPLUS SHADE. This makes possible custom colors, labels, contour levels, etc.

/TRANSPOSE
SHADE/TRANSPOSE

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

/OVERLAY
SHADE/OVERLAY

Causes the indicated shaded plot to be overlaid on the existing plot.

/FRAME
SHADE/FRAME

Causes the graphic image produced by the command to be captured as an animation frame on the device specified by SET MOVIE

/NOLABELS
SHADE/NOLABELS

Inhibits all plot labels except logo written in the lower right.

/LEVELS
SHADE/LEVELS

Specify the SHADE levels or how the levels will be determined.

If the /LEVELS qualifier is omitted FERRET will automatically select reasonable SHADE levels.

To reuse the levels from the last CONTOUR or SHADE plot use SHADE/LEVELS.

To specify the SHADE levels manually use the syntax

```
SHADE/LEVELS=(lo,hi,delta).
```

For example, we might SHADE the sea surface temperature using:

```
yes? SHADE/LEVELS=(10,30,5) SST
```

to produce SHADE levels at 10-15, 15-20, 20-25, and 25-30. Note that a minimum of 2 levels is required.

The arguments on the right hand side of the equal sign can also be used to create unequally spaced SHADE levels. For detailed documentation on this refer to the "LEV" command in the PPLUS manual. If blanks are imbedded within the list of specifiers it will be necessary to enclose the entire list within quotation marks.

/LINE
SHADE/LINE

Causes the shaded plot to be overlaid with contour lines labelling the levels.

When /LINE is specified the key is omitted unless specifically specified via /KEY.

/KEY
SHADE/KEY

Causes a shade level key to be drawn on the plot. By default a key will be drawn unless the /LINE or /NOKEY qualifier is specified.

/NOKEY
SHADE/NOKEY

Suppresses the drawing of a shade level key on the plot.

/I /J /K /L
SHADE/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
SHADE/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/D
/D=data-set
See Context and Data_set

30. SHOW

Show program states and stored values

Command qualifiers for: SHOW

/ALL
Execute all SHOW options.
This command gives a complete description of the current state.

Format:
```
     yes? SHOW/ALL
```

30.1 SHOW AXIS

Show a basic description of the named axis

Format:
```
     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 exact text that will appear on plots and listings "(-)" 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

/ALL
SHOW a brief summary of all axes defined

Format:
```
     yes? SHOW AXIS/ALL
```

30.2 SHOW COMMANDS

Lists commands, subcommands and qualifiers recognized by program FERRET

Format:
```
     SHOW COMMAND    [partial_command]
```

Examples:
```
     yes? SHOW COMMAND S         - show all commands beginning with "S"

     yes? SHOW COMMAND           - show all commands
```

30.3 SHOW DATA_SET

Show information about the data sets which have been SET and indicate the current default data set.

By default the variables and their subscript ranges will also be listed.

Format:
```
     SHOW DATA[/qualifiers]   [set_name_or_number1,set2,...]
```

If no data set name or number is specified then all known data sets will be shown.

Command qualifiers for: SHOW DATA_SET

/BRIEF
SHOW DATA/BRIEF

Show only the names of the data sets; do not describe the data contained in them.

/VARIABLES
SHOW DATA/VARIABLES

In addition to the information given by the SHOW DATA command with no qualifiers this query will also give the grid and world coordinate limits for each variable in the data set.

/FILES
SHOW DATA/FILES

Display the names of the data files for this data set and the ranges of time steps contained in each. Output will be as date strings or as time step values depending on the state of MODE CALENDAR.

/FULL
SHOW DATA/FULL

Equivalent to /VARIABLES and /FILES used together.

Examples: SHOW DATA_SET

SHOW DATA will produce a listing similar to the one below. The output begins with the descriptor file name 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 info.)

```
   1> TMAP:[SETS]GT4D011.DES;14  (default)
        NMC output - 18x20x10x25 points
        Philander/Siegel diagnostic variables:  not available
name     title                  I         J          K          L
TEMP     TEMPERATURE          91:108    35:56       1:10       1:25
U        ZONAL VELOCITY       91:108    35:55       1:10       1:25
V        MERIDIONAL VELOCITY  91:108    35:55       1:10       1:25
W        VERTICAL VELOCITY    91:108    36:55       1:10       1:25
TAU      WIND STRESS          91:108    35:55       1:1        1:25
```

30.4 SHOW EXPRESSION

Show the current default expression

Format:
```
     yes? SHOW EXPRESSION
```

30.5 SHOW GRID

Show the name and axis limits of a grid on which a variable is defined.

Format:
```
     yes? SHOW GRID[/qualifiers]   [var_or_grid1 var_or_grid2 ...]
```

Examples: SHOW GRID

(See SHOW AXIS for an explanation of the columns)

```
yes? SHOW GRID SST[D=GTMNCLNMC]
   GRID NMC1
name      axis              # pts   start                 end
NMCX      LONGITUDE         180 r   20E                   18E
NMCY      LATITUDE           91 r   90S                   90N
normal    Z
TIME1     TIME               12mr   16-JAN-1901 05:00     16-DEC-1901
```

Parameters

The parameter(s) may be the name of one or more grid(s) or variable(s).

If no parameter is given SHOW GRID will display the grid of the last variable accessed. This is the only mechanism to display the grid of an algebraic expression.

Command qualifiers for: SHOW GRID

/ALL
SHOW the names, only, of all grids defined

Format:
```
     yes? SHOW GRID/ALL
```

/X
Displays the coordinates and grid box sizes for the X axis. Optionally, low and high limits may be specified (/X=lo:hi) to restrict the limits of this display.

Format:
```
     yes? SHOW GRID/X[=lo:hi] [variable_or_grid]
```

/Y /Z /T
SHOW GRID/*=coordinate(s)-on-*-axis where "*" is Y,Z or T

... similar to X axis

/I
Displays the coordinates and grid box sizes for the X axis. Optionally, low and high limits may be specified (/I=lo:hi) to restrict the limits of this display.

Format:
```
     yes? SHOW GRID/I[=lo:hi] [variable_or_grid]
```

/J /K /L
SHOW GRID/*=subscript(s)-on-*-axis where "*" is J,K or L

... similar to I axis

30.6 SHOW LIST

Show the current states of the LIST command.

Format:
```
     yes? SHOW LIST
```

30.7 SHOW MEMORY

Show the variables that are currently in memory and designated as permanent.

Format:
```
SHOW MEMORY[/qualifiers]
```

Command qualifiers for: SHOW MEMORY

/ALL
SHOW MEMORY/ALL

Shows all variables currently in memory - permanent and temporary.

/TEMPORARY
SHOW MEMORY/TEMPORARY

List the variables stored in memory and cataloged as temporary (they may be deleted when memory capacity is needed.

/PERMANENT
SHOW MEMORY/PERMANENT

List 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.

/FREE
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. 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
```

30.8 SHOW MODE

Show the names, states and arguments of the FERRET SET MODE command

Format:
```
     SHOW MODE partial_mode_name1,name2,...
```

Example:
```
     SHOW MODE GKS,META
```

30.9 SHOW MOVIE

Show the current states effecting the FRAME command. The same states also effect frames generated by the /FRAME qualifier.

Format:
```
     yes? SHOW MOVIE
```

30.10 SHOW REGION

Show the current default region

Format:
```
     yes? SHOW REGION[/ALL]     [region_name]
```

The region displayed will be 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 will be shown as Y=5S:5N but if the Y axis of the last data accessed is "ABSTRACT" then the Y location will be shown as Y=-5:5.

30.11 SHOW VARIABLES

List diagnostic or user-defined variables.

Format:
```
     SHOW VARIABLES[/qualifier]      [partial_name]
```

Examples:
```
     yes? SHOW VARIABLES    (all user-defined variables)
     yes? SHOW VAR/DIAG Q   (all diagnostic vars beginning with Q)
```

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 will be printed at execution time if such a variable is requested.

Command qualifiers for: SHOW VARIABLES

/ALL
SHOW VARIABLES/ALL

Lists both diagnostic variables (available for the COX/PHILANDER model) and user-defined variables.

/DIAGNOSTIC
SHOW VARIABLES/DIAGNOSTIC [partial_name]

Lists "diagnostic" variables available for the COX/PHILANDER model.

/USER
SHOW VARIABLES/USER [partial_name]

Lists expressions which have been defined by the user as new variables.

This is the default behavior of SHOW VARIABLES with no qualifier.

30.12 SHOW VIEWPORT

Show one or more of the currently defined viewports.

Format:
```
     yes? SHOW VIEWPORT [view_name1,view_name2,...]
```

30.13 SHOW WINDOWS

Lists open window numbers and indicates which is the active one.

Format:
```
     yes? SHOW WINDOWS
```

31. SPAWN

Create a subprocess within VMS to permit DCL commands.

Format:
```
     yes? SPAWN
```

Following the SPAWN command the next LOGOUT command given to DCL will return the user to FERRET at the point that he/she left off.

On Unix systems SPAWN is non-functional. Unix users may use "^Z" to suspend program execution, "fg" to resume.

32. STATISTICS

Compute summary statistics about the data specified.

Format:
```
    STATISTICS[/qualifiers]    expression_1 , expression_2 , ...
```

As of version 2.00 the statistics include:

- o total number of data values in the region specified
- o number of data values flagged as bad data
- o minimum value
- o maximum value
- o mean value(arithmetic mean - not weighted by grid spacing)
- o 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 will be treated in sequence.

The expression(s) will be inferred from the current context if omitted from the command line.

Command qualifiers for: STATISTICS

/I /J /K /L
STAT/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
STAT/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/D
/D=data-set

See Context and Data_set

33. USER

Execute a user-written extension to the FERRET program.

Format:
```
    USER[/qualifiers]    expression_1 , expression_2, ...
```

Parameters
Expressions may be anything described under Expressions.

If multiple variables or expressions are specified they will be passed simultaneously to the user-written routines.

The expression(s) will be inferred from the current context if omitted from the command line..

Implementation
Implementation of a USER command involves modifying the routines XEQ_USER_COMMAND, USER_SUB and possibly creating additional routines to be called by these. The routines, as supplied, are templates; they list simple statistics about the data specified as an example calculation.

The routines must be modified, compiled and relinked into FERRET.

Command qualifiers for: USER

/COMMAND
USER/COMMAND="command string"

The user-written command may use the optional string "command string" to select among a variety of actions. By this mechanism the USER command may perform many different functions.

/OPT1
USER/OPT1[="option 1 string"]

The user-written command may use the optional qualifier /OPT1 to modify the behavior of the USER command or to supply a string with parameters values, etc.

/OPT2
USER/OPT1[="option 2 string"]

... similar to USER/OPT1

34. VECTOR

Produce a vector arrow plot.

Format:
```
    VECTOR[/qualifiers] x_expr,y_expr , x_expr_2,y_expr_2 , ...
```

Parameters

x_expr, y_expr
Algebraic expressions (or simple variables) specifying the x components and y components of the vector arrows. If multiple variable or expression pairs are specified they will be overlaid on the output plot.

The expression pair(s) will be inferred from the current context if omitted from the command line.

(See Variables and Expressions)

Command qualifiers for: VECTOR

/SET_UP
VECTOR/SET_UP

Performs all the internal preparations required by program FERRET for vector plotting but does not actually produce output. The command PPL can then be used to make changes to the plot prior to producing output with the PPLUS VECTOR command. This makes possible custom colors, labels, contour levels, etc.

/TRANSPOSE
VECTOR/TRANSPOSE

Causes the horizontal and vertical axes to be interchanged. By default the X of the data axis will always be drawn horizontal and the Y and Z axes of the data will be drawn vertical. For Y-Z plots the Z data axis will be vertical.

/OVERLAY
VECTOR/OVERLAY

Causes the indicated field(s) to be overlaid on the existing plot.

/FRAME
VECTOR/FRAME

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

/NOLABELS
VECTOR/NOLABELS

Inhibits all label drawing except the plot logo written in the lower right.

/ASPECTS
VECTOR/ASPECT[=aspect_ratio]

Adjusts the direction of the vectors to compensate for differing axis scaling. THE SIZE OF VECTORS IS UNCHANGED - ONLY THE DIRECTION IS MODIFIED. The aspect ratio is (length of Y axis)/(length of X axis).

For example, in a typical XZ plane vector plot the vertical (Z) axis will be in tens of meters while the horizontal (X) axis will be in hundreds of kilometers. This means the vertical scale is greatly magnified in comparison to the horizontal. The /ASPECT qualifier will correspondingly magnify the vertical component of the vector relative to the horizontal while preserving the length of the vector. The magnification factor will be displayed on the plot.

/LENGTH
VECTOR/LENGTH[=value_of_standard]

Control the size of vectors.

If the /LENGTH qualifier is omitted FERRET will automatically select reasonable vector lengths.

To reuse the vector length from the last VECTOR plot use VECTOR/LENGTH.

To specify the vector lengths manually use the syntax
VECTOR/LENGTH=val.

This will associate 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.

For example, we might create a vector arrow plot of velocities using
```
     yes? VECTOR/LENGTH=100 U,V
```
to make 1/2 inch vectors for speeds of 100.

/I /J /K /L
VECTOR/*=subscript(s)-on-*-axis where "*" is I,J,K or L

/X /Y /Z /T
VECTOR/*=coordinate(s)-on-*-axis where "*" is X,Y,Z or T

/D
/D=data-set

See Context and Data_set

Chapter 3: Related Topics

35. Setting up an account

35.1 DEC Ultix setup

To set up your DEC Unix account to run FERRET follow the three steps below:

***** STEP 1:**

Execute interactively or add to your .login file the command

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

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

Explanation:

The FERRET program needs access to several files and directories to run properly. These Unix paths are stored in environment variables defined by the file "ferret_paths" .

To run FERRET your Unix account must also 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 command above.

***** STEP 2 (optional):**

Execute the "cp" command below:

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

Then use a text editor to customize my_ferret_paths. Instructions may be found inside the file.

Explanation:

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, descriptor, and grid definition 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 instructions contained in that file explain its useage.

***** STEP 3:**

Execute interactively or add to your .login file the following commands: (If you are not using your workstation console please read the explanation that follows)

```
    % setenv GKS3Dwstype 211
    % setenv GKS3Dconid 0.0
```

Explanation:

Getting graphical output from FERRET requires that two environment variables, "GKS3Dwstype" and "GKS3Dconid", be defined properly. (These environment variables are optional if either of the files, ~/.GKS3Ddefaults or /usr/lib/GKS3D/.GKS3Ddefaults exists and contains definitions appropriate to your current graphics environment. Those files are ASCII and self-explanatory.)

GKS3Dwstype is an integer identifying to GKS3D, the graphics "system" used by FERRET, what type of graphics device ("work station" in GKS3D parlance: X-windows server, Tektronix terminal, etc.) the output will be sent to. For X-windows outputs (including the monitor of your workstation) this value is "211". Tektronix 4014 terminals (and their emulators) use "70", and Tektronix 4107 terminals use "80". You can set the variable with the command (possibly with a different integer):

```
    % setenv GKS3Dwstype 70
```

The environment variable GKS3Dconid ("connect id") identifies to GKS where the graphical output device is connected on your system or network. For Unix workstations or X-Window terminal screens, its value is "internet_node_name:0.0", where "internet_node_name" is the name of the work station or X-terminal. For example

```
    % setenv GKS3Dconid anorak:0.0
```

If the graphical output will be displayed on VMS VAXstation screens, you may need to use double colons indicating that DECNET is the network transport layer: "decnet_node_name::0.0".

If you are using a Tektronix terminal (or an emulator), GKS3Dconid should be set to indicate the terminal name as revealed by the "who" command. For example, if "who am i" lists your current terminal as "tty08", then

```
    % setenv GKS3Dconid /dev/tty08 .
```

Note: Tektronix output also requires that when FERRET begins running you issue the command

```
SET MODE GKS:TEK4014      (or SET MODE GKS:TEK4107)
```

If this will be your normal environment consider placing the command in the file $HOME/.ferret (see below).

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 up permitting FERRET to be tailored to individual needs and styles. The file must be located in your HOME (login) directory.

FER_DATA - a list of directories to be searched to locate data files. Usualy 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.

FER_DESC - 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. Usualy 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.

35.2 VAX/VMS setup

When running FERRET on a VMS system three logical variables are relevant.

FERRET_JOURNAL (required) The name of the file that will receive a copy of all the commands you give to FERRET. It is normally defined to be FERRET.JNL .

FERRET_HELP (required)
The name of the help file accessed by FERRET. This should be defined by your system manager.

FERRET_INIT (optional)
The name of a file for FERRET to execute immediately upon starting up. This permits FERRET to be tailored to individual needs and styles.

FERRET is designed primarily to be run on a window-managed display. If you run FERRET from a Tektronix terminal (or emulator) begin your session with the command "SET MODE GKS:TEK4014" or "SET MODE GKS:TEK4107", as appropriate. If you find that the graphics screen interacts poorly with the text screen you can begin your session with the alternative commands "CANCEL MODE GKS" and "PPL PLTYPE 1". and use "SET MODE WAIT" if you wish. This method disables GKS metafiles, viewports and

SHADE output. You may wish to put these start up commands into the file pointed to by FERRET_INIT.

36. On-line HELP

36.1 DEC Ultrix 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 will then resume the suspended session.

The Unix commands provided assist 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.

Three 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

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

Ftoc

Usage: Ftoc

Description:

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 occurrance of "string" Note: the string is case sensitive

Fapropos

Usage: Fapropos string

Example: Fapropos regridding

Description:

Fapropos searchs 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 exactly the desired location.

Fhelp

Usage: Fhelp line_number
or Fhelp string

Examples: Fhelp 1136
Fhelp "modulo axis"

Description:

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 above under Ftoc.

36.2 VAX/VMS on-line HELP

On VMS systems help is available via the FERRET HELP command described in Chapter 2 of this manual.

37. Demonstration files

37.1 GO files

A number of demonstration "GO" files (FERRET scripts to be executed with the GO command) have been included with this distribution. These files should be immediately accessible to the FERRET user. The demonstration "GO" files may be executed simply by typing the FERRET command

```
GO demo_name
example: yes? GO spirograph
```

Below is a list of the files provided:

GO files
(located in directory $FER_DIR/fer/examples)

tutorial	Brief tour through FERRET capabilities
spirograph	For-fun plots from abstract functions
splash	For-fun mathematical color shaded plots
wire_frame	Creates a 3D wire frame representation
viewports	Output to viewports
custom_contour	Examples of customized contour plots
reading_files	Example of reading an ASCII file
regridding	Tutorial on regridding data
mathematics	Example of abstract function calculation
statistics	Playing with probability distributions
levitus_demo	Exploring T-S relationships using Sydney Levitus' Climatological Atlas of the World Oceans
coads_demo	A view of global climate using the Comprehensive Ocean-Atmosphere Data Set
fnoc_demo	Using Naval Fleet Numerical Oceanography Center data
relief_demo	Example of global topography

37.2 Sample data sets

A number of demonstration data sets have been 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

SET DATA data_set_name

example: yes? `SET DATA coads`

Data sets
Descriptors are located in directory $FER_DSETS/descr.
Grids are located in directory $FER_DSETS/grids.
Data are located in directory $FER_DSETS/data.

snoopy.dat	Sample ASCII data file (nothin' special)
tsheat021	Numerical model output (run #21) of ocean heat budget terms from a 1982-83 El Nino hindcast in FERRET "TS" format
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)
examp_t_independent	sample descriptor for time-independent data
examp_irreg_t_ax	sample descriptor using irregular time axis
examp_irreg_mod_t_ax	descriptor with irregular modulo time axis

38. Hard Copy

38.1 Hard copy on DEC Ultrix systems

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

> STEP 1:
Within FERRET, enter the command

```
yes? set mode metafile
```

This tells FERRET to generate a file (GKS3D metafile) for each plot created thereafter. To stop making the metafiles type

```
yes? cancel mode metafile
```

> STEP 2:
Produce each plot as you would normally. Each new plot on your screen will generate an additional file named 'metafile.plt.~n~ where 'n' will be incremented for each metafile. Overlay commands do not produce additional metafiles.

> STEP 3:
After EXITing from FERRET two commands are available to obtain hard copy.

SIMPLE METHOD: 'mtp' (metafile translate and print)

Note: The command 'mtp' will function only if it has been custom-tailored for your computer system (presumably by your system manager). See the Installation Guide.

The syntax of 'mtp' is

```
mtp [printer [metafile ...]]
```

Both the printer name and metafile name are optional. With no arguments mtp will route the files metafile.plt* to a (default) system graphics printer.

Examples,

% mtp	- render metafile.plt* on a default printer
% mtp phaser	- render metafile.plt* on printer "phaser"
% mtp lpr my_meta.plt	- render my_meta.plt on printer "lpr"

FULL CONTROL METHOD: 'mtt' (metafile translate)

The command 'mtt' allows you to control the translation of the device-independent metafiles made by FERRET into device-dependent output files. The 'mtt' command uses standard Unix command line syntax.

```
mtt [-d devtype] [-o ofile] [-l line] [-f fill]  [metafile ...]
```

Options

-d devtype — The target device type of the translator. If the -d option is omitted and output is to a file mtt will use devtype 'ps'.

Valid devtype values:

ps	- PostScript
cps	- color PostScript,
phaser	- Tektronix Phaser PX
tek4014	- Tektronix 4014
tek4107	- Tektronix 4107.
(nnn)	- integer GKS3Dwstype (not supported)

-o ofile — The output will be directed to the file ofile. If the -o option is omitted mtt will direct the output to the device specified by the environment variable GKS3Dconid using the device type specified by GKS3Dwstype.

-l line — Line types rendered by a particular device can be specified for a different device, if that device can also render those types. Used for example when sending output to a color workstation screen and the user is interested in knowing how the plot will appear as rendered by a monochrome hard copy device. Valid values for line are 'ps', 'cps', 'tek4014' and 'tek4107'.

-f fill — There are two fill area interior style types. Solid color is used by default by color PostScript and color workstations, and hatching by default by monochrome PostScript and Tektronix 4014 devices. This option is used in a way analogous to the -l option. Valid values for fill are 'color' and 'hatching'.

If the 'metafile' argument is omitted 'mtt' will default to translating and rendering a single file, 'metafile.plt'. After translation by mtt metafiles are renamed with a date stamp. To get hard copy printed, the output file needs to be sent to the appropriate printer.

Examples

To translate the two files, metafile.plt.~1~ and metafile.plt.~4~ into a single device-dependent file, plotfile.tek, ready to be routed to a Tektronix-compatible printer:

```
mtt -o plotfile.tek -d tek4014 metafile.plt.~1~ metafile.plt.~4~
```

To render the metafile metafile, metafile.plt, on the current console:

mtt

38.2 Hard Copy on VAX/VMS

Hard copy may be obtained in 3 ways:

Method A: (Preferred method)
1) within FERRET: enter the command SET MODE GKS,METAFILE
2) produce each plot
3) after EXITing from program FERRET use the GMOM command to route the plot to the desired hard-copy device.
   - This command is documented in Jerry Davison's PLOTPLUS Enhancements manual. Normal use of it is:
     `GMOM PS2 METAFILE.PLT;*`

Method B:
1) enter the command yes? PPLUS PLTYPE 4
   - this will instruct program PPLUS to generate graphic metafiles of the plots which follow. The metafiles will be named ZETA.PLT by default. The PPLUS command PLTNME can be used to change the metafile name (see also PPLUS);
2) produce each plot
3) after EXITing from program FERRET use Don Denbo's MOM command to route the plot to the desired hard-copy device.
   - This command is documented in the PPLUS user's manual. The normal use of it is:
     `$ MOM LN03 ZETA.PLT;*` - for black and white laser output

Method C:
1) with the graphics window fully exposed on the workstation screen enter the command yes? PPLUS PIXMAP.

2) do not obscure any portion of the window until the command prompt has returned indicating that the image has been captured (several minutes - depending on the size of the window)
3) after EXITing from program FERRET use the GMOM command to route the plot to color Versatek via $ GMOM CV PIXMAP.PLT

Method A creates device independent metafiles - graphics produced on a color device is made meaningful on black and white hard copy.

Method B produces a standard PPLUS metafile. No hard copy of the SHADE command is possible by this method. Colored lines will be indistinguishable on black and white hard copy.

Method C is suitable only for color hardcopy of the SHADE command.

39. Animations

39.1 Animations on DEC Ultrix

The command 'mtta' can be used at your workstation screen to display a number of FERRET plots in succession at a speed sufficient to animate them. The complexity of graphics and the resulting animation speed are inversely related.

The mtta command is a modification of the mtt metafile translator and its command syntax is the similar. You may select the metafiles to be shown, and run the animator as in these examples:

```
% mtta metafile.plt.~1~ metafile.plt.~2~ metafile.plt.~3~
% mtta metafile.plt*
```

The first example will select only 3 plots for sequential viewing; the second will select all with names that begin with 'metafile.plt'. If for example you have in your current directory seven plots named 'metafile.plt.~1~' through 'metafile.plt.~7~' to view, invoke

```
% mtta metafile.plt*
```

This results in the 7 files being read in and the plots stored in memory. A plot window will appear on the screen as the first plot is read and the following messages listed in your terminal window:

```
> Reading metafile.plt.~1~
> Reading metafile.plt.~2~
> Reading metafile.plt.~3~
> Reading metafile.plt.~4~
> Reading metafile.plt.~5~
> Reading metafile.plt.~6~
> Reading metafile.plt.~7~
```

Then you will be prompted to begin the sequence:

Enter interframe delay (eg, 100) or <CR> to animate, 'q' to quit:

You may enter a value here to place a delay between plots as they are displayed. Any integer value may be entered; a value of 100 places roughly one second between the display of successive frames. A zero (0) or a carriage return, only, will result in the plots displaying sequentially with no added delay.

The sequence will continue until interrupted - looping back to the first frame after the last is displayed. Interrupt by pressint the '^c' (control-c) key. You will then be prompted again with the same message as before. Enter 'q' to exit the program.

There is one other point to note. FERRET uses "emacs" style version numbers on the output metafiles it generates. For example if 18 FERRET plots are made, the plots will be sequentially named 'metafile.plt.~1~' through 'metafile.plt.~17~', and the 18th, or last, will have the name 'metafile.plt'. As this example shows the most recently made plot will have no version number attached.

Because of the sorting sequence used by the Unix shells, using the command

```
% mtta metafile.plt*
```

may incorrectly order the metafiles ("~11~" will precede "~2~"); the resulting animation will be out of sequence. The FERRET utility 'Fsort' can be used to correct the ordering.

To use Fsort with mtta use the Unix grave accent operator as in

```
% mtta 'Fsort metafile.plt*'
```

The file names will be sorted before they are passed to the animator. (Try 'Fsort metafile.plt*' at the Unix command line to see Fsort in action.)

Fsort can also be used in more complex mtta sequences, for example,

```
% mtta first.plt 'Fsort metafile.plt*' last.plt
```

39.2 Animations on VAX/VMS

A sequence of plots may be stored to form an animation, either on a Panasonic video disk with full color or on the VAXstation screen under X-windows or VWS in black and white.

See also SET MOVIE, CANCEL MOVIE, FRAME and REPEAT.

Examples: Movies

The following 3 commands will create animation of SST:

```
1) yes? SET MOVIE/FILE=MY_MOVIE
2) yes? REPEAT/T="1-JAN-1982":"1-JAN-1983":24 CONTOUR/K=1/@W/FRAME TEMP
3) yes? CANCEL MOVIE
```

Explanation:
Step 1) enables output file MY_MOVIE.MGM
Step 2) generates a sequence of frames contouring SST
Step 3) closes the movie file

40. Unix tools

A number of tools have been 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 the section in this manual on setting up your account if the tools do not appear on-line. They are described below.

Fenv syntax: Fenv

Prints the values of environment variables used by FERRET

Fgo syntax: Fgo name_substring

Searches the list of directories contained in the environment variable FER_GO to find the accessible FERRET GO command files that have names containing the indicated substring. For example,

```
> Fgo rgb
```

will locate the FERRET tools that contain "rgb" (setting up color tables).

Fdata syntax: Fdata data_file_substring

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

```
> Fdata coads
```

will locate the data files containing "coads" in their names.

Fdescr syntax: Fdescr des_name_substring

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

```
> Fdescr coads
```

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

Fgrids syntax: Fgrids gridfile_substring

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

```
> Fgrids fnoc
```

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

Fread1 syntax: 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 generally 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 syntax: 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.

Fpurge syntax: Fpurge filename_template

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

Fsort syntax: Fsort filename_template

Fsort is a support routine for the simple animations available with FERRET metafiles. (See the section discussing animations in this manual.) Fsort exists to reorder 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.

41. GO tools

A number of "script" files of FERRET commands have been provided with the FERRET distribution. These files, which are referred to as "tools", may be executed by the FERRET command GO. For example,

```
yes? GO land
```

will overlay the outline of the continents on your plot. (Note: the environment variable FER_GO must be defined to include the directory of tools supplied with FERRET. Normally this directory is $FER_DIR/fer/go where $FER_DIR is the directory on your system in which FERRET is installed.)

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

```
> Fgo grid
```

will display all of the tools with the substring "grid" as a part of their names.

Below is a table of the tools provided with your FERRET installation. (Others may have been added since this list was compiled in Oct. '91.)

class	tool_name	function performed
-----	---------	----------------
OVERLAYS		
	land	overlays accurate continental boundaries
	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
MATHEMATICAL		
	polar	define polar coordinates, R and THETA
	regressx	define variables for linear regression along X axis
	regressy	define variables for linear regression along Y axis
	regressz	define variables for linear regression along Z axis
	regresst	define variables for linear regression along T axis
	unit_square	sets unit square as default for abstract variables
	variance	define variables to compute variances and covariances
	var_n	refine TVARIANCE with corrected n/n+1 factors
PLOT APPEARANCE		
	show_lines	draws specimens of the available line styles
	show_fill	draws specimens of the available fill styles
	show_symbols	draws specimens of the default symbols
	show_88_syms	draws specimens of all 88 PLOT+ "marks"
	rgb_grayscale	sets the color table to grayscale
	rgb_rainbow	sets the color table to "rainbow" (default)
	rgb_centered	sets the color table appropriate for zero-centered data - red shades for the upper half of the scale - blue shades for the lower half
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
TESTS		
	test	test proper functioning of FER_GO
	ptest	produce a quick test plot

42. Converting data to FERRET formats

Introductory Note:

This write-up assumes that the environment variable FER_DIR is defined in your account. (Try "echo $FER_DIR".) If FER_DIR is not defined consult the section on Setting Up Your Account in this manual.

Typically you will want to create a directory in which to manage data sets used by FERRET - often the directory $HOME/ferret is used. We will assume that this directory is your current directory in the discussions below.

Although FERRET is able to read data from ASCII files and from most simply structured binary files very great benefits may be derived from converting your data into one of FERRET's "direct access" formats. The advantages include faster access, ability to handle vastly larger data sets, automatic labelling of variables, axes and units, improved memory management, friendly query commands for the contents of your data, etc.

The following 4 steps should be followed to convert your data to a FERRET format. Detailed discussion of each step follows.

STEP 1) choose GT or TS format

STEP 2) create grid and descriptor files for your data

STEP 3) modify the conversion program to read your data

STEP 4) link and run the conversion program

DETAILS OF STEP 1) "choose GT or TS format"

Two formats are available for FERRET data: "GT" (grids at timesteps) and "TS" (time series). GT format is best suited to grids with large numbers of points in the X-Y-Z (spacial) dimensions and to cases where the viewing of the data along spacial axes will be as common as the viewing of time series. Conversely, TS format is optimized for data in which the number of points in X,Y, and Z is small compared to the number of points along the time axis. Only performance (computer speed) will be affected by the choice. This write-up discusses only conversion to GT format. Conversion to TS format is similar and support routines are provided.

DETAILS OF STEP 2) "create grid and descriptor files for your data"

Note: Considerable documentation on grid definition files and descriptor files can be found in $FER_DIR/doc. Also numerous sample files are available in the directories listed in the environment variables FER_GRIDS and FER_DESCR. (Use "Fenv" to see the lists of directories.)

Copy the following template files from $FER_DIR/fmt/convrt/gt into your FERRET conversion directory (current directory).

```
convert_to_gt.des
convert_to_gt.grd
```

Rename the files appropriate to your data set. The grid file should retain the ".grd" extension. The descriptor should retain the ".des" extension.

Use an editor to modify the template grid file so it describes the axes and grids on which your data is defined. The file is 132 characters wide. Explanatory documentation is contained inside the file. Further documentation is available in $FER_DIR/doc. Numerous examples of grid files are available in the directories pointed to by the environment variable Fgrids.

Use an editor to modify the template descriptor file so it describes your data set. The descriptor file is in FORTRAN NAMELIST format - an ASCII file. Some lines should not be modified after the data has been converted as FERRET does "sanity" checks between the descriptor files and the converted data files. The list of descriptor lines given at the end of this section will assist you in making choices while editing the descriptor file. Lines not listed do not generally require changes.

DETAILS OF STEP 3) "modify the conversion program to read your data"

Copy the following files from $FER_DIR/fmt/convrt/gt into your FERRET conversion directory:

```
Makefile
convert_to_gt.f
gt.cmn
```

Give the command "make links" to create links required for compiling and linking the code. Check that the links created look reasonable.

With an editor modify the FORTRAN source program convert_to_gt.f so it reads your data and writes the variables named in the descriptor in the same order that they appear in the descriptor. (see internal documentation in convert_to_gt.f)

DETAILS OF STEP 4) "link and run the conversion program"

Execute "make" to compile and link the convert_to_gt program. Correct compiler errors and repeat as necessary.

Execute "convert_to_gt" to convert your data.

If your data files are large you may wish to move them to another directory or device. If so, make sure that the environment variable FER_DATA includes the path to that area so that FERRET will know where to find the files.

DESCRIPTOR FILE LINES THAT MAY REQUIRE CUSTOMIZATION:

This list of descriptor lines will assist you in making choices while editing the descriptor file. Lines not listed do not generally require changes. Lines marked with "*" should not be changed after the data has been converted.

- o first line: comment line describing the descriptor internally
- *o D_EXPNUM = 'xxxx' - any 4 character string you choose
- *o D_MODNUM = 'yyyy' - any 4 character string you choose
- o D_TITLE = 'descriptive title of this data set'
- o D_MOD_TITLE = 'parenthetical remark about this data set'
- o D_T0TIME = 'dd-mmm-yyyy:hh:mm:ss' - the date, if any, to be associated with T=0 on your time axis. If your data set is time-independent set this to 'INDEPENDENT'. S_START, S_END, and S_DELTA, below, are rendered irrelevant in time-independent cases but we recommend setting them to -1.E34. See $FER_DIR/doc/setting_up_a_time_axis.txt for further help.
- o D_TIME_UNIT = 3600 - the number of seconds represented by one unit of your time axis. See $FER_DIR/doc/setting_up_a_time_axis.txt for further help.
- o D_TIME_MODULO = .FALSE. - use ".TRUE." only if your time axis is "modulo" (wraps around)
- o D_TIME_MADE = 'string' - documentation
- o D_WHO_MADEIT = 'string' - documentation
- o D_GRID_FILENAME = name of grid file created above (name, only, the path is in the environment variable FER_GRIDS)

For each variable in your data set:

- *o D_VAR_CODE = 'vvvv' - 4 character code by which this variable will be known
- o D_VAR_TITLE = 'title string for variable'
- o D_VAR_TITL_MOD = 'parenthetical title modification string'
- o D_VAR_UNITS = 'units string for variable'
- o D_GRID_NAME = 'string' - name of grid on which this variable is defined (defined in grid file, above)
- o D_MISSING_FLAG = xxx.xxx - value to be interpreted as missing data
- o D_BAD_FLAG = must be the same as D_MISSING_FLAG
- *o D_ORDERING = 'WE', 'SN', 'UD', 'TI' - indicates that data should be stored on disk as records lying along the X ('WE'- "west to east") axis with successive records progressing along Y ('SN' = "south to north") and successive planes progressing along Z ('UD' = "up to down"). 'TI' must be the final axis. This ordering is relevant for performance, and compactness, only. It is normally altered only for data sets with very short (fewer than 6 points) X axes.
- *o D_GRID_START = 1,1,1,-1 - the starting subscript of the data in this data set along the X,Y,Z, and T axes, respectively, relative to the full extent of the axes on which it is defined. This mechanism allows subsets of very large data sets to easily managed. The "-1" on the T axis indicates that the limits will be determined from "step file" information, below. Typically, "1,1,1,-1" is appropriate.
- *o D_GRID_END = ii,jj,kk,ll as in D_GRID_START, above. Typically "xsize,ysize,jsize,-1", where *size are the number of points on each axis of the variable.

For each block of data files (time step files) with equally spaced time steps:
*o S_FILENAME = 'string' - The data filename. Do not include the path - it will be inferred from the environment variable FER_DATA. The filename extension ".001" is recommended.
*o S_NUM_OF_FILES = nnn - The number of files described by this block. If nnn is greater than 1 the filenames will count up from the .001 extension indicated for S_FILENAME. (Note that values greater than 1 are supported by FERRET but are not supported by the conversion program as of 10/91.)
*o S_START = xxx.xxx - The value of the time word for the first time snapshot of data in this file. D_T0TIME, and D_TIME_UNIT determine the date associated with this time word. See $FER_DIR/doc/setting_up_a_time_axis.txt for further help.
*o S_END = xxx.xxx - The value of the time word for the last time step in the block of time step files described.
*o S_DELTA = xxx.xxx - The delta value of the time word between successive time steps in the data set.

43. Wire frame 3D graphics

Wire frame representations of 3D fields are available from FERRET. These graphs are not strictly a "supported" capability of FERRET but are available as a feature of the PLOT+ graphics which is embedded within FERRET. The wire frame representations can be used on any 2-dimensional field; they are an alternative representation to CONTOUR or SHADE.

To obtain wire frame outputs follow these steps (example below):

i) pass your data to PLOT+ from FERRET using CONTOUR or SHADE;
ii) use the PLOT+ command "window off" (once);
iii) use the PLOT+ command "vpoint" to set the 3D viewpoint ("vpoint x,y,z" where (x,y,z) is the viewing location); and
iv) use the PLOT+ command "view" to produce the plot.

These commands are described in greater detail in the PLOT+ user's manual.

Hard copy may be obtained in the usual way using SET MODE METAFILE and the metafile translator. The plots are interruptible using the control-C key, as usual.

In the following example FERRET and PLOT+ commands will create a wire frame representation of a simple mathematical function in 2 dimensions.

```
!*********************************************************
! define a simple 2D Gaussian function in the XY plane
LET GAUSS = 2*EXP(-1*((x/2)^2 + y^2))

! define an 80 by 80 point region from (-4,-4) to (4,4) in the XY plane
DEFINE AXIS/X=-4:4:.1 xax80
DEFINE AXIS/Y=-4:4:.1 yax80
DEFINE GRID/X=xax80/Y=yax80 g_gauss
SET GRID g_gauss     ! the pseudo-variables "x" and "y" on this grid
SET REGION/X=-4:4/Y=-4:4
```

```
! pass the data to PLOT+
CONTOUR/SET_UP gauss

! tell PLOT+ to view the field from (x,y,z)=(-4,-10,4)
ppl window off
ppl vpoint -4,-10,4
ppl view
!*********************************************************
```

44. Unix file naming

On VMS systems file versions are provided automatically by the file system. On Unix systems FERRET uses a file naming scheme to differentiate successive versions of the graphic metafiles and the journal files it creates. The version numbering scheme is styled after the gnu (Free Software Foundation) emacs editor. The scheme appends the version 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 version suffix appended represents the most recently created version of a file. At the time when the next successive version is created this file will have the suffix ".~nnn~" appended to it, where nnn is the current highest version number plus one.

Two Unix tools have been provided to assist with managing multiple version numbers:

Fpurge - remove (delete) all but the current version of the named file.
Example: `% Fpurge ferret.jnl`

Fsort - sort the versions of a file into increasing numerical order
Example: `% mtta 'Fsort metafile.plt*'`

See further information under Unix Tools in this manual.

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

Go filename
Set DATA filename
Define Grid/file=filename
List/file=filename (do not use relative version's with this command)

The file name 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.

44.1 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 as relative to the current highest version number.

For example,

If the current directory contains the files

```
snoopy.dat   snoopy.dat.~1~  snoopy.dat.~2~  ... snoopy.dat.~99~
```

then the filename snoopy.dat.~0~ refers to the snoopy.dat and the filename snoopy.dat.~-1~ refers to snoopy.dat.~99~.

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

```
yes? go snoopy.dat.~-1~
       or
yes? go snoopy.dat~-1
```

As another example, if we want to access a ferret.jnl file which already exists, the following are all valid:

```
yes? go ferret.jnl.~-4~
       or
yes? go ferret.jnl~-4
       or
yes? go ferret.~-4~
       or
yes? go ferret~-4
```

45. Modifying FERRET

Note: Regrettably DEC RISC/Ultrix workstations are currently supporting two incompatible versions of the f77 compiler (versions 2.1 and 3+). This creates obvious awkwardness supplying the FERRET libraries (archives) in a form ready to link. To avoid these complexities version 2.1 libraries, only, have been included with your FERRET distribution. You may obtain the version 3.0 libraries from Steve Hankin, Internet: hankin@noaapmel.gov, FAX:(206)526-6744, Phone:(206)526-6080. Please specify which compiler you have available to you.

Libraries and source files have been provided together with FERRET to permit a user to make limited modifications to FERRET and to relink the program. The areas targetted for modification are:

1) defining a custom "USER" command

2) modifying the OCEAN model diagnostic variables that are built into FERRET.

3) Customizing file input routines for special formats.

All of the necessary files for relinking may be found in $FER_DIR/fer/link. The steps to follow in making modifications are:

***** STEP 1) modify the source code.**

The USER routine source code is in $FER_DIR/fer/link/user_sub.f. Private subroutines may be called from user_sub.f. The USER command is documented in the FERRET Users' Guide. The OCEAN code is in $FER_DIR/fer/link/ocn. The file IO libraries are available in $FER_DIR/fmt/src. Included COMMON and PARAMETER files are in $FER_DIR/fer/link/common.

After changes are made the main program, $FER_DIR/fer/link/ferret_main.f, should be documented to reflect the changes. DATA statements containing program name and revision number in that file should be modified.

***** STEP 2) "make" FERRET**

A Makefile, $FER_DIR/fer/link/Makeferret, has been provided to perform the linking. If you have added new subroutines you will need to update Makeferret to reflect that fact. Then execute it with the Unix command

```
% make -f Makeferret
```

The resulting executable file will be "custom_ferret".

***** STEP 3) Save all of your changes.**

Archive all of your work to another area where it will be safe. The directory $FER_DIR/fer/link will be over-written at the next FERRET upgrade.

46. Release notes

Rev1.00

Significant changes from GFDL version 2.01:

- o added STATISTICS command
- o added modes CALENDAR,LATIT_LABEL,LONG_LABEL, DEPTH_LABEL and FONT_ASCII to SET MODE
- o added /NOLABELS qualifier to PLOT, CONTOUR and VECTOR
- o added /FORMAT=TMAP option to SET LIST
- o added /NAME qualifier to LOAD and to SHOW MEMORY
- o added /LAST qualifier to SET MODE
- o added double ^Z as an EXIT option to avoid batch job hangs
- o added diagnostic variables: DENS - density based on Knudsen's formula
  CAIR - LEVITUS climatological air temperature
  AIR - air temperature field used by 205 run
- o added calendar-format IO
- o added plot label symbols (See Graphics Plot_labels)
- o added negative delta option to LIST
- o added HELP Tools to explain pre-defined GO options
- o expanded SHOW DATA output to give data set title and GFDL parameters and to take a data set number as an argument
- o SHOW VARIABLES now accepts a variable name argument
- o modified diagnostic variables:
  QRAD - allows 3 parameterizations of radiation - 2 based on SST
  QSEN,QEVA,QFLX - allow parameterized air temperature
- o descriptor files may describe 205 parameters:
  (See Data_set Descriptors)
- o descriptor files/grid files may define new axes and grids and specify variables defined on them
- o corrected sign error in UBPS
- o SHOW VARIABLES and SHOW COMMANDS are interruptible via ^C
- o incorporated logicals GFDL_HELP, GFDL_JOURNAL and GFDL_205RUNS to point to help text, journal filename and the 205 runs dbase
- o eliminated COS correction for when Y axis orientation is not given as SN in a grid file (use orientation "NA")
- o the @SHF transformation can be used only with I,J,K or L
- o output units for axes and variables
- o HELP output waits on scroll after 21 lines
- o eliminated restrictions associated with accessing lines of data ("1-D variables") produced by transformations
- o internal changes:
  - defined "regular" axes (start,n,delta)
  - use GET_CONTEXT_PLANE to set up transformation components
  - "extract" 1-D lines from 2-D grids in memory

Rev1.10

Significant changes from FERRET version 1.00:

new commands and options:

- o DEFINE GRID, DEFINE AXIS and SHOW AXIS allow for defining grids
- o SET WINDOW to create multiple output windows and to resize existing ones
- o SPAWN to spawn out to VMS/DCL
- o CANCEL MEMORY to clear out all memory (useful in batch command files to reduce resource usage)
- o SHOW MEMORY/FREE to show the unused memory
- o SET MODE WAIT helps graphics for non-VAXstation users
- o CONTOUR/LEVELS qualifier preserves last contour levels used
- o @FAV (fill with average) performs missing data filling
- o Regridding is available via the G=grid_or_variable qualifier
- o LISTing to file now performed with /FILE qualifier on the LIST command instead of SET LIST/OUTPUT mode
- o LIST command accepts qualifiers /FILE= and /APPEND

minor changes:

- o FORMAT=TMAP has become FORMAT=GT on the SET LIST command
- o FORMAT=GT now handles multiple variables and time steps
- o The D= data set qualifier now accepts data set names as well as numbers. SHOW DATA data_set_name is also valid.
- o LIST command is interruptible
- o AXLEN and VECSET are reinitialized for each FERRET plot
- o provisions for multiple 205 air temperature climatologies
- o @SBX:even now half-weights the endpoints. No shift to right.
- o invalid data from 205 data base causes warning, only
- o SHOW DATA defaults to /BRIEF instead of /FULL
- o SHOW GRID var[D=name_or_number] has replaced SHOW GRID/D=number SHOW GRID grid_name is also valid

Rev1.20

Significant changes from FERRET version 1.10:

new commands and options:

- o SHADE for color-filled (contour-like) plots
- o SET WINDOW/ASPECT to change aspect ratio of window
- o DEFINE SET SHOW and CANCEL VIEWPORT
- o SET MODE REJECT and SET MODE JOURNAL
- o PPLUS/RESET

other changes

- o added regridding transformations (See Grids Regridding)
- o PPLUS commands AXLEN and SIZE may be used to modify graphics
- o graphical output is now interruptible via ^C
- o SHOW DATA/VARIABLES displays position and time limits
- o SHOW GRID/ALL lists grid names, only

- o error messages are preceded by **ERROR
- o dates in the years 0000 and 0001 are not displayed (for mock climatologies)
- o time axes are automatically "styled"
- o REPEAT command accepts hi:lo:delta with delta
- o MODE GAPS_OK has been eliminated
- o dimensions formerly labelled as "N/A" have been removed
- o FERRET will continue execution even if it cannot create a journal file

new associated packages
- o PPLUS version 1.1 has been incorporated
- o a new version of the TMAP library has been incorporated
  - a time series format is supported
  - grid points need not be a grid box centers
  - grid files may include time axes, modulo axes and lat/long-formatted coordinates

Rev1.21

Significant changes from FERRET version 1.20:
- o fixed program crash using PPLUS TXTYPE DAY
- o allow any air temperature data set defined by GFDL_205RUNS
- o SET/CANCEL the FERRET mode WAIT sets the PPLUS state, too
- o MODE INTERPOLATE defaults to CANCELLED (note: interpolations have bugs when applied to multiple axes or nested with regridding)

Rev2.00

Significant changes from FERRET version 1.21:
new commands and options:
- o user can define variables with DEFINE,SHOW and CANCEL VARIABLE
- o able to read ASCII and unformatted files
- o reverse Polish ordering is no longer required
- o added @LOC transformation (e.g. depth of 20 degree isotherm)
- o added @SBN binomial smoother
- o LIST/ORDER= permits control over axis ordering in listings
- o PLOT/VS accepts expressions with dimensions bigger than lines
- o PLOT/SYMBOL and PLOT/LINE provide simple line style control
- o CANCEL WINDOW/ALL will eliminate ALL graphics windows
- o able to manipulate abstract mathematical quantities
- o plots with multiple variables are labelled with keys
- o SET GRID will define the default grid for abstract variables
- o SET REGION and DEFINE region have /D* (delta) qualifiers
- o GKS metafiles provide device independence (color vs. monochrome)
- o PPLUS PIXMAP command can capture color shaded plots
- o remote graphics displays are provided via MODE REMOTE_X

- o up to 50 times faster than version 1.21
- o 0,1,2,3 and 4 dimensional variables can all be LOADED and LISTED
- o command abbreviations LET (DEFINE VARIABLE) and FILE (SET DATA/EZ)
- o the default region is changed only by SET and CANCEL REGION
- o "D" notation has been eliminated (use SET REGION/DX=... instead)
- o @AVE now averages despite bad data - uses available data
- o SHOW VARIABLES defaults to SHOW VARIABLES/USER
- o LOAD/NAME has been eliminated (replaced by DEFINE VARIABLE)
- o use /OVERLAY for multiple CONTOUR,VECTOR and SHADE variables
- o on-disk movies are now created in X windows format

Rev2.10

Unsupported "BETA" test version on DEC Ultrix

Rev2.20

Significant changes from FERRET version 2.00:

- o run on DEC Ultrix/RISC platforms
- o able to read both IEEE and VAX F-formatted binary files
- o Uses Unix environment variables (FER_DATA, FER_DESCR, FER_GRIDS, ...)
- o supports time-independent GT data sets
- o improved command syntax checking
- o new windowing smoothers added: Hanning (@SHN), Parzen (@SPZ), and Welch (@SWL)
- o minor change to Philander/Seigle model calculations: mimics 205 tau regridding at first and last hour of the month

INDEX

Appendix C. PLOT+ Enhancements

PPL+

PLOTPLUS ***PLUS***: ENHANCEMENTS TO A STANDARD

A User's Guide to the TMAP Modifications of the Plotplus Graphics Package

Jerry Davison
PMEL/TMAP

Oct, 1991

I. PLOTPLUS HISTORY, EVOLUTION

Plotplus is a scientific graphics package with a long history. I have traced it only a small distance, and what I know is sketchy. My present understanding is that a number of users at the Oregon State University department of Oceanography contributed over a number of years to a graphics package with both original and pre-existing algorithms and code; PLOT1, PLOT2, PLOT3 and PLOT4 successively became the current standard. Don Denbo took a strong interest in improving the package; from his work evolved PLOT5. He came to PMEL, improved PLOT5 further, and Plotplus was born. While here he made modifications for the TMAP group to that code to support, in addition to Tektronics, Hewlett-Packard and other devices, the Graphics Kernel System, GKS, an international standard for programming computer graphics applications. This user's guide describes modifications I made to Plotplus to extend the use of GKS within it; this version will no doubt evolve as it is used, as well.

The guide addresses itself only to the TMAP modifications to Plotplus. The Plotplus manual describes all other aspects of the current version as supported by its author and should be consulted for information about using Plotplus.

The TMAP GKS enhancements of Plotplus include modification and addition of several ppl commands, including:

ALINE,
CLSPLT,
LINE,
LIST,
PEN,
PLTYPE,
SHADE,
SHAKEY,
SHASET.

Additionally, there is a new command level utility, mtt, to generate monochrome and color hardcopy of PPL+ plots.

II. ENHANCED COMMANDS DESCRIPTION

ALINE/qualifier line#, minx, miny, maxx, maxy, set

Draws the line associated with the specified line number between 2 points (see PEN for more on this). Two modes are available. In immediate mode the line is drawn when the command is given. Deferred mode permits setting of several lines (with individual endpoints) to be drawn whenever the PLOT command is given. Deferred mode is included so that examples of each linetype used in a plot can be provided as part of a key. The ALINE command does not modify data in the plot buffer; lines drawn can be considered as labels. The ALINE command given with no arguments resets all set lines to OFF.

line#	The line to be drawn will be of the type, thickness, and color associated with this line number.
minx	X-component of the first endpoint.
miny	Y-component of the first endpoint.
maxx	X-component of the second endpoint.
maxy	Y-component of the second endpoint.
set	is optional. If omitted, execution mode is immediate. If ON, sets deferred mode for the specified line number. If OFF, drawing the line is cancelled; specification of the endpoints may be omitted when cancelling ALINE for a line. Execute LIST ALINE to find which lines are set, and their coordinates.

Valid qualifier:

/[no]user determines whether user coordinates or inches will be used in locating the line. Default is /user.

CLSPLT

Modified to be compatible with GKS metafile use. Closes the currently open plot file.

LINE n, mark, use

The original PPL command has been modified. PPL+ uses GKS line bundles to specify line type, thickness, and color. A large number of line bundles are defined for each device type and their characteristics depend on the capabilities of the device. See appendix I for this information. The LINE command use argument no longer specifies the line type -- whether the line is to be dashed or solid. Specification of the use of marks remains the same.

n	line number
mark	data mark
use	line/mark use specification 0 - line connecting points and no mark at points 1 - mark data points 2 - mark end points only 3 - only marks (no line)

LIST arg

New arguments are available to the LIST command to request information about the settings of the added features.

arg	New arguments are ALINE, SHAKEY, and SHASET.

PEN n, ndx

This command has been modified with the use of GKS line bundles in PPL+. It now specifies the line bundle index associated with each line. See appendix I for the type, thickness, and color representation for each line bundle of the supported devices.

n	The line number. If n is 0, sets the pen used to plot the axes and labels.
ndx	sets the line bundle index to be used for line n. Default is 1.

PLTYPE icode WS=wstype META

PLTYPE 3, GKS output, is expanded to support additional devices, both soft and hard copy. These include DEC and VAXstations, the Tektronix 4014 and 4107, the HP 7550, and PostScript device types. Hardcopy can be run off using the mtt command; a separate document covers its use.

icode must be 3 or 4 to use the new features.

wstype The immediate output device. Supported devices are:

VSII	DEC and VAXstations, both monochrome and color
TEK4014	Tektronix 4014 monochrome graphics terminal
TEK4107	Tektronix 4107 color graphics terminal

Use is optional, the default value is VSII. When TEK4014 or TEK4107 is specified, the computer terminal port being used must be specified before beginning the PPL+ session.

A GKS metafile named METAFILE.PLT (with sequential version numbers for subsequent plots) is produced with each plot when META is specified. Metafiles are device independent and can be rendered by specific devices in a manner appropriate to each device.

After the metatype is set, if you wish to deactivate the metafile output, reenter the PLTYPE command without entering META, e.g., PLTYPE 3. To reactivate, reenter PLTYPE including the META specification, .e.g., PLTYPE 3 META. The mtt command referred to above translates the metafiles and generates plots from them.

SHADE/qualifier

Generates a fill area plot of a 2-d field. The results of the command are device specific. On monochrome devices, patterns (or hatching) are used; on color devices, color fill is used. A rectangular grid is defined when visualizing 2-d fields in PPL; a grid cell is associated with each point. The SHADE command fills in each grid cell with a color, or particular pattern, determined by the field value at the grid points. Appendix II outlines the response of the supported device types to this command.

The LEV and LIMITS commands can be used, in a way identical to their use with CONTOUR, to determine the levels shaded, and specify intervals. The SHAKEY and SHASET commands also control the appearance of the plot; default colors (or patterns) and key attributes will be used if not specified. The /[no]wait and /[no]overlay qualifiers are valid, used in the same way as with PLOT and CONTOUR.

SHAKEY do_key, orient, klab_siz, klab_inc, klab_dig, klab_len, kx_lo, kx_hi, ky_lo, ky_hi

This command controls the attributes of the key generated by the SHADE command. The key associates the colors or patterns used in the plot with the field values; its use is optional. LIST SHAKEY will list current settings of the key.

do_key	If 0 the key will not be displayed; if 1 the key will be displayed. Default is 1.
orient	If 0 the key is horizontal (by default on top of the figure); if 1 the key is vertical (by default on the right). Default value is 0.
klab_siz	If non-zero, klab_siz is the height of key label characters in inches. If 0, SHADE selects a reasonable height; default is 0.
klab_inc	If non-zero every klab_inc key level is labelled; if 0, SHADE selects a suitable value. Default value is 0.
klab_dig	is the number of significant digits (klab_dig > 0) or decimal places (klab_dig < 0) in the key. Default is 3.
klab_len	is the maximum number of characters in a key label. Default is 9.

kx_lo	X-coordinate of the left side of the key, in inches.
kx_hi	X-coordinate of the right side.
ky_lo	Y-coordinate of the bottom of the key, in inches.
ky_hi	Y-coordinate of the top.

Example:

```
SHAKEY 1, 1, 0, 3, 4, 8, 9.4, 10.2, 1.4, 7.4
```

Specifies that the SHADE command draw a vertical key with label size selected automatically and every third color of the key labelled. Labels will contain 4 significant digits to a maximum of 8 digits. The entire key will occupy a rectangle from (9.4,1.4) to (10.2,7.4) in inches.

SHASET
SHASET set_pt, red, green, blue
SHASET SAVE=spknme
SHASET SPECTRUM=spknme
SHASET DEFAULT

This command sets the colors used in the plot generated by the SHADE command, and takes several forms. By default the shaded plots use a spectrum of color from blue to red. A user-defined selection of colors can be chosen, saved for future use, and recalled by name. Monochrome devices are not affected by this command.

SHASET uses the following approach in defining a spectrum. The levels set by the LEV command have lower and upper bounds; SHASET defines a scale from 0 to 100 which spans the interval defined by these bounds. With SHASET you may specify a color at any point along the scale by setting a control point on the scale, along with the red, green, and blue fractions (between 0 and 100% of maximum intensity) defining the color at that point.

A spectrum is built up by setting colors at a number of points. The SHADE routine linearly interpolates each red, green, and blue fraction between set points to achieve a smooth transition from point to point.

When you are interested in creating a custom spectrum to use in SHADE plots, first execute SHASET with no arguments. This clears all set points except the bottom and top of the scale; the bottom, at zero, is set to black (red, green, and blue fractions all 0% of maximum intensity). The top, at 100, is set to white (with red, green, and blue all 100%).

SHASET can then be used to set any point in the scale to any color. When defining set points, the colors are realized at execution of the SHASET command if a SHADE plot is on the screen.

set_pt	defines a point in a scale, from 0 to 100, that spans the levels as set in the LEV command. Set_pt can be negative; when set negative, SHASET deletes this set point from the current spectrum, if present. RGB values need not be specified in this case.
red	The intensity of red in the color at the set point, with a value between 0 and 100%.
green	The intensity of green.
blue	The intensity of blue.

You may save a spectrum for later use using the SAVE form of the command, where you give the present spectrum a name. A spectrum can be recalled using the SPECTRUM form of the command. One spectrum, RNB, the PPL+ default spectrum, is always available for recall.

spknme	A name to be associated with a particular spectrum. The spectrum is stored in the current directory as spknme.SPK, and is an ASCII file, which can be edited.

Finally, the DEFAULT form will set the spectrum to the default colors associated with the workstation in use. This is not the PPL+ default but is defined for the device by GKS.

APPENDIX I: LINE BUNDLES ON SUPPORTED DEVICE TYPES

GKS employees the concept of line bundles, where lines may be referred to by an index; the index determines the three characteristics of plotted lines, namely, line type, thickness, and color. Line type means whether the line is solid, dotted, dashed, dashed-dotted, etc. Thickness is measured in units beginning with one; two is twice as thick, followed by three, three times as thick as one, and so on. Colors are indexed by sequential integers beginning with zero.

The values of these characteristics together determine the line representation. One result of bundle use is that the same index may be defined to have different representations on different devices. Each graphics device type has a maximum number of each of these attributes that can be specified, and not all device types support the same attribute values. The following tables of attribute indices and their interpretation are used below in line bundle definitions.

For line type:

1	solid
2	dashed
3	dotted
4	dashed-dotted
-1	dashed and double-dotted
-2	dashed and triple-dotted
-3	long dashed
-4	long and short dashed
-5	double spaced dashed

For thickness:

1	single thickness
2	double
3	triple
...	and so on

For color:

0	black or white, depending on the device type
1	white or black, as above
2	red
3	green
4	blue
5	cyan
6	magenta
7	yellow
8	orange
9	purple
10	pink
11	brown
12	olive
13	beige
14	dark grey
15	light grey

The various devices use combinations of these attribute values to determine a line. A bundle index, selected using the PEN command, may not determine the same line on two different devices. A monochrome device will not use color indices other than zero and one, for example. The line bundle index definitions for the supported devices are explained below:

VSII, monochrome: uses only one color -- index one. Uses eight line types: 1,2,3,4,-3,-4,-5,-1 in that order, with thickness one. Then uses them over with thickness two, and so on to thickness ten, for a total of 8*10=80 line bundle indices.

VSII, color: uses seven color indices, one through seven in that order, the same eight lines types as the monochrome workstation, and four thicknesses. Color cycles first, then line type, so that there are 7*8=56 lines with thickness one. Thickness is then incremented; a total of 4*7*8=224 line bundle indices are available.

TEK4107: a similar progression of color, line type, and thickness is gone through. Color indices used are 1,3,6,9,11,13,0, in that order. Line types used are 1,2,3,4,-3,-4,-5,-2; four thickness values are used. Because the 4107 has a total of only 16 color indices available, colors set by the SHASET command can affect the color value of the line color indices. If the default SHASET argument is set, the colors will correspond to the above table. If not, colors two through 15 will match colors in the spectrum created.

TEK4014: One color only -- one -- is used; there are five line types, 1,2,3,4,-1, and ten thicknesses used, for a total of 5*10=50 indices.

LN03P: There are eight line bundle indices with color one and thickness one, using these line types: 1,2,3,4,-3,-4,-5,-1. Additional indices use only five line types, 1,2,3,4,-1, and ten heavier thicknesses, for a total of 8+5*10=58.

PS: One color, eight line types, 1,-5,3,4,-3,-1,2,-4, and 10 thicknesses; 8*10=80 indices.

APPENDIX II: SHADE COMMAND COLOR, PATTERN AND HATCHING

The internal GKS routine that generates the shaded plots is the fill area call; the areas filled are grid cells. GKS permits use of bundles to determine the representation of a fill area. The advantage of bundles in this context is that on color workstations the colors associated with indices can be changed dynamically. In practice, this means that after a SHADE plot is drawn, the SHASET command may be used to tune the spectrum of colors to your requirements. The changes take place immediately and the plot does not need to be redrawn.

Monochrome devices can not use color to distinguish field values, so patterns or hatching must be used. The supported devices respond to SHADE calls in the following ways.

VSII, monochrome: a grey scale consisting of up to 15 patterns, shading from black to white. This means 15 levels may be distinguished; if more than 15 are depicted, adjacent levels may be represented by the same pattern.

VSII, color: by default up to 64 colors may be set to distinguish field values. The shades can be set using the SHASET command.

TEK4107: The 4107 is capable of producing only 16 colors. The SHADE command uses 14 of these; black (0) and white (1) are not used. SHASET will set these 14 colors.

TEK4014: Up to nine different hatch patterns can be used with the SHADE command. Setting more than nine levels means having some adjacent field values represented by the same pattern.

LN03P: Up to nine different hatch patterns can be used with the SHADE command. Setting more than nine levels means having some adjacent field values represented by the same pattern.

PS: Up to nine different patterns forming a grey scale can be used. Use of more than nine patterns will cause some adjacent field values to use the same pattern.

☆ U. S. GOVERNMENT PRINTING OFFICE: 1992 / 674-568