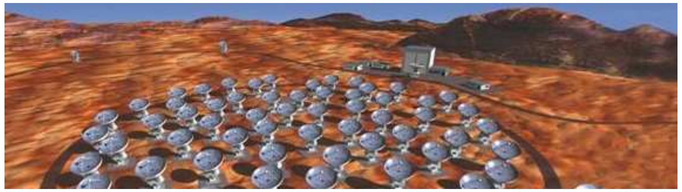
Data Calibration

Frédéric Gueth, IRAM Grenoble

Cycle 0 data reduction tutorial
ESO, January 19-20 2012

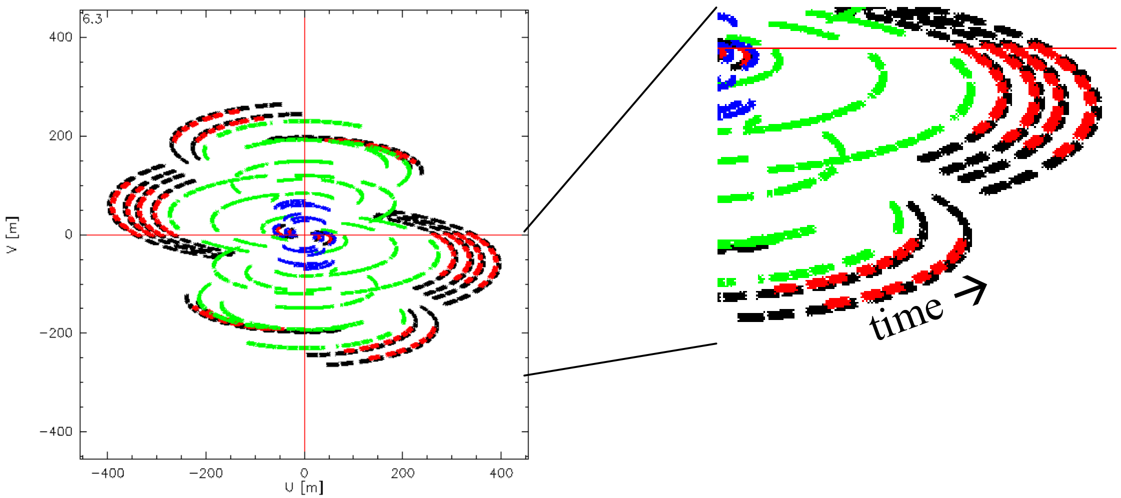
Institut de Radioastronomie Millimétrique

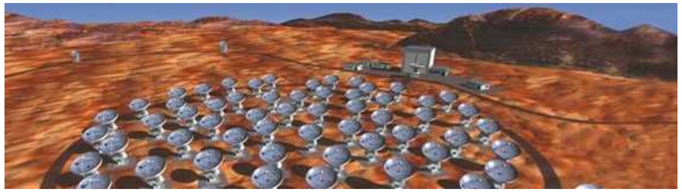
EUROPEAN ARC
ALMA Regional Centre || IRAM



Observations

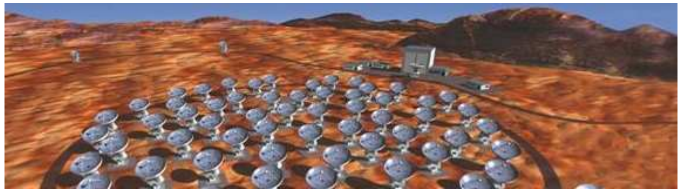
- An interferometer provides visibilities **V = complex number**
- **V(nu,t)** varies as a function of frequency (spectra) and time
  - (u,v) plane dependence not/rarely considered in calibration
  - Time variation = sampling different positions in the uv plane
  - Time calibration critical in interferometry (more than Single Dish) – errors can introduce artifacts in the image!





Observations

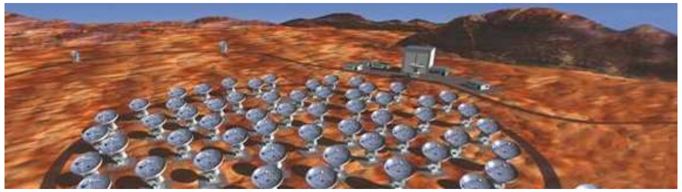
- Observation: $\mathbf{V_{obs} = GV_{true} + N}$
  - V_{true} = true visibilities = FT(sky) - G = “gain”
  - V_{obs} = observed visibilities - N = noise



Observations

- Observation: $\mathbf{V_{obs} = GV_{true} + N}$
  - V_{true} = true visibilities = FT(sky) - G = "gain"
  - V_{obs} = observed visibilities - N = noise

Atmosphere
Antenna
Receiver
Electronics
Correlator



Observations

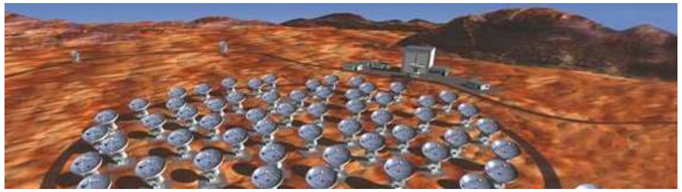
- Observation: $\mathbf{V_{obs} = GV_{true} + N}$
  - V_{true} = true visibilities = FT(sky) - G = "gain"
  - V_{obs} = observed visibilities - N = noise

Atmosphere
Antenna
Receiver
IF transport
Correlator

Antenna-based effects
Baseline-based effects

Frequency variation
Time variation
Polarization

Amplitude
Phase



Observations

- Pointing
- Focus

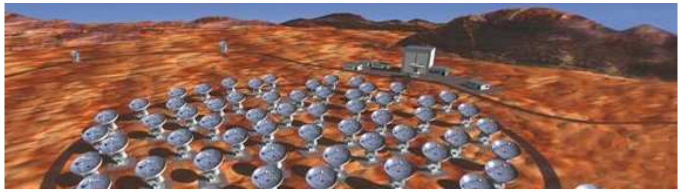
Real-time array calibrations

- Antenna positions
- Delay
- Atmospheric calibration (hot/cold measurements) → **the data are already in correct K scale**

New values can be entered off-line if necessary

- Atmospheric phase (WVR) correction→ **the data are already corrected** for short-timescale fluctuations

Can be reprocessed off-line



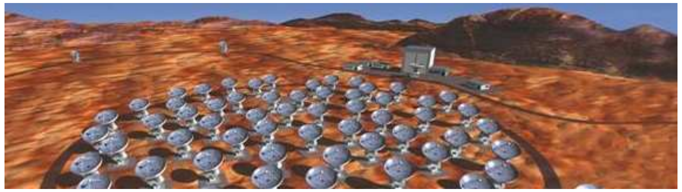
Calibration principles

2 main off-line calibrations:

- **Frequency-dependent response of the system** → **bandpass( )**
  - Receivers intrinsic response
  - Delay offsets (slope on phase)
  - Coaxial cables attenuation
  - Antenna chromatism
  - Atmosphere (O2, O3 lines)

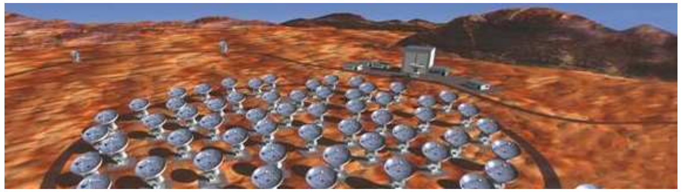
- **Time-dependent response of the system** → **gaincal( )**
  - Atmosphere
  - Antenna position errors (period 24 h)
  - Antenna/electronics drifts

"electronic gain" = cm
"amplitude, phase" = mm



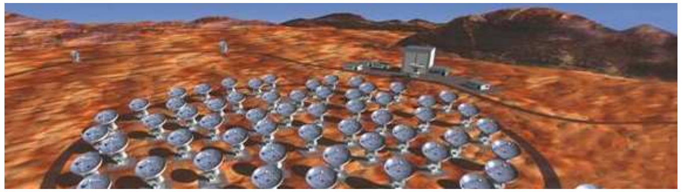
Calibration principles

- Bandpass, gaincal: calibration of Vobs = G V
  - **Observe source with known structure/spectra**
  - Expected visibilities Vmodel are known
  - Solve for G
- In practice: **quasars = continuum bright point sources**
  - Amplitude(nu) = source flux; phase(nu) = zero
  - Amplitude(t) = source flux; phase(t) = zero
  - Taking into account source structure/spectra is also possible but more complex/more model dependant

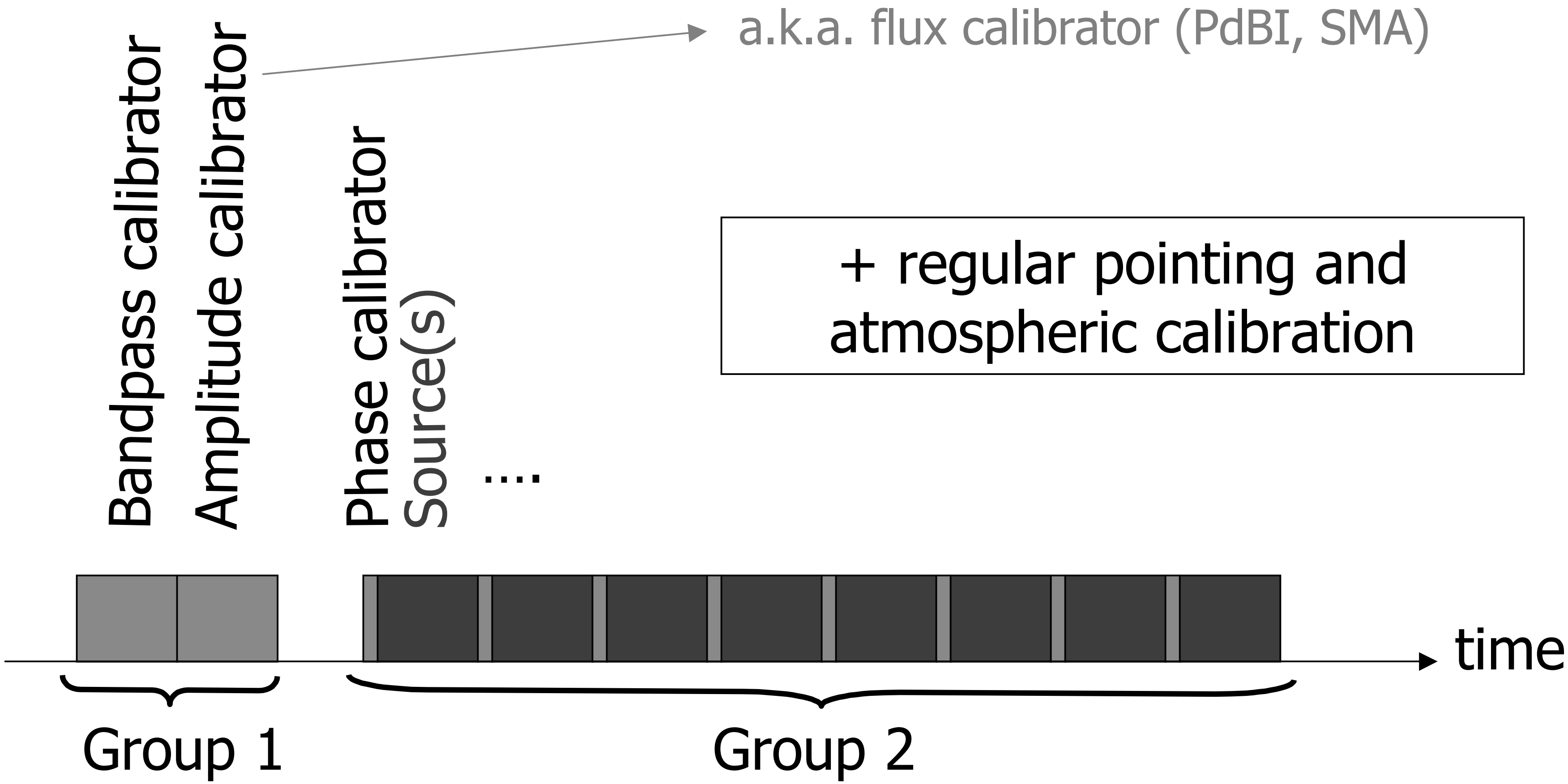


Calibration principles

- Bandpass, gaincal: calibration of Vobs = G V
  - **Observe source with known structure/spectra**
  - Expected visibilities Vmodel are known
  - Solve for G
- In practice: **quasars = continuum bright point sources**
- Quasars are bright but highly variable
  - Need to observe a source of known flux to get absolute flux scale → ALMA: **solar system bodies**

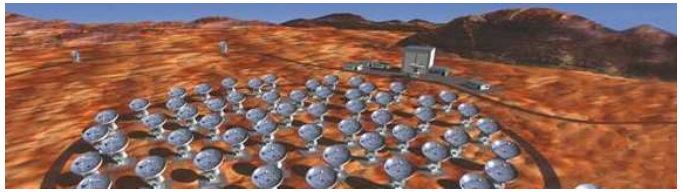


ALMA SB



(anywhere in the sky)

(phase calibrator close to the source; cycle time ~5 minutes)

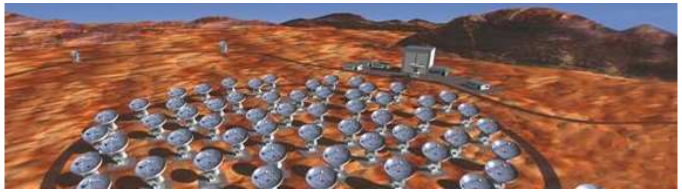


Antenna-based calibration

- Calibrate only temporal or frequency effects, do not consider dependence on (u,v)
- True visibility for baseline ij: $\mathbf{V_{ij}(\nu,t)}$
- Observed visibility:

$$\mathbf{Vobs_{ij}(\nu,t) = G_{ij}(\nu,t)\, V_{ij}(\nu,t) + noise}$$

- $\mathbf{G_{ij}}$ = complex gain (amplitude & phase)
- Scalar description here – no polarization

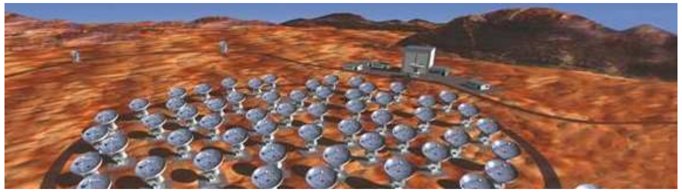


Antenna-based calibration

- **Most of the effects are antenna-based**
  - Pointing, Focus, Antenna position, Atmosphere, Receivers noise, Receivers bandpass…

- Gain decomposition: $\mathbf{Vobs_{ij} = G_{ij}\, V_{ij} = g_i\, g_j\, V_{ij}}$

- The calibration tries to **solve for each antenna gain**, to get a consistent solution over the whole dataset (baselines 12, 13, and 23 are not independent)

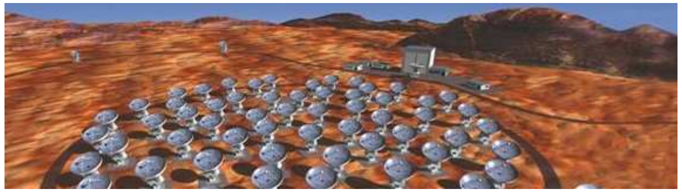


Antenna-based calibration

Advantages of using the antenna-based gains:

1. most of the effects are **truly antenna-based**
   - except if problems in correlator – should not happen
   - except if amplitude loss due to **decorrelation** (short time scale phase fluctuation) **– should be corrected by WVR**

2. precision to which antenna gains are determined is **improved by a factor √N** over the precision of the measurement of baseline gains
   - N complex unknown (one per antenna)
   - N(N-1)/2 equations (one per baseline)
   - System is over-determined



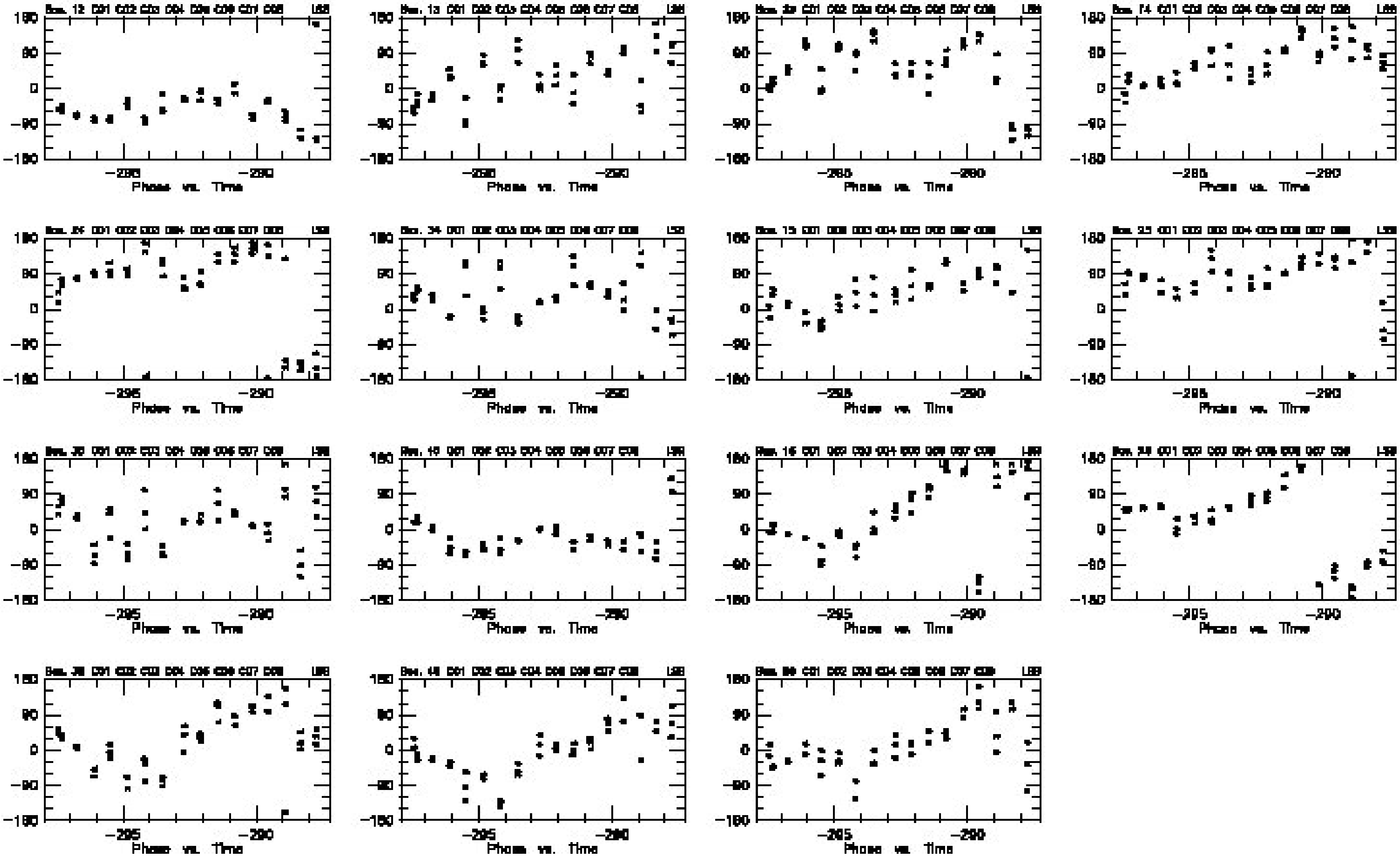
Antenna-based calibration

- Phase closure relation (point source):
  - Antenna-based decomposition: $\varphi 12 = \varphi 1 - \varphi 2$
  - Phase closure: $\boldsymbol{\varphi 12 + \varphi 23 + \varphi 31 = 0}$

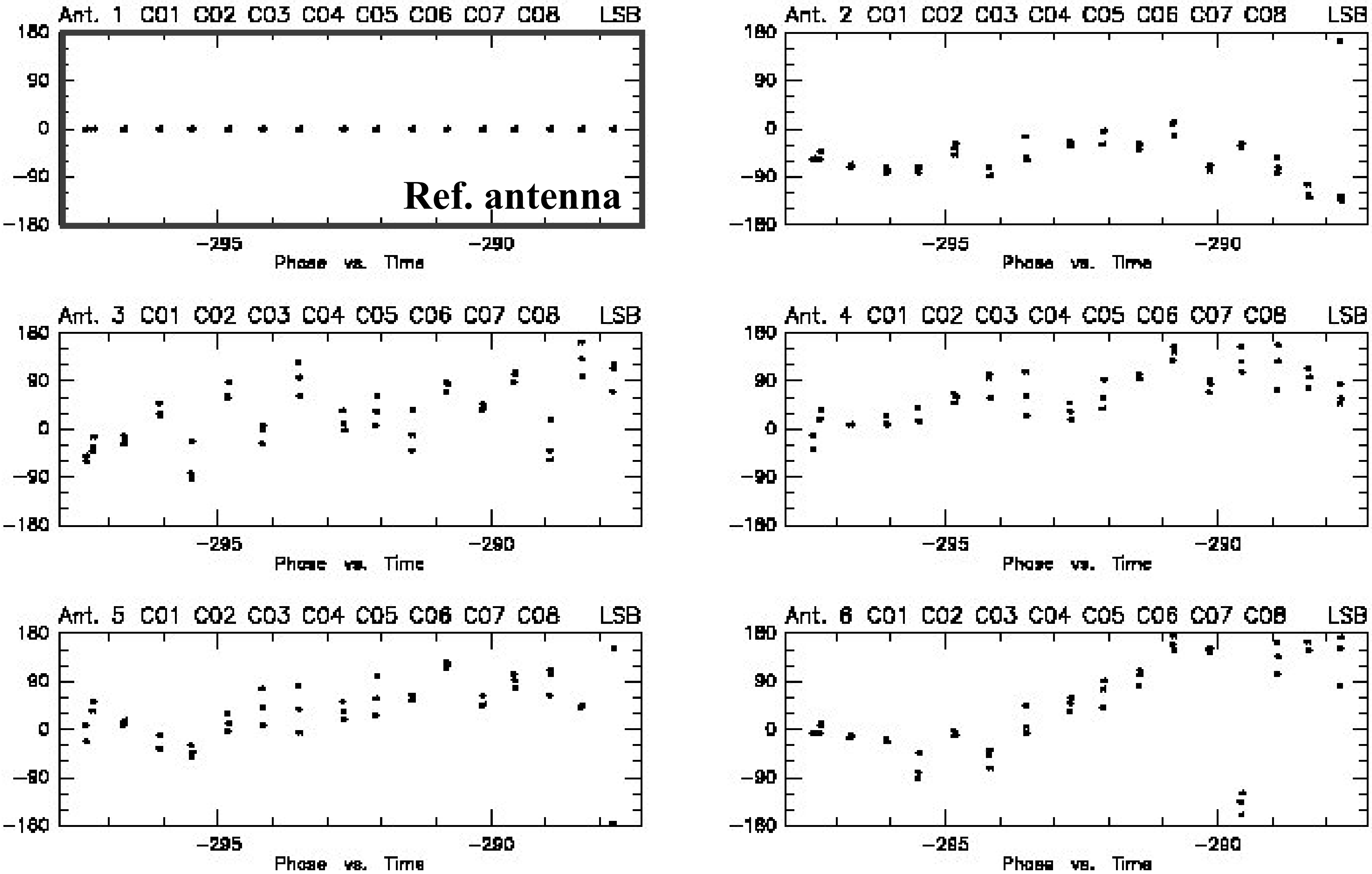
- Very useful relation when phases are too unstable to be directly measured (VLBI, optics)
- Similar relations exists for amplitude ratios

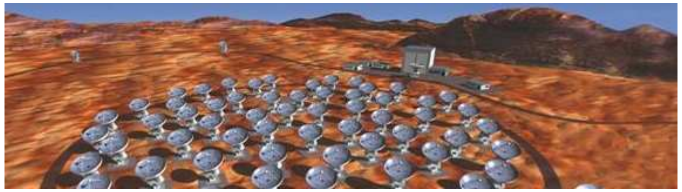
- **The decomposition in antenna-based gains implicitly takes into account the closure relations**

Phase vs. time -- Baselines



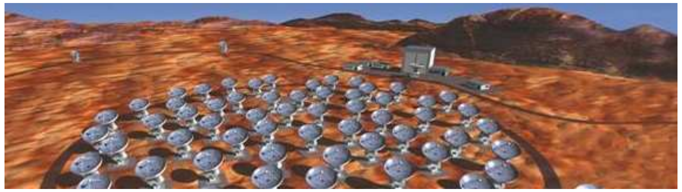
Phase vs. time -- Antennas





Data calibration strategy

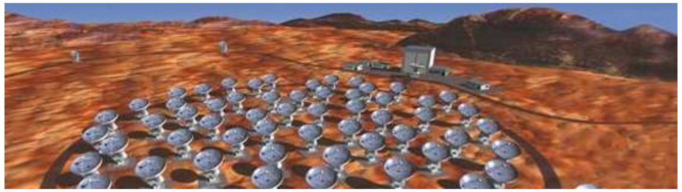
- **Basic assumption: time- and frequency-variations are decoupled**
- Quite robust:
  - Frequency response mostly due to receivers; stable until retuning
  - Time variations (atmosphere, antennas, …) mostly achromatic
- Start with time- or frequency-calibration?



Data calibration strategy

0. Flagging bad data
1. Bandpass
   a. Select brightest quasar
   b. Solve for temporal fluctuation (**gaincal**)
      - Apply gaincal solution
   c. Solve for bandpass calibration (**bandpass**)
2. Gain
   a. Select all calibrators
   b. Apply bandpass calibration
   c. Solve for gain solution (**gaincal**)
3. Find correct flux scale
   a. Set the flux of a reference source (**setjy**)
   b. Derive fluxes of other sources (**fluxscale**)
4. Apply on all data (**applycal**)

Solutions are written in CASA tables

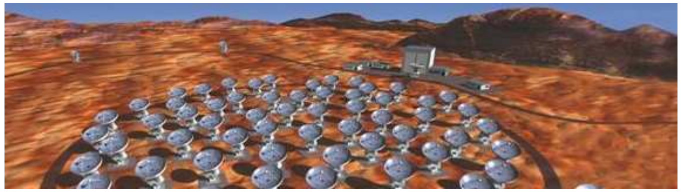


CASA script (1) NGC3256 Band3 SV

 # Solve for gain on bandpass calibrator

gaincal(vis='ngc3256_line.ms', caltable='cal-ngc3256.G1n', spw='*:40~80', field='1037*', selectdata=T, solint='int', refant='DV04', calmode='p')

- caltable = output
- field, spw = source, spectral window selection
- refant = reference antenna
- calmode = do it for phase only (‘p’) → calibration only used to phase up the data

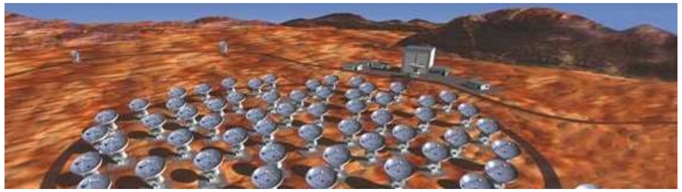


CASA script (2)

 # Bandpass solution

```
bandpass(vis = 'ngc3256_line.ms', caltable = 'cal-ngc3256.B1n',
    gaintable = 'cal-ngc3256.G1n',
    timerange='<2011/04/16/15:00:00', field = '1037*',
    minblperant=3, minsnr=2, solint='inf', combine='scan',
    bandtype='B', fillgaps=1, refant = 'DV04', solnorm = T)
```

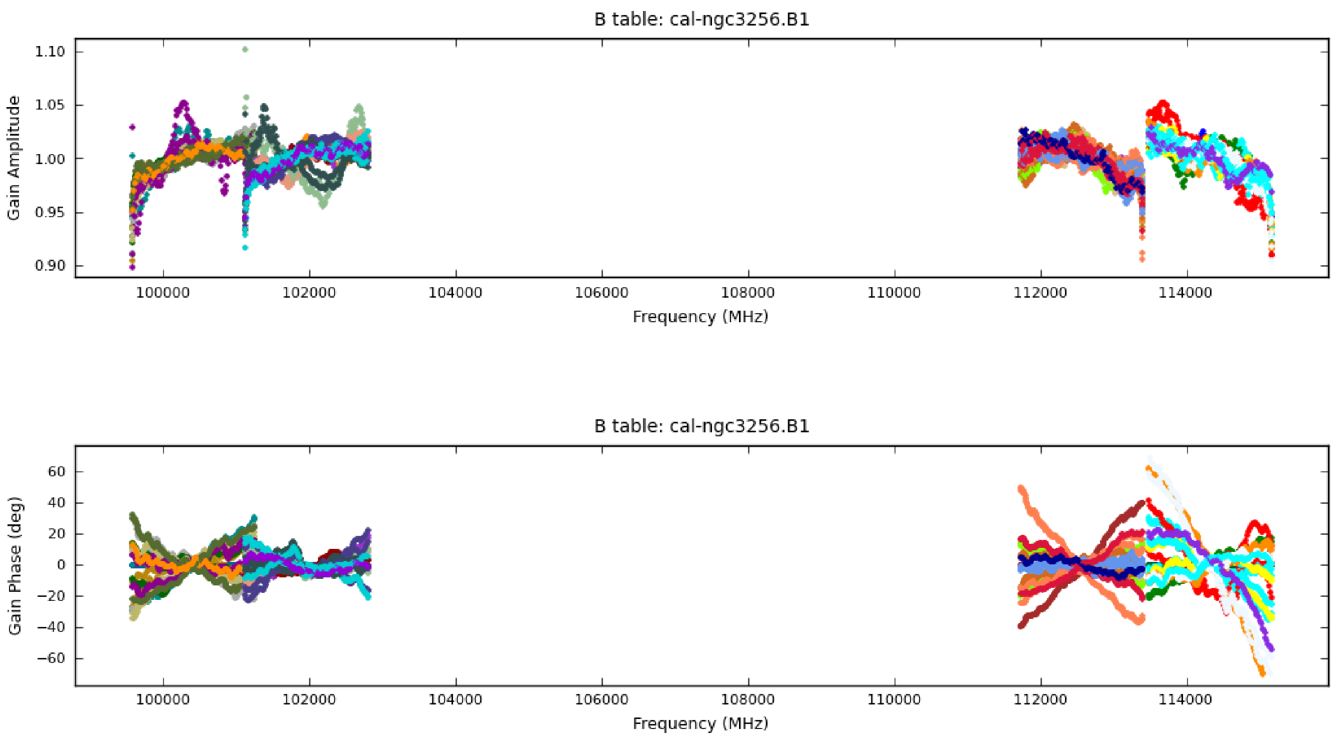
- caltable = output
- gaintable = previous gain table, to be applied on-the-fly
- minblperant = minimum number of baseline per antenna
- solint = get a single bandpass solution, no time dependence
- bandtype = B → get solution point per point (will be later interpolated) rather than fitting a polygon (BPOLY)



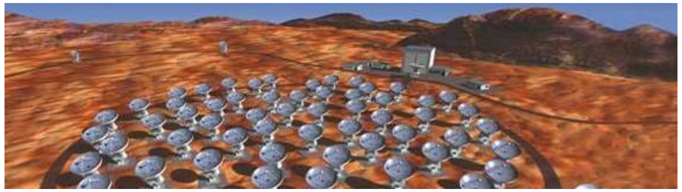
CASA script (3)

 # Plot calibration results

plotcal(caltable = 'cal-ngc3256.B1', xaxis='freq', yaxis='phase',
spw='',subplot=212, overplot=False, plotrange = [0,0,-70,70],



<u>Caution</u>: plots the calibration, not the data, not the calibrated data

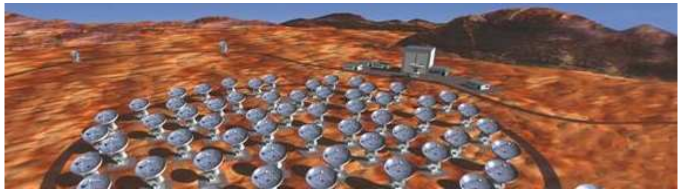


CASA script (4)

 # Solve for gain

```
gaincal(vis = 'ngc3256_line.ms', caltable = 'cal-ngc3256.G2n', spw
  ='*:16~112', field = '1037*,Titan', minsnr=1.0, solint= 'inf',
  selectdata=T, solnorm=False, refant = 'DV04', gaintable = 'cal-
  ngc3256.B1n', calmode = 'ap')
```

- gaintable = bandpass calibration, to be applied on-the-fly
- caltable = output
- do it for both amplitude and phase (ap)

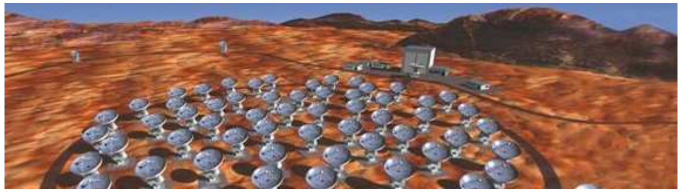


CASA script (5)

 # Use Titan as flux reference

setjy(vis='ngc3256_line.ms', field='Titan', standard='Butler-JPL-Horizons 2010', spw='0,1,2,3')

- Set Titan flux from internal catalogue/model
- Flux scale accuracy far from final ALMA goals: solar system bodies not so well known in the mm domain
- Mars, Uranus, satellites, a few asteroids…
- Caution:Titan has strong, broad CO & HCN lines → may need to flag these channels

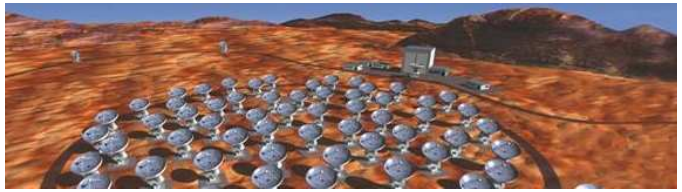


CASA script (6)

 # Solve for flux scale

fluxscale(vis="ngc3256_line.ms", caltable="cal-ngc3256.G2n",
fluxtable="cal-ngc3256.G2n.flux", reference="Titan",
transfer="1037*", refspwmap=[0,1,1,1])

- Use Titan flux as a reference to derive antenna efficiencies (Jy/K) factors & apply them to source 1037* (phase calibrator)
- Caltable is the calibration table created by gaincal
- Fluxtable = output = flux-corrected gain table

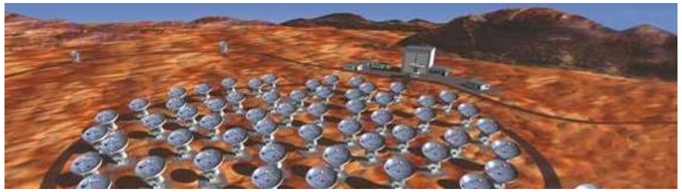


CASA script (7)

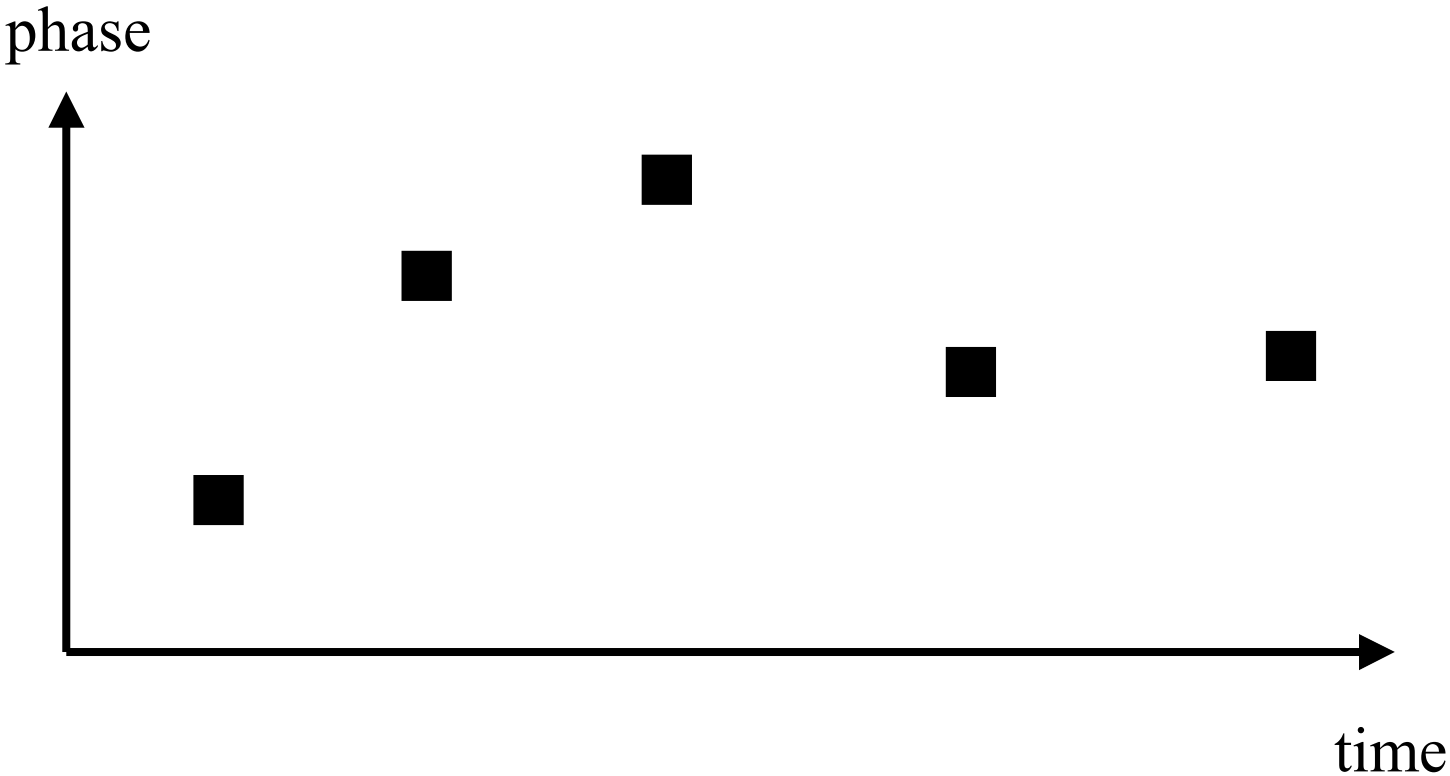
 # Apply solutions

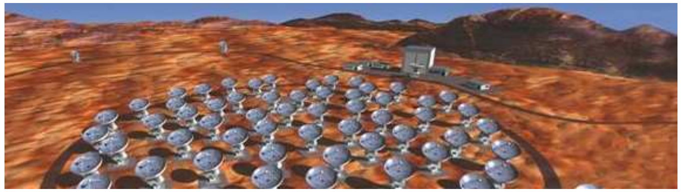
applycal(vis='ngc3256_line.ms', flagbackup=F,
field='NGC*,1037*', interp=['nearest','nearest'], gainfield = ['1037*', '1037*'], gaintable=['cal-ngc3256.G2n.flux', 'cal-ngc3256.B1n'])

- Now apply bandpass & gain calibration to the data, including at last the source (NGC*)
  - <u>table.flux contains flux-corrected gain</u>
  - <u>table.B1n contains bandpass</u>
- Calibrated data now in CORRECTED_DATA column
- interp = time interpolation technique: nearest or linear

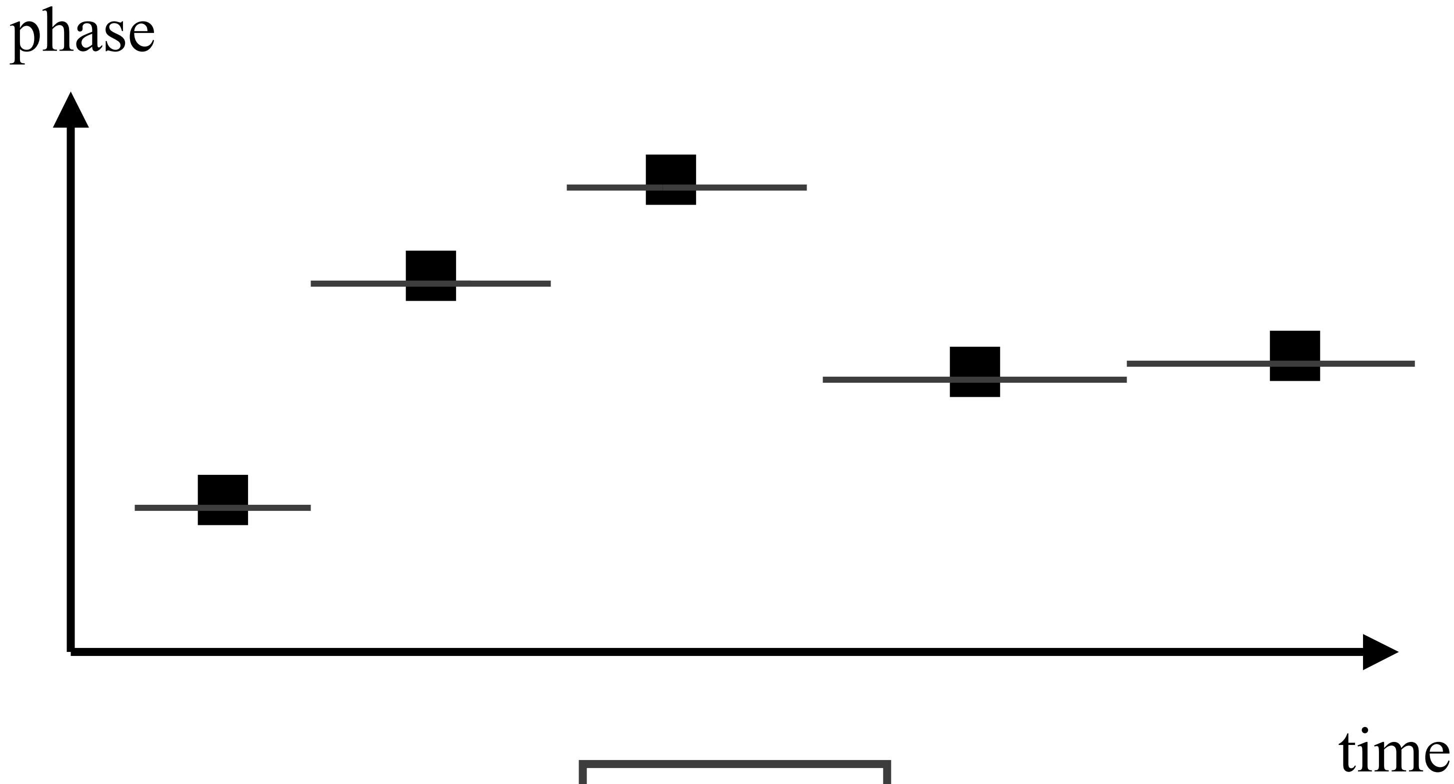


Time calibration

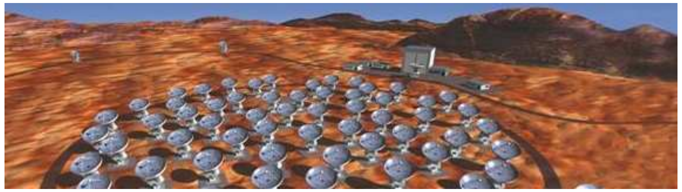




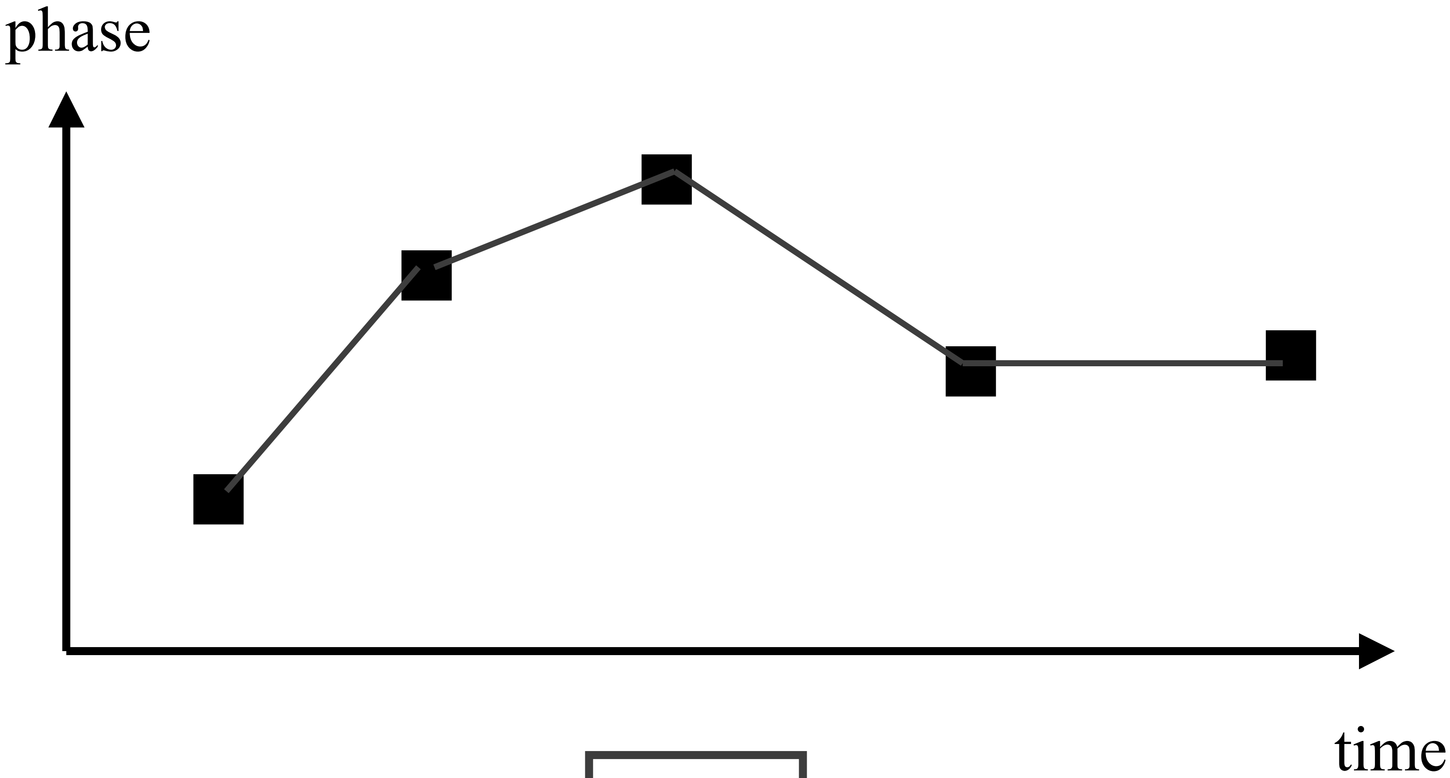
Time calibration



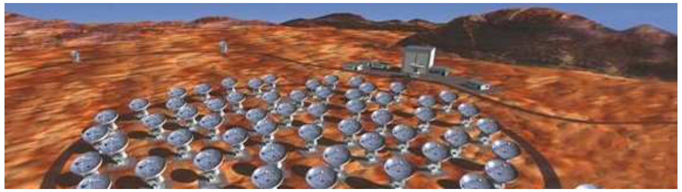
Nearest



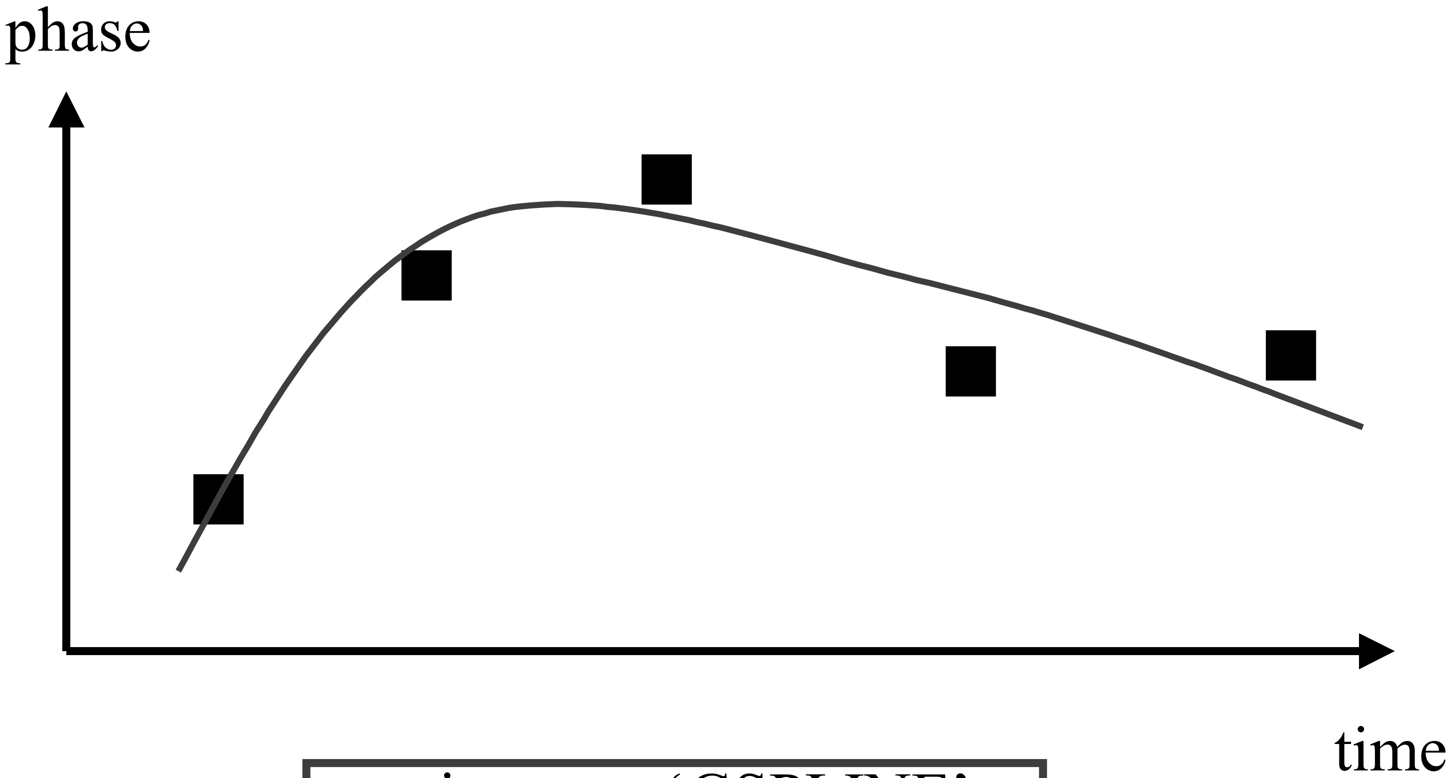
Time calibration



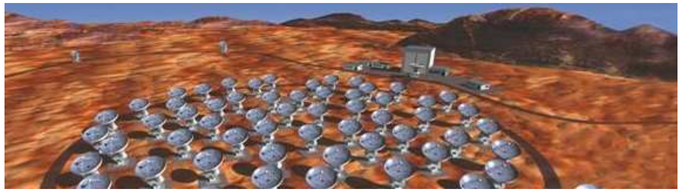
Linear



Time calibration



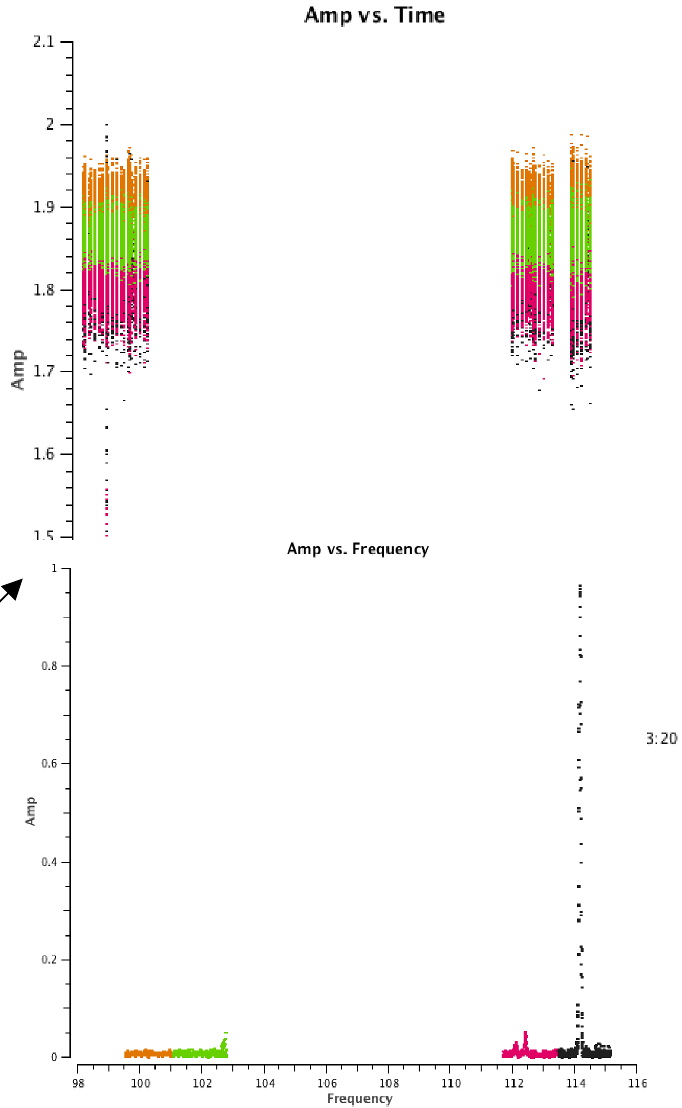
gaintype = 'GSPLINE'
better if low SNR

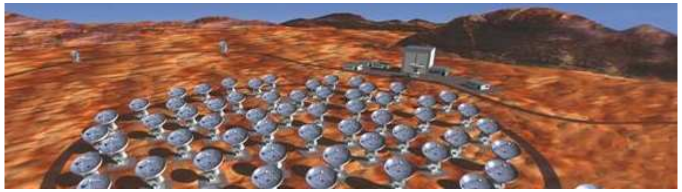


CASA script

- Other useful tasks:
  - **flagdata()** … to flag the data
  - **split()** … to split/extract/modify a MS:
    - extract subset of sources
    - bin the data on another resolution
  - **plotms()** … to plot/inspect the data

E.g. plot calibrated data (datacolumn ='corrected') vs time or frequency





CASA script

 # Solve for gain

gaincal(vis = 'ngc3256_line.ms', caltable = 'cal-ngc3256.G2n', spw ='*:16~112', field = '1037*,Titan', minsnr=1.0, solint= 'inf', selectdata=T, solnorm=False, refant = 'DV04', gaintable = 'cal-ngc3256.B1n', calmode = 'ap')

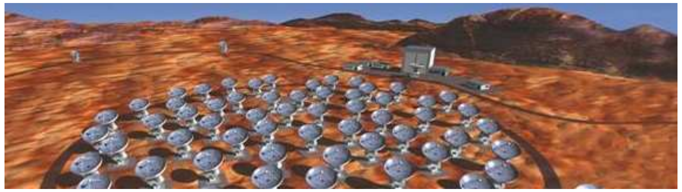
The devil is in the details = many inputs/tuning parameters ➔ need to practice (download SV data, attend a tutorial, etc.)

 # Solve for

fluxscale(vi 256.G2n",
fluxtable= ',
transfer="

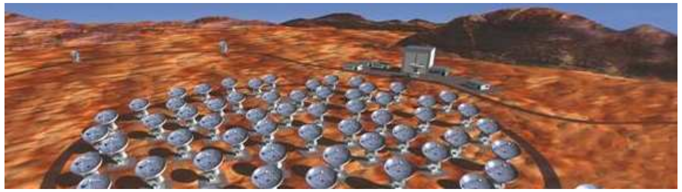
 # Apply solutions

applycal(vis='ngc3256_line.ms', flagbackup=F, field='NGC*,1037*', interp=['nearest','nearest'], gainfield = ['1037*', '1037*'], gaintable=['cal-ngc3256.G2n.flux', 'cal-ngc3256.B1n'])



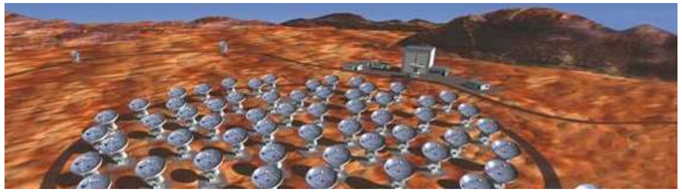
bandpass()

```
#  bandpass :: Calculates a bandpass calibration solution
vis                 = 'ngc5921.demo.ms' #  Nome of input visibility file
caltable            = 'ngc5921.demo.bcal' #  Name of output gain calibration
                                        #   table
field               =       '0'         #  Select field using field id(s) or
                                        #   field name(s)
spw                 =        ''         #  Select spectral window/channels
selectdata          =     False         #  Other data selection parameters
solint              =     'inf'         #  Solution interval
combine             =    'scan'         #  Data axes which to combine for solve
                                        #   (scan, spw, and/or field)
refant              =      '15'         #  Reference antenna name
minblperant         =         4         #  Minimum baselines _per antenna_
                                        #   required for solve
solnorm             =     False         #  Normalize average solution amplitudes
                                        #   to 1.0 (G, T only)
```



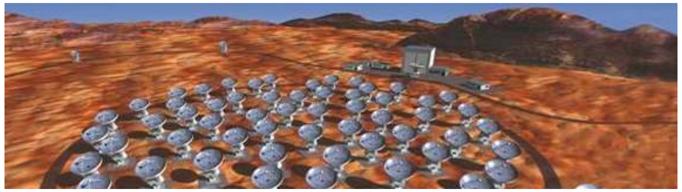
bandpass()

```
bandtype        =    'B'     #  Type of bandpass solution (B or
                             #   BPOLY)
     fillgaps   =      0     #  Fill flagged solution channels by
                             #   interpolation
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 bandpass(...)
```



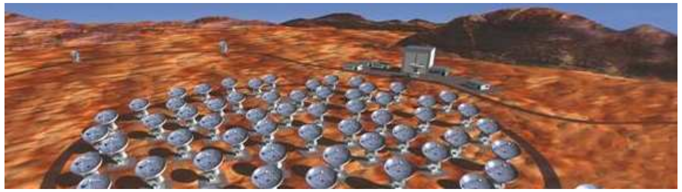
plotcal()

```
#   plotcal :: An all-purpose plotter for calibration results
caltable            = 'ngc5921.demo.bcal' #  Name of input calibration table
xaxis               =        ''        #  Value to plot along x axis(time,chan
                                       # ,freq,antenna,amp,phase,real,imag,sn
                                       #   r)
yaxis               =    'phase'       #  Value to plot along y axis
                                       #   (amp,phase,real,imag,snr,antenna)
poln                =        ''        #  Antenna polarization to plot
                                       #   (RL,R,L,XY,X,Y,/)
field               =       '0'        #  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             =       212        #  Panel number on display screen (yxn)
```



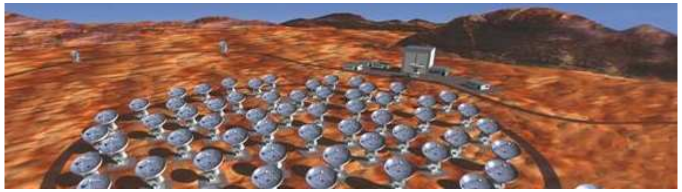
plotcal()

```
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(...)
```



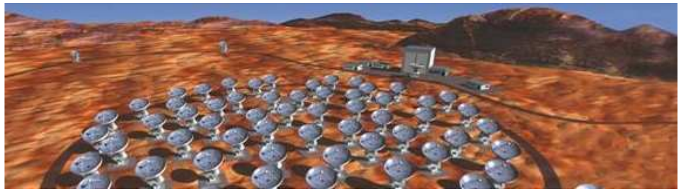
gaincal()

```
#  gaincal :: Determine temporal gains from calibrator observations
vis                 = 'ngc5921.demo.ms' #  Nome of input visibility file
caltable            = 'ngc5921.demo.gcal' #  Name of output gain calibration
                                        #   table
field               =      '0,1'        #  Select field using field id(s) or
                                        #   field name(s)
spw                 =   '0:6~56'        #  Select spectral window/channels
selectdata          =      False        #  Other data selection parameters
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              =       '15'        #  Reference antenna name
minblperant         =          4        #  Minimum baselines _per antenna_
                                        #   required for solve
minsnr              =        1.0        #  Reject solutions below this SNR
solnorm             =      False        #  Normalize average solution amplitudes
                                        #   to 1.0 (G, T only)
```



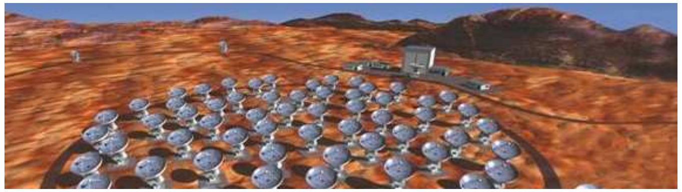
gaincal()

```
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    = 'ngc5921.demo.bcal' #  Gain calibration table(s) to apply
                                   #   on the fly
gainfield    =         ''          #  Select a subset of calibrators from
                                   #   gaintable(s)
interp       =  'nearest'          #  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(...)
```



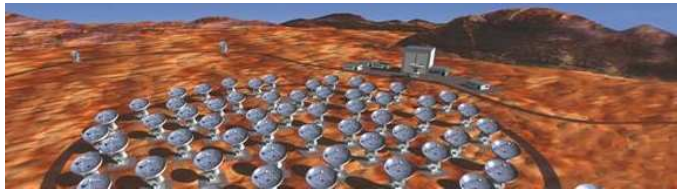
setjy()

```
#  setjy ::
vis                = 'ngc5921.demo.ms' #  Name of input visibility file (MS)
field              = '1331+305*'       #  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(...)
```



fluxscale()

```
#  fluxscale :: Bootstrap the flux density scale from standard calibrators
vis                  = 'ngc5921.demo.ms' #  Name of input visibility file (MS)
caltable             = 'ngc5921.demo.gcal' #  Name of input calibration table
fluxtable            = 'ngc5921.demo.fluxscale' #  Name of output, flux-scaled
                                         #   calibration table
reference            =    '1331*'        #  Reference field name(s) (transfer
                                         #   flux scale FROM)
transfer             =    '1445*'        #  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(...)
```



applycal()

```
#  applycal :: Apply calibrations solutions(s) to data
vis                   = 'ngc5921.demo.ms' #  Nome of input visibility file
field                 =        '0'        #  Select field using field id(s) or
                                          #   field name(s)
spw                   =         ''        #  Select spectral window/channels
selectdata            =      False        #  Other data selection parameters
Gaintable             = ['ngc5921.demo.fluxscale', 'ngc5921.demo.bcal'] #  Gain ca
                                         #   libration table(s) to apply on the
                                          #   fly
gainfield             = ['0', '*']        #  Select a subset of calibrators from
                                          #   gaintable(s)
interp                = ['linear', 'nearest'] #  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(...)
```