NOTE 205 – AIPS++ DEMONSTRATION SCRIPTS

Tim Cornwell

September 12 1997

Contents

1 Purpose

The purpose of this document is to suggest interactive demonstrations of AIPS++.

2 General reminders

- Set DISPLAY
- Use a stable version of AIPS++, *e.g.* /aips++/beta
- Ensure that the netscape is local to the machine being used or can see the same disks

- Work in a clean directory. Know what is in the active .aipsrc and .glishrc files. I recommend that you put the following in your .aipsrc file:

  ```
  objectcatalog.default: gui
  ```

- Do a trial run!

3 General capabilities

- Say what AIPS++ is:
  - Astronomical Information Processing System
  - C++, Scripting, GUIs, libraries, toolkits, and applications
  - Designed by a team of astronomers and programmers
  - Developed by an international consortium of radio observatories
- Start aips++
- Talk about logger windows, clients.
  - Show options on File and Options menu.
  - Show how help works

    ```
    help()
    help('aips')
    help('display')
    web()
    ```

  - Show *about AIPS++* window, talk about configuration of AIPS++ (aipsrc variables)
  - Show Object Catalog GUI. Talk about objects in AIPS++.
- Talk about Glish. Show capabilities of Glish:
  - Simple vectors:

    ```
    a:=1:1000
    b:=sin(pi*(a/360)^2)
    print a[1:10] , b[1:10]
    ```

  - Plotting:

```
mypg:=pgplotter()
mypg.plotxy(a, b, "Chirp")
```

– Servers and FFTs. Do a help first:

```
- help('*fft*')

There are 10 matches. Please choose from:
  aips.mathematics.fftserverdemo
  aips.mathematics.fftservertest
  aips.mathematics.fftserver
  aips.mathematics.fftserver.complexfft
  aips.mathematics.fftserver.realtocomplexfft
  aips.mathematics.fftserver.convolve
  aips.mathematics.fftserver.crosscorr
  aips.mathematics.fftserver.autocorr
  aips.mathematics.fftserver.shift
  synthesis.synthesisdos.imagesolver.setfft
F
- help('*fftserver')

 fftserver -- Object  -- aips.mathematics

  FFTs and related operations

 Methods
   complexfft   Full complex-to-complex in-place FFT of an array
   realtocomplexfft     Real to Complex FFT of an array
   convolve     Convolve a model with a psf
   crosscorr    Cross-correlate two real arrays
   autocorr     Auto-correlate an array
   shift        Shift an array some number of pixels with an FFT

You may find more information in the on-line documentation available
via your web browser.  Type the command

   web()

to view more about aips.mathematics.fftserver.

F
```

```
- web()
```

Now use it:

```
myfft:=fftserver()
c:=myfft.realtocomplexfft(b)
mypg.clear()
mypg.plotxy(a, abs(c), "FT of Chirp")
```

- Point out that `myfft` and `mypg` show up in the Object Catalog. Use the Object Catalog to delete `myfft`, since we no longer need it. Zoom on plot to show detail.
- Talk about the plotter. Show the plotting capabilities given by the PGPLOT tk widget (courtesy of Martin Shepherd and Tim Pearson!) as bound to Glish. To do this, go to the menu of the plotter and under Options, use the Add Commands option. Scroll until you find demo, then select it and go.
- Aipsview. Make and display an array:

```
a:=array(1:1000, 100, 100)
dd.array(a)
```

4 Synthesis

- Start the File Catalog to see the files as they are generated: either use the Object Catalog to Show dc or enter by hand:

```
dc.gui()
```

- Set up an `imager` using the `imagertester` constructor. This will create a table 3C273XC1.ms and set the state correspondingly.
- Refresh the File Catalog and Browse 3C273XC1.ms. Remember to close the table. Talk about data access in AIPS++: you can always get at it and edit it if you like.
- First deconvolve an an image using the `clean` method. The default inputs should be ok.
- Second, restore the image using the `restore` method.
- Refresh the File catalog and View the image clean.restored

- Set levels on aipsview (-0.03, 0.05 works). Go through all planes. Show the axes.
- An option is to use `simpleimage` to show wizard-like processing. You'll be prompted for answers to a few questions.

  ```
  simpleimage()
  ```

5 Measures GUI

- Find Local Apparent Sidereal Time from Greenwich Mean Time. First start measures gui by using the Object Catalog Show button after selecting dm.
  - Select Position gui
  - Under OBS, select your observatory e.g. GB
  - Select output as IRTF
  - Press − >Convert to convert to IRTF
  - Press − >Frame to define this as a frame
  - Select Epoch gui under main measures gui
  - Enter the string 'today' (no quotes) in the slot on the left
  - Select output as LAST
  - Press − >Convert to convert to LAST (presto)

6 Single Dish

- Start the single dish environment, dish

  ```
  include 'dish.g'
  ```

  If someone has previously used dish from the same account, then dish will start up in the same state as it was when last used. This can be pointed out after the dish GUI appears. If you want to guarantee a pristine, default state then you must remove the directory which holds that state - $HOME/aips++/dishstate. Note that if you do start with some previously set state which was arrived at after following this demo, you may be able to skip the steps involving opening data sets (i.e. check first to see if it is already opened). Also, it is probably a good idea to clean out the Results Manager periodically if giving this demo repeatedly so that it doesn't become too cluttered with the results of past demonstrations. Just select what you want to forget and press the "Delete" button in the Results Manager.

- Create the demo data tables for reading

  ```
  include 'dishdemo.g'
  dishdemo()
  ```

  This will create the demo data in the current directory. If the demo data already exists this will do nothing. It is quite safe to repeat it if you aren't sure the demo data exists. It takes a while. It sends messages to the logger as it processes each SDFITS file with fits2table.

  The current demo data consists of the following tables:

  - `dishdemo1` and `dishdemo1`. These are simulated data sets which are especially useful for demonstrating the averaging operations.
  - `dishparkes`. This is some sample data from Parkes. This is useful for showing how the dish plotter displays multi-polarization data.
  - `dishmopra`. This is some sample Mopra data. This is used in the command line portion of this demonstration. It consists of 4 scans of position switched total power data. The first two scans are "on" scans at two separate frequencies and the last two scans are the corresponding "off" scans at those same frequencies.
  - `dishspecproc`. This is some GBT Spectral Processor data taken using the 140' earlier this year. It is useful for showing real data. It can be used in the selection and averaging part of this demonstration but because it was used without the benefit of all of the GBT Monitor and Control devices and software it lacks several important header parameters (no telescope position, no system temperature information, and no velocity frame and rest frequency information).

- Load the Parkes table (`dishparkes`) using the File menu (select Open and Read Only)
- Browse it

  - Select it in the Results Manager

  - Press the “Browse” button.
  - Once the browser has come up, you can move around in the browser by selecting different records, by using the up and down arrow keys as well as the Home and End keys.
  - Look at a number of records and see how the plot changes
  - The first and fourth records have two polarizations.
  - Play with plotting options in DISHPLOT (Styles, Coordinates, Statistics, and Options are the most useful).
  - Play with the browser options.
  - Select “Browse record” in the browser.
    * Change to a different coordinate system for “Direction” by pushing the button labeled “B1950” and selecting “J2000” (button label should change).
    * Change to a different format for the Time display by pushing the button labeled “Time” and selecting “dmy”.
    * Change to a different time system by pushing the button labeled “UTC” and selecting “GAST” for example. Note that some conversions will not be possible with this data set because the telescope position is not in the header (e.g., a conversion to LST is impossible).
    * Experiment with the other buttons.
    * Display the data values by pressing the “Browse arr” button.
    * Show the history by pressing that button (scroll down first).
  - Click on any number of other records in the “SD Working Set Browser” window to see them update the record browser window and the plotter.

- Dismiss the browser windows.

- Open the `dishspecproc` data set.

- Select two operations: Selection and Averaging (Operations menu).

  - Press some of the “All” buttons on the Selection window to show all of the unique values for those quantities in this data set. Make sure you push the “Object” field’s “All” button.
  - In the Object entry enter “1328*” to select both 1328+305 and 1328+305SBT

- Turn on the Object selector by pressing the box between the combobox down pointer and the "All" button.
- Press the "Apply" button on the Selection window. This produces a new working set (in the Results Manager as well as in the Selection window).
- In the Averaging window, turn off the "Make selection before averaging" option since we have just made the selection (it turns out that it doesn't matter in this case since the act of making the selection clears the selectors and when the selectors are clear, no selection happens and the entire current working set is used - which is what we want).
- Press the "Apply" button in the averager and look at the result (the result will appear in the plotter and be selected in the result manager). You may want to manually adjust the Y axis range on the plotter.
  * Under the Options menu select Scaling and then Manual
  * Enter the appropriate limits for the Y axis. For this example, MinY of 0.4 and MaxY of 0.6 seem to work well.
  * Press return in the MaxY or MinY entry fields or press the Redraw button to have these ranges take effect. Be sure to point out during this operation that cursor-driven zoom is expected very soon.
- Browse (this just brings up the individual record browser when the focus is on an individual record). Look at the history of this result.
- Note that because of the limitations of this data set discussed above, if you want to play with other alignment options it is best to use one of the fake data sets (dishdemo1 and dishdemo2). For this data set, you really can only choose no alignment or align by x-axis. You can not choose tsys weighting because there are no tsys values.
- You can browse the working set created as a result of the selection operation. You can browse more than one thing at a time.

- Any of these results can be examined at the glish prompt. For example, type "print average1.hist" at the glish prompt. This shows the history. Type "print average1.data.desc". This shows the axis description.

- You can rename any variable in the results manager by typing in a new name and hitting return.
- Now do some smoothing (turn "off" the other operation GUIs to conserve real estate on your desktop by pressing their "Dismiss" buttons).
  - Select a record (either the result of a previous operation, or through the browser on a working set). This operation works on the most recently displayed record (as do all operations which work on single records).
  - Try different types of smoothing.
  - Try Hanning or Boxcar with and without "decimation"
  - Do a Gaussian smooth.
    - Select "Gaussian"
    - Make sure that the displayed units are "Channels" and not "x-axis units" (press that menu to select Channels).
    - Enter a width in channels
    - Convert it to x-axis units by pushing the "Convert" button.
    - Enter a different width.
    - Oops, you meant that to be in channels, push the menu button labeled "X-axis Units" and select "Channels" (the value you just typed in has not changed, but the label has).
    - Actually do the smooth.
- Fit some baselines.
  - Select "Polynomial" ("Sinusoidal" also works, but, as with most sinusoidal fitters, its fairly sensitive to the initial conditions. It tends to work best if the initial guess at a period is longer than the suspected period in the data.)
  - Choose an order.
  - Chose the x-axis ranges to include in the fit, using the plotter.
    - click on "Cursor active" at the bottom of the Baselines GUI.
    - Mark some ranges with the cursor - left mouse to start and end, right mouse to cancel.
    - The height of each range box is 2x the RMS within that box, the center Y of a range box is the local mean.

    * Notice that these ranges are also displayed in the Baselines GUI.
    * You can convert to X-axis units in the Baselines GUI in DISH.
  - With the "Recalculate" and "Show" options selected, press "Apply".
  - Note the RMS in the GUI, the fit overlaying the data on the plotter and the original record still selected. The RMS value is the RMS of the data with respect to the fit over the baseline regions.
  - Change the order and/or the ranges
  - Ranges can be edited in the Baselines GUI entry - these changes will not appear on the plotter until the "Apply" button is pressed or you hit the Enter/Return key. The contents of the ranges entry are used "as is" whenever "Apply" is pressed (even if you haven't previous hit return).
  - When you are satisfied that this is a baseline you want to remove, turn off the "Recalculate" button and select the "Subtract" option and press "Apply". Note that if manual scaling is in effect as recommended above then you will need to re-scale at this point, -0.1 to 0.1 seems to work well.

- Function on data - with a record selected and plotted try the following (comments appear after the `#`), you might want to return to the original record before each application of a function on the data.
  - `ARR*ARR #` square of the data array, press "Apply"
  - `ARR * 10 #` scale the data by 10.0
  - `sqrt(ARR) #` take the sqrt of the data
  - Any valid glish can be used here. `ARR` is replaced by the data array, `HEADER` by the `header` record, `NS_HEADER` by the `ns_header` record, `DESC` by the `desc` record, and `DATA` by the `data` record. The operations must return an array of the same shape as the input data array. Any global glish variable can be used here (including functions).

- Turn on the "Trace" option and try some of these operations again.

- Select "Open Eval Window" from the "Tools" menu.

- Type any valid glish in the eval window.
- Press "Eval Contents"
- marvel at the result which appears in the glish window where you started from.

- Try the Multi-Operation Operation.
  - Clicking on one of the "Add:" buttons adds that operation to the operation cache. Clicking on "Do multi-op sequence" executes this sequence, doing those operations in order using the parameters as currently set in each operation window (even those which aren't currently displayed).
  - Add the operations "Average", "Baseline", and "Smoothing" to the cache and do that sequence.
  - Store that sequence by giving it a name by which it will be known in the "Store Cache" entry and press the "Store Cache" button.
  - Reverse the order of the baseline and smoothing operations by first deleting baseline and then adding it back in.
  - Do this sequence and save it with a different name.
  - Restore the first sequence by selecting it (clicking on it) in the list of stored caches.
  - Our current thinking is that once the user gets various parameters set up that DISH will be often "driven" from this multi-operation operation menu.

- A simple command-line example. In this example you will be taking two total power scans, one on-source and one off-source, constructing the difference spectra and placing the result back in the results manager where GUI operations on it could continue.
  - Open the `dishmopra` data set
  - At the glish command line, type the following (the comments are not necessary, obviously)

```
dishmopra.setlocation(1)  # point the iterator at record 1
on := dishmopra.get()     # get the ``on source'' data
dishmopra.setlocation(3)  # point the iterator at record 3
off := dishmopra.get()    # get the ``off source'' data
result := on              # initialize the result to have
```

```
                                # the same structure and header
                                # values as on
    result.data.arr :=
       (on.data.arr - off.data.arr)/off.data
                                # the difference spectra
    dish.rm().add('result'      # add it to the results manager by name
                  'Difference of rows 1 and 3'   # description
                  result        # the value
                  'SDRECORD')   # its type - dish knows how to plot these
```

– In the Results Manager you should see a new variable named "result1". Select it. It should now be displayed.
– Some manual scaling is necessary - the values at the end are unreliable due to the overall bandpass shape of on and off. A range of -0.01 to 0.01 works well here.

7 Utilities

- Show system information:

```
sysinfo()
sysinfo().numcpu()
sysinfo().host()
```

- GUI stuff: each one of these lines does something interesting:

```
f:=infowindow('Eat at Bodos', 'Dining recommendations') # Informative only
choicewin('Type of bagel',"plain wheat sourdough")      # Choice returned to Gli
g:=guiframework() # Framework of GUIs
```