ParselTongue

Mark Kettenis, JIVE

December 3, 2008

ParselTongue

Software infrastructure for the ALBUS project (a RadioNet JRA)

- to develop and implement new calibration algorithms
- to implement pipelines that produce new/better data products

Binding classic AIPS (and possibly other packages) to Python.

Useful for any science project that uses AIPS for data processing: EVN, VLBA, MERLIN, VLA, GMRT, WSRT ...

Uses parts of the Obit package written by Bill Cotton, NRAO.

http://www.jive.nl/dokuwiki/doku.php/parseltongue:parseltongue

http://www.radionet-eu.org/rnwiki/ParselTongue

Why Python?

- Extensive support for "Scientific Computing"
- Supports procedural programming, object oriented programming and (some) functional programming
- Integration of C/C++ code is simple (SWIG, Boost)
- Widely adopted by (radio)astronomy: PyRAF, casapy, pyrap, MeqTrees

ParselTongue Capabilities

- Running AIPS tasks
- Access to the AIPS catalog
- Access to AIPS images and UV data
- Access to AIPS extension tables
- Logging
- Access to the AIPS TV
- XML-RPC interface allows distributed computing

Running AIPS verbs is impossible; but ParselTongue offers alternatives

ParselTongue Philosphy

Do things the Python way as much as possible:

- use Python iterators:

```
for row in table:
    print row
```

- Python-style array indexing

Keep things people like in AIPS/POPS:

- Minimal match
- Array indices that match AIPS documentation (APARMS, BPARMS, . . .)

Running AIPS Tasks

```
>>> fitld = AIPSTask('FITLD')

>>> fitld.infile = "/home/sipior/data/N1066.fits"
>>> fitld.outname = "N1066"

>>> fitld.go()     # fitld() will work just as well

>>> help(fitld)
FITLD
Type: Task
Use: FITLD loads both maps and UV data from tape (or disc) to disc. It
    will only load FITS files, if any other type of file exist on the tape
                                  .....
```

Wizardry

Wizardry versions of ParselTongue modules allow read/write access to tables and full access to UV data/image pixels.

```
from Wizardry.AIPSData import AIPSUVData
from AIPS import AIPS

from pylab import plot, show

data = AIPSUVData('MULTI', 'UVDATA', 1, 3)
u = []; v = []
for visibility in data:
    u.append(visibility.uvw[0])
    v.append(visibility.uvw[1])

plot(u, v, '.')
show()
```

Excellent way to prototype new algorithms!

News

- NumPy support
- Support for running external tasks
- Parallelization support
  - Remote copy of AIPS data, directly into the catalogue.
  - Parellel execution of tasks
- Improved documentation
- Lots of bug fixes

Most of this work was done by Michael Sipior!

ParselTongue Usage

- MERLIN Imager on AstroGrid (Richards et. al.)
- Ionospheric Calibration (Anderson)
- Widefield Imaging (Wucknitz, Bourke)
- EVN Pypeline (Reynolds)

EVN Pypeline

Written by Cormac Reynolds; evolution of POPS script from Phil Diamond. Used by support scientists at JIVE to check data quality

- Handles EVN and VLBA data; should be easily adaptable for other instruments.
- Pipeline products:
  - Lots of diagnostic plots
  - Calibration tables
  - Dirty maps
  - Crude maps (a few rounds of selfcal)
- Checkpointing: Pypeline can be restarted at various point.
- Automatic defaults based on metadata.
- Reusable building blocks.

http://www.jive.nl/dokuwiki/doku.php/parseltongue:grimoire

EVN Pypeline Steps

1. Load and sort the data
2. A priori data flagging
3. Plot the raw data
4. Amplitude calibration and parallactic angle correction
5. Fringe fitting
6. Bandpass calibration
7. Plot the results after ampcal, fringe fitting and bandpass
8. Split
9. Create multi files and make dirty maps and first clean maps
10. Continue mapping
11. Plot the final data
12. Calculate the antenna sensitivities
13. Save useful data and plot final map

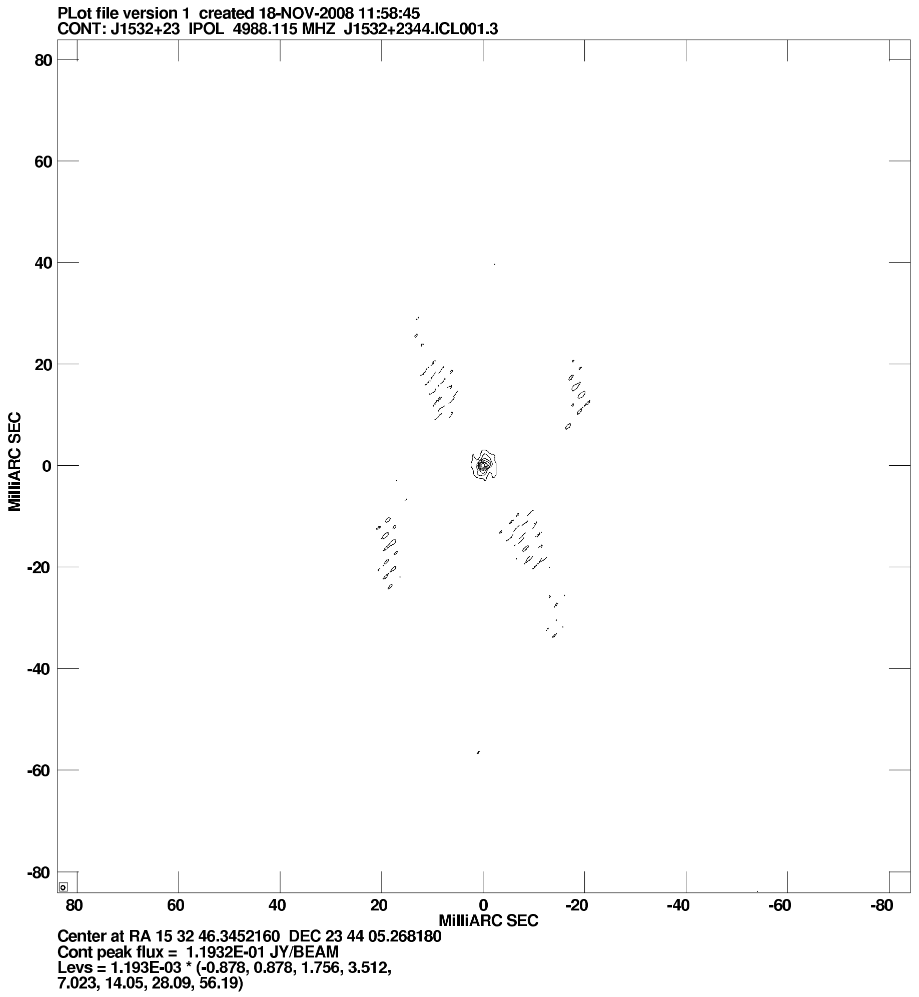
EVN Pypeline

The pipeline is driven by a simple input file:

```
# tmask is a range of steps to be carried out. tmask=1 will run the whole
# pipeline.
tmask = 1

# if a multiple pass experiment then append the pass number to the experiment
# name (e.g. n05c3_1)
experiment = n05c3
userno = 3602

# refant is a prioritised list of reference antennas
refant = Ef, Mc, Nt

# plotref is a list of antennas, baselines to which will be plotted.
plotref = Wb

# bpass is a list of sources for bandpass calibration
bpass = 3C345, 3C454.3

# phaseref and target must be set if phase referencing. Each source in target
# list will be phase calibrated by the corresponding source in phaseref list.
phaseref = 3C454.3
target = J2254+1341

# solint defaults to typical scan length on phase calibrator. Must be set if
# not phase referencing.
#solint = 0
```

EVN Pypeline

Final image of J1532+2344 from the EVN Pypeline:



Future

- Support for multiple (local) AIPS TV's
- Support for running POPS scripts

ParselTongue development & support will continue in ALBiUS