



Predicting visibilities in MeqTrees with UVBrick

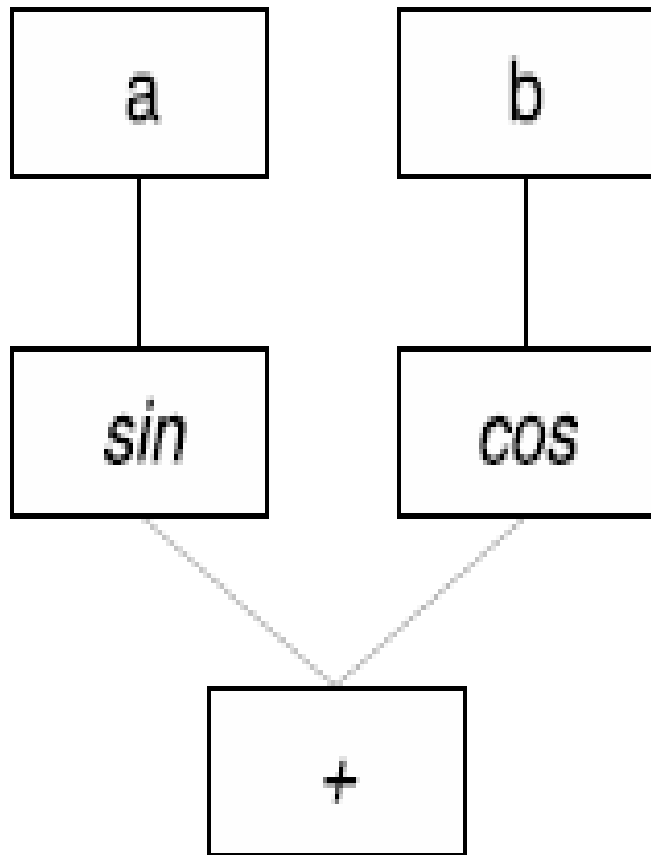
N. Iniyan, Dr. Kurt van der Heyden
3GC II, Algarve, 2011

Outline



- MeqTrees software structure
- UVBrick – Introduction
- Current implementation
- Future path for UVBrick

MeqTrees Design and Impl.



- MeqTrees makes use of tree structures to represent mathematical expressions.
 - In the traditional view, a tree data structure consists of a hierarchy of nodes with parent-child relationship with the leaf nodes consisting of values that are propagated upwards (or downwards) towards the root node.
- (O.M.Smirnov, MeqTrees, 2010)

MeqTrees Design and Impl.



- In MeqTrees, the result of each node is a *function* of N real/complex variables.
- For our purpose, we can consider our functions to be dependent on the two variables, t and v .
- A *Request* object in MeqTrees consists of these variables represented as vectors (among other parameters).
- A *Result* object that is returned by a node consists of values that populate the grid formed by these vectors and the input vectors themselves.



UVBrick



UV Brick is not...

W-projection



UVBrick is...

a collection of code (that forms a part of MeqTrees)
that aims to predict uv-data from a given image
including corrections for DDEs.



UVBrick are here!

Applications Places System 24 °C Sun Sep 25, 12:25 iniyan

MeqBrowser - meqserver.6373 (idle) - example-sim.py

MeqTrees IDL Debug View Bookmarks Help

(Trees) Tabs (Grid) Reload TDL Options

Tabbed Tools

Messages Snapshots example-sim.py

Exec Options | L:186 C:1

```
# -*- coding: utf-8 -*-
#% $Id$
#
# Copyright (C) 2002-2007
# The MeqTree Foundation &
# ASTRON (Netherlands Foundation for Research in As
# P.O.Box 2, 7990 AA Dwingeloo, The Netherlands
#
# This program is free software; you can redistribute it
# it under the terms of the GNU General Public License
# the Free Software Foundation; either version 2 of the
# (at your option) any later version.
#
# This program is distributed in the hope that it will be
# but WITHOUT ANY WARRANTY; without even the implied
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
# GNU General Public License for more details.
#
# You should have received a copy of the GNU General
# along with this program; if not, see <http://www.gnu.org
# or write to the Free Software Foundation, Inc.,
# 59 Temple Place, Suite 330, Boston, MA 02111-1307
#
# standard preamble
from Timba.TDL import *
from Timba.Meq import meq
import math
```

[r/o] example-sim.py

TDL Compile-time Options

MS: <none>
Polarization: XX XY YX YY
Correlations to use: 2x2
 Start Purr on this MS
Simulation mode: simonly

Measurement Equation options

Image-plane components

Sky model

- Use 'Siamese.OMS.gridded_sky' module
- Use 'Siamese.OMS.transient_sky' module
- Use 'Siamese.OMS.fitsimage_sky' module

FITS image file: <none>
Padding factor: 1.2

- Use 'MeowLSM' module
- Export sky model as kvis annotations
- Use Ncorr Jones (n-term correction)
- Use Z Jones (ionosphere)
- Use L Jones (dipole projection)
- Use E Jones (beam)

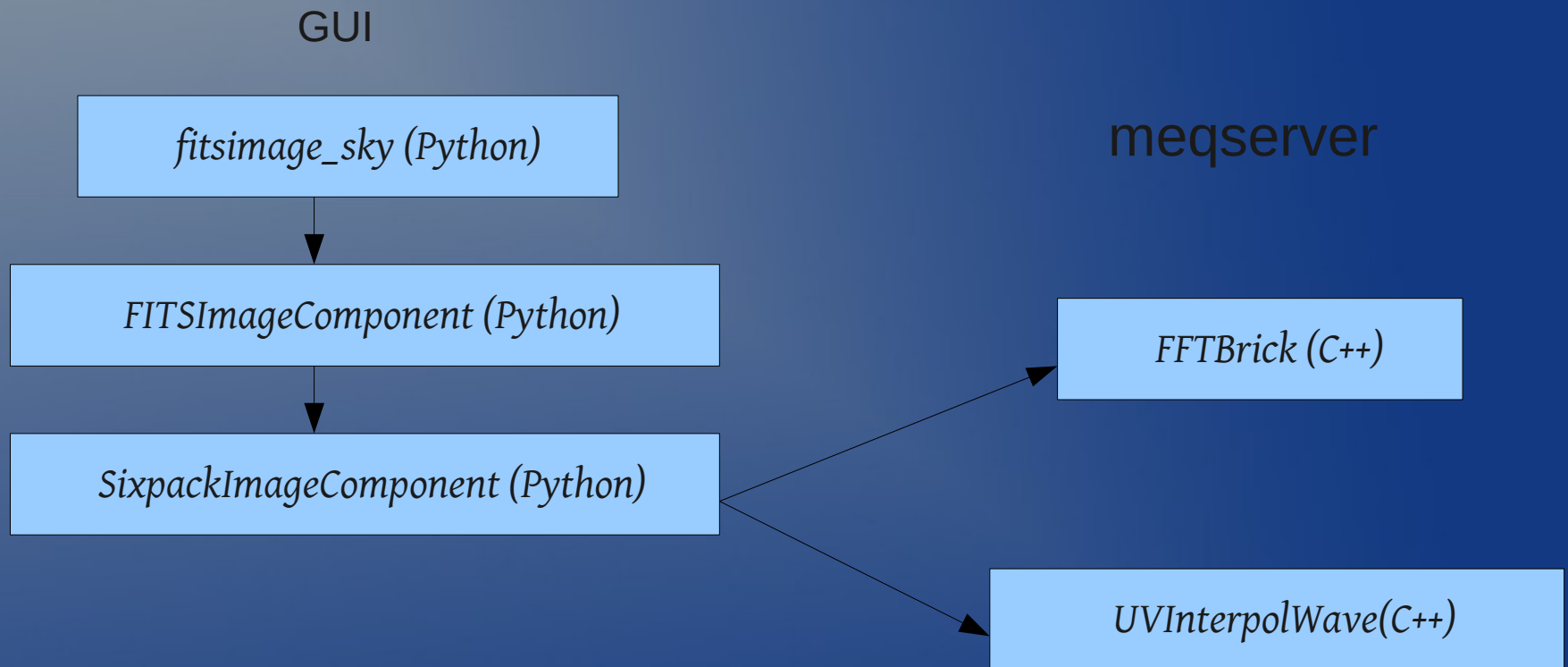
UV-plane components

Use Jones (feed angle)

Compile Load Save Cancel

VSZ:332M RSS:24M

The UVBrick is currently structured as follows:



$$\vec{I}(u, v, w) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} A(l, m) \vec{I}(l, m) \exp^{-2\pi i [ul + vm + w(\sqrt{1-l^2-m^2}-1)]} \frac{dldm}{\sqrt{1-l^2-m^2}},$$

(Nijboer, UVBrick, 2005)

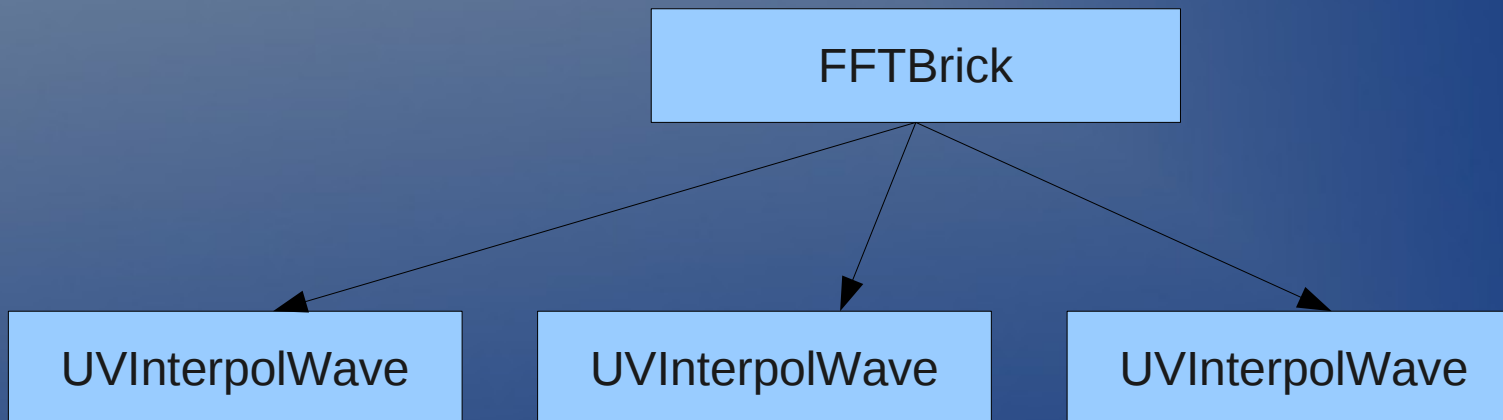
- *FFTBrick* is the part of UVBrick that takes as input an image in the sky plane and performs a 2D FFT on it to produce a uv plane image.
- The input lm-plane image and the padding factor to perform FFT are passed on by the user as compile-time options in the meqbrowser.

$$\vec{I}(u, v, w) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} A(l, m) \vec{I}(l, m) \exp^{-2\pi i [ul + vm + w(\sqrt{1-l^2-m^2}-1)]} \frac{dldm}{\sqrt{1-l^2-m^2}},$$

(Nijboer, UVBrick, 2005)

- *FFTBrick* outputs a grid of visibilities as a function of u and v . This brick is created only once.
- This is passed on to *UVInterpolWave* along with a vector of required frequencies.

- *UVInterpolWave* takes the uv-grid and returns the interpolated visibilities as a function of frequency and time.



UV/Brick - Future



Frequency dependence
W-projection capability
Forward application of other DDEs

Frequency dependence



- As it stands, UVBrick accepts a sky image of constant frequency.
- The first functionality that will be added to uvbrick is to enable it to accept a multi-frequency image cube as input and interpolate for the in-between frequencies.

Eg. Take an image with two frequencies and interpolate for N frequencies in between.

w-projection in the brick



$$V(u, v, w) = \int \frac{I(\ell, m)}{\sqrt{1 - \ell^2 - m^2}} G(\ell, m, w) e^{-2\pi i [u\ell + vm]} d\ell dm$$
$$G(\ell, m, w) = e^{-2\pi i [w(\sqrt{1 - \ell^2 - m^2} - 1)]}.$$

- The w-projection algorithm (Cornwell et al.) enables one to calculate $V(u, v, w)$ from $V(u, v, w=0)$ by convolution with the F.T of the known function $G(l, m, w)$.

$$V(u, v, w) = \tilde{G}(u, v, w) * V(u, v, w = 0)$$

w-projection in the brick



- Currently, uvbrick does not include corrections for w-term effects while predicting the visibilities.
- Choosing a suitable convolution kernel $F(G)$ that takes the w-term effects into account will be the next step.

Extending to other DDEs



$$\begin{aligned}V_{pq}(t) &= X_{pq}[t](u_{pq}(t)), \\X_{pq}[t] &= U_p[t] \circ X \circ U_q^H[t], \\X &= \mathcal{F}B, \quad U_p[t] = \mathcal{F}E_p[t]\end{aligned}$$

(OMS, RIME, 2011)

- In the presence of DDEs, each baseline 'sees' a different apparent sky.
- DDEs are multiplicative terms in the sky plane whose effects can be represented using convolution functions in the uv domain.

Extending to other DDEs



- This means that by choosing appropriate convolution kernels while degridting, the direction-dependent effects can be introduced in the predicted visibilities.
- This method is called AW-projection (Bhatnagar et al.), which is a generalization of the more specific w-projection algorithm.
- By using multiple such convolutional kernels during the predict stage, other DDEs can be applied to the visibilities in uvbrick.



Thank You