OPeNDAP Matlab Interface Handbook
Version 1.15

Tom Sgouros

December 21, 2004

Preface

This document describes version 6.0 of the Open Source Project for a Data Access Protocol (OPeNDAP) Matlab Graphical User Interface (GUI), a system intended to allow researchers transparent access to earth science data—stored in any of several different file formats—across the Internet. The OPeNDAP Matlab GUI was originated by Deirdre Byrne and Peter Cornillon.

This document is relevant to the following software versions:

OPeNDAP Matlab GUI	6.0
OPeNDAP data access protocol	2.0
loaddods (Matlab command-line client)	3.4.5
loaddap (enhanced Matlab command-line client)	3.4.4

We are interested in feedback and bug reports about the OPeNDAP Matlab GUI Please send them to: support@unidata.ucar.edu.

Using the OPeNDAP Matlab GUI, a researcher can browse and retrieve earth science data from many different sources, importing it directly into Matlab, where it is available for display or for further manipulation. The OPeNDAP Matlab GUI provides a graphical environment within which a researcher can browse many datasets to find appropriate data, can select or subsample that data, and can display it as well, all by using a simple point-and-click user interface.

An overview of the system's use is presented, and specific tasks illustrated, for data providers as well as for users.

This document also covers the Open Source Project for a Data Access Protocol Matlab command-line clients, with which a researcher can download data directly into Matlab, for further analysis. The OPeNDAP Matlab GUI provides a graphical front-end to such a client, but you can use them directly as well, which can be of use in automating data analysis, for example.

0.1 Tasks Illustrated in this Guide

For a quick start to getting, installing, and using OPeNDAP Matlab GUI software, see the list below of tasks described in this document.

- Quick start. (page 2) (You may also appreciate *OPeNDAP Quick Start Guide*for a quick overview of the Open Source Project for a Data Access Protocol.)
- Getting and installing the OPeNDAP Matlab GUI software. (page 53)
- Installing the rest of the OPeNDAP software. (page 53)
- Updating the dataset list to include the most recent versions of all the datasets. (page 15)
- Reading your own datasets into the browser. (page 35)
- Using OPeNDAP from the Matlab command line. (page 10)

0.2 Who is this Guide for?

This guide is for people who wish to use Matlab to browse and retrieve data from existing on-line OPeNDAP datasets. Scientists who wish to share their data with colleagues may also find this a useful system, since it is a relatively simple matter to set up a server that can allow remote access to your data.

This documentation assumes that the readers are familiar with computers and the internet, but are not necessarily programmers. More than a passing familiarity with Matlab will be useful, as will an understanding of elementary internet concepts, such as URLs and http.

0.3 Organization of this Document

This book contains both introductory and reference material. There is also a description of the installation procedure.

Chapter 1 provides an overview of the entire system, including an introduction to its use, and some tutorial-style examples.

Chapter 2 Provides a reference to the menus and operations used in the OPeNDAP Matlab GUI.

Chapter 3 Describes how to add your own datasets to the OPeNDAP Matlab GUI.

Appendix A on page 53 Contains the instructions for installing the OPeNDAP Matlab GUI, and the related OPeNDAP software you may require.

0.4 Conventions

The typographic conventions shown in Table 1 are followed in this guide and all the other DODS documentation.

Table 1: Typographic Conventions

`Literal text`	Typed by the computer, or in a code listing.
`User input`	Type this precisely as written.
Variables	Used as a place holder for a user-specified or variable value. Choose an appropriate value and use that in place.
Button Text	Used to indicate text on a GUI button.
Menu Name	This is the name of a GUI menu.

When referring to a button in a menu, we will often use the notation: **Menu,Button**. For example, **Options,Colors,Foreground** would indicate the **Foreground** button in the Colors menu, selected under the Options menu.

Contents

List of Figures

List of Tables

1

The OPeNDAP Matlab Graphical User Interface

The OPeNDAP Matlab GUI combines OPeNDAP, a powerful data access protocol, with a convenient set of display functions, and the flexibility and power of the Matlab mathematics programming environment. The effect of the OPeNDAP Matlab GUI is to turn Matlab into a powerful network browser, capable of reading specialized data messages from the Internet. For this reason, we often refer to it as an Open Source Project for a Data Access Protocol "browser."

This chapter contains an overview of the browser, which will be useful for most browser users. It starts in Section 1.2 on page 3.

1.1 Quick Start

Some find it easiest to jump right in.

If you are already somewhat familiar with Matlab and OPeNDAP, you may find it simplest to jump right in. (You might want to consult *OPeNDAP Quick Start Guide* before this book, for a brief description of how OPeNDAP works.) After the OPeNDAP Matlab GUI has been installed, and the necessary environment variables set (See Appendix A on page 53), run Matlab.

```
$ matlab
```

When Matlab has started up, start the browser by typing:

```
>> browse
```

Three windows will pop up. One big one, for selecting geographic regions in which to search for data, and for displaying it, and two smaller ones. One of these is for selecting datasets to browe or display, and one for selecting the variables you want to display.

Use the mouse to select a dataset to look at. (Click once to selet a dataset. Click twice to open a dataset folder.) You'll see that the lists of available variables changes to list only those variables in that dataset. Move to the Variable window, and choose a variable from the list presented. Now, switch to the big main window and use the **Set Data Range** button to outline a geographic rectangle and a time and depth range to look at. Click **Get Details** to get an indication about where the data is, and some information you can use to estimate the size of the dataset you're requesting. Click **Get Data**, and follow the directions, and you should soon see a map of your data in the display window. The browser imports data in a consistent set of units, so that data from any source in the list will appear in the same units.

Subsequent requests for data will not overwrite the original ones. Each set of downloaded data is stored with a unique Matlabvariable name, so that many requests can be made in each Matlabsession.

You can also begin by setting a data range to specify a time period or geographic area of interest, and then selecting a dataset and variable, or the other way around. All of these parameters must be set before making a data request, but they can be set in any order.

You will find a useful reference to the OPeNDAP Matlab GUI buttons and menus in Chapter 2.

1.2 What is the OPeNDAP Matlab GUI?

The OPeNDAP Matlab GUI display consists of a map, a timeline, a depth range, a small array of pushbuttons, and several drop-down menus (Figure 1.1).

You can use the OPeNDAP Matlab GUI for four distinct operations: setting the parameters of a request for data, querying a dataset about its contents, issuing a request for data, and displaying the results of a request.

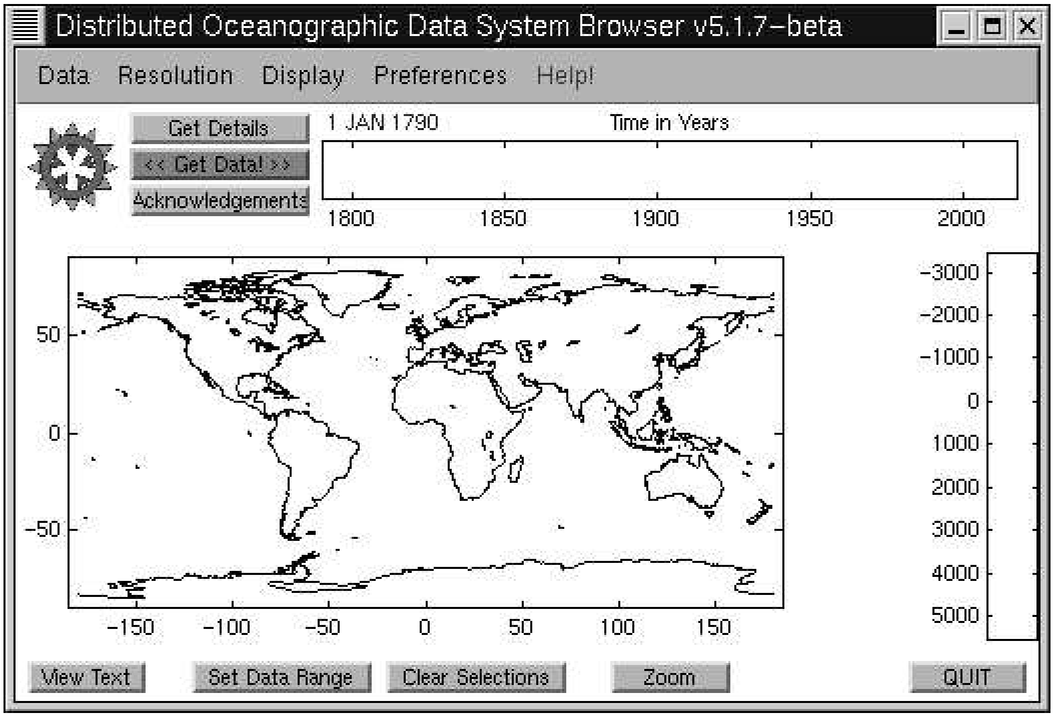


Figure 1.1: The OPeNDAP Matlab GUI Main Window

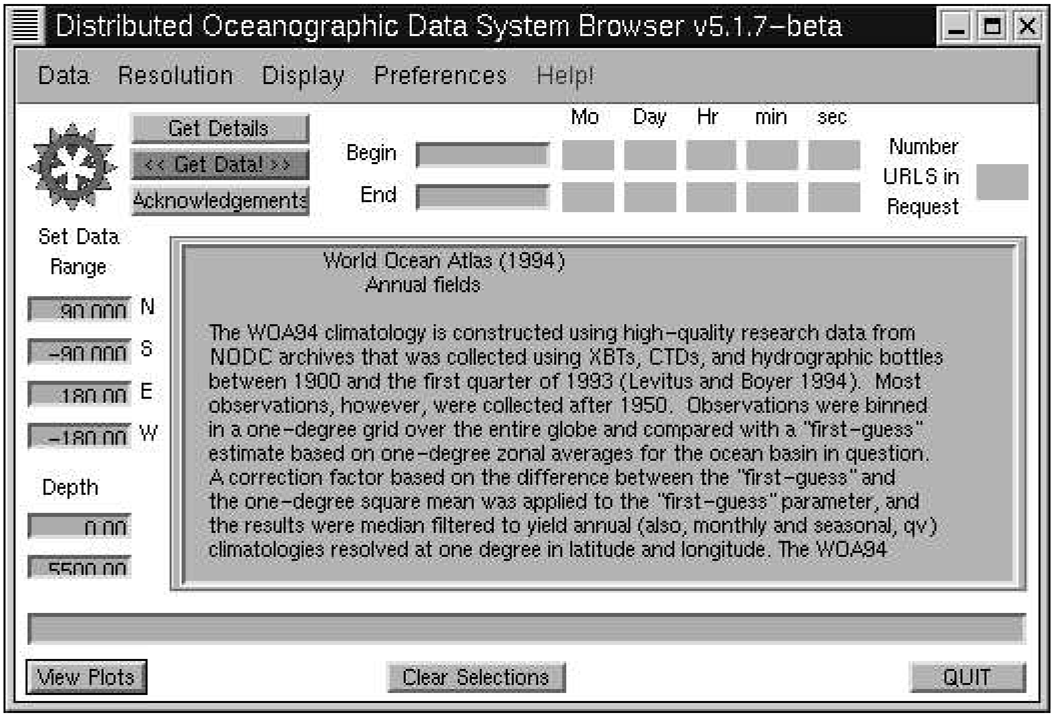
You can also see a text version of the window, by clicking on the **View Text** button. Return to the graphical view by clicking **View Plots**.

1.2.1 Setting up a data request

Choose the dataset and variable you want to see.

To set up a data request, you must select the variable and dataset you wish to examine. Select these from the variable and dataset lists that appear when you start the browser. If you've closed those windows, you can recall them with the **Data,Show Data Bookmarks** button.

The Variable and Dataset menus depend on each other. Once you've selected something from one menu, the other menu changes to display the corresponding



90.000 N
-90.000 S
180.00 E
-180.00 W

Depth
0.00
5500.00

World Ocean Atlas (1994)
Annual fields

The WOA94 climatology is constructed using high-quality research data from NODC archives that was collected using XBTs, CTDs, and hydrographic bottles between 1900 and the first quarter of 1993 (Levitus and Boyer 1994). Most observations, however, were collected after 1950. Observations were binned in a one-degree grid over the entire globe and compared with a "first-guess" estimate based on one-degree zonal averages for the ocean basin in question. A correction factor based on the difference between the "first-guess" and the one-degree square mean was applied to the "first-guess" parameter, and the results were median filtered to yield annual (also, monthly and seasonal, qv) climatologies resolved at one degree in latitude and longitude. The WOA94

Figure 1.2: The OPeNDAP Matlab GUI Main Window

entries. For example, if you are interested in temperatures, you can select it in the Variables menu. After this, the datasets that contain temperature data will be highlighted (displayed with bold font) in the Bookmarks menu. Once these have been selected, the spatial and time ranges of the selected dataset are indicated on the display map. You can use the **Zoom** button to examine any of the data displays more closely.

You can also begin your data selection by selecting the region, time, or depth you wish to examine. The Datasets and Variable menus will be updated to show only datasets that contain data in those regions.

The OPeNDAP Matlab GUI uses locally-cached archives of “metadata” to describe datasets which may be stored on any machine connected to the Internet. This allows the OPeNDAP Matlab GUI to download its data in a consistent set of units, allowing comparisons among datasets. You can use the **Data,Update Bookmarks** button to update the local archives with any new datasets or changes in existing datasets.

1.2.2 Querying a Dataset

After a dataset, variable, and region (space and time) are chosen, you can find out roughly how much data (and what kind) you have selected with the **Get Details** button. Clicking this will send a query to the dataset to find out how much of it fits your chosen ranges; for example, how many images fall within your time range, or how many XBT casts fall within your geographic selection box.

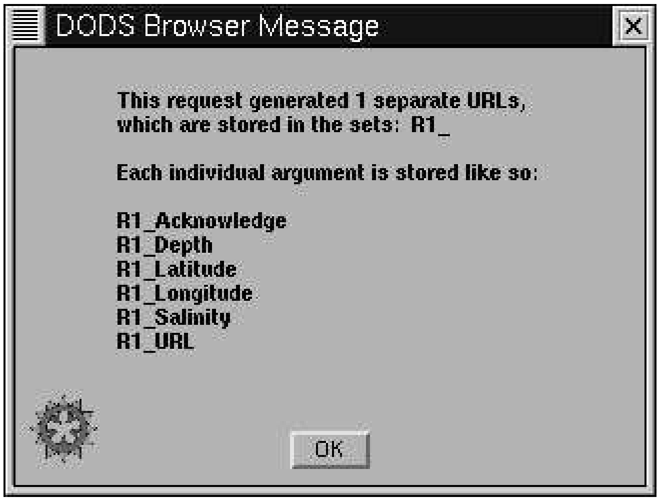
For gridded data, the **Get Details** button will display a star in the Time window for each grid available. That is, if you are looking at a dataset consisting of a series of complete grids, one per day, you will see a star drawn for each day in the requested time range. If you're looking at a dataset containing non-gridded data, you'll see stars in the geographic and depth windows, too.

1.2.3 Issuing a Data Request

Once the parameters of a data request have been specified, you can issue the request by clicking the **Get Data** button.

Sometimes, inadvertently or not, you ask for a lot of data. The browser has a data size threshold you can set to notify you when you make a request like this. The data source you've selected may give the browser an estimate of how much data is on the way, and the browser will ask you for confirmation in the event that the estimated size of the data requested takes up more memory than the data size threshold specified in the **Preferences,Get Data Threshold** popup menu.

If you approve the request (or if you weren't asked), the requested data are retrieved, loaded into Matlab, and displayed according to the directions (you will be asked for these directions, see section 1.2.4). The browser displays a message telling you what has been loaded into Matlab (figure 1.3).



This request generated 1 separate URLs,
which are stored in the sets: R1_

Each individual argument is stored like so:

R1_Acknowledge
R1_Depth
R1_Latitude
R1_Longitude
R1_Salinity
R1_URL

Figure 1.3: Data returned to Matlab

Note that along with the data you requested, the URL that pointed to that data, as well as the text of any credits that apply to the data are also returned. To duplicate the request just made, say in some later session, you could issue the following command:

The request returns the URL, which you can use later.

```
>> loaddods(R1_URL)
```

(This assumes that `R1_URL` has not changed. Since the browser is likely to re-use that name in future sessions, it is good practice to save it in a different name if you think you might request that data again.)

Of course, once the data are incorporated into Matlab variables, the entire power of the Matlab analysis package is available. For many users, this is the real goal. Other users may find the data display functions of the OPeNDAP Matlab GUI valuable.

1.2.4 Displaying the Data

Choose the display axes.

Each time you use the browser to request data, it will ask how the data are to be displayed. The first thing that happens is that the browser makes a guess as the X, Y, and Z coordinates you want to display. It confirms the guess by presenting a dialog to you like the one in figure 1.4. Using this dialog, you can select the three coordinates to be displayed from among all the variables returned by the data request.

Using this Display dialog, you can modify your earlier choices, and you can also select the type of display you want (color image, contour plot, quiver plot, and so on), and further select the returned data.

You can call this dialog up with the **Data, Plot acquired Rxx** button.

You can use the Display Dialog to subsample or modify data arrays.

In some circumstances, the Z data might be returned as a three-dimensional array. To display it, the array must be subsampled. You can do this by typing a Matlab expression right into the Z-data text box. For some datasets, the Matlab matrix transposition operator (`'`) is important to use here. Others might require a matrix sampling.

The dialog contains the following fields to modify:

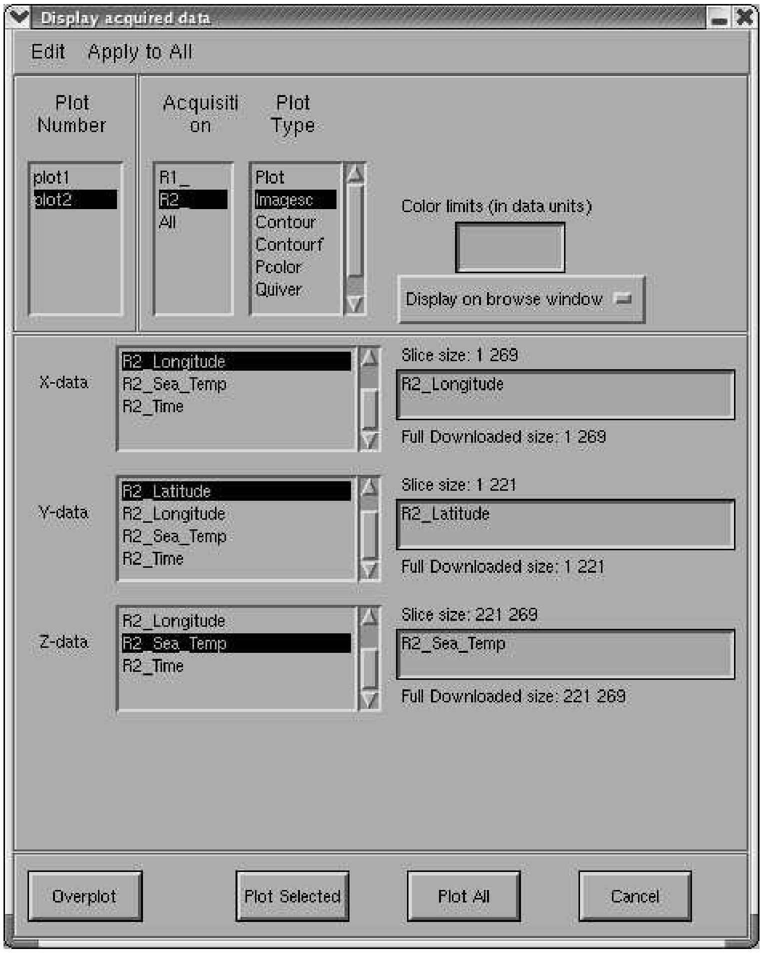
Plot Number This is the number of the plot, assigned and incremented by the browser.

Acquisition Number Requests for OPeNDAP data are numbered within a Matlab session. Names of data variables from the first request are prefixed with `R1_`. The second request makes variables with `R2_`, and so on. This keeps data from being overwritten as requests are made after other requests.

Plot Type A list of available plot types. These are standard Matlab types.

Plot Data Depending on the plot type selected, the display dialog will present an assortment of selection lists to set plot parameters.

Display Window A selection list is presented to allow you to specify whether the plotting should be done in a new window, in the existing browser window, or in a window to be specified.



Slice size: 1 269
R2_Longitude
Full Downloaded size: 1 269

Slice size: 1 221
R2_Latitude
Full Downloaded size: 1 221

Slice size: 221 269
R2_Sea_Temp
Full Downloaded size: 221 269

Figure 1.4: The Display Dialog

X-data, Y-data, Z-data, V-data The data to be plotted must be specified in these fields. If any of these arrays must be trimmed, processed, transposed, or sampled, you can enter valid Matlab expressions to do so in the text boxes. For example, if you have an array that needs transposing, you can type the Matlab array inversion operator (') after the array name.

At the bottom of the display dialog, there are four buttons. Use the **Cancel** button if you've thought better of the whole thing, and wish to make another data request. The data will still be available inside Matlab, but the browser will skip plotting it. The **Plot Selected** button will plot the data selected in the dialog. If you want to plot several layers at once, use **Plot All**, and all the data will be plotted at once. A **Next Plot** button will appear on the browser window to allow you to flip through the plots.

The **Overplot** button is a toggle button. When it's activated, the browser will not clear the previous plot when drawing a new one. Click it to activate, and click it again to de-activate this mode.

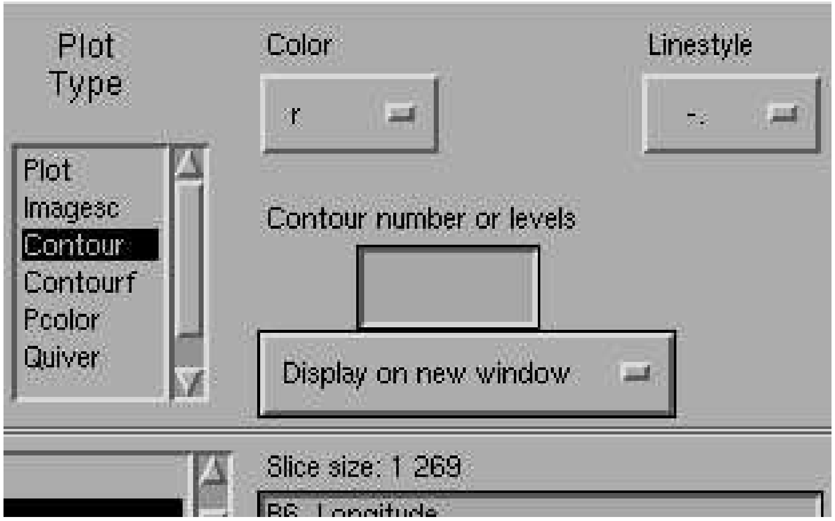


Figure 1.5: The Display Dialog, Contour plotting

Several of the plotting modes require different settings. When you select the plot mode, drop-down lists and dialogs will appear to allow these settings. See figure 1.5 for an example.

1.3 Example Sessions

The simplest way to begin to use the OPeNDAP Matlab GUI is to follow along with a simple session. Two examples follow: the first uses the OPeNDAP Matlab GUI window to display data, and the second uses the browser only as a mechanism to retrieve data, which is then examined with some other Matlab functions.

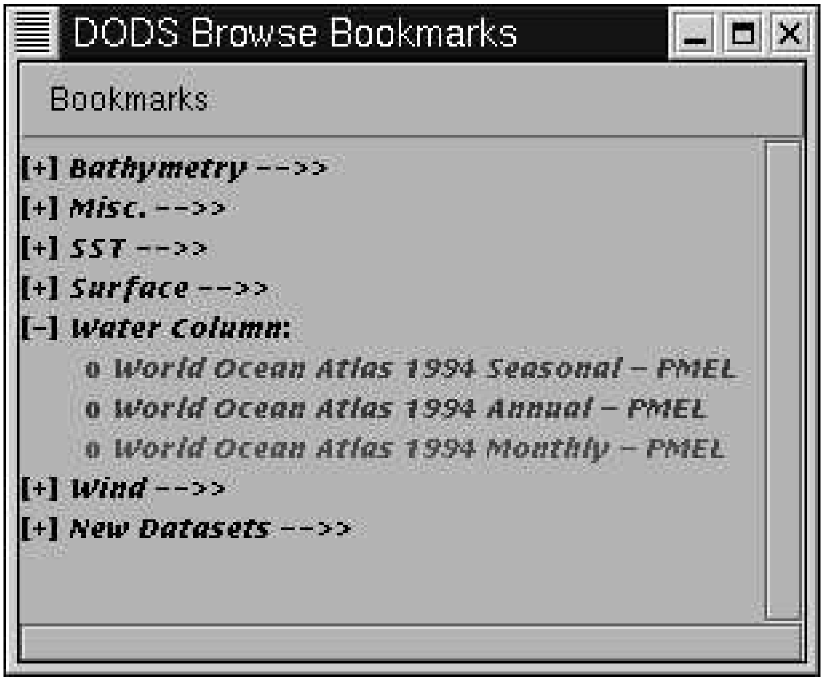
1.3.1 Drawing a Simple Picture

Select the dataset and the variable.

Start Matlab, and start up the browser. (See page 2).

❶ The Bookmarks and Variables menus should already be visible. If not, use the **Data,Show Data Bookmarks** button. The Bookmarks menu will look roughly like figure 1.6. Look under “SST,” then click on the entry marked “Reynolds Daily Long Term Mean SST.”

NOTE: When you click on this dataset, you'll see a little message come up labeled "Policy." This describe where you can download this data yourself. Data providers often require messages like this when programs like the OPeNDAP Matlab GUI incorporate their data.



[+] *Bathymetry -->>*
[+] *Misc. -->>*
[+] *SST -->>*
[+] *Surface -->>*
[−] *Water Column:*
o *World Ocean Atlas 1994 Seasonal – PMEL*
o *World Ocean Atlas 1994 Annual – PMEL*
o *World Ocean Atlas 1994 Monthly – PMEL*
[+] *Wind -->>*
[+] *New Datasets -->>*

Figure 1.6: The Bookmarks Menu

This dataset contains sea surface temperatures for the entire globe, in one degree squares, for each day since about 1800. You will see a rectangle in the main window outlining the entire globe. The time window contains a small line starting around 1800, and moving forward.

Zoom in to select the time.

❷ Click the **Zoom** button, and click the left mouse button near the Reynolds time line a few times, until the time is measured in days or months. (If you find the default colors hard to see, click on the **Preferences,Colors** button to adjust them to your liking. Use the **Preferences,Save Preferences** button to save your colors for the next time.) The left mouse button zooms in, and the right button zooms out. You can also select a small range by clicking and dragging, and that range will be mapped over the timeline.

❸ Click on **Set Data Range**, and select a time range by clicking and dragging in the time window (be patient). Remember that this dataset has a global grid recorded every day. Be careful not to select more than you'r willing to wait for.

While you are selecting the range, you might notice the Datasets or Variables menus changing. These menus are dynamically loaded with all the datasets and variables that might overlap the data range you select. However, the changes will not affect your selection.

> **NOTE:** If you select a date or geographic range not covered by the dataset you've selected, the browser will unselect it. Messages for such events are printed in the main Matlab window.

Select the geographic range.

❹ Now select a geographic range by clicking and dragging in the main window. By default, the whole world is selected, so if you skip this step, you will still get data. When selecting a time range, remember that this dataset is made up of weekly means, so be careful you don't ask for too much.

❺ Click the **Get Details** button. This will put a star at each point on the time line where data is available. You may see that you've selected 29 points, which could bring a lot of data back. You can refine your selection at this point to select the few data points of interest.

Get the data.

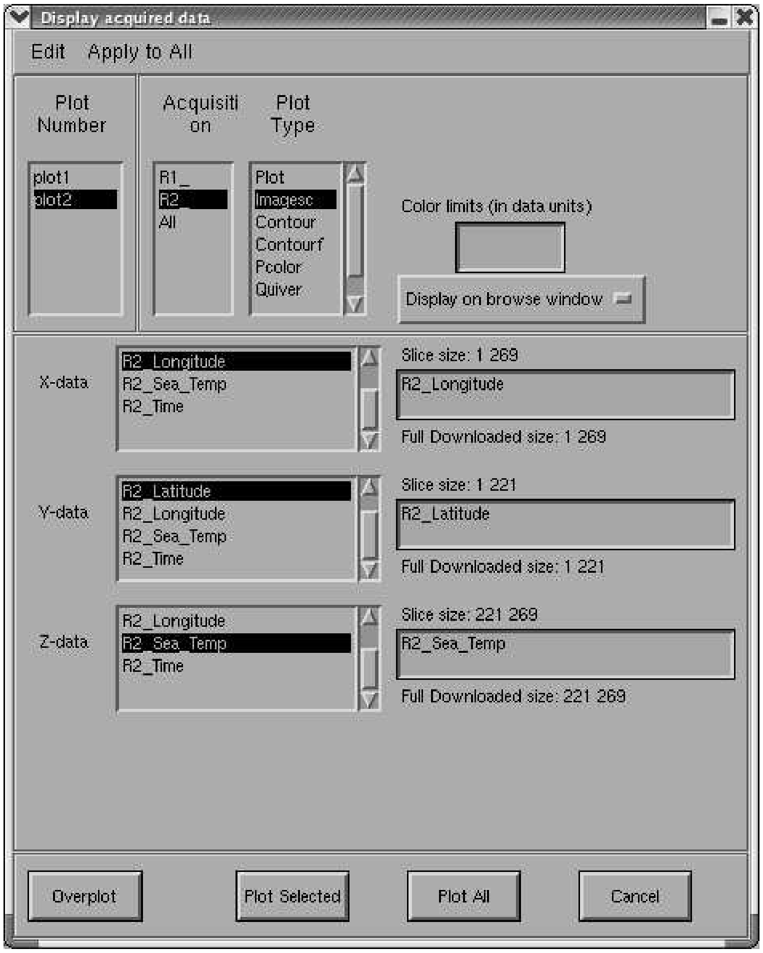
❻ Click the **Get Data** button. You will see a dialog box that says: "Hang on, transferring data," and some other stuff, like a dialog announcing what number this request is. Eventually, the browser will pop up the Data Display dialog (figure 1.7), asking how you wish to display the data returned. Unless you've been careful in selecting your time range, it's likely that the temperature array returned will have three dimensions. The browser is designed to allow this, so it's not a problem. If you want to see a plot of something besides the first time array, you may have to click on the text box for the Z values, and add a Matlab indexing expression. That is, if the box reads `SST`, and you want to see the twelfth array in the series, you should change the Z value text box to something like `SST(:,:,12)` to get it.

❼ Watch the data display. A message will announce when the data have been returned to the Matlab workspace, and what the new variable names are. You can examine and display these variables like any other Matlab variable.

A Short But Edifying Digression About the Internet

The GUI is just a specialized form of a web browser like Netscape.

The OPeNDAP Matlab GUIgets its data over the internet, and the way it specifies the location of the data it wants is with a URL very similar to the URL you might use in a web browser, such as `Netscape`. In fact, the OPeNDAP Matlab GUI can be thought of as a very specialized web browser, that can only look at a very particular kind of web page. When you type a URL into a web browser, the browser "de-references" the URL to produce the page you can read. When you



Slice size: 1 269
R2_Longitude
Full Downloaded size: 1 269

Slice size: 1 221
R2_Latitude
Full Downloaded size: 1 221

Slice size: 221 269
R2_Sea_Temp
Full Downloaded size: 221 269

Figure 1.7: The “Display Data” menu

enter a URL into an OPeNDAP Matlab program called `loaddods`, that program de-references the URL to produce data in the Matlab workspace.

It’s a lot to type, but try typing the following at the Matlab prompt (all one line):

```
    >> loaddods('http://ferret.wrc.noaa.gov/cgi-bin/nph-nc/data/
              COADS_climatology.nc?SST[7:7][0:1:89][0:1:179]')
```

When `loaddods` dereferences this long URL, it prints something like this in the terminal window (not in the Matlab window, but in the terminal you were using when you started Matlab):

```
Reading: http://ferret.wrc.noaa.gov/cgi-bin/nph-nc/data/COADS_climatology.nc
  Constraint: SST[7:7][0:1:89][0:1:179]
Server version: dods/2.15
Creating matrix SST (90 by 180) with 16200 elements.
Creating scalar TIME.
Creating vector COADSY with 90 elements.
Creating vector COADSX with 180 elements.
```

and it puts the four arrays (`SST`, `TIME`, `COADSX`, and `COADSY`) into your Matlab workspace.

```
>> whos
  Name                  Size           Bytes  Class

  COADSX              180x1             1440  double array
  COADSY               90x1              720  double array
  SST                  90x180         129600  double array
  TIME                  1x1                8  double array
```

The `loaddods` program is the heart of the OPeNDAP Matlab GUI. The rest of the OPeNDAP Matlab GUI is simply a “front-end” to the `loaddods` program that you use to construct URLs and make some housekeeping chores somewhat easier.

You can use this to your advantage, as is shown in the next example.

Incidentally, when the OPeNDAP Matlab GUI returns data, one of the returned variables is the exact URL that was used to issue the request for that data. This provides a unique identifier should you want to recreate a request or expand on it, perhaps in your own Matlab script.

1.3.2 Creating a Time Series

It is often the case that the data you wish to see cannot be easily displayed in the OPeNDAP Matlab GUI window. OPeNDAP allows much richer access to the data than the browser can accommodate. This means that you can often use the OPeNDAP Matlab GUI to help construct a URL, even though you may not use it to submit the data request.

You can also use OPeNDAP from the Matlab command line.

The following example shows how you might go about constructing and displaying a time series from the OPeNDAP Matlab GUI datasets, even though there is limited browser capacity for displaying this kind of data. The OPeNDAP Matlab GUI actually is flexible enough to plot this example, but there are others for which this technique is useful. It’s also useful to become familiar with `loaddods` in case you want to include access to OPeNDAP datasets in your own scripts.

❶ Start up the browser and select the “Surface - COADS Monthly Datasets - PMEL” dataset.

❷ Select “Air Temperature” from the Variables menu. (See figure 1.8.) If it’s not there, you haven’t selected a dataset containing that variable.

❸ Using the **Zoom** and **Select Data Range** buttons, select a very small geographic range. This dataset is arranged in two-degree squares, so pick a two-degree square somewhere. Pick a wide time range.

❹ Click on **Get Details**. Select a time at the beginning of the range. Click on **Get Data**.

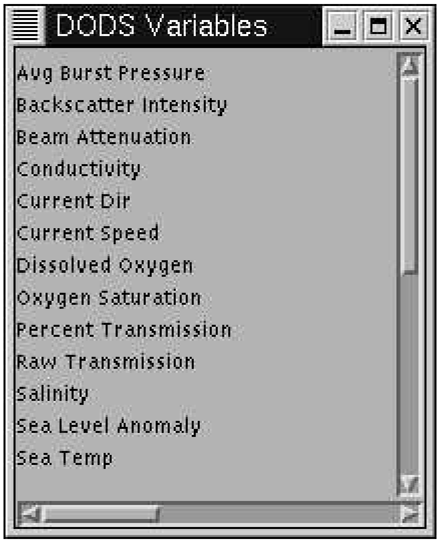


Figure 1.8: The Variables Menu

❺ Repeat the process for a time at the end of the time range. Click on **Get Data**.

❻ Look at the Matlab session. You will see two messages like this:

```
Reading: http://ferret.wrc.noaa.gov/cgi-bin/nph-nc/data/coads_air.nc
  Constraint: AIR[1316:1316][57:1:57][140:1:140]
Server version: dods/2.15
Creating matrix AIR (2 by 2) with 4 elements.
Creating scalar TIME.
Creating vector LAT with 2 elements.
Creating vector LON with 2 elements.

          ..... data transfer complete

This request generated 1 separate URLs,
which are stored in the sets:  R3_

Each individual argument is stored like so:

R3_Time R3_Longitude R3_Latitude R3_Air_Temp
```

The above URL indicates that your request asked for time index 1316 in the dataset, latitude index 57, and longitude index 140. (This corresponds to September, 1963, at about 24°N and 78°W.)

❼ Type the following at the Matlab prompt (all on one line):

```
>> loaddods('http://ferret.wrc.noaa.gov/cgi-bin/nph-nc/data/
coads_air.nc?AIR[1316:1500][57:57][140:140]')
```

You will see a message like this:

```
Reading: http://ferret.wrc.noaa.gov/cgi-bin/nph-nc/data/coads_air.nc
  Constraint: AIR[1316:1500][57:57][140:140]
Server version: dods/2.15
Creating vector AIR with 185 elements.
Creating vector TIME with 185 elements.
Creating scalar LAT.
Creating scalar LON.
```

❽ You now have, in the Matlab workspace, a vector of temperatures, `AIR`, whose entries correspond to a vector of times, `TIME`. The scalars `LAT` and `LON` give the location of the data. You can display the data with Matlab functions. Type the following at the Matlab prompt:

```
>> figure
>> plot(TIME,AIR)
```

CAUTION: There are some pitfalls to using `loaddods` directly. The OPeNDAP Matlab GUI does quite a bit of housekeeping for you, including some unit conversions. When you use the browser, you get your data back in consistent units, whatever dataset you are looking at. When you use `loaddods`, you get the data are returned in the units they are stored in. The COADS time data, used in the above example, comes in units of days since 1700.

Similarly, different data providers use different values within their datasets to indicate “bad” or “missing” data points. Through the browser, these data flags are transformed into Matlab `NaN`s. (See Matlab ‘help nan’ if you are not familiar with the `NaN` symbol.) Again, downloading the data directly with `loaddods` skips this transformation.

If you like, you can save your data with the Matlab `save` command and restore it with the `load` command. See the Matlab documentation for more information about these commands.

1.4 Updating the Dataset List

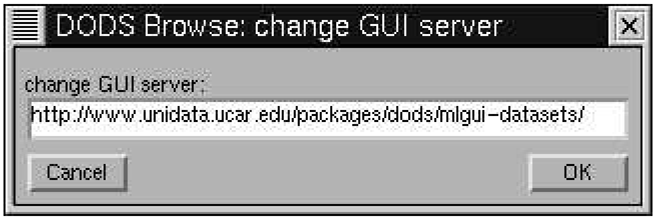
Maintaining your set of dataset bookmarks.

The OPeNDAP Matlab GUI consists of two sets of files: the M-files and associated data used to run the display and user interface, and a second collection of Matlab M-files used to describe the datasets available through the OPeNDAP Matlab GUI. These “metadata” files (also called the “archive M-files ”) contain descriptions of the datasets, and retrieval and catalog functions tailored specifically for them.

The metadata files are stored in a sub-directory of the main OPeNDAP Matlab GUI directory, called `DATASETS`. Generally speaking, most users will be quite unlikely to want to edit the OPeNDAP Matlab GUI M-files. However, there are several reasons why you might have to edit the archive M-files: adding a dataset, adding comments to a dataset, or otherwise editing the metadata. Chapter 3 contains information about the structure of the archive M-files and their associated functions.

For example, the archive M-file for the declouded Hatteras to Nova Scotia 1.1km resolution AVHRR dataset is called `htn_decld.m`. You will find this script in the browser subdirectory `DATASETS`.

The dataset descriptions are stored in Matlab M-files, but the way a browser user selects these M-files is through a “bookmark” file, comparable to the bookmarks used in a web browser like `Netscape`. You can edit and rearrange these bookmarks into folders like in `Netscape`, too. If you want to add your own M-file dataset descriptions to the bookmark file, you can just add it with the bookmark editor.

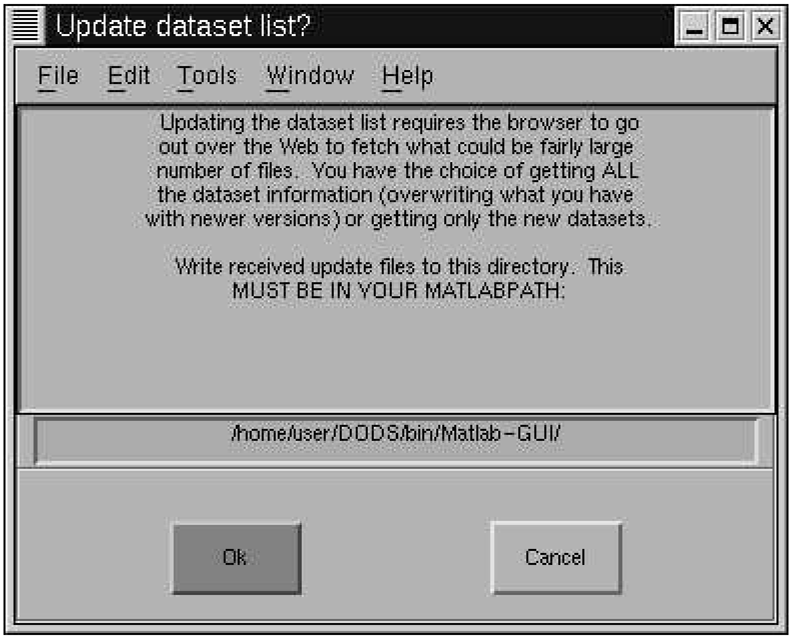


change GUI server:
http://www.unidata.ucar.edu/packages/dods/mlgui-datasets/

Figure 1.9: GUI Server Dialog

The collection of datasets the browser can access is continually updated at the OPeNDAP Matlab GUI central archive. You can update your browser with the contents of that archive by clicking the **Data,Update Datasets** button. If you know of another archive of GUI dataset descriptions, you can use the **Preferences,Change GUI Server** option to point the browser at that server instead.

When you want to update the dataset descriptions in your copy of the OPeNDAP Matlab GUI, click on **Data,Update Datasets**, and the dialog in figure 1.10 will appear. If you want to continue, simply click the **Update** button.



Updating the dataset list requires the browser to go out over the Web to fetch what could be fairly large number of files. You have the choice of getting ALL the dataset information (overwriting what you have with newer versions) or getting only the new datasets.

Write received update files to this directory. This MUST BE IN YOUR MATLABPATH:

/home/user/DODS/bin/Matlab-GUI/

Figure 1.10: Update Datasets Dialog

The update function downloads a new MANIFEST file, and then retrieves all the new files mentioned in that file, as well as new versions of already-existing M-files in your DATASETS directory. The update process also updates any bookmarks you might have that point to datasets in the MANIFEST. If there are bookmarks that you've added (that match no entry in the MANIFEST), they are left untouched. If there are datasets in the MANIFEST that are not in your bookmarks file, they are added to your bookmarks, in a folder called "New Datasets."

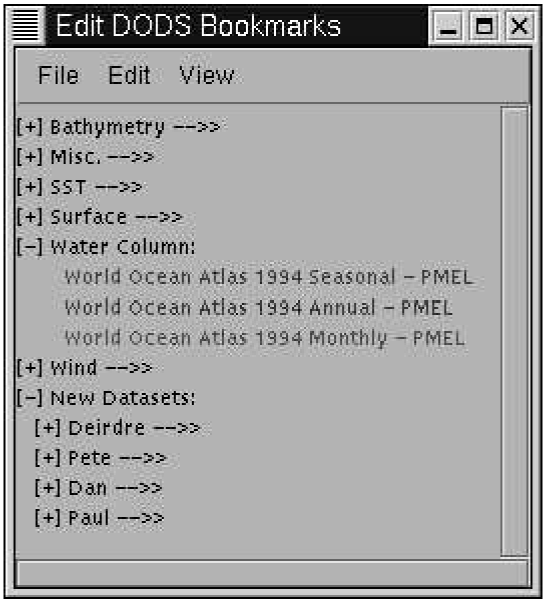
1.5 Editing Bookmarks

The OPeNDAP datasets presented to the user in the OPeNDAP Matlab GUI are described by a list of "bookmarks." These bookmarks are comparable to the bookmarks in the Netscape WWW browser.

You can edit and augment the bookmark list with the bookmark editor. Invoke the editor with the **Data,Edit Bookmarks** button.

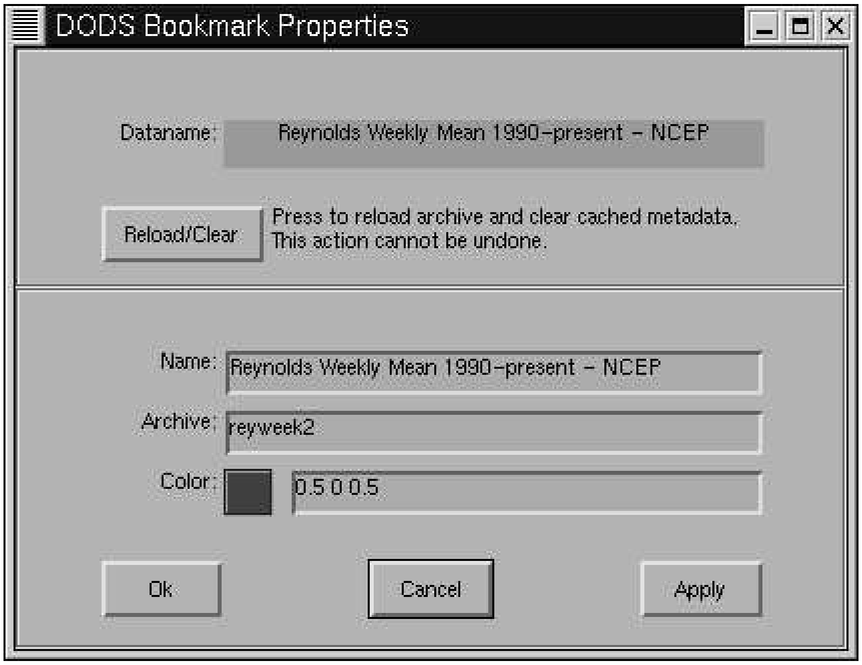
The editor window is shown in figure 1.11. It looks a great deal like the bookmark window (see page 9), but the menu along its top is different.

To see a bookmark entry, highlight the bookmark entry, and click on **Edit,Properties**. You'll see an entry like the one in figure 1.12.



```
[+] Bathymetry -->>
[+] Misc. -->>
[+] SST -->>
[+] Surface -->>
[-] Water Column:
      World Ocean Atlas 1994 Seasonal - PMEL
      World Ocean Atlas 1994 Annual - PMEL
      World Ocean Atlas 1994 Monthly - PMEL
[+] Wind -->>
[-] New Datasets:
  [+] Deirdre -->>
  [+] Pete -->>
  [+] Dan -->>
  [+] Paul -->>
```

Figure 1.11: Bookmark Editor Window



Dataname: Reynolds Weekly Mean 1990-present - NCEP

Press to reload archive and clear cached metadata.
This action cannot be undone.

Name: Reynolds Weekly Mean 1990-present - NCEP

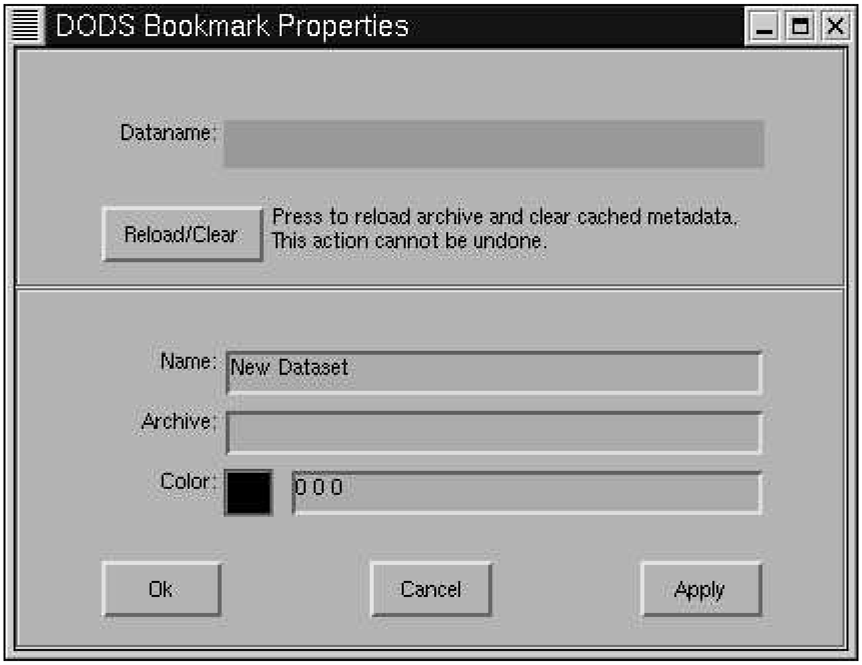
Archive: reyweek2

Color: 0.5 0 0.5

Figure 1.12: Bookmark Properties Window

The “Dataname” at the top of the window is the name given to the data by the M-file describing that dataset. The “Name” field is the name that appears in the bookmark window, and the “Color” is the color in which that name is displayer. The “Archive” field names the M-file that described the dataset in question. You can edit fields here and save the changes with the **Ok** or **Apply** buttons.

If you want to add a dataset, and have the M-file ready (see Chapter 3), select **Edit,Insert New**, and you'll get a dialog like the properties dialog, but empty. (Figure 1.13)



DODS Bookmark Properties

Dataname:

Press to reload archive and clear cached metadata.
This action cannot be undone.

Name: New Dataset

Archive:

Color: 0 0 0

Figure 1.13: Adding Bookmarks

To add the dataset, make sure the M-file is in place in the `DATASETS` directory, fill out this dialog, and click **Apply**, then **Reload/Clear**. When the correct Dataname appears in the top window, you'll know your dataset has been registered.

2

The DODS GUI Menu Reference

This chapter contains a description of each of the menus and buttons that make up the OPeNDAP Matlab GUI. They are arranged by function, and by containing menu. The buttons and menus are also referenced in the index.

2.1 The Buttons

The following buttons are available on the main screen of the OPeNDAP Matlab GUI. (See figure 1.1.) They are roughly arranged by function.

Set Data Range Before a query can be made to a dataset, the dataset and variables must be selected, and a range must be selected in space and time. If you are looking at the graphical view, you can click this button, and then select an area in any of the selection windows (global location, time, and depth) by clicking and dragging the left mouse button.

You can also enter the boundaries of the selected range by hand. Simply switch to the text view with the **View Text** button, and enter the values in the entry windows provided.

Zoom Use this button to examine any of the displayed axes more closely. When the zoom operation is in effect, the left mouse button zooms in to a closer view, while the right mouse button zooms out to a more distant view. If you click and drag, the enclosed area will be enlarged to fill the selection window. (This is the standard Matlab “zoom” function.)

> **NOTE:** Zooming in on the map, time line, or depth line does not select a region. In order to submit a request for data, you must select a range with the **Set Data Range** button.

Get Details Once you have selected a dataset, variables, and range, you can click the **Get Details** button for information about the volume and location of the data in the dataset. For example, if you select the Reynolds Weekly Sea Surface Temperature dataset, **Get Details** will produce an asterisk at each time within the selected time range where there is data. If you zoom in on these asterisks, you will see one of them every seven days. If multiple asterisks appear in the range you have selected, you might choose to refine your range to make a smaller request before you submit the data request. Of course, you can request data from several times, in which case there will be multiple accesses to the remote dataset. Each request will be interned separately into the Matlab workspace.

Get Data Use this button to submit a request for data from the remote server. If your data range contains several data points, your data may be returned in several Matlab arrays, or in a multi-dimensional array that will need to be subsampled to display. A message box will announce the names under which the data are to be stored.

For very large requests, a warning window may appear to remind you that the request you are about to make is a large one. Press “OK” to confirm you still want the request to be submitted. See page 25 for more information.

Next Plot This button only appears when you’ve plotted many variables at once. You can use this button to flip through the plots, one at a time, until the end.

Acknowledgements Selecting this button displays a text window containing text describing the source and authors of the dataset currently selected. You can use this text to cite the source of the data, should you end up using it in a context where this is appropriate, such as a publication. If you make data products available to other scientists using this data, it is also considered good form to include this citation with that data in some way.

View Text The text version of the main GUI window allows you to type the boundaries of the data range, and displays the dataset comments, which contain detailed descriptions of the datasets.

Press this button if you are looking at the graphic version of the main window and you want to be looking at the text version.

View Plot The graphic version of the main GUI window allows you to select the search boundaries by pointing and dragging with the mouse. This is also where you can see the displayed results of a search.

Press this button if you are looking at the text version of the main window and you want to be looking at the graphic version.

Clear Selection This button clears all the selections you have made: data range, variable and dataset. The Datasets and Variables menu windows are restored to their initial conditions.

Quit This button terminates the browser. Matlab continues running after the browser finishes; you have to type `quit` at the Matlab prompt to end your Matlab session.

2.2 The Menus

The menu bar over the matlab main window (figure 2.1) contains drop-down menus with several useful functions. The functions in the OPeNDAP Matlab GUI menus allow you to select the datasets and variables to search for and display, to control many aspects of how the data are displayed, and to adjust the overall appearance of the main window.

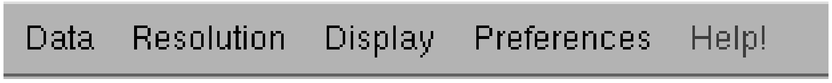


Figure 2.1: The GUI Menubar

2.2.1 Data

The Data menu (figure 2.2) contains buttons that manage aspects of the data bookmarks and downloaded data.

Show Data Bookmarks

This button causes a list of the data bookmarks and the variable browser window to appear. You must select a dataset from the bookmarks window, as well as a variable, in order to submit a data request.

If you click on one of the variable names in the variable browser window, the Datasets menu window will only display the datasets that contain that variable. Similarly, if you have already made a selection of a dataset or a range, the Variables menu only displays the variables of datasets still on the dataset menu.

The Variables menu is persistent; after you have displayed it, it will remain on your screen until you close it. Sometimes the menu will seem to disappear; this is because your window manager software causes the browser window to cover over the new Variables menu.

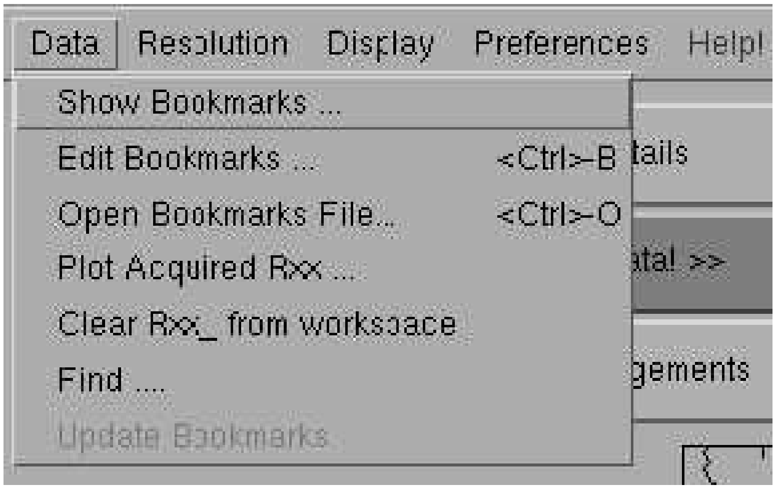


Figure 2.2: The Data Menu

Edit Bookmarks

Clicking this button invokes the bookmark editor. You can use this to edit items in the list of dataset bookmarks offered in the bookmark window, or you can add your own. See Section 2.2.7 on page 30 for a description of the bookmark editor menus, and see Section 1.5 on page 16 for an introduction to the operation of the editor.

If you want to add a bookmark, you should probably look at Chapter 3.

Open Bookmarks File

The OPeNDAP Matlab GUI supports having multiple files of bookmarks. Use this button to change the bookmark file currently in use.

Plot Acquired Rxx

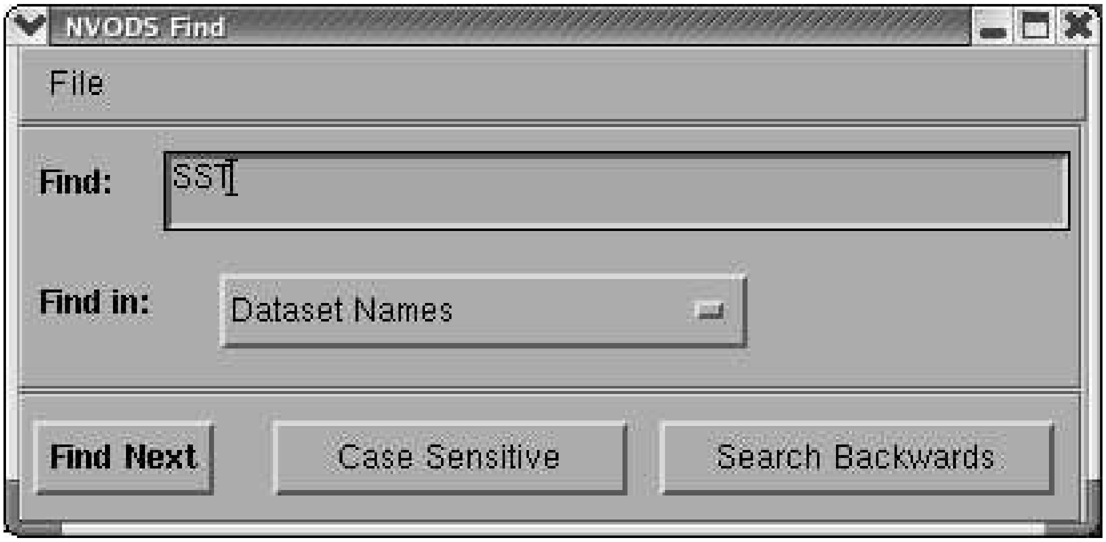
If you want to replot data you've already downloaded, use this option, and you'll see the plot dialog you originally saw when you downloaded the data in the first place. See section 1.2.4 for more information about plotting.

Clear Rxx˙ from workspace

Selecting this option causes all Matlab variables with names like R20_Pressure that have been created during the current browse session to be deleted. Effectively, this clears all variables created by remote data requests. Of course, if an Rxx variable has been copied to a variable of a different name, the new variable will not be deleted.

Find

You can use this button to search for a text string. You can search the dataset names in the bookmark file, the variables made available by any of those datasets, the dataset descriptions, or the acknowledgements attached to each dataset. The searcher returns the first result it finds, displaying the associated dataset in the Text view. Click **Find Next** to see the next match.



Find: SST

Find in: Dataset Names

Figure 2.3: The (Text) Search Dialog Menu

Update Bookmarks

The archive M-files that come with the OPeNDAP Matlab GUI are kept up-to-date in a central location on the DODS OPeNDAP Matlab GUI server, currently at URI. You can update your copies of these files at any time with this button. (See Section 1.4 on page 15) Clicking this button pops up the Update menu, which gives you further instructions.

2.2.2 Resolution

This menu allows you to select the "resolution" at which you wish to see a dataset displayed. The resolutions used indicate (roughly) the distance between adjacent values of latitude, in kilometers. Of course, this only applies to data lying on a rectangular grid.

NOTE: The resolution applies equally to the X and Y directions. However, the browser expresses resolution in units of distance, which (for a dataset whose locations are recorded in longitude and latitude) only remains constant in the North-South direction.

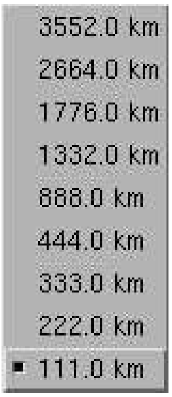


Figure 2.4: The Resolution Menu

The resolution choices for the Reynolds Weekly mean dataset are shown in figure 2.4. This dataset is recorded on a global one-degree grid, so the bottom choice (111 km) approximately corresponds to one-degree of latitude. Through the browser, the resolutions available are the following multiples of the fundamental resolution of the dataset: 2,3,4,8,12,16,24, and 36. (This can be adjusted by editing the saved preferences file. See Section 2.3 on page 32.)

The resolution corresponds directly to the “stride” value you use in requesting gridded data with DODS. (Refer to *The OPeNDAP User Guide* for more information about grids, stride values, and how they fit together.) Therefore, if the available resolutions are not what you need, you can use `loaddods` to request a different resolution. See page 10 for more information about that program.

2.2.3 Display

This menu, shown in figure 2.5, contains various options that allow you control the way data are displayed, if at all.

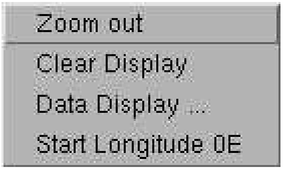


Figure 2.5: Display Menu

It contains several buttons:

Zoom Out If you've zoomed in on some part of the world map, this button restores the map to cover the whole world. It also restores the time line and the depth range to their original values.

Clear Display This button erases the displayed data from the browser main window.

Reset Display Erases the displayed data from the browser main window, and zooms out to view the whole world, the complete time line, and all depths.

Start Longitude This changes the display map longitude boundaries. Start the longitude at 180W (the international date line) if you want to see the Atlantic, but start at 0E (the greenwich meridian) if you want to see the Pacific entire. This is a toggle switch.

2.2.4 Preferences

The preferences menu is shown in figure 2.6.

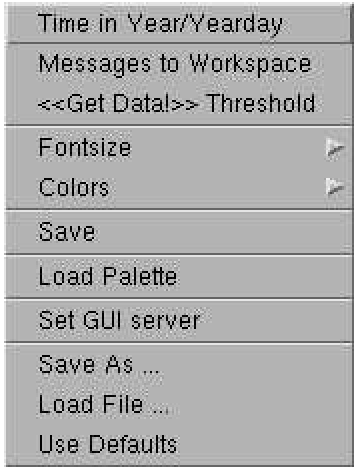


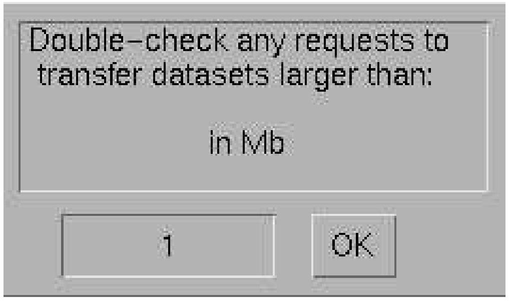
Figure 2.6: Preferences Menu

Time The date can be displayed as Year/Month/Day triples, or as the year and the ordinal number of the day for that year. This is a toggle switch.

Messages Messages to the user can be directed either to pop-up windows ("Messages to Pop-up Window") or to the Matlab command-line interface ("Messages to Workspace"). This is a toggle switch, click it to change its value.

Get Data Threshold This button calls up a dialog box that controls the data request threshold. (See figure 2.7.)

This threshold marks the maximum size of data file that the Matlab DODS browser will fetch without prompting the user for confirmation. For example, if the threshold is set at 1 megabyte of data, the user will be prompted for confirmation of any data request likely to return more than 1 megabyte of data. Because the OPeNDAP Matlab GUI cannot know exactly how big a dataset is until it has actually downloaded the entire data request, this threshold works with estimates.



Double-check any requests to transfer datasets larger than:

in Mb

1

Figure 2.7: The Data Threshold Dialog. Use this dialog to control the *estimated* size of the data request.

> **NOTE:** A further problem with this threshold is that the estimates are about how much memory the data will occupy in Matlab memory, *not* the size of the the transmitted information. The memory occupied by data in Matlab can be much larger than the amount of data transmitted. Any number in Matlab occupies eight bytes of storage space. Many DODS data—especially satellite data—are transmitted as single byte numbers. Further, the OPeNDAP Matlab GUI has no way to know whether the transmitted data were compressed or not. This can increase the size of this discrepancy. In other words, this threshold is meant to be a guide to an informed user, not a precise prediction.

Fontsize This button pops up a submenu you can use to select the size of the font used to display titles, legends, and so on.

Color This button pops up a submenu you can use to select the colors of various browser features. See figure 2.8.

Save Save the preferences to the file from which they were loaded. See page 31.

Load Palette Opens a dialog allowing you to load a palette for mapping. See figure 2.9.

The colormap is an ASCII file containing a standard Matlab RGB colormap. This is a 3-column matrix with values between 0 and 1. The

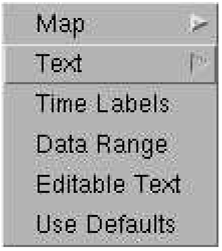
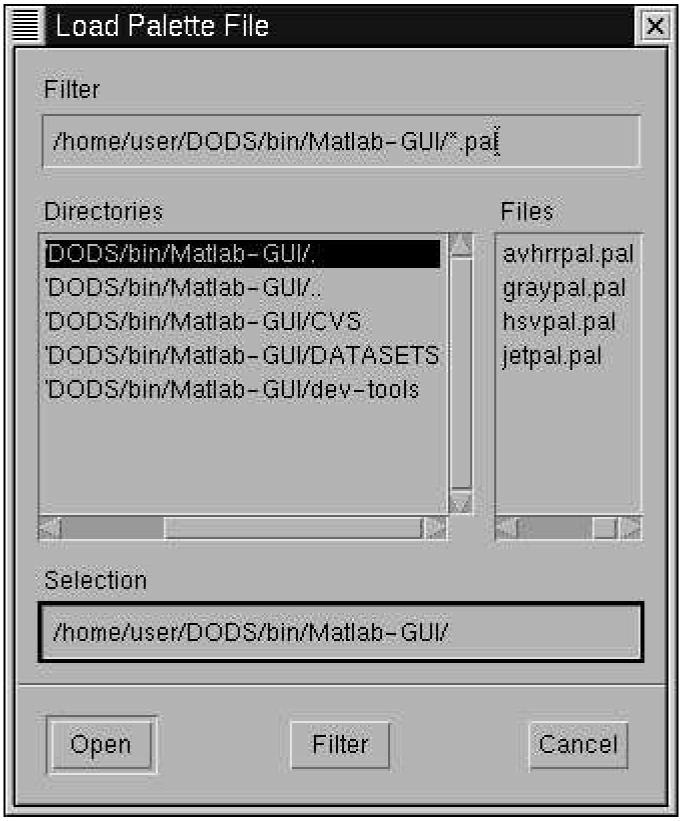


Figure 2.8: Color Menu



Filter: /home/user/DODS/bin/Matlab-GUI/*.pal

Directories:
'DODS/bin/Matlab-GUI/.
'DODS/bin/Matlab-GUI/..
'DODS/bin/Matlab-GUI/CVS
'DODS/bin/Matlab-GUI/DATASETS
'DODS/bin/Matlab-GUI/dev-tools

Files:
avhrrpal.pal
graypal.pal
hsvpal.pal
jetpal.pal

Selection: /home/user/DODS/bin/Matlab-GUI/

Figure 2.9: Color Palette Dialog

columns represent the relative strength of the red, green, and blue components, respectively. Matlab also has a number of built-in colormaps (type `help colormap` at the Matlab prompt for more information). These can be specified from the Matlab command line.

The `avhrrpal.pal` file (the default color palette) consists of 255 lines like this:

```
...
0.2000 0.0000 0.3000
0.3700 0.0000 0.5000
0.4516 0.0000 0.7000
0.4548 0.0323 0.8710
0.4548 0.2323 1.0000
...
```

The file represents a $255x3$ array. You can think of it as a list of 255 colors, each of which is specified as a combination of red, green, and blue values (from left to right). Each value represents the fraction of intensity, with 1.0 equal to full intensity of that color component.

To use one of the colormaps that come with Matlab, you can create a palette in an external file with the following steps:

❶ Create a color map with the following command:

```
>> newpalette = hsv(256);
```

Any of the following could substitute for the `hsv` in the above example. They are all colormaps supplied with Matlab:

hsv Hue-saturation-value color map.

hot Black-red-yellow-white color map.

gray Linear gray-scale color map.

bone Gray-scale with tinge of blue color map.

copper Linear copper-tone color map.

pink Pastel shades of pink color map.

white All white color map.

flag Alternating red, white, blue, and black color map.

lines Color map with the line colors.

colorcube Enhanced color-cube color map.

jet Variant of HSV.

prism Prism color map.

cool Shades of cyan and magenta color map.

autumn Shades of red and yellow color map.

spring Shades of magenta and yellow color map.

winter Shades of blue and green color map.

summer Shades of green and yellow color map.

❷ Save the result into an ASCII file:

```
>> fid = fopen('newpalette.pal');
>> fprintf(fid,'%g %g %g\n',newpalette');
>> fclose(fid);
```

Set GUI Server You can use the dialog summoned by this button (see figure 1.9) to update your master list of datasets from some other source than the DODS central archive.

Save As If you have changed preferences such as color, font, threshold or anything else, you can save these values by selecting this option. When the OPeNDAP Matlab GUI is next started, these same values will be read from the startup file, by default called `browsopt.m`. You can control the name with the Save As dialog box. To reload a preference file, use the **Preferences,Load Prefs File** button. Browser preference files are simple Matlab scripts and may be edited by hand.

The browser options saved with this command are described in Section 2.3 on page 32.

Load Prefs File This button allows you to load a set of user preferences saved from a previous browser session. Clicking the button brings up the dialog.. The browser will execute the given preferences file, expecting it to be a file such as are saved by the **Save Preferences** command. Note that the preferences file *must* be somewhere on your Matlab path. If you are unsure, type `path` at the Matlab prompt and examine the result.

The browser preferences loaded with this command are described in Section 2.3 on page 32. Saves the current preferences in a file of your choosing.

Use Defaults Selecting this option will restore to their default values:

- The size and screen position of the browse windows
- The resolution factors (multiples of the dataset stride available through the **Resolution** menu)
- The font size
- The color scheme for the user display
- The time format (this is year/month/day)
- The axis limits
- The data size threshold (1Mb)
- The color limits (0 255)
- The color palette.
- The limits of the four browser axes (time, depth, latitude, longitude).

Clicking this button also clears the ranges, dataset and variable selections. See Section 2.3 on page 32 for more information about the options.

2.2.5 DODS Help

This menu allows you to see several short hints addressing frequently asked questions. You can also select **DODS GUI User Manual**, which will start an html browser on your system, reading a hypertext version of this document. Your

Matlab file, `docopt.m` must be properly configured to use the appropriate browser. See your system administrator if it is not set correctly. Similarly, you can edit the `browsopt.m` file (part of the OPeNDAP Matlab GUI) to control the URL submitted to the browser. The default is to read the copy of the documentation on the DODS server: `http://opendap.org/support/docs.html/user/mgui-html/`.

If you have a local copy of the hypertext documentation, you may want to refer to that copy instead, to make browsing the text faster.

2.2.6 Display Data

The data display dialog is described in Section 1.2.4.

There are two pull-down menus on the data display dialog:

Edit

This menu contains three buttons:

Add New Plot Create a new, empty, plot window.

Delete Plot Delete one of the existing plots.

Delete All Plots Delete all the existing plots.

Apply to All

If you have a set of plots and wish to operate on them all simultaneously, you can change some aspect of the Display Dialog, and then select the appropriate button on this menu, and that aspect of all the plots will be updated.

2.2.7 Bookmarks

The bookmark editor looks like the bookmark selection window, but with a different menu bar. The menu bar has three pull-down menus, File, Edit and View.

File

The bookmarks File menu looks like figure 2.10. Use it to manage the bookmarks file itself. It has the following buttons:

Open Bookmarks File Select a different bookmarks file to edit.

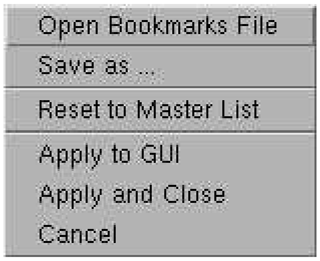


Figure 2.10: Bookmarks File Menu

Save As Save this bookmark file under a different name.

Reset to Master List Restore the current bookmarks to the "default" state, which is defined by the current version of the master list downloaded from the GUI server. (See Section 1.4 on page 15.)

Apply to GUI Apply the changes to the OPeNDAP Matlab GUI, but don't close the bookmark edit window.

Apply and Close Apply the changes to the OPeNDAP Matlab GUI, and close the bookmark edit window.

Cancel Close the bookmark edit window without applying the bookmark changes to the OPeNDAP Matlab GUI.

Edit

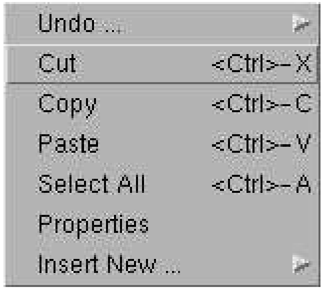


Figure 2.11: Bookmarks Edit Menu

The bookmarks File menu looks like figure 2.11. Use it to manage the bookmarks file itself. It has the following buttons:

Undo Allows you to undo bookmark edits. You can undo either the last change you made, or all the changes since you last applied changes to the OPeNDAP Matlab GUI.

Cut Cut the selected bookmark or range of bookmarks. Store them in an invisible buffer, to be pasted if desired.

Copy Copy the selected bookmark or range of bookmarks to an invisible buffer, to be pasted if desired.

Paste Copy any contents of the invisible copy buffer to the bookmark list.

Select All Select all the bookmarks in the list.

Properties Display the properties of the highlighted bookmark. See figure 1.12.

Insert New Insert a new bookmark or bookmark folder into the bookmark list. See Section 1.5 on page 16.

View

There is only one option under the View menu, which is **Master List**. Selecting this button provides a view of the current version of the master list, from which entries can be copied and pasted.

2.3 Browser Preferences

The OPeNDAP Matlab GUI uses a set of preferences to control several aspects of its use, including selection ranges, font sizes, colors and so on. These preferences can be loaded with the **Preferences,Load File** button, and saved with the **Preferences,Save** or **Preferences, Save As** buttons (See page 29). These commands load and save a Matlab script whose function is to define the values of the various preferences. You may find it convenient to edit the file by hand; it is only an ASCII file of Matlab commands.

If the preferences file is called `browsopt.m`, and is stored somewhere in the Matlab path, it will be loaded automatically when the browser starts up.

Following is a list of all the options a preference file may contain.

`timetoggle` This option (controlled with the **Display,Time** button) controls whether times are displayed as Year/Month/Day or as the Year/Yearday, where Yearday is the ordinal number of the day of that year.

`dods_colors` This is the set of colors, expressed in RGB values in a 10×3 array, used to display various aspects of the browser window. The colors are used as follows:

`dods_colors[1]` Default button color.

`dods_colors[2]` Axis and window background.

`dods_colors[3]` Axis and window foreground.

`dods_colors[4]` Non-interactive window and axis text.

`dods_colors[5]` Edit boxes, where users may type.

`dods_colors[6]` Edit box text and button labels.

`dods_colors[7]` The **Get Data!** button.

`dods_colors[8]` Text and axis lines that change.

`dods_colors[9]` The “whizzo” color, used for drawing attention to special options. The Help is displayed in this color, among others.

`dods_colors[10]` The selection range box color.

`fontsize` The point size of the displayed type.

`figsizes` The default size and position of the browser window, expressed in the pixel locations of the upper left and lower right corners.

`axes_vals` The starting axis values for the displayed axes, expressed four pairs of minimum and maximum values for (in this order), longitude, latitude, depth, and time:

```
axes_vals=[minLon maxLon minLat maxLat minDepth maxDepth minTime maxTime]
```

`ranges` Specifies the default selection ranges, in the same manner as the `axes_vals` preference.

`num_rang` An eight element vector of ones and zeros, indicating which of the selection ranges have been set by the user. A one means the corresponding item in the `ranges` variable has been set by the user. If not all the ranges are set, you can’t ‘get data’ or ‘get details’. If a dataset is selected, and the depth or geographic ranges have not been set, these are set to those of the dataset.

`color_limits` A two element vector indicating the data values to assign to the top and bottom colors in the data palette.

`palettefile` The name of a file containing the data palette. This file must be in the Matlab path.

`check_level` This is the number of megabytes for the “Get Data” threshold. (Set with the **Preferences,Get Data Threshold** button, see page 25 for more information.)

`available_resolutions` This is a vector containing the available resolutions offered with the Resolutions menu, expressed as multiples of the dataset’s basic resolution. The default value is:

```
available_resolutions = [32 24 16 12 8 4 3 2 1];
```

This option is not changable through the browser, but you can edit the preferences file directly to change it.

`manURL` The URL of the OPeNDAP Matlab GUI manual. By default, this points to `http://opendap.org/support/docs.html/user/mgui-html/`. If you have a local copy of the manual served, you can substitute that URL.

`guiserver` The URL to use for updates to the dataset metadata files.

`userlist_file` The name and location (on the local file system) of the user's bookmarks file.

3
Adding Your Own Datasets to the OPeNDAP Matlab GUI

There are a few reasons you may need to add a dataset to the OPeNDAP Matlab GUI.

- You may be interested in an OPeNDAP data source that hasn't been put into the OPeNDAP Matlab GUI already,
- You may want to add to or modify some of the metadata supplied for one of the OPeNDAP Matlab GUI datasets, or
- You may have some of your own data you want to make available via OPeNDAP.

It is possible to add datasets to the ones that come with the OPeNDAP Matlab GUI, and to edit or add to the metadata for the datasets that do come with it. This will require some minor programming in Matlab, but much of the time it's fairly simple.

Start with getting your data served by an OPeNDAP server.

The OPeNDAP Matlab GUI is designed for the retrieval and display of data served by an OPeNDAP server. This means that in order for the OPeNDAP Matlab GUI to read your data, there must be an OPeNDAP server, on a machine connected to yours somehow, with access to that data. The machine serving the data could be connected to yours on a local network or by the internet, or it could even be on your own computer. Either way, the data must come from an OPeNDAP server.

Making your data available by an OPeNDAP server is a fairly straightforward affair for datasets stored in one of the several supported formats. The server itself is a typical web server (`httpd`), adapted with a set of OPeNDAP CGI programs. To set up a server for your data, please refer to *OPeNDAP User Guide*. You can find it and all the other OPeNDAP documentation at OPeNDAP Home page.

Once a OPeNDAP server is set up to serve your data, the dataset may be added to the OPeNDAP Matlab GUI by providing the following pieces of information. Many datasets can use minimally modified versions of one or more of these pieces from existing datasets. Each of these are detailed in the sections that follow.

- An “archive” M-file, to describe the data in the dataset.
- A *Get* function that accepts a data range and retrieves data from the remote dataset. This is also called the `getxxx` function. Most often it’s possible to use (or adapt) one of the existing `getxxx` functions.

3.1 The Archive File

The archive M-file is a Matlab program that sets a number of variables describing various aspects of some dataset. In order for the browser to display the geographic and temporal bounds of a dataset, they must be specified in this file.

The archive M-file describes a dataset to the GUI.

Many archive M-files exist in the DATASETS directory. If your dataset resembles (in data structure) one of the datasets already in the OPeNDAP Matlab GUI dataset collection, it may be simplest to copy the corresponding archive M-file and modify it (and its corresponding `getxxx` function) to suit the new dataset. For example, the Reynolds Weekly Sea Surface Temperature data are stored in a single file, containing a global one-degree grid of temperature for each week covered. If your data are stored as a similar grid, it will likely be easier to copy the Reynolds M-files and adapt them than to write new ones from scratch.

3.1.1 An example

The following archive M-file describes the Levitus ocean atlas dataset. (You can find it in the DATASETS directory, called `woa94mon.m`.) Many of the M-files follow the same basic structure. Most of the variables set here are necessary for the operation of the browser. A few of them, however, are set only for the convenience of the `getxxx` function (called `getrectg` here). If you have to write a new `getxxx` function, you may not need them.

```
%  World Ocean Atlas Monthly Climatology

% FUNCTIONS -- REQUIRED
GetFunctionName = 'getrectg';

% VARIABLES -- OPTIONAL
Server = ['http://ferret.wrc.noaa.gov/cgi-bin/nph-nc',...
   '/data/Ocean_atlas_monthly.nc'];
axes_order = [4 3 2 1];
% force time axis to correct day [Y M D H M S]
Time_Offset = [0 0 -1 0 0 0];

% VARIABLES -- REQUIRED
LonRange = [20.000 380.000];
LatRange = [-90.000 90.000];
TimeRange = [1800 str2num(datestr(date,10))];
DepthRange = [0.0 1000.0];
DepthUnits = 'Meters';
Resolution = 111.0;

DataName = 'Water Column - World Ocean Atlas 1994 Monthly - PMEL';
SelectableVariables = str2mat('Sea_Temp', 'Salinity');
DodsName = str2mat('TEMP','SALT');

Acknowledge = sprintf('%s\n%s\n%s\n%s\n%s', ...
Comments1 = sprintf('%s\n', ...
```

The following variables are for use by the browser.

`GetFunctionName` This is the name of the `getxxx` function that will return both the catalog information as well as the actual data. See Section 3.2 on page 42. The `getrectg` function is a common one used to return gridded multi-dimensional data. The `getxxx` function must be available in your `MATLABPATH`.

`LonRange`, `LatRange`, `TimeRange`, `DepthRange` These four two-element vectors describe the spatial bounds of the dataset's data. This dataset is stored on a grid; the first elements of these range vectors indicate the positions of the first row (column) in the dataset. The latitude and longitude are given in decimal degrees. Times are given in decimal years. In the example here, the data is a climatology, an average over many years This means that the entire time range isn't particularly meaningful (which is why it's set to the maximum), though being able to select by time is important, so you can select the appropriate month.

`DepthUnits` This is a character string naming the units in which the data stores depth information. The OPeNDAP Matlab GUI has no knowledge of units, so it's really just an axis label.

`Resolution` This is a value, in kilometers, of the distance between adjacent horizontal values in the grid. The OPeNDAP Matlab GUI is only set up for grids with the same resolution in the X and Y dimensions. For georeferenced grids, the resolution refers to the distance between adjacent X values at the equator.

`DataName` This is the name of the dataset, to be used in the bookmark list.

`DataRange` An array of data pairs, giving an upper and lower bound to the data. The entries in the array correspond to the entries in the `DataNull` array. This matrix is used by the browser plot functions to create a consistent color scale for the imaging of multiple returning data slices within one dataset. It is *only* used for `image` and `pcolor` plots. An entry of `[NaN NaN]` indicates the total range of that variable is not known and the browser should scale each image using the local minimum and maximum. The temptation to write in non-NaN scale entries for longitude should be avoided, since the browser converts longitude to match the current window view upon return. Not required.

`Comments` A detailed description of the dataset. To avoid Matlab limitations on the length of a command, data descriptions may be broken up into several input lines, which are then concatenated:

```
Comments1 = sprintf('%s\n', ...
    '        Levitus (1982) Climatological Atlas of the World Ocean', ...
    '    ', ...
    'Annual climatologies of salinity and temperature on 1x1 degree ');

Comments2 = sprintf('%s\n', ...
    '    ', ...
    'Levitus, S., Climatological Atlas of the World, NOAA/ERL GFDL', ...
    'Professional Paper 13,  Princeton, N.J.,  (NTIS PB83-184093),', ...
    '1982.');
Comments3 = sprintf('%s\n', ...
    '    Time_Period_of_Content = ', ...
    '         Beginning_Date:1935 ', ...
    '         Ending_Date:1978 ');

Comments = [Comments1 Comments2 Comments3];
```

`Acknowledge` A text string indicating how uses of this data should acknowledge its originator.

`Data_Use_Policy` A text string indicating what limitations are set on the use of this data.

3.1.2 Getrectg function variables

A few variables aren't used by the OPeNDAP Matlab GUI, but are used by `getrectg`, the `getxxx` function used here. They are documented here because the `getrectg` function is flexible enough to be used for many different gridded datasets. See section 3.2 for more information about the `getxxx` API.

`Server` The server from which the data in question will come.

`axes_order` The dataset may store its variables in a different order than the OPeNDAP Matlab GUI expects them. You can use this vector to rearrange the variables before `getrectg` returns them.

`Time_Offset` Data providers often disagree whether a time value marks the beginning of the interval, or the middle, and whether the first unit is number one or number zero. You can use this six-element vector to provide a constant with which to adjust the time values from the dataset. The elements are adjustments to the year, month, day, hour, minute, and second, in that order. For example, the offset vector in the example instructs `getrectg` to subtract one day from the day values in the dataset before returning them to the OPeNDAP Matlab GUI.

`LongitudeName`, `LatitudeName`, `TimeName`, `DepthName` These identify the names of the spatial quantities as used in the dataset. Data requests to the OPeNDAP server will be made with these names. The variables may contain empty strings if no meaningful information is returned by the server about any of these variables.

`SelectableVariables` The names of the variables in the dataset. These are the names that will be used in displaying the data to a user. Wherever possible, you should use a name that is already in use in the browser. For example, ‘Salinity’ is already in use. If you are adding a dataset containing salinity measurements, don’t call it ‘Salt’, unless there is some reason you don’t want to see your dataset in a list of datasets containing salinity measurements.

`DodsName` This is the name of the selectable variable as used by the dataset server. Data requests to the dataset server will be made using these names. For example, though the OPeNDAP standard is to use ‘Salinity’ (see above), the dataset server itself uses ‘SALT’, and requests to that server must be made with that name. Each entry on this list must exactly correspond to an entry on the `SelectableVariables` list.

`DataNull` This is a vector of null values, indicating the “no data” value for each of the spatial variables (in order: Time, Longitude, Latitude, and Depth), and each of the selectable variables (in the same order as the `SelectableVariables` list. As data from the remote dataset is interned into Matlab, any values matching the appropriate element of this array will be changed into the Matlab NaN (“not a number”) value.

`DataScale` A 2-column array, to be used in a linear conversion from the data as it is received from the remote dataset. The first column is the intercept and the second the slope. The rows the array correspond to the entries in the `DataNull` array. Entries of `[NaN NaN]` or `[0 1]` do no scaling. Within the

OPeNDAP Matlab GUI an effort is made to present the data to the user in SI units. For example, density would be presented in kg/m^3. Not required.

NOTE: While technically not required, `DataScale` and `DataNull` are in practice almost essential for the return of intelligible results to the user. In addition, for datasets that use extreme values as a bad or missing data flag, (e.g., -9e42) , no meaningful plotting will be able to be done within the browser unless `DataNull` is declared.

3.1.3 Another example

In the Levitus Ocean Atlas example above, the time axis is faked using the start time, the stop time and the interval. Some datasets consist of many similar files, corresponding to different times. These datasets can benefit from using a catalog server to indicate the time at which data was measured.

A catalog server is an OPeNDAP server, serving a file that associates time data or other data values with OPeNDAP data URLs. (See the *OPeNDAP Server Installation Guide*for more information about setting up servers.)

For example, the ASCII reply (see *OPeNDAP User Guide*) of the catalog server corresponding to the Total Ozone Mapping Spectrometer (TOMS) project looks like this:

```
TOMS.type, TOMS.year, TOMS.month, TOMS.day, TOMS.DODS_URL
"ep", 2000, 1, 1, "http://daac.gsfc.nasa.gov/..."
"ep", 2000, 1, 2, "http://daac.gsfc.nasa.gov/..."
"ep", 2000, 1, 3, "http://daac.gsfc.nasa.gov/..."
"ep", 2000, 1, 4, "http://daac.gsfc.nasa.gov/..."
```

The OPeNDAP Matlab GUI (the `getrectg` function, actually) contains a provision for accessing this kind of catalog, but nominating a function to deal with this data.

The archive M-file for the TOMS project looks like this:

```
%      Total Ozone Mapping Spectrometer Daily Dataset

% FUNCTIONS -- REQUIRED
GetFunctionName = 'getrectg';

% FUNCTIONS -- OPTIONAL
Cat_m_File = 'ffcsquery';
CatalogServer = [ 'http://daac.gsfc.nasa.gov/daac-bin/nph-ff/DODS/catalog/', ...
                'toms-cat.dat'];

% VARIABLES -- REQUIRED
LonRange = [-180 180];
LatRange = [90 -90];
TimeRange = [1978.83287671233,2000.5];
DepthRange = [0 0];
DepthUnits = '';
Resolution = 138.75;

DataName = 'TOMS Global Level 3 Daily Gridded - GSFC';
LongitudeName = 'LONGITUDE';
LatitudeName = 'LATITUDE';
TimeName = '';
DepthName = '';
SelectableVariables = str2mat('Total_Ozone','Reflectivity');
DodsName = str2mat('TOTAL_OZONE','REFLECTIVITY');

DataNull = [nan nan nan nan 0 999];

Data_Use_Policy = '';

Acknowledge = '';

Comments1 = sprintf('%s\n', ...
```

The `ffcsquery` function is packaged with the OPeNDAP Matlab GUI and is called by `getrectg`, using the `CatalogServer` identified. To adapt this, you could create a catalog server, and use `ffcsquery`, or it might be simpler to write a catalog function (`Cat_m_File`) that cheats and has the URLs and time values encoded within.

3.2 The "Get" Function

The `getxxx` function, another Matlab program, serves as the connection between the OPeNDAP browser and individual datasets, hiding their idiosyncracies and providing a uniform interface between the browser and the data. Some get functions may serve for more than one dataset, such as `getrectg.m`, a function that works for a number of gridded datasets. The `getxxx` function can be named anything, but convention dictates that the name be of the form `get`*xxx*`.m`, where *xxx* is some sequence of characters that indicates the dataset (or class of datasets) for which the `getxxx` function is used. Note that several `getxxx` functions are provided with the OPeNDAP Matlab GUI; be careful to avoid those names.

The browser uses the getxxx function both to get data, and information about data.

A `getxxx` function has 3 modes of operation (or 3 subroutines). Each mode receives the same input arguments, although for some arguments and some datasets, not all the incoming information will be used. The modes differ in their action, *and in the data they return to the calling function.* The input arguments to the `getxxx` function are as follows:

```
function [arg1,...]  = getxxx (mode, ranges, dataset, vars, stride, num_urls,
        georange, variables, archive, whichurl, url_info)
```

mode A string argument which may be `cat` (short for 'catalog'), `datasize`, or `get`. The return values of this function differ depending on the *mode* argument. They are outlined in the following sections.

ranges A 4x2 matrix containing the current user-selected longitude, latitude, depth and time ranges. The first column is the minimum and the second the maximum.

dataset An index number into an ordered list of datasets contained within the browser. You can use this to check whether the dataset is the same one as was specified in the previous request.

vars A vector of indexes into an ordered list of user-selectable variables contained within the browser. You can use this to check whether the request has changed since the last request made, but use the variable list for the names of the requested variables.

stride A user-selected subsampling interval for gridded datasets. A stride value of two requests every other data value, three gets you every third, and so on.

num_urls The number of URLs in a data request.

georange A two-element vector indicating the geographic orientation and ranges of the browser display — typically this will be `[-180 180]` or `[0 360]` but can be set to other values.

variablelist A character array of the variable names selected by the user.

archive The name of the *archive*`.m` file for the selected dataset.

whichurl – For a “catalog” mode request, this is a structure containing information about the data to be found at the URL. For a “get” mode request, this is an index into a list of URLs that comprise a data request. The list will have been returned by a `cat` mode request to the `getxxx` function.

In some modes, a particular input argument will not contain valid information. For example, the input number of URLs is not valid for a catalog request, since part of the function of the catalog mode is to determine the number of URLs needed to satisfy a request. However, all arguments are passed in each time for the sake of consistency.

3.2.1 `getxxx` Function Modes

The `getxxx` function operates in three different modes. One mode returns URLs corresponding to a dataset, another returns estimates of the request size, and the third actually retrieves data. The following sections describe the three modes.

Catalog mode

In *catalog mode* (or `cat` mode) the browser is requesting information about location in time and space of the indicated dataset. The function takes the location of an indicated geo-spatial region (using the *ranges* matrix), and returns a list of the points where the queried dataset contains data.

Catalog mode is for determining the location of a dataset in space and time.

The return arguments from such a request are:

```
[ x, y, z, t, n, URL ] = getxxx( ...  )
```

x, y, z, t Are one-dimensional vectors containing the locations and times corresponding to individually addressable URLs available to the browser user. (Latitude and longitude in decimal degrees, depth in meters from the surface, time in decimal years.) These vectors will be used to plot tick marks on the browser display, and the user will be able to select one or more of these tick marks to get data. All of the arguments must be returned, although those that are not appropriate for a given dataset may be returned empty.

For example, a dataset containing buoy data might return a series of (x, y) pairs, a vector of depths indicating instrument depths, and an empty time vector. The browser will display the buoy locations, and the depth levels, allowing the user to select specific buoys or specific depths. A dataset containing satellite images taken daily might return only a time vector, indicating the times of each image.

n is the total number of URLs identified by the catalog function. The browser will call the `getxxx` function (in `get` mode) this many times to retrieve all the data referenced in the `cat` mode request.

URL A URL directed at the dataset, or at a catalog server, if one was used to make the `cat` mode request. This is displayed in case you would like to make this request (or a related one) independent of the OPeNDAP Matlab GUI, say with `loaddods`.

Sometimes the only way to provide the data requested in a catalog request is to contact the dataset's OPeNDAP server for data, or contact a catalog server for information about the dataset (see Section 3.2.2 on page 47). For some datasets, such as monthly climatologies, an internet request does not need to be made to satisfy the request — instead a Matlab function can be written which examines the user-selected time and geographic ranges and determines how many of the climatologies fall within these ranges. For other kinds of datasets, such as satellite datasets stored along a groundtrack, which repeat in a predictable but irregular pattern in time and space, reference satellite groundtrack locations and data densities can be stored in a local `.mat` file.

During a catalog request, the *dataset*, *vars* and *ranges* values should be stored internally (i.e., in a global variable). This is so that for future `get` and `datasize` mode requests, the `getxxx` function will be able to determine whether or not the existing catalog is up-to-date. If a `get` or `datasize` request is made where the data parameters have changed from the last catalog request, the `getxxx` function should make a new catalog request internally, before satisfying the new request.

The `getxxx` function should also internally store an ordered list of URLs based on this catalog request, or enough information to recreate those URLs when requested by the browser.

Datasize Mode

The browser uses datasize mode to estimate the size of a data request.

The function of the *datasize mode* is to provide the browser with an estimate of the size of a formulated data request as it will appear in the Matlab workspace (as a double-precision float). When a data request is issued, the browser checks to see if the estimated datasize is below a threshold level (a level which defaults to 1Mb but can be set by the user). If above that level, the browser requires the user to affirm the data transfer. This is to prevent users from unintenionally requesting very large data transfers. (See Section 2.2.4 on page 25.)

The previous values (from the most recent catalog request) of the *dataset*, *variable* and *ranges* variables should first be compared with the new values, to see if the catalog request is current. If any of these values has changed, the `getxxx` function should first call itself in `cat` mode to get the number of data points or URLs or whatever the pertinent operative may be that causes the size of a data request to change.

The return arguments from a `datasize` request are:

```
[ datasize, num_urls ] = getxxx [ ...  ]
```

datasize The approximate size (in megabytes) of the data that will be returned by the specified data request.

num_urls The number of URLs in the request. (This is in case a new catalog request was required during the `datasize` request. If no new catalog request was required, the input *num_urls* argument can simply be returned.)

Get Mode

The `get` mode request (also “data request”) queries the remote dataset for data. The browser makes a `datasize` request before each data request, and a `cat` request before each `datasize` request. This series is to be repeated if the user changes any of the search parameters, like the geographic ranges or times.

And the get mode is for getting data.

The data request therefore need only operate on the ordered list of URLs generated by the most recent `cat` request — the browser simply increments the *url_index_number* from one to the total number of URLs and waits for each result. However because the input information from the browser (such as the text list of selected variables) is passed in each time, the `getxxx` function does not actually have to construct a URL until this point[1].

Sometimes it is appropriate to return a count of one URL to the browser while actually using more than one, internally. This can occur when, for example, the user requests all data from longitudes 40W to 40E from a gridded dataset with global coverage but the grids are stored with a split at 20E (that is, the longitude in the dataset ranges from 20E to 380E). The `getxxx` function in this case must either transfer the whole dataset, then subset and reformat it, or dereference two URLs, one from 40W to 20E and one from 20E to 40E, and combine the result. In other words, the number of derefernced URLs returned by the `getxxx` function should match what the browser expects based on the result of the catalog mode request.[2]

The data should be scaled according to the *DataScale* information in the *archive*`.m` file before being returned.

The return arguments from the `get` mode are as follows:

```
[ data, sizes, names, index, URL ] = getxxx[ ...  ]
```

[1] This is a time-saver. It is not necessary to construct URLs that won’t be used.

[2] If, on the other hand, the catalog function returned a count of two URLs for this particular request, those URLs should be dereferenced and the results returned to the browser separately.

data is a $M \times 1$ vector of data, where M is the total number of data measurements returned. That is, all the data are reshaped to a single column, with their original sizes recorded in the 'sizes' argument.

sizes is an $N \times 2$ array of array dimensions, indicating the sizes of the returned arrays. This requirement must be met: `sum(sizes(:,1).*sizes(:,2)) = M`.

names is a list of character strings with the same number of rows as *sizes* — each variable as it is unpacked will be given the corresponding name. The current convention is to rename returning variables with the common names used in the browser. If 'Longitude' and 'Latitude' (exactly so) are not present, the browser will not allow plotting on the browse window. Some of the `getxxx` functions now written wll create longitude and latitude vectors for gridded datasets using the information in the archive M-files. This is an occasionally useful tactic for troublesome datasets.

> **NOTE:** For the benefit of Matlab Version 4 users, variable names should be no longer than 19 characters.

index The index number is the index into this list:

```
Time
Longitude
Latitude
Depth
SelectableVariables[1]
.
.
.
SelectableVariables[n]
```

The browser uses this index into the `DataRange` array in the *archive*`.m` file for mapping multiple images within the same dataset to a unified color scale, and into the `DataScale` array for scaling the returned data.

URL This variable should contain the URL dereferenced to retrieve the data provided as a result of the `get` request.

This variable is to be a single string — if several URLs have actually been dereferenced to satisfy what the browser has treated as a single URL, they should all be included, concatenated end-to-end, each separated by a space, i.e.:

If *TimeName* in the *archive*`.m` file is empty because the server doesn't actually return a time, you can insert a time made up from the catalog request. However this is not required; every `getxxx` function should be able to handle empty variables correctly.

If any of the variable names resulting from dereferencing a URL is longer than 19 characters, the data contained in that variable must be requested separately for Matlab 4 users and the result mapped to the first 19 characters of that variable. That is, you must first do:

```
loaddods(URL)
```

then

```
[longname(1:19)] = loaddods(URL);
```

This is a workaround for a bug concerning long variable names interned into Matlab v4.

3.2.2 `getxxx` Function Tools

OPeNDAP provides several tools with which to construct your `getxxx` function. The tools fall into two broad classes:

- `loaddods`, the function used to dereference an OPeNDAP URL, and
- a set of Matlab utility functions, useful for the sorts of operations likely to be needed by a `getxxx` function.

The `loaddods` Function

loaddods is at the heart of every getxxx function.

The loaddods[3] function is the OPeNDAP client for the Matlab browser. The function accepts a URL from the user, sends it out over the internet, and creates Matlab scalars, vectors, and matrices to hold the data that is returned from that request. The `loaddods.m` help file contains several examples of its use. (Type `help loaddods` at the Matlab command prompt.)

The function is called like this:

```
function values = loaddods([switches,] URL [per-url switches] [,
    URL [per-url switches]])
```

switches These options are all off by default except for `-v` which is on by default.

`-v`: Verbose output. (This option is on by default; use `+v` to disable it).

[3] loaddods is a Matlab MEX-file, and appears in your directories as `loaddods.mex`*xxx*, where *xxx* is a suffix defined by Matlab for a particular machine architecture. For example, `loaddods.mexglx` would be for a GNU/Linux machine. There is also a `loaddods.m` function, which contains a help message for `loaddods`.

`-g`: Use the Tcl/Tk progress and error reporting GUI. (This displays a bar graph showing transmission activity, and is not to be confused with the OPeNDAP Matlab GUI. See *OPeNDAP User Guide* for more information about this feature.).

`-k`: This switch causes `loaddods` to concatenate two or more variables with same name. When reading from multi-file datasets, use this option to concatenate the values read from several accesses into a single Matlab variable. This is intended for `getxxx` functions which build up lists of URLs and pass them to `loaddods` for evaluation.

You *must* supply the URLs whose variables are to be concatenated in a single invocation of `loaddods`. The `-k` switch will not work across two or more calls to loaddods. That is,

```
loaddods('-k', '<URL1> <URL2')
```

will work, while calling `loaddods` twice with `-k` (once for URL1 and once for URL2) won't work.

`-n`: This command switch causes Matlab variable names to be built using the complete name of the variable, including all the constructor types which contain it. Without `-n`, variables are named using the name of the leaf variable. This means, for example, that two separate *Structures* who each happen to have a member with the same name will create a name conflict.

`-s`: Return a vector of variable names along with the variables themselves. The `-s` option can only be used when assigning the values accessed by `loaddods` to a vector. If `-s` is used in this situation, then the first variable in the vector will be set to a vector of strings which name the successive variables.

`-V`: Print version information for loaddods and any programs it uses.

`-t[abclmpstu]`: Enable HTTP tracing (only for the hardcore...).

`-F`: Force all string variables to be translated to float. The `-F` option is needed when reading from datasets that return as string values that should be interned as numbers.

> **CAUTION:** This is a change from previous versions of `loaddods` and introduces potential incompatibilities with version 1.5 and prior. In other words you'll need to change your software if it used version 1.5 or earlier.

`-p`: Recognize the DDMMSS strings and convert them (takes precedence over `-F`).

`-A`: The `-A` option provides a way to access the DAS object of a dataset. Given with a URL, `-A` will cause all attributes to be interned in the Matlab workspace. Given with `-c` only the contents of named attribute *container*[4] in a dataset will be interned. The latter is much safer since many DAS objects contain long names and names that repeat in several containers. Choosing an attribute container known to have ‘Matlab safe’ identifiers is a good idea. (See *OPeNDAP User Guide* for more information about the DAS structure, and attributes in general.) Note that the attributes are interned into the Matlab workspace under their own names, not the name of the container[5].

> **NOTE:** While DAS objects are separate from the DDS (data) objects in OPeNDAP, once loaddods reads the information from the writeval client program they are treated no differently than data objects. Each attribute container is interned as if it was a Structure variable in the DDS object.

per-url switches These options are all off by default.

`-r` *`var`*`:`*`new name`*.: Rename a variable. The `-r` option provides a way to rename variables so that variables read will not overwrite ones already present in a Matlab session.

`-c` *`expr`*: Supply a constraint without using the `?` notation.

URL The `loaddods` function should be called with a OPeNDAP URL which references data, in which case the command will access the dataset and create Matlab data structures to hold those data. If the *URL* argument is a `*` or `?`, `loaddods` will cause Netscape to start up and open the default OPeNDAP URL-builder. If the `*` or `?` is followed by a URL, then that URL will be used instead of the URL builder. Finally, when called without an argument, the command will wait for a URL to be passed to it by Netscape or another program[6].

loaddap

The `loaddods` function is at the core of the OPeNDAP Matlab GUI. However, for backwards compatibility, it was originally written not to take advantage of the advanced data structures now available in Matlab. The `loaddap` function is a newer version, that makes use of Matlab data structures in an attempt to preserve the structure of datasets.

For example, if the target dataset contains three arrays in a structure, the `loaddods` function will bring them in as three arrays. The `loaddap` function will

[4] Note the distinction between attributes and attribute containers. A variable called “v” in some dataset will correspond to an attribute container called “v” in the dataset’s DAS. That container may contain attributes such as “precision” or “scale factor” or “units.”

[5] This is due to the variable name length limitations of Matlab.

[6] See the OPeNDAP `urlqueue` program for information about this feature.

import the data as three arrays in a structure, corresponding to the original arrangement.

The use of `loaddap` is identical in all other respects to `loaddods`.

If you find that the OPeNDAP Matlab GUI is more work than is necessary for your application, and you use the command-line client, you might consider using `loaddap`.

Utility Functions

Several functions used in the OPeNDAP Matlab GUI are general enough to be of use to people who have to write new `getxxx` functions[7]. This section contains a short list of some of the functions likely to be useful. The list is not complete, since the whole collection is fairly fluid, as new datasets are constantly being added to the archive. We encourage function writers to browse the OPeNDAP Matlab GUI directories before starting in on what may seem like a common problem.

These functions have been useful in writing the getxxx functions provided with the GUI. They may be useful to you, too.

Many of the utility functions are listed below. Use the Matlab `help` command for details of their invocation.

`day2year` Converts an integer day into a decimal year.

`dods_dbk` Removes trailing blanks from a string, and translates http escape sequences (e.g. `%20`) into underscores, so that `Surface%20Temperature` gets translated into `Surface_Temperature`.

`dods_ddt` (“de-dot”) Removes the parent structure name from compound names. For example, `Cast.Temperature` is changed to `Temperature`. Actually, it removes all of a string up to and including the right-most period.

`findnew` This function returns the position of non-duplicate data in a vector. It is useful for pruning duplicate information from a string or vector.

`isleap` Accepts an integer year AD, and returns 1 if it is a leap year according to Pope Gregory’s newfangled calendar.

`mminmax` Finds the minimum and maximum of a vector or a matrix, and returns them in a two-element vector. The function is robust with respect to NaN.

`strntok` Given a string and a token separator, quickly extracts tokens from short input strings.

[7] The OPeNDAP Matlab GUI is divided into two directories: the main directory, typically called something like `ml-toolbox`, and the `DATASETS` subdirectory contained in it. Roughly speaking, the operational division between these two directories is between the generally useful functions and dataset-specific functions. If a function written for a specific dataset appears to be useful for other datasets, it will be “elevated” to the main directory.

`year2day` Converts a decimal year into an integer day. (Note that `1985.0` will return day 0. To convert to the more common calendrical representation, where days are numbered from one, you must add one to the returned day integer.)

3.3 Adding a New Dataset

Here's an example of how to add a gridded dataset to the OPeNDAP Matlab GUI.

We start with just a URL:

```
http://ferret.pmel.noaa.gov/cgi-bin/dods/nph-dods/data/
  PMEL/large_trenberth_moncl.nc';
```

We will do this iteratively, improving our archive M-file as we go. For the first step, we'll use the following. (We've shortened the URL here for formatting reasons. Use the whole URL shown above.)

```
GetFunctionName = 'getrectg';
Server = 'http://ferret.pmel.noaa.gov/...

LonRange = [-180 180];
LatRange = [-90 90];
TimeRange = [1800.0 2000.0];
DepthRange = [0.0 0.0];
Resolution = NaN;
DataName = 'Trenberth Wind Stress - PMEL';

SelectableVariables = str2mat('Wind Stress U', 'Wind Stress V');
```

It's missing the DodsName variable, but we can find that information using `loaddods`. At the Matlab command line, type:

```
DAS =loaddods('-A','http://ferret.pmel.noaa.gov/... ')
```

You can look at the DAS variable now, and you'll see a `TAUX` and a `TAUY`, so those must be the names for the stresses.

So we'll add the following to the bottom of our tentative archive M-file:

```
DodsName = str2mat('TAUX', 'TAUY');
```

Now open the bookmarks editor (control-B is a shortcut), select a wind dataset and copy and paste it to get a copy. Open the properties editor for the new entry, and erase the name and archive. Pick a new color, too. Click **Apply** to make the changes in the bookmarks, and **Reload** to force the old archive files to be cleared, and to be reread, along with this new archive. The dataset name should be filled in automatically with the information from the new (but still tentative) archive M-file.

Now click OK to dismiss the properties dialog. Once back in the bookmark editor window, Save (and apply to the GUI) and Close.

Once back in the browser window, select the new dataset. Select the entire time range and click Get Details. You can examine the URLlist returned, and you'll see that the URL returned has only twelve data points in the time dimension. This would imply that this is a climatology, containing monthly averages.

```
http://ferret.pmel.noaa.gov/...?TAUX[0:1:11][0:1:72][0:1:143]...
```

We still don't know the resolution or actual geographic extent, but we can correct the original guess for TimeRange. Substitute the following line for the original time range guess:

```
TimeRange = [1800 str2num(datestr(date,10))];
```

To update the bookmarks, you can press Ctrl-B to bring up the bookmarks editor, select the new dataset, Ctrl-I to get the properties editor, Press Reload/Clear, then OK to dismiss this window, Ctrl-S to save the bookmarks and apply them to the GUI again.

Now select June 1947 and Wind Stress U and V and click Get Details. You will probbaly need to identify the axes again because reloading the archive clears the cached metadata. Check the URL again, and you'll see the time slice which is selected in the URL:

```
http://ferret.pmel.noaa.gov/...?TAUX[5:1:5][0:1:72][0:1:143]...
```

Month 5 (starting at zero) is June, so the time axis is set correctly. Had it not been, we could use the `Time-Offset` feature of `getrectg` to correct it.

Now we can click Get Data. Go with the default plot option, and you'll see that the extent of the dataset is indeed global.

To figure out the resolution, type this at the Matlab command prompt:

```
>> diff(R1_Latitude)
```

This gives a value of 2.5 (so does the same test with Longitude), so we'll replace the Resolution line in the archive file with this one (111 km is the distance of one degree of longitude at the equator):

```
Resolution = 2.5*111;
```

To reload the new file, hit Ctrl-B to get the bookmarks editor, Ctrl-I get the properties, 'Reload/Clear' one more time, dismiss the properties, Ctrl-S, save and close. Select the new dataset, reselect June 1947, and this time, a Resolution of 555.0 km. Click Get Data, re-identify the axes one last time when prompted, and change the quiver plot color to 'r' (red). Select "overplot" to plot on top of the previous ploy, and you'll see that the scaling is now different. But you also see that you have every other data point and they are also clearly one per 5 degrees, so the addition has been a success.

A

Getting Started with the OPeNDAP Matlab GUI Software

This appendix contains instructions for acquiring the OPeNDAP Matlab GUI software. It also contains instructions for a quick start, in case you feel like diving in without reading the rest of this document. If you are very familiar with Matlab, this may be just fine.

The OPeNDAP Matlab GUI consists of the following pieces:

Program Files These are the M-files and data for the browser itself. These are usually stored in a single directory (the "Matlab directory") that must be accessible on the Matlab path. These files are the same on all platforms.

Archive Files M-files describing the data archives accessible to the browser. These are found in the `DATASETS` subdirectory of the Matlab directory. You can add your own files to the browser by adding M-files to this directory. (See page 51) These files are the same on all platforms.

Helper Programs Three helper programs: `loaddods`, `writeval`, and `geturl`. These programs are part of the DODS "Matlab Client" distribution. The directory with loaddods in it must be on the Matlab path, and the directory with writeval and geturl in it must be on the shell's $PATH. These programs differ among platforms. You must either retrieve and compile the source distribution, or make sure to secure the binary distribution appropriate for the computer you are using.

In addition to these programs, you will need a copy of the GNU gzip utility to uncompress the archives. The OPeNDAP Home page contains links to an up-to-date version of that utility.

The OPeNDAP Matlab GUI does not come with Matlab itself. You will have to secure a copy of Matlab in order to run any of this.

A.1 Getting the OPeNDAP Matlab GUI Software

All the OPeNDAP software can be downloaded from OPeNDAP Home page, which has a link at the top reading **Download**. Click there and follow the directions.

A.2 Unpacking the OPeNDAP Matlab GUI Software

Once you have the files downloaded they must be uncompressed with the `gzip` program, and un-tarred with the `tar` program. The following command executes both at the same time:

```
gzip -d -c GUI-stuff.tar.gz | tar -xvf -
```

or, if you use GNU `tar`:

```
tar -xvzf GUI-stuff.tar.gz
```

Move to the directory where you wish to store the OPeNDAP Matlab GUI, repeat this for each `tar` file you need to install.

If you are installing this software from a binary distribution, you may move on to the next section, Section A.3 on page 55.

After the `tar` files have been unpacked, set the `$DODS_ROOT` environment variable. The way to do this will differ depending on the shell you use[1]. For csh and tcsh, use:

```
setenv DODS_ROOT /usr/local/OPD-2.8
```

(Make sure to use the correct version number.) For ksh and bash, use:

```
export DODS_ROOT=/usr/local/OPD-2.8
```

After you have set the environment variable (this might be a good time to add the variable definition to your `.login`, `.cshrc` or `.profile` initialization file), finish the configuration with the following shell commands:

```
cd $DODS_ROOT
./configure
make
make install
```

If you already have some OPeNDAP software installed on your system, and are simply adding the Matlab functions, you need not execute `configure` and `make`

[1] Type `echo $SHELL` to find out which shell is active on your system.

in the $DODS_ROOT directory, but can change your directory to the $DODS_ROOT/src/matlab-GUI directory, and run configure and make install there.

A.3 Installing the OPeNDAP Matlab GUI Software

To run the OPeNDAP Matlab GUI, change your working directory to the directory where the OPeNDAP Matlab GUI files are kept ($DODS_ROOT/bin/matlab-gui), and start Matlab. Then issue the browse command. In order to run the OPeNDAP Matlab GUI from other directories, you only need to set three environment variables:

- Be sure that the $DODS_ROOT environment variable is set, as described in Section A.2 on page 54.
- The $PATH environment variable must include an entry for the $DODS_ROOT/bin directory containing the writeval and geturl programs.
- The $MATLABPATH environment variable must include entries for the directories containing the Matlab client program (loaddods), the browser programs and data, as well as the directory containing the data archive files. These are usually $DODS_ROOT/bin, $DODS_ROOT/matlab-GUI and $DODS_ROOT/matlab-GUI/DATASETS[2]

If you didn't do it before, put these variable definitions into your shell's initialization file (.login, .profile, bashrc or whatever), so you won't have to set them by hand.

You are now ready to run the OPeNDAP Matlab GUI. Simply start Matlab, and type 'browse' at the prompt to see the browser window appear:

```
>> browse
```

[2] If you prefer, you can set the Matlab search path from within Matlab with the path() or addpath commands. You can put this command into a startup.m file to set the path automatically.

A.4 The Archive Files

The OPeNDAP Matlab GUI software is not updated as frequently as are the archive M-files in the central DODS directory. This means that it is generally a good idea to issue an “update all” command (**Get All** on the Update menu) soon after you complete the installation. This is also a good test of the installation.

Index