Calibration in CASA

ALMA Workshop, McMaster University, June 1-3, 2009

George Moellenbrock

•Atacama Large Millimeter/submillimeter Array
•Expanded Very Large Array
•Robert C. Byrd Green Bank Telescope
•Very Long Baseline Array

NRAO

Overview

- A demonstration of calibration essentials
  - Concentrate on calibration strategy and effects
  - Continuum total intensity
- Get a flavor of how to use CASA to calibrate
  - Let's not get bogged down in syntax and subtlety today (we'll do that tomorrow)
- Fill, plot, flag, (plot), solve, (plot), apply, (plot), (image)

The interferometer in practice: the need for calibration

- For the ideal interferometer the phase of a point source at the phase center is zero, because we can correct for the known geometric delay
- However, there are other sources of path delay that introduce phase offsets that are telescope-dependent; most importantly at submillimeter wavelengths these are water (vapour and liquid) in the troposphere, and electronics
- The raw amplitude of the visibility measured on a given baseline depends on the properties of the two telescopes (gains, pointing, etc.), and must be placed on a physical scale (Jy)
- There may be frequency-dependent amplitude and phase responses of the electronics
- There may also be baseline-based errors introduced due to averaging in time or frequency
- We must observe calibration sources to derive corrections for these effects, to be applied to our program sources

NRAO

Contributors to the Calibration Problem

- Telescope-based components from:
  - Ionospheric Faraday rotation (important for cm, not for submm)
  - *Water, Oxygen in the troposphere (important for submm)
  - Parallactic angle rotation
  - Telescope voltage pattern response
  - Polarization leakage
  - *Electronic gains
  - Bandpass (frequency-dependent amplitudes and phases)
  - Geometric (delay) compensation (important for VLBI)
- Baseline-based errors due to:
  - Correlated noise (e.g., RFI; important for cm)
  - Frequency averaging
  - Time averaging (important for submm because of the troposphere)

Evaluating the calibration

- Are solutions continuous?
  - Noise-like solutions are just that—noise
  - Discontinuities indicate instrumental glitches
  - Any additional editing required?
- Are calibrator data fully described by antenna-based effects?
  - Phase and amplitude *closure errors* are the baseline-based residuals
  - Are calibrators sufficiently point-like? If not, self-calibrate
- Any evidence of unsampled variation? Is interpolation of solutions appropriate?
  - Reduce calibration timescale, if SNR (and data) permits
- Evidence of gain errors in your final image?
  - Phase errors give asymmetric features in the image
  - Amplitude errors give symmetric features in the image

The Observation:

- Science:
  - 7mm continuum observations of SNRs and HII regions in the starburst galaxy M82
- VLA B-configuration (also D-configuration, later)
- Sources (B-config):
  - Target source: M82
  - Near-target calibrator: 0958+655 (4.1 deg from target; unknown flux density, assume 1 Jy initially)
  - Flux density calibrator: 3C147 (0.91 Jy), also observations of 0713+438 (0.29 Jy) at the end
- Signals:
  - RR, LL (and RL, LR) correlation, total intensity
  - Two 50 MHz "channels", centered at 43.315 and 43.365 GHz
    - [aside: for the EVLA and ALMA *all* observations will have many more than two channels!]

The Data: listobs, plotxy

- listobs : a dataset (MS) summary
- plotxy : data (MS) plotting
  - will be superceded by plotms soon

The Data: plotxy

```
CASA <30>: inp plotxy
---------> inp(plotxy)
#  plotxy :: An X-Y plotter/interactive flagger for visibility data.
vis                 =         ''        #  Name of input visibility dataset
xaxis               =     'time'        #  X-axis: def = 'time': see help for options
yaxis               =      'amp'        #  Y-axis: def = 'amp': see help for options
     datacolumn     =     'data'        #  data (raw), corrected, model, residual (corrected - model)

iteration           =         ''        #  Plot separate panels by field, antenna, baseline, scan, feed
selectdata          =      False        #  More data selection parameters (antenna, timerange etc)
spw                 =         ''        #  Spectral windows:channels: ''==>all, spw='2:5~57'
field               =         ''        #  Field names or field index numbers: ''==>all, field='0~2,3C286'
averagemode         =   'vector'        #  Select averaging type: 'vector', 'scalar'
     timebin        =        '0'        #  length of time in seconds to average, default='0', other options: 'all'
     crossscans     =      False        #  have time averaging cross over scans?
     crossbls       =      False        #  have averaging cross over baselines?
     crossarrays    =      False        #  have averaging cross over arrays?
     stackspw       =      False        #  stack multiple spw on top of each other?
     width          =        '1'        #  number of channels to average, default: '1', other options: 'all', 'allspw'

restfreq            =         ''        #  a frequency quanta or transition name. see help for options
extendflag          =      False        #  have flagging extend to other data points?
subplot             =        111        #  Multipanel display scheme (yxn)
plotsymbol          =        '.'        #  Options include . : , o ^ v > < s + x D d 2 3 4 h H | _
plotcolor           =  'darkcyn'        #  Plot color
plotrange           = [-1, -1, -1, -1]  #  The range of data to be plotted (see help)
multicolor          =     'corr'        #  Plot in different colors: Options: none, both, chan, corr
selectplot          =      False        #  Select additional plotting options (e.g, fontsize, title,etc)
overplot            =      False        #  Overplot on current plot (if possible)
showflags           =      False        #  Show flagged data?
interactive         =       True        #  Show plot on gui?
figfile             =         ''        #  ''= no plot hardcopy, otherwise supply name
async               =      False        #  If true the taskname must be started using plotxy(...)

CASA <31>:
```

The data (M82 B-config)

- 24 antennas… 276 baselines
- …with 2 "spectral windows" at each timestamp
- …with 1 "channel" in each spectral window
- …and 4 correlations: RR,RL,LR,LL (we care only about RR, LL today)
- 3.33s integrations for 2h50m
- → ~1.5M spw/baseline/time points for each correlation
- Four radio sources:
  - 0542+498 = 3C147 = flux density calibrator: one scan, ~270s
  - 0958+655 = 09588+65339 = gain calibrator: 73 scans, ~50s each
  - 09559+69408 = M82 = science target: 73 scans, 110s each
  - 0713+438 = alternate flux density calibrator: one scan (ignored)

Flagging: flagdata

- Manually-selected flagging
  - "Known" bad data, e.g., from observing logs
- Auto-flagging
  - "quack" (bad integrations at scan beginnings)
  - "clip" (excessively discrepant data)

NRAO

Flagging: flagdata (mode='quack')

```
CASA <7>: inp flagdata
--------> inp(flagdata)
#  flagdata ::  All purpose flagging task based on selections
vis                 =         ''         #  Name of file to flag
mode                =    'quack'         #  Mode (manualflag,autoflag,summary,quack)
     autocorr       =      False         #  Flag autocorrelations
     unflag         =      False         #  Unflag the data specified
     quackinterval  =        0.0         #  Quack n seconds from scan beginning

spw                 =         ''         #  spectral-window/frequency/channel
field               =         ''         #  Field names or field index numbers: ''==>all, field='0~2,3C286'
selectdata          =       True         #  More data selection parameters (antenna, timerange etc)
     antenna        =         ''         #  antenna/baselines: ''==>all, antenna = '3,VA04'
     timerange      =         ''         #  time range: ''==>all, timerange='09:14:0~09:54:0'
     correlation    =         ''         #  Select data based on correlation
     scan           =         ''         #  scan numbers: ''==>all
     feed           =         ''         #  multi-feed numbers: Not yet implemented
     array          =         ''         #  (sub)array numbers: ''==>all
     uvrange        =         ''         #  uv range: ''==>all; uvrange = '0~100klambda', default units=meters

async               =      False         #  If true the taskname must be started using flagdata(...)

CASA <8>:
```

Flagging: flagdata (mode='manualflag')

```
CASA <4>: inp flagdata
--------> inp(flagdata)
#  flagdata :: All purpose flagging task based on selections
vis                 =         ''        #  Name of file to flag
mode                = 'manualflag'      #  Mode (manualflag,autoflag,summary,quack)
     autocorr       =      False        #  Flag autocorrelations
     unflag         =      False        #  Unflag the data specified
     clipexpr       =   'ABS RR'        #  Expression to clip on
     clipminmax     =         []        #  Range to use for clipping
     clipcolumn     =     'DATA'        #  Data column to use for clipping
     clipoutside    =       True        #  Clip outside the range, or within it

spw                 =         ''        #  spectral-window/frequency/channel
field               =         ''        #  Field names or field index numbers: ''==>all, field='0~2,3C286'
selectdata          =       True        #  More data selection parameters (antenna, timerange etc)
     antenna        =         ''        #  antenna/baselines: ''==>all, antenna = '3,VA04'
     timerange      =         ''        #  time range: ''==>all, timerange='09:14:0~09:54:0'
     correlation    =         ''        #  Select data based on correlation
     scan           =         ''        #  scan numbers: ''==>all
     feed           =         ''        #  multi-feed numbers: Not yet implemented
     array          =         ''        #  (sub)array numbers: ''==>all
     uvrange        =         ''        #  uv range: ''==>all; uvrange = '0~100klambda', default units=meters

async               =      False        #  If true the taskname must be started using flagdata(...)

CASA <5>:
```

Calibration: solve then apply

Calibration (inverse) applied to target (and calibrators)

Observed data

$$V_{ij}^{obs} = G_{ij} V_{ij}^{model} \quad \rightarrow \quad V_{ij}^{cal} = G_{ij}^{-1} V_{ij}^{obs}$$

Solved for

Models supplied for calibrators

Calibrated data for imaging!

Calibration models: setjy

- For point sources, a flux density, if known
  - (If flux density is unknown, use 1.0 Jy)
- For resolved sources, a model image
  - E.g., supplied by observatory
  - Self-calibration…

Calibration models: setjy

```
CASA <11>: inp setjy
---------> inp(setjy)
#  setjy ::
vis                 =          ''         #  Name of input visibility file (MS)
field               =          ''         #  Field name list or field ids list
spw                 =          ''         #  Spectral window identifier (list)
modimage            =          ''         #  File location for field model
fluxdensity         =          -1         #  Specified flux density [I,Q,U,V]; -1 will lookup values
standard            = 'Perley-Taylor 99' #  Flux density standard
async               =       False         #  If true the taskname must be started using setjy(...)

CASA <12>:
```

'Gain' calibration: gaincal

- Time-dependent amplitude and phase
  - per antenna, spectral window, polarization
- Antenna-based solutions:

$$V_{ij}^{obs} = G_{ij} V_{ij}^{model}$$
$$= G_i G_j^* V_{ij}^{model}$$

$$N_{ant}\ G_i s\ :\ \frac{N_{ant}(N_{ant}-1)}{2}\ observations$$

'Gain' calibration: gaincal

- With prior calibration (upstream and downstream terms)

$$V_{ij}^{obs} = G_{ij}^{d} G_{ij} G_{ij}^{u} V_{ij}^{model}$$

$$\left(G_{ij}^{d} \right)^{-1} V_{ij}^{obs} = G_{ij} \left(G_{ij}^{u} V_{ij}^{model} \right)$$

$$V_{ij}^{corrected \sim obs} = G_{ij} V_{ij}^{corrupted \sim model}$$

- Basic calibration solver is generic
- A matrix equation: beware of commutation rules
  - E.g., polarization terms which are not diagonal
  - CASA calibration enforces commutation rules for you

‘Gain’ calibration: gaincal

I/O

Data Selection

Solving parameters

Prior calibration

```
CASA <12>: inp gaincal
---------> inp(gaincal)
#  gaincal :: Determine temporal gains from calibrator observations
vis                 =         ''        #  Nome of input visibility file
caltable            =         ''        #  Name of output gain calibration table
field               =         ''        #  Select field using field id(s) or field name(s)
spw                 =         ''        #  Select spectral window/channels
selectdata          =       True        #  Other data selection parameters
     timerange      =         ''        #  Select data based on time range
     uvrange        =         ''        #  Select data within uvrange (default units meters)
     antenna        =         ''        #  Select data based on antenna/baseline
     scan           =         ''        #  Scan number range
     msselect       =         ''        #  Optional complex data selection (ignore for now)

solint              =      'inf'        #  Solution interval (see help)
combine             =         ''        #  Data axes which to combine for solve (scan, spw, and/or field)
preavg              =       -1.0        #  Pre-averaging interval (sec)
refant              =         ''        #  Reference antenna name
minblperant         =          4        #  Minimum baselines _per antenna_ required for solve
minsnr              =        0.0        #  Reject solutions below this SNR
solnorm             =      False        #  Normalize average solution amplitudes to 1.0 (G, T only)
gaintype            =        'G'        #  Type of gain solution (G, T, or GSPLINE)
calmode             =       'ap'        #  Type of solution" ('ap', 'p', 'a')
append              =      False        #  Append solutions to the (existing) table
gaintable           =       ['']        #  Gain calibration table(s) to apply on the fly
gainfield           =       ['']        #  Select a subset of calibrators from gaintable(s)
interp              =       ['']        #  Interpolation mode (in time) to use for each gaintable
spwmap              =         []        #  Spectral windows combinations to form for gaintables(s)
gaincurve           =      False        #  Apply internal VLA antenna gain curve correction
opacity             =        0.0        #  Opacity correction to apply (nepers)
parang              =      False        #  Apply parallactic angle correction
async               =      False        #  If true the taskname must be started using gaincal(...)

CASA <13>:
```

Calibration solution evaluation: plotcal

- Provisional calibration plotting task
  - to be superceded by plotms soon!

Calibration solution evaluation: plotcal

```
CASA <23>: inp plotcal
---------> inp(plotcal)
#  plotcal :: An all-purpose plotter for calibration results
caltable            =        ''        #  Name of input calibration table
xaxis               =        ''        #  Value to plot along x axis (time,chan,freq,antenna,amp,phase,real,imag,snr)
yaxis               =        ''        #  Value to plot along y axis (amp,phase,real,imag,snr,antenna)
poln                =        ''        #  Antenna polarization to plot (RL,R,L,XY,X,Y,/)
field               =      ['']        #  field names or index of calibrators: ''==>all
antenna             =        ''        #  antenna/baselines: ''==>all, antenna = '3,VA04'
spw                 =        ''        #  spectral window:channels: ''==>all, spw='1:5~57'
timerange           =        ''        #  time range: ''==>all
subplot             =       111        #  Panel number on display screen (yxn)
overplot            =     False        #  Overplot solutions on existing display
clearpanel          =    'Auto'        #  Specify if old plots are cleared or not
iteration           =        ''        #  Iterate plots on antenna,time,spw,field
plotrange           =        []        #  plot axes ranges: [xmin,xmax,ymin,ymax]
showflags           =     False        #  If true, show flagged solutions
plotsymbol          =       'o'        #  pylab plot symbol
plotcolor           =    'blue'        #  initial plotting color
markersize          =       5.0        #  Size of plotted marks
fontsize            =      10.0        #  Font size for labels
showgui             =      True        #  Show plot on gui
figfile             =        ''        #  ''= no plot hardcopy, otherwise supply name
async               =     False        #  If true the taskname must be started using plotcal(...)

CASA <24>:
```

NRAO

M82 Calibration Sequence

1. Fast, per-integration provisional phase-only solution, for use in slower amplitude solution
2. Per-scan amplitude-only solution
3. Per-scan phase-only solution (fast solution discarded)
4. fluxscale to correct for unknown gain calibrator flux density
5. applycal to generate calibrated data on calibrators and science target

Flux density calibration: fluxscale

- Correct for incorrect flux density assumptions in gaincal solutions
  - Enforce *constant* mean gain amplitude among calibrators (referred to a flux density calibrator)

$$V_{ij}^{obs} = G_i G_j^* V_{ij}^{model} = G_i' G_j'^* \frac{V_{ij}^{model}}{S_\nu} = G_i' G_j'^* 1$$

$$\left|G_i\right| = \frac{\left|G_i'\right|}{\sqrt{S_\nu}} \qquad \rightarrow \qquad \sqrt{S_\nu} = \frac{\left\langle \left|G_i'\right| \right\rangle}{\left\langle \left|G_i\right| \right\rangle_{f.d.calibrator}}$$

  - Yields flux density estimate for calibrator

Flux density calibration: fluxscale

```
CASA <13>: inp fluxscale
---------> inp(fluxscale)
#  fluxscale :: Bootstrap the flux density scale from standard calibrators
vis                 =         ''        #  Name of input visibility file (MS)
caltable            =         ''        #  Name of input calibration table
fluxtable           =         ''        #  Name of output, flux-scaled calibration table
reference           =       ['']        #  Reference field name(s) (transfer flux scale FROM)
transfer            =       ['']        #  Transfer field name(s) (transfer flux scale TO), '' -> all
append              =      False        #  Append solutions?
refspwmap           =       [-1]        #  Scale across spectral window boundaries.  See help fluxscale
async               =      False        #  If true the taskname must be started using fluxscale(...)

CASA <14>:
```

• TBD: selection of specific timerange(s) and antennas to limit scaling calculation

Calibration accumulation: accum

- Optional
- Accumulates calibration solution tables into a single cal table, prior to (optional) application to data
  - Transfer among fields (e.g., calibrator to target)
  - Sampling, interpolation

$$G_i^{net} = G_i^a G_i^b G_i^c \cdots G_i^{template}$$

- In this demo, we use accum to illustrate calibration transfer and interpolation

NRAO

Calibration accumulation: accum

```
CASA <21>: inp accum
---------> inp(accum)
#  accum :: Accumulate incremental calibration solutions into a calibration table
vis                 =         ''        #  Name of input visibility file (MS)
tablein             =         ''        #  Input (cumulative) calibration table; use '' on first run
     accumtime      =        1.0        #  Timescale on which to create cumulative table

incrtable           =         ''        #  Input incremental calibration table to add
caltable            =         ''        #  Output (cumulative) calibration table
field               =       ['']        #  List of field names to process from tablein
calfield            =       ['']        #  List of field names to use from incrtable.
interp              =   'linear'        #  Interpolation mode to use for resampling incrtable solutions
spwmap              =       [-1]        #  Spectral window combinations to apply
async               =      False        #  If true the taskname must be started using accum(...)

CASA <22>:
```

Calibration application: applycal

- Generates calibrated data
  - Transfer among fields
  - Interpolation

Calibration application: applycal

```
CASA <14>: inp applycal
---------> inp(applycal)
#  applycal :: Apply calibrations solutions(s) to data
vis                 =         ''        #  Nome of input visibility file
field               =         ''        #  Select field using field id(s) or field name(s)
spw                 =         ''        #  Select spectral window/channels
selectdata          =       True        #  Other data selection parameters
     timerange      =         ''        #  Select data based on time range
     uvrange        =         ''        #  Select data within uvrange (default units meters)
     antenna        =         ''        #  Select data based on antenna/baseline
     scan           =         ''        #  Scan number range
     msselect       =         ''        #  Optional complex data selection (ignore for now)

gaintable           =       ['']        #  Gain calibration table(s) to apply on the fly
gainfield           =       ['']        #  Select a subset of calibrators from gaintable(s)
interp              =       ['']        #  Interpolation mode (in time) to use for each gaintable
spwmap              =         []        #  Spectral windows combinations to form for gaintables(s)
gaincurve           =      False        #  Apply internal VLA antenna gain curve correction
opacity             =        0.0        #  Opacity correction to apply (nepers)
parang              =      False        #  Apply parallactic angle correction
calwt               =       True        #  Calibrate weights along with data for all relevant calibrations
async               =      False        #  If true the taskname must be started using applycal(...)

CASA <15>:
```

Other calibration tasks, options

- bandpass: channel-dependent gaincal
- polcal: instrumental polarization
- blcal: *baseline-based* gains (use with great care!)
- smoothcal: smooth calibration solutions
- uvcontsub/uvcontsub2: continuum estimation/subtraction

- Additional calibration options:
  - combine : combine data on field, spw, scan axes
  - solnorm: normalize solutions
  - gaintype='T', 'GSPLINE'; bandtype='BPOLY'
  - spwmap: transfer calibration among spws
  - opacity: correct for absorption
  - gaincurve: correct for antenna efficiency elevation-dependence

Self-calibration

- Calibration transfer is not ideal because it does not sample the calibration effects in exactly the same direction at exactly the same times
- With sufficient SNR on appropriate timescale(s), can iterate calibration solving on the target
  - Deconvolution provides models against which to solve for calibration

- Bookkeeping exercise:
  - split out nominally calibrated data first
  - apply sequences of self-calibration solutions as needed

Calibration TBD

- Antenna-position corrections
- Other priors, like Tsys
- Selection in fluxscale
- Delay-aware gaincal (wide bandwidths)
  - c.f. fringe-fitting
- gaintype='GSPLINE' improvements
- Generalized instrumental polarization calibration (linears)
- Improved cal table definition, better integration with MS
- Calibration plotting in plotms
- Performance improvements
- Command-based calibration flagging (flagcal)

NRAO

NRAO