Introduction to scientific visualisation
Part 3

Petra Heil
Australian Antarctic Division
&
Antarctic Climate and Ecosystems CRC
University of Tasmania
Hobart, TAS 7001
Australia

October 16, 2006

This course is designed to introduce students of environmental science studies (e.g., oceanography, atmospheric sciences, environmental science) to *state of the art* computer-based graphic tools for scientific analysis as well as digital presentation of derived results. The course emphasis is on presenting available software tools and how to use and apply these in academic research where the focus is on computational fluid dynamics. Not all software packages currently available have been included in this course, instead the it is attempted to provide the student with an overview of the various types of software available and their potential benefit within fluid dynamics applications.

Course objectives are to:

- analyse and interpret large data sets derived either from experiments or computer simulations;
- visually summarise multi-dimensional properties (e.g., extrema, eigenvalues) of the data set;
- understand application and virtue of discussed software tools;
- prepare a well-structured project using the student's data set.

Syllabus

Part 1:

- 1. Background and introduction.
- 2. Graphical presentation of data.
- 3. Basic visualisation algorithms.
- 4. Volume visualisation.

Part 2:

- 5. Visualisation systems: a) Matlab.
- 6. Visualisation systems: b) IDL.

Part 3:

- 7. Visualisation systems: c) GMT.
- 8. Visualisation systems: d) Ferret.
- 9. Visualisation systems: e) Vis5d+.
- 10. Other application examples and useful data sources.

Chapter 1

Specialized visualisation packages

Visualisation software can be divided into three basic groups:

- Modular visualisation environments (MVE), complete environments for the development of visualisation.
- Software toolkits, such as libraries, that can be helpful when creating visualisation applications.
- Specialised visualisation software.

Modular visualisation environments
The user has direct control of the visualisation process, often by interacting with a visual programmable interface. Software in this category are:

- AVS/Express
- OpenDX
- IRIX Explorer (from NAG)

Toolkits
The user can use toolkits to develop a number of different visualisation applications by using various visualisation techniques. An example is VTK[1], an open-source visualisation toolkit for 3D computer graphics, image processing, and visualisation.

Specialised visualisation software
These specialised softwares are generally produced for a certain type of data visualisation, and application might be limited.

[1] http://public.kitware.com/VTK/

- Vis5d
- PCmodel

1.1 GMT

The *Generic Mapping Tool*[2] is a free, public-domain tool that allows the visualisation of two and three-dimensional data by manipulating two- and three-dimensional data set. The manipulation procedures include griding, map projections, filtering etc. GMT provides data on coastline and rivers and supports 25 map projections. The graphical output from GMT is stored in form of an (encapsulated) post-script file.

The *Generic Mapping Tool* has been developed by Pål Wessel[3] and Walter Smith[4]. GMT is, in part, supported by the National Science Foundation of the USA. It is released under the GNU General Public License[5]. For the most current version and information on new releases and fixes see the *GMT* release notes[6].

1.1.1 *GMT* commands

GMT **wrapper:**	
fixgmtdefs	Script to make file gmt 3.0 compatible.
gmt	Wrapper for *GMT* programs.
gmtdefaults	Lists current *GMT* defaults.
gmtset	Edits settings in the .gmtdefaults file.
grdinfo	Gets information about the contents of a grd file.
makecpt	Makes *GMT* colour palette tables.
minmax	Gets extreme values in data tables.
spectrum1d	Computes auto- (and cross-) spectra from one (or two) time-series.

[2]http://gmt.soest.hawaii.edu/
[3]http://www.soest.hawaii.edu/pwessel/pwessel.html
[4]http://ibis.grdl.noaa.gov/SAT/people/walter.html
[5]http://www.gnu.org/copyleft/gpl.html
[6]http://gmt.soest.hawaii.edu/gmt/gmt_releases.html

Plotting of 1- and 2-dimensional data:	
grdcontour	Contours a 2-D gridded data set.
grdimage	Creates grayshaded or coloured image from a 2-D netCDF grd file.
grdvector	Plots vector fields from grdfiles.
psbasemap	Plots PostScript base-map.
psclip	Sets up polygonal clip paths.
pscoast	Plots land-masses, water-masses, coastlines, borders, and rivers.
pscontour	Contours xyz-data by direct triangulation.
pshistogram	Bins data and plot histograms.
psimage	Plots Sun raster files on a map.
psmask	Clips or mask areas of no data on a map.
psmegaplot	Creates poster-size PostScript plots from page-size plot.
psrose	Plots (length, azimuth) as sector or rose diagram or polar histogram.
psscale	Creates gray scale or color scale for map.
pstext	Plots text string on map.
pswiggle	Plots anomaly along track on a map.
psxy	Plots lines, polygons, and symbols on map.
psxyz	Plots lines, polygons, and symbols in 3-D.

Gridding of 1- and 2-dimensional data:	
nearneighbor	A "Nearest neighbor" griding algorithm.
surface	Adjustable tension continuous curvature surface griding algorithm.
triangulate	Performs optimal Delauney triangulation and griding.

Map-transformations and grid projections:	
grdproject	Projects gridded data on a new coordinate system.
mapproject	Forwards map transformation of 2-D coordinates.
project	Projects data along a line or great circle.

Filtering of 1- and 2-dimensional data:	
blockmean	Filter to block average (x,y,z) data by L2 norm.
blockmedian	Filter to block average (x,y,z) data by L1 norm.
filter1d	Time domain filtering of 1-D time series.
grdfilter	Filters a .grd file in the Time domain.
splitxyz	Filter to divide (x,y,z[,distance,heading]) data into (x,y,z) track segments.

Deriving trends in 1- and 2-dimensional data:	
fitcircle	Finds mean position and pole of best-fit great (or small) circle to points on a sphere.
grdtrend	Fits a polynomial trend to a grd file.
sample1d	Resamples 1-D data sets
trend1d	Fits a (weighted) polynomial or Fourier model for y = f(x) to xy[w] data.
trend2d	Fits a (weighted) polynomial model for z = f(x,y) to xyz[w] data.

Further 2-dimensional grid functions:	
grd2cpt	Reads a grdfile and make a colour palette file.
grd2xyz	Converts a grdfile to an ASCII or binary table.
grdclip	Clips the range in grdfiles.
grdcut	Extracts a subregion out of a .grd file.
grdedit	Modifies the header in a .grd file.
grdfft	Performs mathematical operations on grdfiles in the freq. domain.
grdgradient	Computes directional gradient from .grd files.
grdhisteq	Histogram equalisation for .grd files.
grdlandmask	Creates mask .grd file from coastline data base.
grdmask	Creates mask grdfiles from xy paths.
grdmath	Reverses Polish calculator for grd files.
grdpaste	Pastes together two .grd files along a common edge.
grdreformat	Converts between different grdfile formats.
grdsample	Resamples a grd file onto a new grid.
grdtrack	Samples of a 2-D grdfile along 1-D trackline.
grdvolume	Calculates volume under a surface within a contour.
xyz2grd	Converts an ASCII or binary table to grd file format.

1.1.2 Data analysis with GMT

GMT may be used for the signal analysis of data series. Routines are available to pre-process the raw data (filter1d, sample1d, or spectrum1d) and to project sparse data on an even grid. Figure 1.1 was produced to compare the surface air temperature at two coastal Antarctic stations for the year 1973. (Temperature data are courtesy of the Australian Bureau of Meteorology and have been obtained through BAS metlog data base[7].)

The GMT script used to produce figure 1.1, including all the data analysis required, is listed in Appendix A.

1.1.3 Mapping with GMT

GMT provides the user with a range of mapping options, including numerous geographical projections.

[7]http://www.antarctica.ac.uk/cgi-bin/metdb-form-1.pl

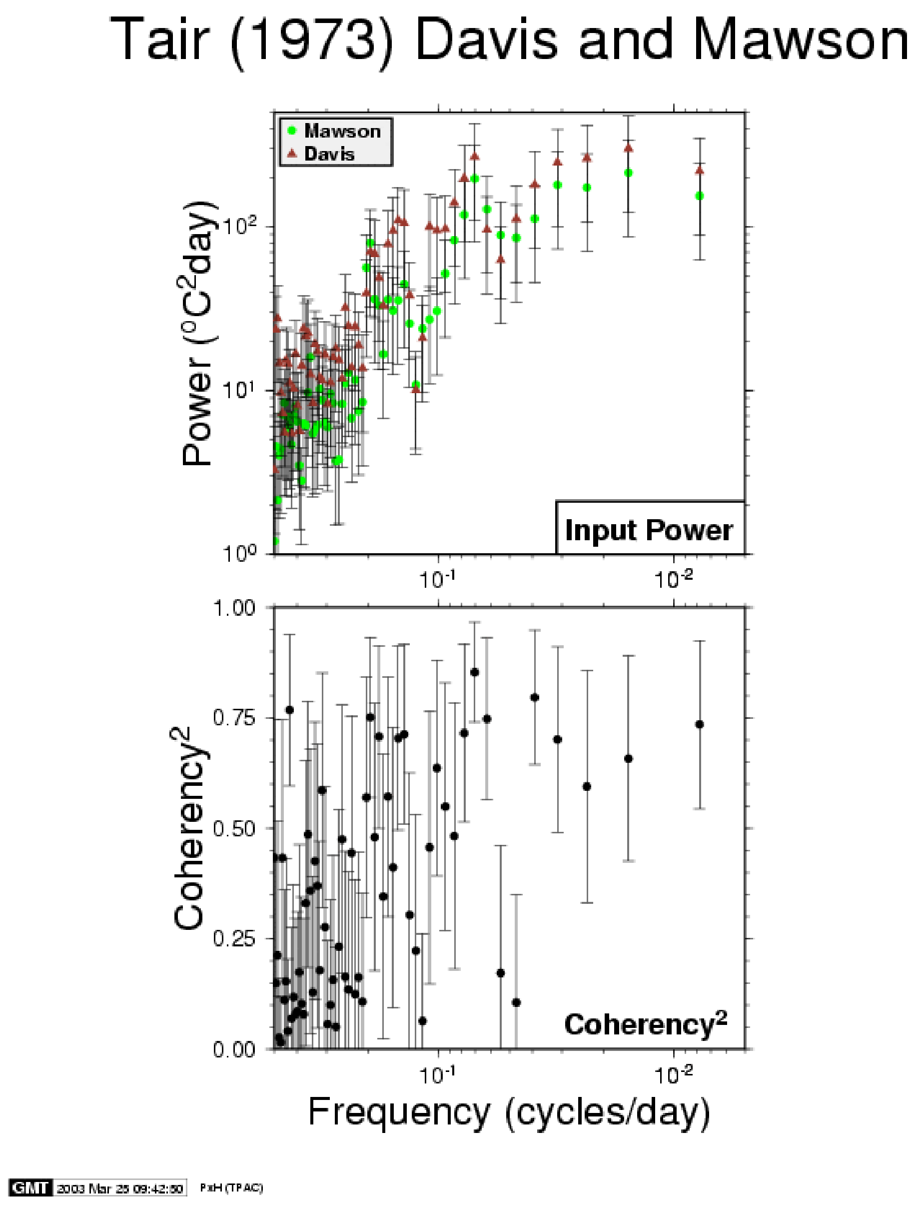

Figure 1.1: Surface air temperature at Davis and Mawson stations for 1973.

Geographical projections with GMT

GMT supports a range of geographical projections:
(Note: The following list does not give the complete list of GMT-supported projections. Please refer to the GMT man pages.)

- Cylindrical projections:
  - Cassini (-Jc*lon0/lat0/scale*)
  - Mercator (-Jm*scale* or -Jm*lon0/lat0/scale*)
  - Plate Carree (-Jq*lon0/scale*)
  - Transverse Mercator (-Jt*lon0/scale*)
  - Universal transverse Mercator (-Ju*zone/scale*)
- Azimuthal projections:
  - Lambert (-Ja*lon0/lat0/scale*)
  - Equidistant (-Je*lon0/lat0/scale*)
  - Orthographic (-Jg*lon0/lat0/scale*)
  - General stereographic (-Js*lon0/lat0/scale*)
- Conic projections:
  - Albers (-Jb*lon0/lat0/lat1/lat2/scale*)
  - Lambert (-Jl*lon0/lat0/lat1/lat2/scale*)
- Non-geographical projections:
  - Linear, logarithmic and power scaling (-Jx*x-scale[l—ppow][/y-scale[l—ppow]]*)
  - Linear projection for polar (theta,r) coordinates (-Jp*scale*)
- Miscellaneous projections:
  - Hammer (-Jh*lon0/scale*)
  - Sinusoidal (-Ji*llon0/scale*)
  - Eckert VI (-Jk*lon0/scale*)
  - Robinson (-Jn*lon0/scale*)
  - Winkel Tripel (-Jr*lon0/scale*)
  - Mollweide (-Jw*lon0/scale*)

References:

For further information on the Generic mapping tool, please refer to these links:

Wessel, P., and W. H. F. Smith, *The Generic Mapping Tools (GMT) version 3.0 Technical Reference & Cookbook*, SOEST/NOAA, 1995.
(Updated version: The Generic Mapping Tools Version 4 Technical Reference and Cookbook, version 4 beta[8].)

Wessel, P., and W. H. F. Smith, New Version of the Generic Mapping Tools Released,*EOS Trans.*, **76**, 329, 1995.
(WWW document[9], Copyright 1995 by the American Geophysical Union.)

Wessel, P., and W. H. F. Smith, Free Software Helps Map and Display Data, *EOS Trans.*, **72**, 441, 1991.

[8]http://gmt.soest.hawaii.edu/gmt4/gmt_services.html

[9]http://www.agu.org/eos_elec/95154e.html

1.2 NCAR Graphics or GrADS

In the climate research community the following packages are commonly used to display 4-dimensional data consisting of 3 spatial coordinates (longitude, latitude, vertical level) and time as the 4th coordinate: NCAR Graphics[10] and GrADS[11].

1.2.1 GrADS

Climate and atmospheric data can be accessed, displayed and analysed using the Grid Analysis and Display System (GrADS), a free-ware[12] available for a number of operating systems as an interactive tool. Accepted input formats are binary, GRIB, NetCDF or HDF-Scientific Data Sets. Gridded data may be regular, of variable resolution, non-linearly spaced, or Gaussian.

GrADS commands[13] are similar to Fortran expressions, and are entered at the command line. Pre-built functions are available, and additional functions can be added as external programs. The GrADS scripting language[14] allows detailed analysis and visualisation of the data sets.

The user interface is programmable, and scripts may be provided to it for execution.

Starting GrADS and the help function

From the Unix prompt GrADS is started up by typing **grads**. The user will be prompted for portrait or landscape display mode, and the user interface will be started up.

Further tutorials

http://grads.iges.org/grads/gadoc/tutorial.html
http://www.egs.uct.ac.za/workshops/mm5/grads_exercise.html

[10]http://grads.iges.org/grads/
[11]http://grads.iges.org/grads/
[12]http://grads.iges.org/grads/downloads.html
[13]http://grads.iges.org/grads/gadoc/reference_card.pdf
[14]http://grads.iges.org/grads/gadoc/reference_card_scl.pdf

1.3 Ferret

A further interactive display and analysis software is provided by Ferret[15]. It is preferentially used by the oceanographic scientific community to analyse large (3- or 4-dimensional) data sets and to produce high-quality graphical output. Ferret is free-ware with binaries[16] available for a number of operating systems.

1.3.1 Introduction to Ferret

Similar to Matlab (where data are presented in the form of regular matrices), Ferret assumes that all data are defined on regular grids. Ferret relies on grid information to address data in space, time or other data dimensions.

Data files may be loaded up into Ferret using the **set data** or the **use** commands, followed by the data-file name. The **let** command may be used to create new variables within Ferret.

To obtain Ferret output the user needs to specify a data set and a data region, to define a data variable or expression, and to request an output mode.

1.3.2 Starting up Ferret

To run ferret under the Unix operating system, type **ferret** at the command line prompt. The active window will return the Ferret prompt (**yes?**).

To make sure that correct paths are set for Ferret, it is necessary to source these before starting up Ferret. The *ferret_paths* file can be customised by the user.

An example for a *ferret_paths* file is given here:

```
# ferret_paths_template
# Template for setting up the FERRET environment
#
# If you are installing FERRET for the first time at your site we ask that
# that you please notify us with an Internat mail message to ferret@noaapmel.gov
# [192.68.161.0] (or call Steve Hankin at (206)526-6080) - no strings attached.
# Please specify if you would like to be on a mailing # list for FERRET updates.
#
# Modify this to suit your system and place the result as file ‘ferret_paths’ in
# another directory set up to hold login initialization scripts. ‘/usr/local’ is
```

[15]http://ferret.wrc.noaa.gov/Ferret/

[16]http://ferret.wrc.noaa.gov/Ferret/Downloads/ferret_downloads.html

```
# the suggested choice. Users of FERRET should include the command
#  `source /usr/local/ferret_paths' in their .login file.
#
# The environment variable FER_DIR should be the pathname of the directory named
# `ferret' you created to install the FERRET software.

# You may want to customize the pathname of the `ferret' directory:
         setenv FER_DIR /usr/local/ferret

# The environment variable FER_DSETS should be the pathname of
# the directory named `fer_dsets' you created to install the FERRET program.

# !!!! You must customize the following line:
# This directory will contain FERRET demonstration data files (30+ Mbytes)
         setenv FER_DSETS /usr/local/ferret/fer_dsets

# System Manager: Check this PATH modification for your system security.
# If you prefer not to modify PATH here you may comment out the following few
# lines and execute the file $FER_DIR/bin/install_ferret_links wich will
# create ferret links in /usr/local/bin.
# This logic will replace any previous $FER_DIR in PATH so this
# file may be sourced repeatedly in a single session
   if ( $PATH !~ *ferret* ) then
       setenv PATH {$PATH}:$FER_DIR/bin
   else
        echo replacing ferret path
      setenv PATH `echo $PATH | awk -F: '{for (i=1; i<=NF; i++)
                   {if ($i \!~ /ferret\/bin/) {printf "%s:",$i}}}'`
      setenv PATH {$PATH}$FER_DIR/bin
   endif
# *** end of PATH modifications
# =========== Initially make no modifications below this line ===========
#  Default Ferret document browser
   setenv FER_WEB_BROWSER "netscape -ncols 60"
   setenv FER_EXTERNAL_FUNCTIONS "$FER_DIR/ext_func/libs"
   setenv FER_GO        ". $FER_DIR/go $FER_DIR/examples $FER_DIR/contrib"
   setenv FER_DATA      ". $FER_DSETS/data $FER_DIR/go $FER_DIR/examples
                           $FER_DIR/contrib"
   setenv FER_DESCR     ". $FER_DSETS/descr"
   setenv FER_GRIDS     ". $FER_DSETS/grids"
   setenv TMAP            "$FER_DIR/fmt"
```

```
   setenv PLOTFONTS       "$FER_DIR/ppl/fonts"
   setenv SPECTRA         "$FER_DIR/ppl"         # for old ferret versions
   setenv FER_PALETTE     ". $FER_DIR/ppl"       # palette search list
# SunOS uses shared libraries
if ( `uname` == SunOS ) then
#  IF USER DOES NOT HAVE LD_LIBRARY_PATH POINT TO X11
   if (! $?LD_LIBRARY_PATH) then
      setenv LD_LIBRARY_PATH "/usr/openwin/lib"
   endif
endif

# Mod for AIX Ferret users -- needed for namelist reads
if (`uname` == AIX) then
   setenv XLFRTEOPTS namelist=old
   setenv LANG C
endif

# Faddpath tool to add another path to the search lists quickly
#     usage: Faddpath new_path
alias Faddpath 'if \!$ != Faddpath setenv FER_GO "$FER_GO \!$";
                if \!$ != Faddpath setenv FER_DATA "$FER_DATA \!$";
                if \!$ != Faddpath setenv FER_DESCR "$FER_DESCR \!$";
                if \!$ != Faddpath setenv FER_GRIDS "$FER_GRIDS \!$"'
```

1.3.3 Help in Ferret

At the **ferret** command line help is available. Type:

show commands	to obtain a list of Ferret commands type
show functions	to obtain a list of Ferret functions type
show transform	to obtain a list of Ferret transforms type

A comprehensive guide on **ferret** is available online[17].

[17] http://www.ferret.noaa.gov

1.3.4 Graphics generation

Setting a graphics window

In Ferret up to eight graphics windows may be opened and used at any time. The user activates the various windows via:

```
yes? set window n_id
```

where *n_id* is a number between 1 and 8.

The windows can be created at various sizes and aspect ratios:

```
yes? set window/size=0.5/aspect=0.5 n_id
```

A graphics window may be subdivided by using the **viewport** command. Options are:

```
yes? set viewport upper
yes? set viewport lower
yes? set viewport left
yes? set viewport right
yes? set viewport ul
yes? set viewport ur
yes? set viewport ll
yes? set viewport lr
```

To reset the ferret graphics window to a full view use:

```
yes? clear viewport
```

Ferret commands

Data may be uploaded, manipulated and displayed using some common Ferret commands:

set data or **use**	Facilitates uploading of a data set
show data	Presents summary of a data set
show grid	Presents grid information
set region	Sets the region to be analysed
list	Presents a listing of loaded data
let	Defines a new data variable
plot	Produces a plot
contour	Produces a contour plot
fill	Produces a filled contour plot
shade	Produces a shade plot
vector	Produces a vector plot
polygon	Produces a plot of polygonal regions
define	Allows definition of new symbols, axes, or grids
statistics	Presents summary statistics of data variables and expressions
save	Writes data into a netCDF file
go	Executes Ferret commands given in a journal file

2D plots

A sine plot equivalent to the one presented in the Matlab section (see part 1 of this course) is created by:

```
yes? define axis/x=0:6.2832:0.1 xaxis   ! Define new x-axis
yes? define grid/x=xaxis demo_2d        ! Define new grid
yes? set grid demo_2d                   ! Set current grid to
                                        ! user-defined grid.
yes? let p=1/15*sin(x[g=demo_2d]/2*3.1415)
yes? plot/title='Sine curve'/SYMBOLS=2/LINE/VLIMITS=-0.1:0.1:0.01 p
yes? set grid abstract                  ! Restore default index grid.
```

which leads to figure 1.2.

```
yes? let z2=exp(-2./i)*sin(1/25*3.1415*i)*cos(1/25*3.1415*j)
yes? let z3=exp(-2./k)*sin(1/25*3.1415*i)*cos(1/25*3.1415*j)
yes? set viewport ur
yes? contour/title='Contour plot'/i=1:25/j=1:25 z2
yes? set viewport ul
yes? shade/title='Shade plot'/i=1:25/j=1:25 z2
yes? set viewport ll
yes? fill/title='Fill plot'/i=1:25/j=1:25 z2
```

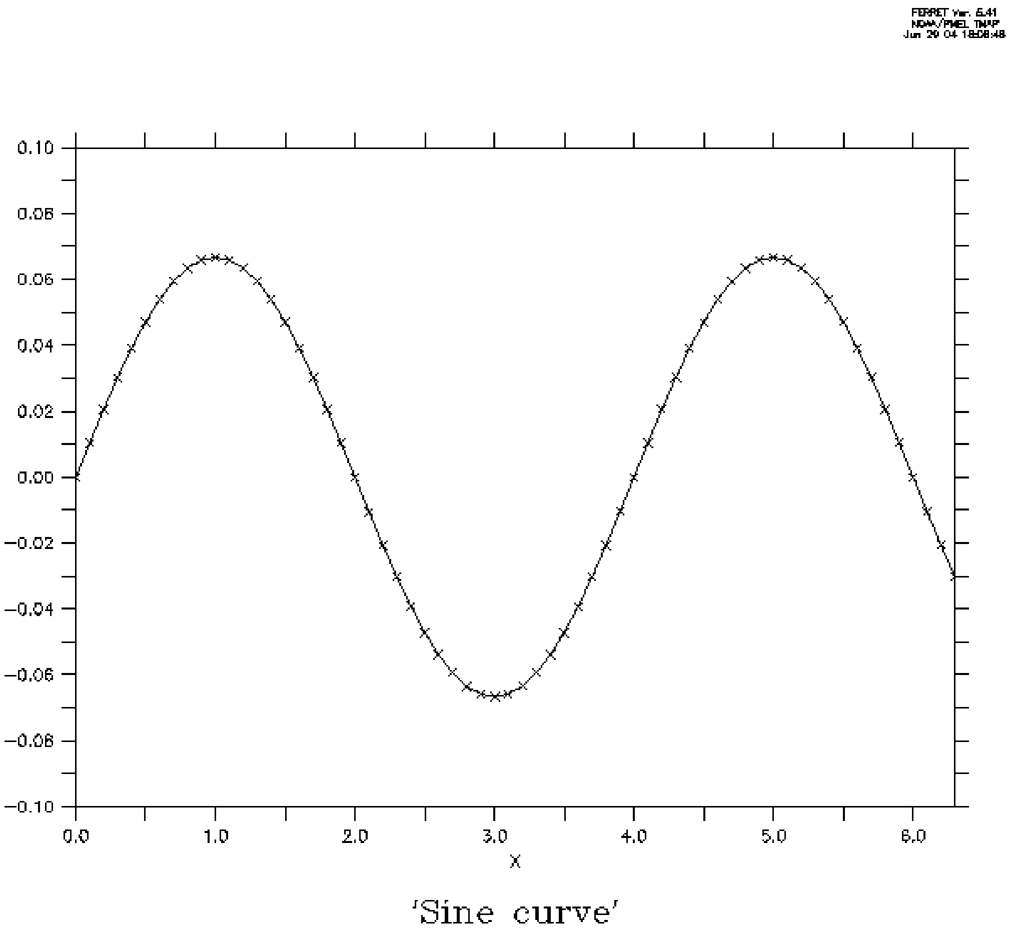

Figure 1.2: Simple sine plot done in Ferret.

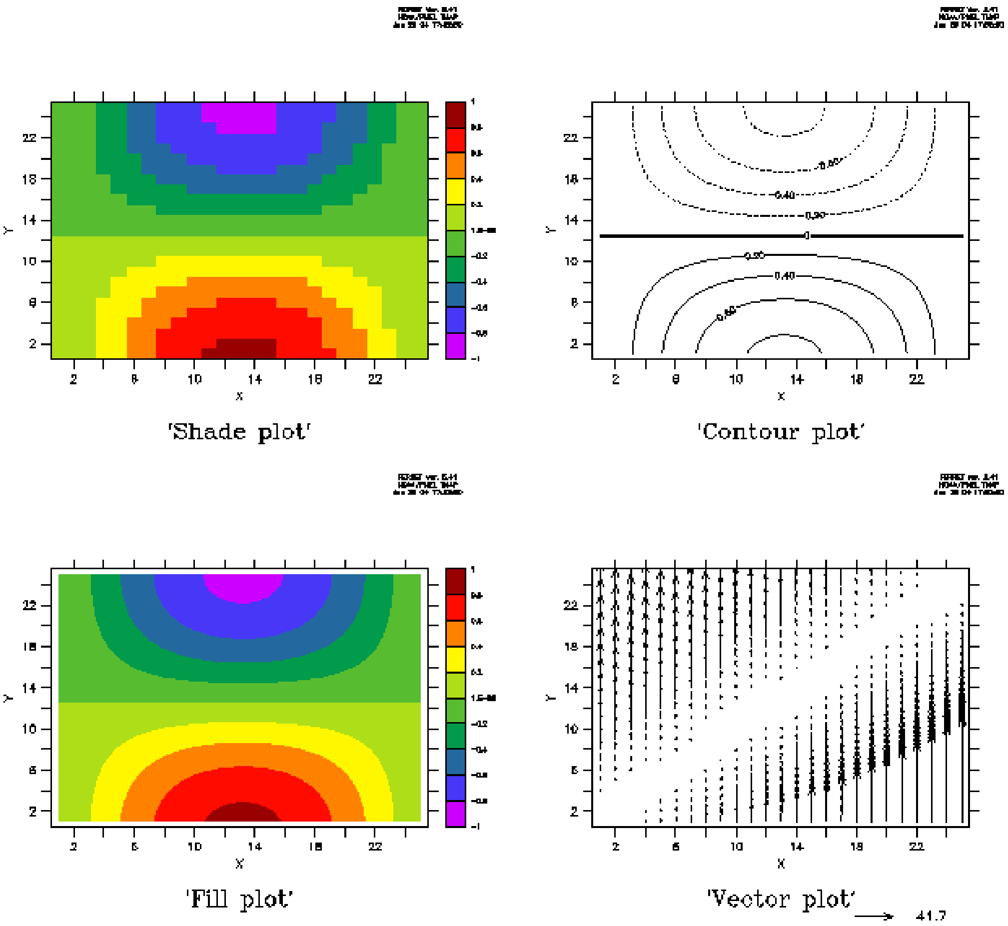

Figure 1.3: Simple contour plot done in Ferret.

```
yes? set viewport lr
yes? vector/title='Vector plot'/i=1:25/j=1:25 z2,(i-j)^2/5
```

which leads to figure 1.3.

3D plots

```
yes? cancel viewport                              ! Reset window to 1x1 figure
yes? define axis/x=-25:25:1 xaxis                 ! Define new x-axis
yes? define axis/y=-25:25:1 yaxis                 ! Define new y-axis
yes? define grid/x=xaxis/y=yaxis demo_3d          ! Define new grid
yes? set grid demo_3d                             ! Set current grid to
                                                  ! user-defined grid.
```

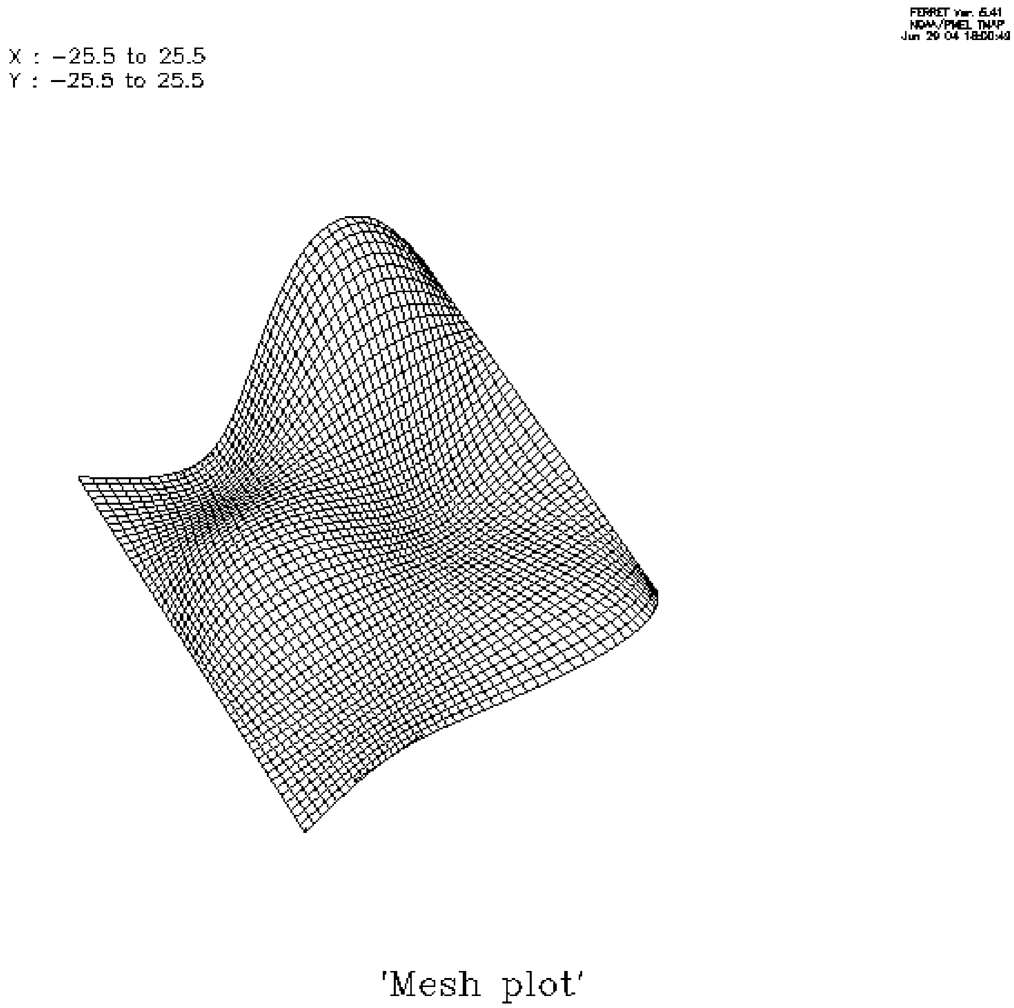

Figure 1.4: Mesh plot done in Ferret.

```
yes? let z3=sin(1/25*3.1415*x[g=demo_3d])*cos(1/25*3.1415*j[g=demo_3d])
yes? wire/viewpoint=-10,-15,5 z3          ! Produce a mesh plot.
```

which leads to figure 1.4.

In the following we present an example of displaying data from a NCEP/NCAR-2[18] file of short-wave radiation. Data for this example can be obtained from the NOAA-CIRES Climate Diagnostics Center (USA).

```
yes? use dswrf.sfc.gauss.2000.nc
```

[18] http://www.cdc.noaa.gov/cdc/data.ncep.reanalysis2.html

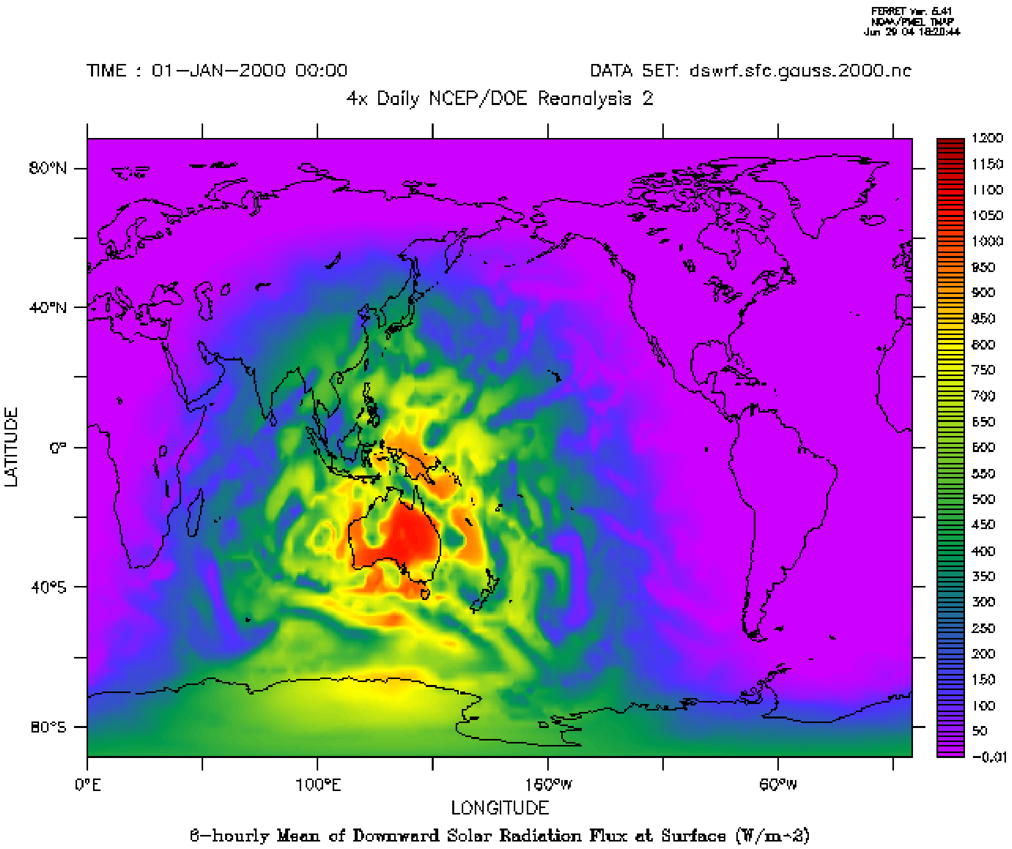

Figure 1.5: Example displaying data from NOAA-CIRES Climate Diagnostics Center (USA).

```
yes? show data                        # Display summary of loaded data sets
  currently SET data sets:
   1> ./dswrf.sfc.gauss.2000.nc  (default)
   name     title                             I      J      K      L
   TIME_BNDS
            Time Boundaries                  1:2    1:1    1:1    1:1464
   DSWRF    6-hourly Mean of Downward Solar  1:192  1:94   1:1    1:1464
yes? fill/lev=(-.01,1200,10) dswrf[l=1]
yes? go land                          # Add a land mask
```

which leads to figure 1.5.

Ferret provides simple means to derive sections across a 2D data set, e.g.:

```
yes? set window/aspect=0.55              # Reset aspect ratio of window
```

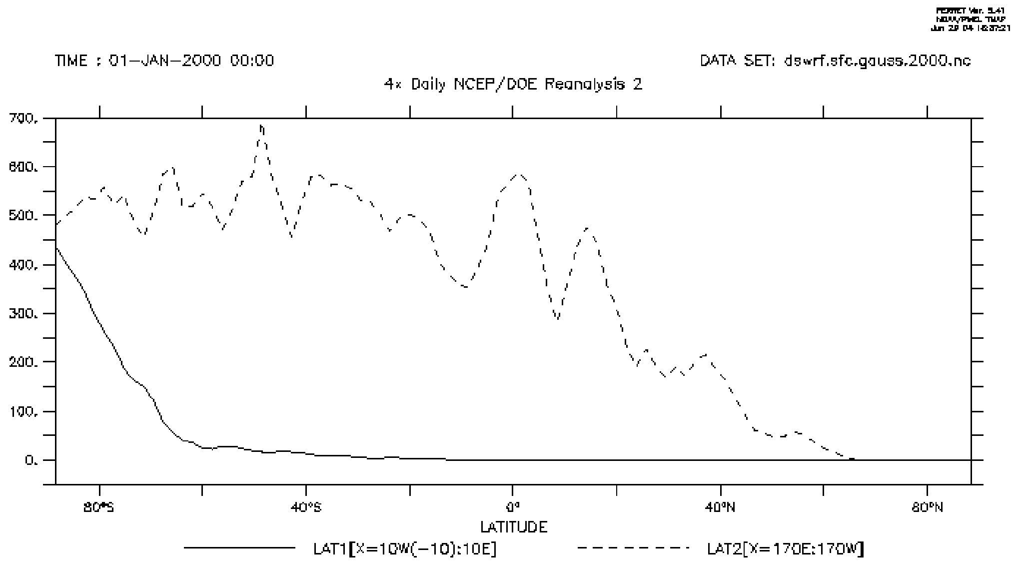

Figure 1.6: Example showing sections across a 2D data set.

```
yes? let lat1=dswrf[X=-10E:10E@AVE,l=1]  # Derive meridional average for
                                         # 20\degr band centered at the
                                         # equator at timestep l=1
yes? let lat2=dswrf[X=170E:190E@AVE,l=1]
yes? plot lat1,lat2
```

which leads to figure 1.6.

1.3.5 Journal files

The user can create his or her own scripts (journal files), which contain Ferret code and be executed from the Ferret command line using the **go** command.

User-created Ferret journal files may be:

- Scripts to perform an operation on data in the Ferret workspace.
- Functions that ingest one or more user arguments and return a set of output.

A simple example for a Ferret journal file, named *test.jnl* is shown here:

```
cancel mode verify
! Description: A simple example of a Ferret journal file.
! 28.06.2004
```

```
say   A demonstration of a Ferret journal file.
! Eliminate the program state left by previous activities
cancel variables/all          ! Delete all previous variables from session
cancel data/all               ! Delete all previous data from session
cancel region                 ! Reset region
cancel viewports              ! Reset window to 1x1 figure
set win/size=0.25             ! Set default window
set grid abstract             ! Set default grid
set mode calendar:days
set mode verify               ! Display command file lines at the screen
!-----------------------------------------
message

define axis/x=0:6.2832:0.1 xaxis    ! Define new x-axis
define grid/x=xaxis demo_2d         ! Define new grid
set grid demo_2d                    ! Set current grid to user-defined grid.
let p=1/15*sin(x[g=demo_2d]/2*3.1415)
plot/title='Sine curve'/LINE/VLIMITS=-0.1:0.1:0.01 p
!-----------------------------------------
message

plot/title='Sine curve'/SYMBOLS=2/LINE/VLIMITS=-0.1:0.1:0.01/symbols=2 p
!-----------------------------------------
cancel mode verify
! Reset Ferret
set mode/last verify
```

In Ferret this script is then called as

```
go test
```

1.3.6 Generating output data files

Ferret supports the generation of netCDF output, and appropriate axis and grid information may be generated:

The following provides an example for output of global 1 degree by 1 degree data variable (e.g., named *var1*) from Ferret into a netCDF data file:

```
yes? define axis/x=1:360:1/units=degrees xaxis
yes? define axis/y=1:180:1/units=degrees yaxis
```

```
yes? define grid/x=xaxis/y=yaxis var_grid
yes? let var1=exp(-2./i)*sin(1/25*3.1415*i)*cos(1/25*3.1415*j)
yes? contour/title='Contour plot'/i=1:360/j=1:180 var1
yes? set variable/title="Variable 1" var1
yes? save/file=ferret_test_output.nc var1
```

To append data to a netCDF output file, Ferret uses the **append** command:

```
yes? set variable/title="Variable 3" var3
yes? save/file=ferret_output.nc var3
```

Note that the data field named *var3* needs to be given on the same grid (called *var_grid*) as the previous data fields. I.e. in this example on a global 1 degree by 1 degree grid.

Ferret supports the following netCDF attributes:

- title
- history
- units
- long_name
- Fillvalue
- missing_value
- scale_factor
- add_offset
- x_axis
- y_axis
- z_axis
- time_axis
- my_axis

Further information on Ferret support for netCDF I/O may be obtained from either the Ferret home page[19] or from the Unidata[20].

Useful links for the use of Ferret may be found at:

- http://ferret.pmel.noaa.gov/Ferret/Documentation/ferret_tour.html
- http://ferret.wrc.noaa.gov/Ferret/Documentation/Users_Guide/current/fer_html.htm

[19]http://www.ferret.noaa.gov/noaa_coop/coop_cdf_profile.html
[20]http://www.unidata.ucar.edu/packages/netcdf/convention

1.4 Vis5d+

Vis5d+[21] is the enhanced version of Vis5d (Visualisation of 5 Dimensional data), a free OpenGL-based multidimensional visualisation program for scientific datasets in three and more dimensions. The program was developed to display output from atmospheric numerical models, which produce data fields of several time-dependent physical quantities, which extend over the three-dimensional space. It allows interactive visualisation, and includes splicing and volume rendering on a three-dimensional grid. Vis5D has the capability to animate temporally varying data, and includes tools to manage data grids and conversion of temporally varying three-dimensional data sets into Vis5D format. Vis5d has been developed but the University of Wisconsin, Madison.

Vis5D has been extended to a virtual reality version, the Cave5D[22], which can be used for application on the CAVE and ImmersaDesk. This version is currently under development, updates[23] are available.

1.4.1 Tutorial

Vis5D works on data in the form of a 5D matrix. The data must be real numbers, and available at all grid points, e.g. sparse matrices are not allowed. The five dimensions are composed of three spatial dimensions, one time dimension, and one dimension for numerical value of the parameter.

Visualising a file

To use Vis5d the data set needs to be available in a special Vis5d format, and calls to Vis5D utility functions need to be issued in a Vis5d program. On command line the application is activated by the call:

vis5d file.v5d [options]

Vis5d has a Gui control panel which is used to choose between the various visualisation methods and types.

Calling **vis5d** without arguments will produce a listing of all command line options and keyboard functions.

[21] http://vis5d.sourceforge.net/
[22] http://www-unix.mcs.anl.gov/ mickelso/CAVE2.0.html
[23] http://www-unix.mcs.anl.gov/ mickelso/CAVE2.0.html

Display features

- Isosurfaces
- Contours
- Coloured slices
- Trajectories
- Volume rendering
  This allows rendering of three-dimensional data fields using colour opacity.
- Vertical sounding, SkewT
- Data probing (in viewing mode)
- Text labels (in viewing mode)

Snapshot of the control panel

Vis5d has a GUI control panel which is used to choose between the various visualisation methods and types (see figure 1.7).

Viewing modes

- **Normal** Mouse-operated mode to rotate, zoom, and pan the graphics in the 3-D window.
- **Trajectory** Creates and displays trajectories.
- **Probe** Probes values of single grid points by moving the cursor through the 3-D grid.
- **Slice** Horizontal and vertical slices can be manipulated.
- **Sounding** Display a vertical sounding and SkewT at the location of the cursor.
- **Label** Create and edits text labels in a 3-D window.

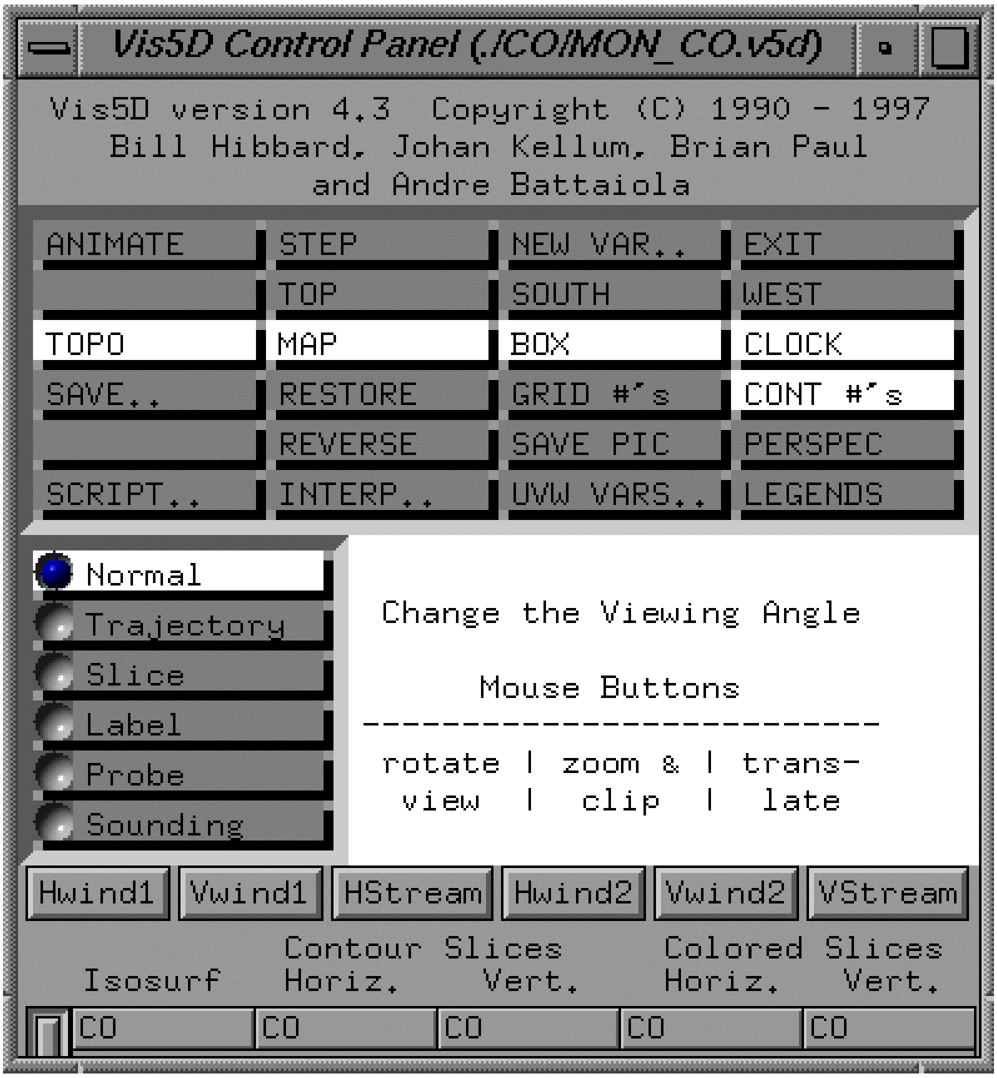

Figure 1.7: Vis5d GUI control panel.

Normal Mode
In the this mode mouse movement changes the viewing aspect of the three-dimensional data. The left mouse button controls the rotation of the three-dimensional image. The center mouse enables multiple functions. Pressing the center mouse button in combination with mouse motion to or away from the operator zooms the active scene out or in. Pressing the mouse button and moving it towards the right activates a *clipping plane* to move into the active scene.

Slice Mode

- Activations of the slice mode is by left-clicking of the *slice* widget button.
- This mode enables a plane view of a two-dimensional slice that is cut out of the three-dimensional data. The slice orientation is limited to horizontal or vertical, and can show isolines, filled countours, streamlines or vectors.
- Slice positioning can be changed interactively with the mouse.

1.4.2 HDF5 and tools for data conversion

Scientific data can be stored in HDF5, a general purpose library and file format. HDF5 differentiates between datasets and groups. A dataset is defined as a multidimensional array of data elements, while a group is defined as a structure for organising objects in an HDF5 file. Those two building elements are used to construct storage for any kind of scientific data, including data arrays, grid data, or even images. More details on HDF5[24] are available.

[24] http://hdf.ncsa.uiuc.edu/whatishdf5.html

1.5 AVS or OpenGM

The **Advanced Visualisation System** (AVS[25]) provides an interface to utilise multiple processors to enable a three-dimensional visualisation. Licenses are available for AVS 5, AVS/Express, and the AVS/Express Multipipe Edition. The AVS tools are designed to enhance the user's ability to integrate, analyse and present a variety of different data types. Plug-in modules written in Fortran or C++ are used to expand and customise user-specific applications. A library of such modules is available through the International AVS Center[26].

Please refer to Dr A.C. Pineda's page[27] for a short tutorial on AVS, or to the Texas Advanced Computing Center[28] for a more extensive guide.

[25] http://www.avs.com

[26] http://www.iavsc.org/

[27] http://www.hpc.unm.edu/~acpineda/AHPCC_Vis/html/OSCnAHPCCAVS/ahpcc-avs/mhpcc/maui_avs-Frame_11.html

[28] http://www.tacc.utexas.edu/resources/software/avs_5.5/tutguide.pdf

Chapter 2

Data sources

The internet provides access to a large number of useful data sources for scientists in marine or atmospheric research. Some of these data sites provide data under the DODS/OPeNDAP protocol.

2.1 Digital libraries

- NASA's Global Change Master Directory Portal (GCMD)[1] provides DODS/OPeNDAP related data sets for agriculture, atmosphere, biosphere, climate indicators, human dimensions, hydrosphere, land surface, oceans, paleoclimate, snow & ice, and spectral (engineering).
- The National Virtual Ocean Data System (NVODS)[2] provides data catalogues, collections and individual data provider sites.
- The TPAC Digital Library for Oceans and Climate[3] provides data archives of marine climatologies, satellite altimetry, Antarctic weather stations, and acts as a mirror for NCEP's[4] DOE Reanalysis 2 and WOCE 3.0.

[1] http://gcmd.gsfc.nasa.gov/Data/portals/dods/index.html
[2] www.nvods.org
[3] http://digitallibrary.tpac.org.au
[4] http://www.cdc.noaa.gov/cdc/data.ncep.reanalysis2.html

2.2 Coastline and topographic data

The National Geophysical Data Center [NGDC] provides shoreline & coastline data[5] for upload. Included data sets are:

- The British Oceanographic Data Centre's Gebco[6] bathymetry data.
- Coastline Generator[7] Supported data formats are:
  - Arc/Info Ungenerate
  - Mapgen
  - Matlab
  - Splus
- GSHHS - A Global Self-consistent, Hierarchical, High-resolution Shoreline Database[8]
  This data set has been developed by the GMT developers and provides data at various resolution (full, high, intermediate, low, and crude).
- TerrainBase[9]
- NOAA's Medium Resolution Digital Vector Shoreline Database[10]
- Coastal Relief Model at NGDC[11]

[5] http://www.ngdc.noaa.gov/mgg/shorelines/shorelines.html
[6] http://www.ngdc.noaa.gov/mgg/gebco/gebcoproducts.html
[7] http://rimmer.ngdc.noaa.gov/mgg/coast/getcoast.html
[8] http://www.ngdc.noaa.gov/mgg/shorelines/gshhs.html
[9] ftp://ncardata.ucar.edu/datasets/ds759.2/data/tbase.Z
[10] http://www.ngdc.noaa.gov/mgg/shorelines/noaamrdvs.html
[11] http://www.ngdc.noaa.gov/mgg/coastal/coastal.html

Appendix A

```
#/local/bin/bash
# FFT-routine with commandline and gmt commands.
#
# PxH, 06.03.2003
# Purpose: Take one year of meteorological data and derive the
#               power spectrum for atmospheric variables.
# GMT progs: filter1d, gmtset, minmax, sample1d,
#  spectrum1d, trend1d, pshistogram, psxy, pstext
# Unix progs: awk, cat, echo, head, paste, rm, tail
#
# This example begins with data files "ship.xyg" and "sat.xyg" which
# are measurements of a quantity "g" (a "gravity anomaly" which is an
# anomalous increase or decrease in the magnitude of the acceleration
# of gravity at sea level).  g is measured at a sequence of points "x,y"
# which in this case are "longitude,latitude".  The "sat.xyg" data were
# obtained by a satellite and the sequence of points lies almost along
# a great circle.  The "ship.xyg" data were obtained by a ship which
# tried to follow the satellite's path but deviated from it in places.
# Thus the two data sets are not measured at the same of points,
# and we use various GMT tools to facilitate their comparison.
# The main illustration (example_03.ps) are accompanied with 5 support
# plots (03a-f) showing data distributions and various intermediate steps.
#
# The minmax utility will report the minimum and maximum values for all columns.
# We use this information first with a large -I value to find the appropriate -R
# to use to plot the Davis_met_1973_3h.dat data.
# YYYY MM DD HH   DoY   Pres   Tair  Tdew  RH  Wspd  Wdir  ic  SWin SWinc
# 1973  1  1  0   1.00  994.4   0.0  -5.0  69   1.0 360.0   7 356.5 213.2
# 1973  1  1  3   1.12  994.0   0.0  -4.0  74   1.5 270.0   7 642.1 384.0
```

```
#
\rm Davis_Tair_1973_3h.dat
\rm Mawson_Tair_1973_3h.dat
awk '{print $5,$7}' Davis_met_1973_3h.dat >> Davis_Tair_1973_3h.dat
awk '{print $5,$7}' Mawson_met_1973_3h.dat >> Mawson_Tair_1973_3h.dat
cat Davis_Tair_1973_3h.dat Mawson_Tair_1973_3h.dat| minmax -I10/2 -C > $$
xmin=`awk '{print $1}' $$`
xmax=`awk '{print $2}' $$`
ymin=`awk '{print $3}' $$`
ymax=`awk '{print $4}' $$`
gmtset MEASURE_UNIT INCH
\rm GMT_FFT_Davis_Tair_1973a.ps
psxy -R$xmin/$xmax/$ymin/$ymax -JX8i/5i -Ba50f50:"Time (DoY [1973])":
   /a5f10:"Surface air temperature (degC)": -U/-1.75i/-1.25i
   /"PxH" -X2i -Y1.5i -K -W1p Davis_Tair_1973_3h.dat
   > GMT_FFT_Davis_Tair_1973a.ps
psxy -R -JX -O -W1/0/0/255tap Mawson_Tair_1973_3h.dat
   >> GMT_FFT_Davis_Tair_1973a.ps
#
# To facilitate comparison of the two with a cross-spectral analysis using
# "spectrum1d", we resample both data sets to get the temporal interval of
# sampling. Data spacing by using awk to get the delta-p between points and
# view it with "pshistogram".
\rm GMT_FFT_Davis_Tair_1973b.ps
awk '{ if (NR > 1) print $1 - last1; last1=$1; }' Davis_Tair_1973_3h.dat |
   pshistogram  -W0.1 -G0 -JX3i -K -X2i -Y1.5i -B:."Davis": -U/-1.75i/-1.25i
     /"PxH" > GMT_FFT_Davis_Tair_1973b.ps
awk '{ if (NR > 1) print $1 - last1; last1=$1; }' Mawson_Tair_1973_3h.dat |
   pshistogram  -W0.1 -G100 -JX3i -O -X5i -B:."Mawson":
   >> GMT_FFT_Davis_Tair_1973b.ps
#
# This experience shows that the sampling interval for both stations are spaced
# fairly evenly, with delta-p exactly 0.125\,day at Mawson and 0.125 and 0.2\,
# day at Davis. This means that when we want 0.125\,day even sampling,
# "sampled" can be used to interpolate the met data. We will want to use
# "paste" to put the two sampled data sets together, so they must start and
# end at the same point, without NaNs. So we want to get a starting and ending
# point which works for both of them. Thus we need to start at max( min(ship.p)
# , min(sat.p) ) and end # conversely. "minmax" can't do this easily since it
# will return min( min(), min() ), so we do a little head, paste awk to get
# what we want.
```

```
#
head -1 Mawson_Tair_1973_3h.dat > Mawson_Tair_1973_3h.dat.extr
head -1 Davis_Tair_1973_3h.dat > Davis_Tair_1973_3h.dat.extr
paste Mawson_Tair_1973_3h.dat.extr Davis_Tair_1973_3h.dat.extr |
   awk '{ if ($1 > $3) print int($1); else print int($3); }' > sampr1
tail -1 Mawson_Tair_1973_3h.dat > Mawson_Tair_1973_3h.dat.extr
tail -1 Davis_Tair_1973_3h.dat > Davis_Tair_1973_3h.dat.extr
paste Mawson_Tair_1973_3h.dat.extr Davis_Tair_1973_3h.dat.extr |
   awk '{ if ($1 < $3) print int($1); else print int($3); }' > sampr2
sampr1=`cat sampr1`
sampr2=`cat sampr2`
#
# Use sampr in awk to make a sampling points file for sample1d:
awk 'BEGIN { for (i='$sampr1'; i <= '$sampr2'; i++) print i }' /dev/null
   > samp.x
#
# Resample the projected metdata for the 2 stations:
#
sample1d Mawson_Tair_1973_3h.dat -Nsamp.x > samp_Mawson_Tair_1973_3h.dat
sample1d Davis_Tair_1973_3h.dat -Nsamp.x > samp_Davis_Tair_1973_3h.dat
#
# Plot the projected metdata to check that they are correct:
#
\rm GMT_FFT_Davis_Tair_1973c.ps
psxy -R$xmin/$xmax/$ymin/$ymax -JX8i/5i -Ba50f50:"Time (DoY [1973])":
   /a5f10:"Surface air temperature (degC)":
   -X2i -Y1.5i -K -W1p samp_Davis_Tair_1973_3h.dat -U/-1.75i/-1.25i/"PxH"
   > GMT_FFT_Davis_Tair_1973c.ps
psxy -R -JX -O -W1/0/0/255tap samp_Mawson_Tair_1973_3h.dat
   >> GMT_FFT_Davis_Tair_1973c.ps
## psxy-option: -Sp0.03i
#
# Now to do the cross-spectra, assuming that Davis has the input and Mawson is
# the output data:
tail -n364 samp_Davis_Tair_1973_3h.dat >samp_Davis_Tair_1973_3h.datt
##paste samp_Davis_Tair_1973_3h.datt samp_Mawson_Tair_1973_3h.dat |
##   cut -f2,4 | spectrum1d -S56 -D1 -C >& /dev/null
paste samp_Davis_Tair_1973_3h.datt samp_Mawson_Tair_1973_3h.dat |
   cut -f2,4 > tmpp
spectrum1d tmpp -S128 -D1 -C -V
#
```

```
# Plot the spectra.
# The following commands will plot the ship and sat power in one diagram and
# the coherency on another diagram, both on the same page.
#
\rm GMT_FFT_Davis_Tair_1973.ps
psxy spectrum.coh -Ba1f3p:"Frequency (cycles/day)":
   /a0.25f0.05:"Coherency@+2@+":WeSn -JX-4il/3.75i -R0.005/0.5/0/1 -U/-2.25i
   /-1.25i/"PxH (TPAC)" -P -K -X2.5i -Sc0.07i -G0 -Ey/2 -Y1.5i
   > GMT_FFT_Davis_Tair_1973.ps
echo "3.85 0.3 18 0.0 1 11 Coherency@+2@+" |
   pstext -R0/4/0/3.75 -Jx1i -O -K >> GMT_FFT_Davis_Tair_1973.ps
cat << END > box.d
9.375 0.45
2.4   0.45
2.4 0.00
END
psxy -R0.005/0.5/1/500 -JX-4il/3.75il -O -K -W1.5p box.d
   >> GMT_FFT_Davis_Tair_1973.ps
psxy -Sc0.07i -G0/255/0 -O -Ba1f3p/a1f3p:"Power (@+o@+C@+2@+day)"::
   ."Tair (1973) Davis and Mawson":WeSn spectrum.ypower -R0.005/0.5/1
   /500 -JX-4il/3.75il -Y4.2i -K -Ey/2 >> GMT_FFT_Davis_Tair_1973.ps
psxy -V spectrum.xpower -R0.005/0.5/1/500 -JX-4il/3.75il -St0.10i -G155/055
   /044 -O -K -Ey/2,color=100/155/100 >> GMT_FFT_Davis_Tair_1973.ps
##echo "3.9 3.6 18 0.0 1 11 Input Power" |
##   pstext -R0/4/0/3.75 -Jx -O -K >> GMT_FFT_Davis_Tair_1973.ps
echo "3.9 0.3 18 0.0 1 11 Input Power" |
   pstext -R0/4/0/0.75 -Jx -O -K >> GMT_FFT_Davis_Tair_1973.ps
psxy -R -Jx -O -K -W1.5p box.d >> GMT_FFT_Davis_Tair_1973.ps
psxy -R0/4/0/100 -Jx -O -K -G240 -L -W1.5p << END
   >> GMT_FFT_Davis_Tair_1973.ps
0.05 3.3
1.0 3.3
1.0 3.7
0.05 3.7
END
echo "0.15 3.6" | psxy -R -Jx -O -K -Sc0.07i -G0/255/0
   >> GMT_FFT_Davis_Tair_1973.ps
echo "0.25 3.6 12 0.0 1 5 Mawson" | pstext -R -Jx -O -K
   >> GMT_FFT_Davis_Tair_1973.ps
echo "0.15 3.4" | psxy -R -Jx -O -K -St0.10i -G155/055/044
   >> GMT_FFT_Davis_Tair_1973.ps
```

```
echo "0.25 3.4 12 0.0 1 5 Davis" | pstext -R -Jx -O
   >> GMT_FFT_Davis_Tair_1973.ps
# Clean up:
#\rm -f $$ box.d report samp* *.extr spectrum.* .gmtcommands
exit
```