Using DSS Data in MATLAB with VISTA

This document assumes that the user has a working knowledge of MATLAB scripting and DSS naming conventions.

A number of functions have been created in MATLAB to allow for importing data from DSS and exporting data to DSS. These functions are based on the VISTA java classes and, as such, also allow for the use of the classes and functions within MATLAB using MATLAB’s conventions for the use of java classes. Because of the need for java support, these scripts do not function in versions of MATLAB earlier than MATLAB 6.

# Enabling the VISTA Classes

In order to use the VISTA classes in MATLAB and utilize the DSS extraction scripts, the following steps must be followed:

Note: These steps only have to be performed once. Subsequent uses of MATLAB, once these steps are performed, will have access to the VISTA functionality

1. Install the latest version of VISTA on your computer

The latest version of VISTA can be found at:

<ftp://ftpmodeling.water.ca.gov/pub/delta/IEP/DSM2/source/vista/vista-bin.zip>

1. Add the VISTA files to the MATLAB classpath

Edit the MATLAB classpath.txt file (this file should be located in the \toolbox\local directory in the directory where MATLAB was installed)

Add the paths to the following files into the classpath.txt file after the header comments (lines starting with ##) and the line defining the local path (.)

vista.jar (located in vista\lib)

pd.jar (located in vista\lib)

misc.jar (located in vista\lib)

jython.jar (located in vista\jython)

If VISTA is installed at D:\vista, the lines entered would be:

d:/vista/lib/vista.jar

d:/vista/lib/pd.jar

d:/vista/lib/misc.jar

d:/vista/jython/jython.jar

1. Add the location of the scripts to the MATLAB path

In MATLAB, select the File menu and then select Set Path… This launches the Set Path dialog box. In the dialog click on the Add Folder… button. This launches a file browser. Use this browser to select the location where you have copied the MATLAB VISTA scripts. Once you have selected the folder, click the OK button. Then click on the Save button on the Set Path dialog box. Click on the Close button on the dialog and this step is complete.

1. Add the VISTA library directory to the system path (If it has not already been done)

Note: These directions are for Windows XP. If you have a different system, you will have to make the appropriate changes.

Right click on My Computer, either the icon on the Windows desktop or under the Windows Start Menu. Select Properties (left click). In that menu, select the Advanced tab and click the Environment Variables button. In the User variables list select PATH and click the Edit button (If there is no existing path defined, click on the New button). Add the location of the \vista\lib directory (d:\vista\lib for this example and be sure to separate the directories with a semi-colon) and click on the OK buttons until all of the newly opened dialog boxes are gone.

1. Close MATLAB if it is currently open and restart it.

The VISTA scripts should now work within MATLAB.

# Retrieving Data From DSS

The simplest script to retrieve data from DSS simply returns it as a MATLAB array. This function is createArrayFromDSS

usage: createArrayFromDSS(file,path,timewindow,[isDaily])

file - name of DSS file *string*

path - DSS pathname /a/b/c/d/e/f/ (individual path parts may be omitted)

*string*

timewindow - format: ddMMMyyyy hhmm - ddMMMyyyy hhmm *string*

isDaily - (optional) - 0 = no change, 1 = time series will be converted to daily

*integer*

# Exporting Data To DSS

The following extra information must be provided with the array to properly store it in a DSS file.

writeArrayToDSS(array,file,path,startTime,interval,attr)

array - MATLAB array containing the data

file - name of the file to add the DSS data to

path - DSS pathname /a/b/c/d/e/f/ (d and e parts may be omitted)

startTime - the time for the first data point (format: ddMMMyyyy hhmm)

interval - the time interval for the data (eg. 15MIN, 1HOUR, 1DAY, 1MON)

attr - a DataSetAttr object that can be created using createDSSAttr function.

If using default type of INST-VAL, may enter units as a string instead

createDSSAttr(yunits,ytype)

yunits - units of the data (eg. FEET, CFS, TAF, UMHOS/CM)

ytype - (optional) type of data (eg. INST-VAL, PER-AVER) default is INST-VAL

Alternately, the data can be loaded into a RegularTimeSeries object and saved like that. This is done by first creating a new RegularTimeSeries object.

writeRTSToDSS(rts,file,path)

rts - RegularTimeSeries object containing the data

file - name of the file to add the DSS data to

path - DSS pathname /a/b/c/d/e/f/ (d and e parts may be omitted)

RegularTimeSeries(name,startTime,interval,array,flags,attr)

name - name for the time series (can fill in an empty string ‘’)

startTime - the time for the first data point (format: ddMMMyyyy hhmm)

interval - the time interval for the data (eg. 15MIN, 1HOUR, 1DAY, 1MON)

array - MATLAB array containing the data

flags - MATLAB array containing the flags for the data

attr - a DataSetAttr object that can be created using createDSSAttr function.

If using default type of INST-VAL, may enter units as a string instead

# Other VISTA Functionality

This is an advanced topic that requires knowledge of java and the VISTA classes.

To use other VISTA classes include the classes you need as you would any other java import.

import vista.time.\*

Classes are constructed as normal, except the new keyword is omitted, and if the constructor (or function) has no parameters passed to it, the parenthesis are completely omitted. It should also be noted that MATLAB is not type safe and so variables are neither declared nor cast.

This document was created by Tawnly Pranger ([redwood@water.ca.gov](mailto:redwood@water.ca.gov)) on September 25, 2003. Please report and problems or inaccuracies to [redwood@water.ca.gov](mailto:redwood@water.ca.gov).

Last updated September 25, 2003.