APSYNSIM and APSYNTRU - version 2.3-beta
Instructions and exercises

Iván Martí Vidal
Departament d'Astronomia i Astrofísica
Universitat de València (Spain)

June 22, 2023

1 USAGE OF APSYNSIM

APSYNSIM (*AP*erture *SYN*thesis *SIM*ulator) is a simple interactive tool to help the students visualize and understand the basics of the Aperture Synthesis technique, applied to astronomical interferometers. This simulator is based on `Python`, and uses the `Numpy/Scipy` and `Matplotlib` (open source) packages. The *APSYNSIM* code is also open (distributed under the GPL version 3) and can be downloaded from:

`https://launchpad.net/apsynsim`

of from (recommended):

`https://github.com/marti-vidal-i/APSYNSIM.git`

The user can load many different (connected) interferometer arrays and source models (and also create their own), change the observing conditions in real time (e.g., source coordinates, observing wavelength, antenna location, integration time, etc.), and even deconvolve the interferometric images and corrupt the data with gain errors (in amplitude and phase).

1.1 Installation

GNU/Linux

1. Install the needed packages, i.e.:

   - `python3`
   - `python3-numpy`
   - `python3-scipy`
   - `python3-matplotlib`
   - `python3-tk`
   - `python3-mpltoolkits.basemap`
   - `python3-astropy`

   Please, tell me if something else is missing in your system.

   If you have conda, just run:

   - `conda create -n apsyn python=3.8`
   - `conda activate apsyn`
   - `conda install matplotlib scipy astropy basemap`

2. Just run the script `APSYNSIM.py` (located in the `SCRIPT` directory):

   ```
   python APSYNSIM.py
   ```

3. You can also run `APSYNTRU` (i.e., the real data imager!), which is located in the `APSYNTRU/SCRIPT` directory.

Mac OS

The instructions are similar to those in GNU/Linux. Many thanks to Silvio Fuchs for providing them!
IN THE TERMINAL:

1. Create conda environment:

   ```
   conda create -n apsyn python=3.10
   ```

2. Activate it:

   ```
   conda activate apsyn
   ```

3. Install packages:

   - `conda install numpy`
   - `conda install scipy`
   - `conda install matplotlib`
   - `conda install tk`
   - `conda install basemap-data`
   - `conda install basemap-data-hires`
   - `conda install astropy`

4. If needed (depends on the release you have downloaded), change line number 3166 in the `APSYNSIM.py` script:

   from: `root.attributes('-zoomed',True)`

   to: `root.state('zoomed')`

Windows

Unfortunately, I do not have any Virtual Machine with the newest MS Windblows versions, which means that I cannot generate binary files as before. But, fortunately, there are ways of running Python scripts in Windows (I'm afraid you will hae to "google" for them).

1.2 Main Window. The GUI

The main APSYNSIM window is shown in Figure 1. The window can be divided in six parts. From left to right (and top to bottom):

- **ARRAY CONFIGURATION**. Shows the spatial distribution of the interferometer elements (antennas). The user can pick and drag the antennas to change their locations. When an antenna is moved, all the other plots are refreshed in real time. The index of the selected antenna, and its current coordinates, are shown in the upper-left corner of the figure.

  If the subarraying feature is being used (so that there are effectively two interferometers observing the same source) each subarray with be plotted in a different color. A slider will also appear at the bottom of the array configuration, where the user can finetune the relative weight between the two subarrays. The quantity that can be set with the slider is "log(W1/W2)", i.e. the logarithm (base 10) of the weight ratio between the two subarrays.

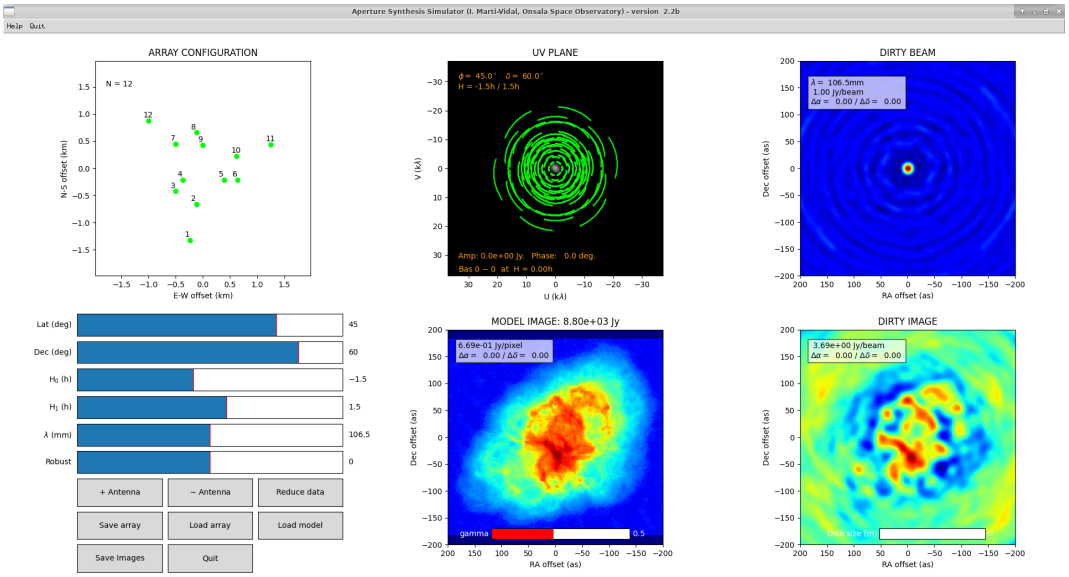

Figure 1: Main window of *APSYNSIM*.

- **UV PLANE**. Shows, in gray scale, the amplitude of the Fourier transform of the observed source. It also shows the tracks of the projected baselines in the UV plane (i.e., the UV points sampled by the interferometer). If subarraying is used, different colors are assigned to the different subarrays.

  In the upper-left corner of the figure, a text shows the current latitude of the observatory, the source declination, and the hour-angle coverage of the observations. If the user clicks on any point in the UV plane, the program prints the value of the Fourier transform at that point (amplitude and phase).

  If the click is done over a UV track, the program also prints the baseline (and hour angle) for that particular point. A red line in the **ARRAY CONFIGURATION** plot, joining the antennas of the baseline, is also shown when the user clicks on a UV track.

- **DIRTY BEAM**. The PSF of the interferometer. It depends on the UV coverage and the visibility-weighting scheme. If the user clicks on any point in the image, the program prints its coordinates (RA and Dec offsets with respect to the image center) and intensity (brightness, in Jy/beam) in a text at the upper-left corner.

- **Program controls**. With these controls, the user can change:

  – The latitude of the observatory.

  – The source declination.

  – The hour-angle coverage (automatically constrained to the time range when the source is above the horizon).

  – The observing wavelength.

  – The visibitlity weighting scheme (based on Briggs), defined from -2 (uniform weighting) to $+2$ (natural weighting).

  Other controls (buttons) are:

  ✓ **+ Antenna**. Adds an antenna to the array. The default position of the antenna is at the array center. The user can drag it to any other position.

  ✓ **− Antenna**. Removes an antenna from the array (the one with the highest index number). The program remembers the latest position of this antenna (in case the user wants to recover it by pressing **+ Antenna**).

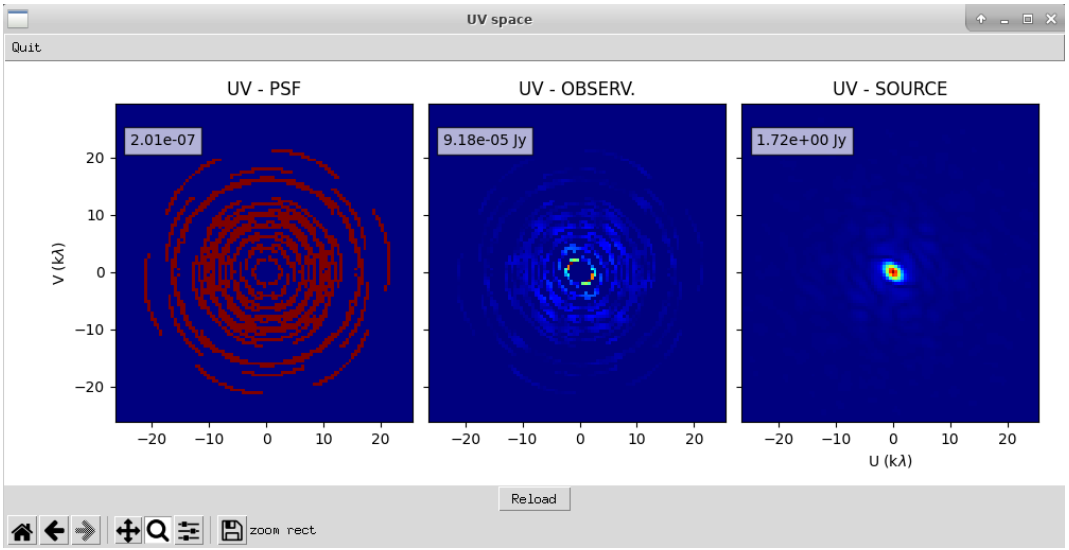

Figure 2: Fourier transforms of the Main Window images (it is shown by pressing "u" with the focus on the Main Window).

- ✓ **Save array**. Saves the current array configuration into an ASCII file.
- ✓ **Load array**. Allows the user to load any array configuration from an ASCII file.
- ✓ **Load model**. Allows the user to load any model source from an ASCII file.
- ✓ **Save Images**. Saves all the images (PSF, Convolved model and CLEAN reconstruction, if it exists) in FITS format.
- ✓ **Quit**. Quits the program. This is the recommended way to quit the application (together with the *Quit* menu button at the top of the window).
- ✓ **Reduce data**. This will open the GUI for the deconvolution (i.e. the CLEAN algorithm) and/or for the data corruption (see next section).

Some special keys (these work whenever the focus is set on the main window) are:

- **Z** (i.e., capital "Z", or `Shift+z`). Pressing this key will zoom in at any of the images in the main window. The zoom will be centered on the position of the mouse cursor.
- **z** (i.e., small "z"). Pressing this key will zoom out any of the images in the main window. The zoom will be centered on the position of the mouse cursor.
- **c**. Pressing this key will change the color palette (from hue to grayscale and viceversa).
- **u**. Pressing this key will show a new window with the FFTs of the source image, the PSF, and the dirty image (see Fig. 2). The user will have to press the "Reload" button in this window to refresh the figures, if any change is made in the main window.

1.3 CLEAN and gain corruption. The GUI

This window (see Fig. 3) shows two figures: the image of the CLEAN residuals (left) and the image of the convolved CLEAN model (right). The user can zoom in and out using the same keys as with the main window (i.e., "Z" zooms in and "z" zooms out).

For the CLEANing, the user can add an indefinite number of CLEAN masks (windows), by clicking and dragging on the residuals image with the *left* mouse button. The masks can be removed by clicking and dragging with the *right* mouse button.

At the left of the GUI window, the user can set the main CLEAN parameters (i.e., the loop gain, the number of iterations, and the CLEAN threshold). The threshold is given in Jy/beam. Negative values for the threshold allows the program to CLEAN negative components. The rest of buttons found at the left of the GUI are:

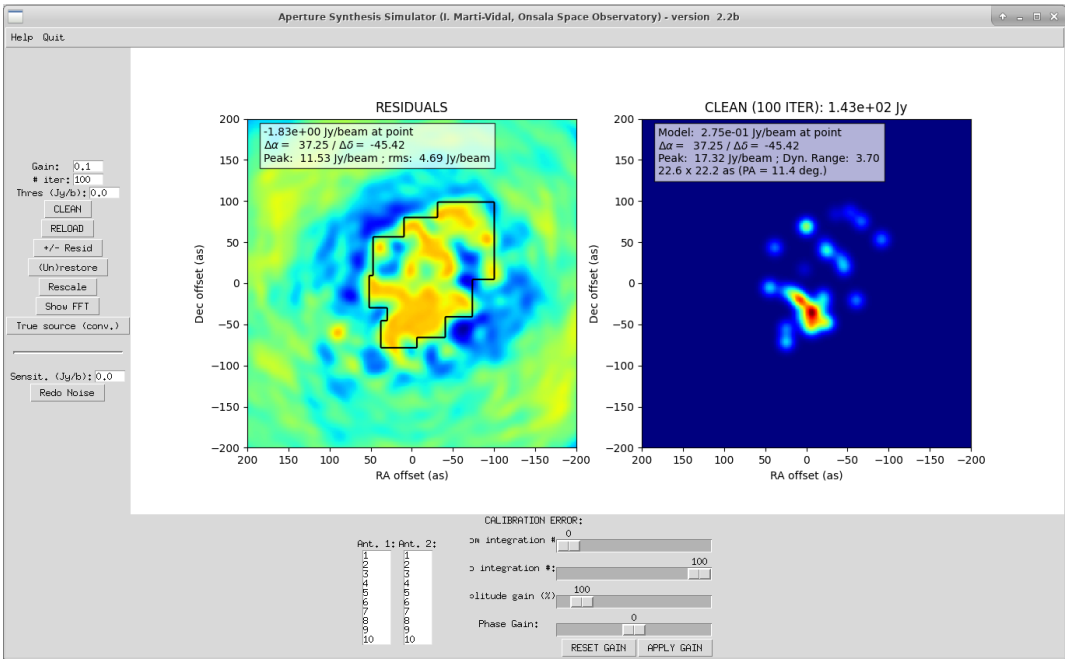

Figure 3: The CLEAN, noise and gain-corruption window.

- **CLEAN**. Will apply the required number of CLEAN iterations, updating the two figures (i.e., residuals and model) in real time.
- **RELOAD**. If anything is changed in the Main Window (e.g., antenna locations, observing wavelength, visibility weighting, etc.), the user must press this button to update the CLEAN GUI images accordingly.
- **+/- Resid**. Adds (or removes) the residuals from the CLEAN image (this may be useful to check the achieved dynamic range by CLEAN). This button only works if the CLEAN model is being restored with the CLEAN beam when plotting (i.e., button "(Un)restore").
- **(Un)restore**. By default, the CLEAN model is restored with the CLEAN beam when it is shown in the CLEAN plot. This button allows the user to plot the unrestored CLEAN model instead (e.g., just the point sources found by CLEAN). If the model is not restored, the residuals cannot be added to the model.
- **Rescale**. Rescales the color palette in the image of the residuals. This is useful to check better the structure of the residuals when the dynamic range is large.
- **Show FFT**. Will load another GUI window, showing the Fourier transforms of the PSF (i.e., basically, the visibility weighting used in the visibility gridding), the residuals (this can be useful to study the effect of corrupting gains in the deconvolution), and the convolved CLEAN model (useful to check the quality of the CLEAN interpolation in the UV plane). An example of this window is shown in Figure 4. Clicking on any point in the figures will show the UV coordinates and intensities in Fourier space (in Jy). These values will be printed at the upper-left corners in the plots.
- **True source (conv.)**. Will open a new window showing the source structure convolved with the CLEAN beam (i.e., the *MODEL IMAGE* in the Main Window convolved with the CLEAN beam). This is useful to check the fidelity of the CLEAN deconvolution.
- **Sensit**. The program can add random noise to each visibility (all baselines are assumed to have the same sensitivity) such that the final image (natural weighting assumed) will have an rms equal to the desired sensitivity (in Jy/beam). The noise will be added to the main window as well (so the user can check how the noisy image changes with antenna positions, hour-angle coverage, visibility weighting, etc.).

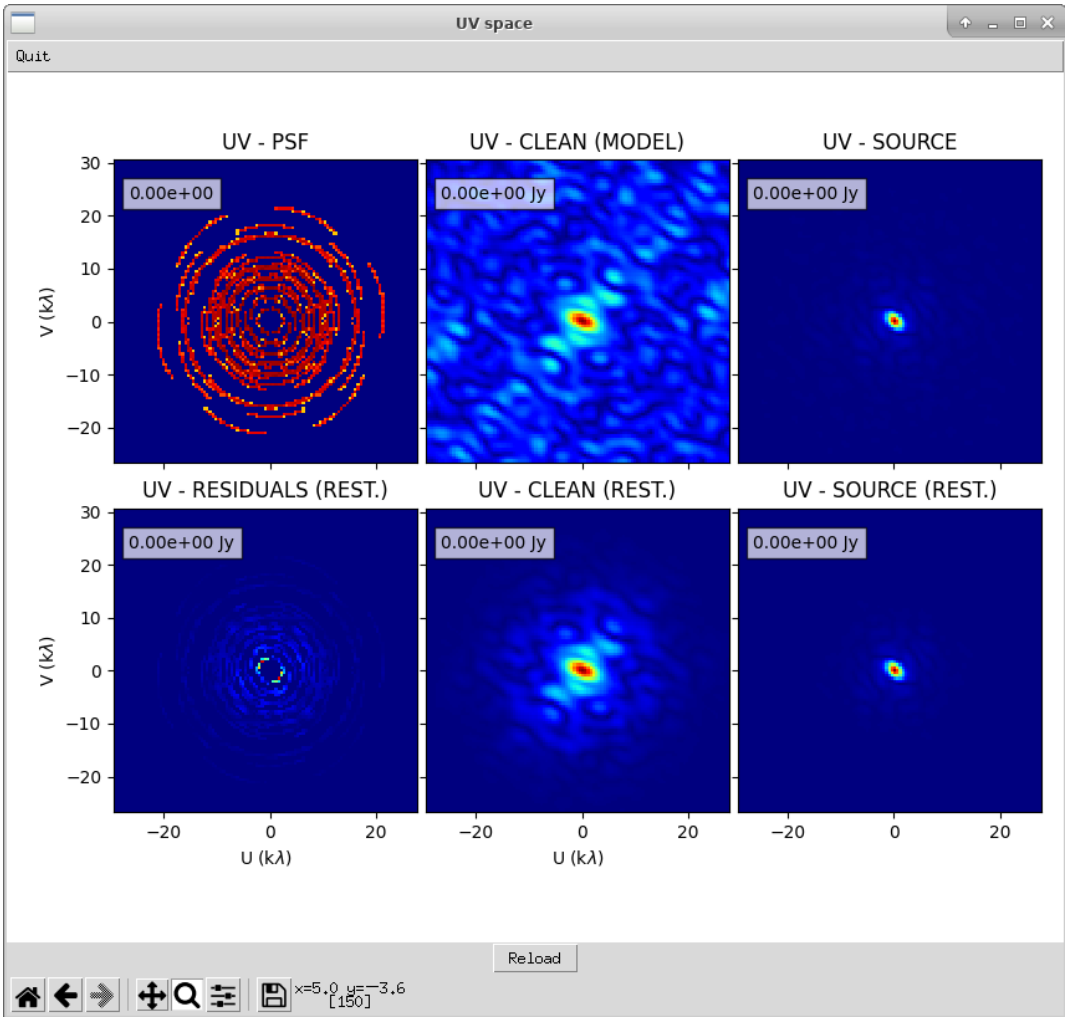

Figure 4: FTs of the PSF, CLEAN residuals, true source, and CLEAN model (it is shown by pressing the "Show FFT" button on the CLEAN window).

1.3.1 Adding gain corruptions

The controls to define the corrupting gains are located in the lower part of the CLEAN window (see Fig. 3). To apply a given gain to an antenna, just select the antenna index from the "Ant 1" list. If the selections of "Ant 1" and "Ant 2" are the same (or if nothing is selected in the "Ant 2" list), all the baselines related to the selected antenna will be modified by the corrupting gain. Otherwise (i.e., if the selected "Ant 1" and "Ant 2" are not the same) only the selected baseline will be modified by the corrupting gain.

The simulated observations are divided in integration times (100 integration times homogeneously distributed between the initial and final hour angles). The initial and final integrations to be affected by the corrupting gain are set with the sliders named "From inte gration" and "To integration", respectively. The amplitude gain (in %) and the phase gain (in degrees) are set using the sliders named "Amplitude gain" and "Phase gain", respectively.

Pressing the **APPLY GAIN** button will insert the corrupting gain into the data. Pressing the **RESET GAIN** button will undo the data corruption.

Notice that if a new array is loaded (and/or if any antenna is added or subtracted from the array) the corrupting gain is removed from the data. Notice also that only one corrupting gain is applied at a time.

2 EXERCISES WITH APSYNSIM

2.1 Diffraction Limit

1. Open the `Point-source.model` and the `Two-antennas Long.array`. Look at the shape of the PSF and the dirty image. Discuss the results.

2. Study the width of the PSF fringe as a function of wavelength and baseline length.

3. This is a snapshot observation. Now, let the Earth rotate. Increase the observing time (i.e., the hour-angle coverage), so that it runs from `-5h` to `5h`. What happens to the shape of the PSF?

4. How does the new PSF depend on observing frequency?

2.2 Image Deconvolution

1. Observe the `Five-Gauss.model` with the `Two-antennas_Long.array` at a wavelength of 10 mm and with an hour-angle coverage between, say, `-5h` and `5h`.

2. CLEAN only one of the source components (set a mask around it). Look at the CLEAN model and the residuals (also in UV space).

3. Now, add more CLEAN masks around the other sources and continue the CLEANing. Look at the level of the residuals (pressing **Rescale** will help see better the residuals).

4. Repeat the above exercise, but now CLEANing all sources simultaneously (reset the CLEANing by pressing the **RELOAD** button). Compare the residuals with those from the previous part of the exercise.

5. Look at how the CLEAN model compares to the actual FFT of the source, by pressing the **Show FFT** button. Comment on the results (remember that this interferometer only has *one* baseline!).

2.3 Basic Image Observables: Flux Density vs. Surface Brightness

1. Observe the `Point-and-Gauss-ALMA.model` with the `Two-antennas-Long.array` (with a time coverage between `-5h` to `5h`) at a wavelength of 10 cm.

2. Measure, from the dirty image, the peak intensities of the Gaussian and the point source.

3. CLEAN the image and measure the peak intensities on the final CLEAN model. Look at the peak amplitude of the CLEAN model in the UV plane. How does it compare to the total CLEANed flux? Comment on the results.

4. Now, decrease the wavelength to 6 cm and re-CLEAN. What happens to the peak intensities of the two sources (point and Gaussian) in the image plane? Why?

5. Change the wavelength to 2 cm and repeat the exercise.

6. Take another look at the UV-CLEAN model. Notice how the interpolation of the source FFT has been done!

2.4 Missing Flux and Dynamic-Range Limitation

1. Observe the `One-Disc.model` with the `E-W.array` at 50 mm, using an observing time from `-4h` to `4h`.

2. CLEAN the image. Compare the surface brightness of the CLEAN image to that of the model (convolved with the CLEAN beam). Compare the total recovered flux density to the total flux density of the disc.

3. Change the observing wavelength to 10 or 11 mm and CLEAN again. Compare the recovered flux density to that in the previous case and discuss the results (a look at the UV PLOT might help understand what is going on).

4. Compare also the dynamic range achieved in the CLEAN images at 50 mm and at the original wavelength.

5. Observe the `Nebula.model` with the `Golay_12.array` at 1.5 cm. CLEAN it and look at the residuals. Compare the CLEAN image with that of the source model convolved with the CLEAN beam. Discuss the results.

6. Increase the observing wavelength by, say, a factor 2-3 and re-CLEAN. Discuss the results (look at the residuals in UV space).

7. Observe the `Discs.model` with the `Long_Golay_ 12.array` at 2 cm. CLEAN it and look at the residuals and CLEAN model in UV space. How does it compare to the **UV PLANE** plot in the main program window? (hint: make a large zoom in by pressing capital "Z" in the **UV PLANE** plot). Discuss the results.

8. Increase the wavelength to the maximum and re-CLEAN. Discuss what you find.

2.5 Earth Rotation Synthesis

1. Load the `E-W.array` and observe the `Double-source.model` at 50 mm, with a time coverage from `-5h` to `5h`.

2. Change the latitude of the observatory and check how that affects the UV sampling of the interferometer. Comment on the results.

3. Change the declination of the source and check how that affects the UV sampling of the interferometer. Comment on the results.

4. Now move some of the antennas toward North (or South), in order to build a 2D array distribution. Study how this position changes affect the UV coverage when you change source declination and/or latitude of the observatory.

2.6 Visibility weighting, primary beam, and subarraying

1. Observe the `Discs.model` with the `Long_Golay_12.array` at 50 mm. Change the **Robustness** parameter and check how it affects the dirty beam and the dirty image. Comment on the results.

2. Now, observe the same source with the `WSRT.array`, with a time coverage from `-5h` to `5h` and at a wavelength of about 50 mm. Change the **Robustness** parameter and check how it affects the dirty beam and the dirty image. Compare with the previous case. Comment on the results.

3. Observe the `Nebula_small.model` with the `ALMA-ACA-Cycle1-Conf5.array`. Change the **Robustness** parameter and check how it affects the dirty beam and the dirty image.

4. Increase the weight of the ACA with respect of the main array (i.e., decrease W1/W2) and comment on the results.

5. Now shorten the observing wavelength and play with W1/W2 (e.g., 0.7 mm with log(W1/W2) = -3.0). The dish diameters of the antennas are 12 m (main array) and 7 m (ACA). Based on this information, comment on what you see in the dirty image.

2.7 Noise and Weighting

1. Observe the `Point-source2.model` with the `E-W.array` during a reasonable time (say,from `-3h` to `3h`).

2. Open the **CLEAN GUI** (i.e., press the **Reduce data** button) and add a noise of about 0.2 Jy/beam.

3. You will see that the noise has also been added to the dirty image in the main window. Now, go to the main window and change the Robust parameter, from uniform (i.e., -2) to natural (i.e., $+2$) and check how well you detect the point source, depending on the weighting.

2.8 Calibration Error Recognition

1. Observe the `Point-source.model` with the `E-W.array` at 10 mm, with a time coverage from `-5h` to `5h`. CLEAN the source and add the residuals to the CLEAN model.

2. Now, corrupt antenna #1 with an amplitude gain factor of, say, 7. Apply that gain and re-CLEAN.

3. Add the residuals to the CLEAN model. Take a look at the residuals in UV space. Comment on the results.

4. Now apply that corrupting gain to only a fraction of the experiment (e.g., from integration time #40 to #60). CLEAN again and look at the residuals in both image plane and UV space. Comment on the results.

5. Apply a phase-only gain of 90 degrees to antenna #1 and re-CLEAN. Look at the new residuals. Comment on the results.

6. Repeat all the previous exercises, but using now the `Five-Gauss.model` source. Comment on the results.

2.9 Array Design

1. What do you think are the main Figures of Merit (FoM) to design an interferometer with high fidelity imaging capabilities?

2. Study some special array distributions, like Golomb and Golay. Looking at the uv-coverage from snapshot observations may help you understand what is special in these distributions. Discuss on the pros and cons of these arrays, in terms of UV coverage and PSF shape.

3. Observe a source of your choice with one of the VLA configurations. Start with a snapshot and apply some Earth rotation synthesis afterwards. Discuss on the pros and cons of the VLA antenna distribution.

4. Observe a given source with the `ALMA-ACA-Cycle1-Conf5.array`. Use natural weighting (Robust 2) and look at the Fourier transform of the PSF. Comment on what you see. As an add-on, you can also compare with uniform weighting (Robust -2).

5. Design your own array. For instance, create spiral arms (to your taste) or random distributions and check the resulting PSFs.

3 QUICK ANSWERS TO THE EXERCISES

3.1 Diffraction Limit

The PSF for a snapshot with two antennas looks like a plane-wave on the sky (the PSF is the Fourier transform of the UV coverage). If we let the Earth rotate, the UV coverage becomes a portion of an ellipse, hence producing a more point-like PSF.

If we change the wavelength, the UV coverage will be scaled by a constant factor, but the shape of the UV tracks will remain the same. Hence, the PSF will just be scaled, keeping its shape. The PSF becomes narrower for shorter wavelengths.

3.2 Image Deconvolution

CLEAN is a non-linear algorithm. The result of a CLEAN deconvolution not only depends on the masks used in the CLEANing, but also on when are they introduced during the process.

If we only CLEAN one source, and then the other, the residuals will look worse than when both sources are CLEANed at the same time. The reason is simple: in the first case, there is a danger of "over-cleaning" a sidelobe from the other source, whereas in the second case, the sidelobes from both sources are being iteratively removed.

3.3 Basic Observables

At low frequencies, both the Gaussian and the point source have similar peak intensities. Our short baselines are able to recover most of the extended emission of the Gaussian.

As the observing frequency increases, all baselines become longer, in units of the observing wavelength, so we begin to over-resolve the extended Gaussian emission (i.e., the extended part of the Gaussian becomes invisible to the interferometer, because the baselines are not short enough). Thus, the peak intensity of the Gaussian decreases, while that of the point source remains always the same, independent of the observing frequency.

APSYNSIM shows the Fourier Transform (FT) of the CLEAN image (i.e., the CLEAN model convolved with the CLEAN beam). Hence, the intensity units of that image are Jy/beam. This shall not be confused with the units of an unconvolved image model, which are Jy/pixel.

The FT of the convolved image, computed at the origin of Fourier space, is the pixel sum of the CLEAN image, which is, of course, a value much larger than the flux density of the source. The actual source flux density would be the pixel sum divided by thnumber of pixels within a CLEAN beam (indeed, to be exact, this quantity happens to be the pixel sum of the CLEAN beam).

3.4 Missing Flux and Dynamic Range Limitation

The CLEAN image of the disc is not very smooth; the intensity varies across the disc. This is a well-known limitation of CLEAN, when dealing with smooth structures: *CLEAN tends to deconvolve a smooth brightness distribution as a clumpy image.*

The total CLEANed flux density is lower than its true value, since there are spatial scales in the disc that are not being sampled by our baselines. If we decrease the observing frequency, our short baselines will become sensitive to larger spatial scales, and we will thus recover more flux density from the source.

A similar thing happens with the Nebula model. There is a lot of extended emission that cannot be recovered by the interferometer, since our shortest baselines are not short enough to become sensitive to these large spatial scales.

The problem of missing extended flux can be readily seen in Fourier space, by plotting the FT of the image residuals. If the extended emission cannot be modeled well by CLEAN, we will see large residuals at the shortest baselines.

In the case of the "Discs.model", there is one disc with a huge size, which is being completely over-resolved by the interferometer. There is only one hint of it in the dirty image, close to the disc edges. This is normal, since the edges of the disc are sharp (i.e., they have signals at high spatial frequencies), being thus detectable by our baselines.

3.5 Earth Rotation Synthesis

In an E-W array, the UV coverage does not depend on the latitude of the interferometer. It only depends on the hour-angle coverage and the source declination. For declination 0 (i.e., source on the Celestial Equator), the UV coverage degenerates to a line, whereas with the source at the Poles it becomes a set of concentric circles.

If there are baselines in the N-S direction, the latitude of the interferometer affects the UV coverage. As the angular distance between the interferometer latitude and the source declination changes, the UV coverage gets stretched in different ways. For a source at the Celestial Equator, the UV coverage becomes a set of parallel horizontal lines, whereas for a source at the Poles, it becomes, again, a set of concentric rings.

3.6 Visibility Weighting, Primary Beam, and Subarraying

If our baseline configuration has a minimum redundancy (as it the case of the Golay array), then, after gridding the UV plane, there will not be many pixels with a number of visibilities much larger than in others. Basically, the use of different weighting schemes will not affect the PSF substantially.

However, if the baselines have a lot of redundancy (as it is the case of the WSRT), there will be pixels with many more visibilities than others, which will be weighted much less if natural weighting is being used. Hence, in these cases, the use of different weighting schemes will affect more the PSF.

ACA has very short baselines, compared to the main array. Hence, over-weighting the ACA will add a "wide" component into the PSF, which will end up dominating the whole PSF if the ACA weight gets too high.

The nebula is quite extended, related to the primary beam of the telescopes. Increasing the observing frequency makes the primary beam even narrower, so the smearing of the nebular emission off the pointing center becomes more clear. Sinece the ACA primary beam is slightly larger than that of the main array, this effect is less important for the ACA.

3.7 Calibration Error Recognition

An amplitude error introduces symmetric convolution-like effects in the image residuals. If the sources are compact, these residuals can be seen quite well most of the times. If the sources are very extended, they are more difficult to detect.

The FT of the image residuals is, by construction, equal to the gridded residual visibilities. Therefore, if one antenna has a wrong calibration, its baselines should "shine" in the FFT of the image residuals.

If the calibration error is phase-like, the convolution-like residuals tend to be anti-symmetric. This is usually a bit more tricky to see, especially if the baselines involved are mostly long.

If we are observing several compact sources, this kind of residuals will appear around each one of them, due to its convolution-like nature.

3.8 Array Design

If we want an interferometer with good image fidelity, it is necessary to have a dense UV coverage, with no large gaps between the UV tracks and a smooth density distribution of baseline lengths.

The use of arrays with minimum redundancy (like Golomb rulers or Golay arrays) maximizes the uv coverage, but the sampling is far from random-like, hence introducing relatively large sidelobes (a regular UV sampling introduces sidelobes in the PSF, as a kind of "resonance" effect in the Fourier inversion).

The VLA configuration has a very good angular and radial coverage of the UV plane, even in just one snapshot. However, the snapshot PSF has quite large radial-like sidelobes.

More randomized antenna arrays, with a Gaussian-like baseline distribution, produce quite good PSFs, even in a snapshot. When the number of antennas is large (e.g., the case of ALMA), the image fidelity with these distributions can be quite good.

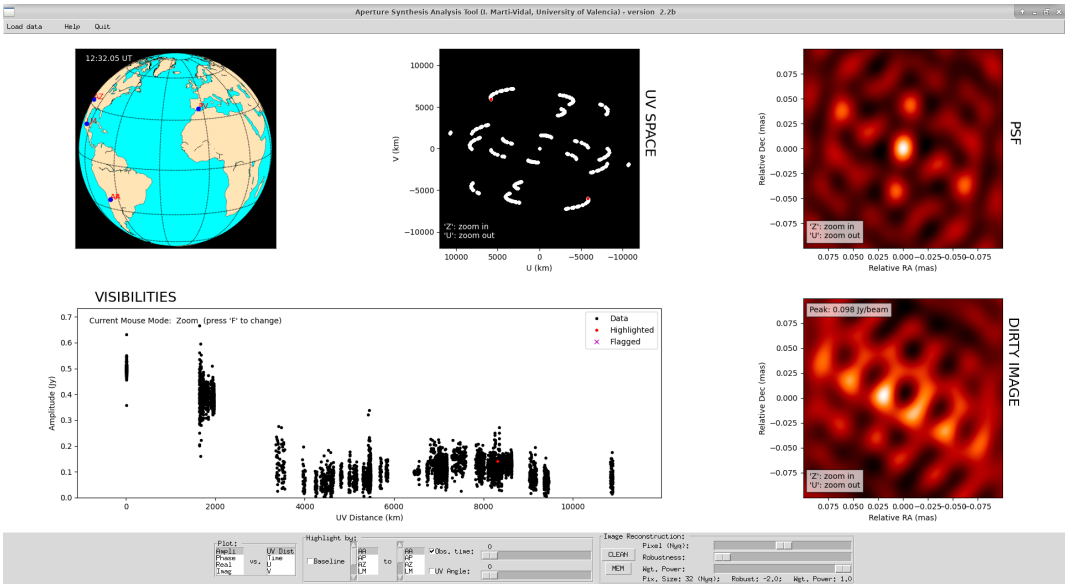

Figure 5: Main Window of *APSYNTRU*, showing the EHT data of M87*.

4 USAGE OF APSYNTRU

APSYNTRU is a simple interactive tool for the imaging of real interferometry data. It reads visibility files in "uvfits" format, which is the standard output of `AIPS` and `Difmap` (and it can also be produced by `CASA`, with the task *exportuvfits*).

This program is part of the distribution of *APSYNSIM*. You can find it in the subdirectory called "APSYNTRU". If you are able to run *APSYNSIM* with your Python3, then you should have no problem in running *APSYNTRU*.

4.1 Main Window. The GUI

Once you launch the program, it will load the default dataset (the Event Horizon Telescope observations of the black hole in M87*, taken on April 2017)[1]. A window will appear shoring some interesting metadata related to the uvfits file. Once the data are loaded and arrange, you will be able to click on the *OK* button to close the information window. Then, the focus will be set to the Main window (see Fig. 5), which is arranged as follows (from left to right and from top to bottom):

- The Earth, with the antenna distribution, as it is seen from the source. Press "z" to zoom in (in the direction of the mouse pointer) and "u" to zoom out (yes, I apologize for the different key combinations w.r.t. *APSYNSIM*).

- **UV SPACE**. All visibility locations are shown in white and the *selected* visibilities are shown in red. You can zoom in pressing "z" and unzoom by pressing "u".

- **PSF**. The Point Spread Function. As before, "z" and "u" can be used to zoom in and out, if the mouse pointer is located inside the plot.

- **VISIBILITIES**. A very useful interactive plot! Click and drag the mouse (but do not exit the area of the plot, or the dragging will be reset!), in order to either zoom in, flag (i.e., remove) or unflag the visibilities. You can change the mouse mode by pressing "f". If you want to reset the zoom of the plot, just replot the data (see below).

[1] EHT Collaboration et al. 2019, ApJL, 875, 1−6

Figure 6: Main controls of *APSYNTRU*.

- *DIRTY IMAGE*. The FT of the visibilities. It depends on the weighting scheme, the power of the uncertainties, and the pixel size. All these quantities can be set with the corresponding slide rulers (see below).

The bottom part of the Main Window (see Fig. 6) contains a set of GUI controls that can be used to select several plotting and imaging parameters. From left to right:

- **Plot**. Here, the user can select the quantities to be shown in the X and Y axes of the visibility plot. For instance, *Ampli* vd. *UV Dist* (the default plot) or *Phase* vs. *Time*, etc.

- **Highlight by**. The data are shown as black dots in the **VISIBILITIES** plot and white dots in the **UV SPACE** plot. However, the user can select subsets of the data and plot them as *red points* in both plots. The data can be selected in different ways:

  ✓ **Baseline**: Just select the pair of antennas of the baseline. If the second antenna is set to "ALL" (look at the bottom of the list), then all the visibilities related to that antenna (i.e., all baselines containing that one antenna) will be highlighted.

  ✓ **Obs. time**: The data are divided into observing times and/or scans. This slide ruler allows you to highlight all the data of a given observing time or scan. Actually, as you change the observing time, you will see the Earth plot updating to the selected time (i.e., the Earth will rotate, as it is seen from the source).

  ✓ **UV Angle**: Will highlight visibilities according to their position angle in Fourier space. This can be very pedagogical, when observing structures like core-jets, double sources, etc.

- **Image Reconstruction**: Here, you can set different parameters that will affect the image reconstruction from the visibilities, and you can also launch two different deconvolvers (CLEAN and MEM, see below). The slide rulers that allow you to fine-tune the imaging parameters are:

  ✓ **Pixel (Nyq)**: The size of the pixel, in units of the Nyquist Sampling Criterion. I'd suggest you to set it to relatively large values (e.g., 20−30), in order to have the main lobe of the PSF well sampled.

  ✓ **Robustness**: The Briggs Robustness parameter (the same as in *APSYNSIM*).

  ✓ **Wgt. Power**: When you have very heterogeneous arrays (like the EHT) it may be a good idea to not use the natural power of the visibility weights (which is 1), but lower it down, so that the extremely sensitive baseline (i.e., those to ALMA, in the case of the EHT) will not dominate the whole image, hence degrading the quality of the PSF.

Last but not least, you have two buttons, called **CLEAN** and **MEM**, which allow you to launch these two deconvolvers.

4.2 CLEAN Deconvolver

Disclaimer: This CLEAN implementation is **very** simplistic. It does not perform major cycles à la Cotton-Schwab, so the results should be taken as what they actually are: pedagogical products.

The CLEAN window of *APSYNTRU* is shown in Fig. 7. It is very similar to that of *APSYNSIM*, so we are not going to explain the details again. The only difference that is worth noticing is the button called **SAVE IMAGE**, which allows you to save the CLEAN convolved model as a FITS file.

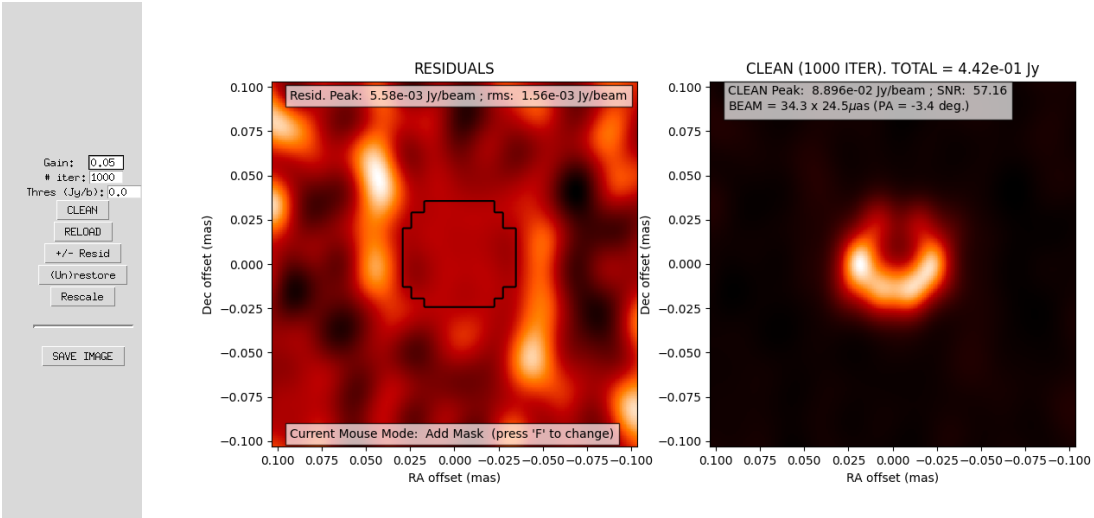

Figure 7: The CLEAN window of *APSYNTRU*.

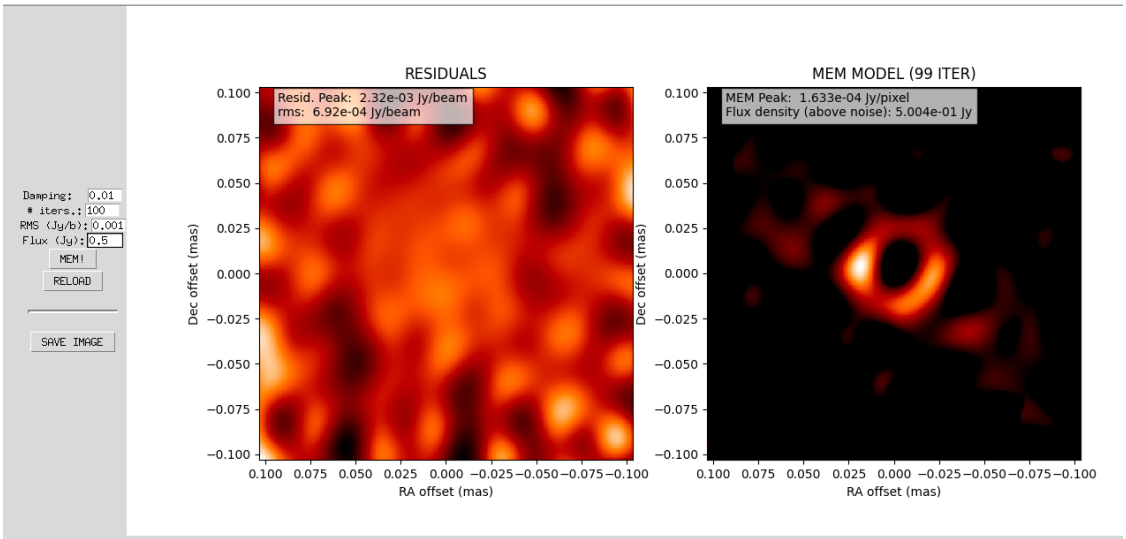

Figure 8: The MEM window of *APSYNTRU*.

4.3 MEM Deconvolver

Disclaimer: This MEM implementation is **very** simplistic. For instance, it determines χ^2 gradients from gridded visibilities, to accelerate the process. Hence, the results should be taken as what they actually are: pedagogical products.

The Maximum Entropy deconvolver[2] (shown in Fig. 8) needs an estimate of the total source flux density (you can guess it from the *Ampli* vs. *UV Dist* plot) and an estimate of the image noise (you can guess it, kind of, from the CLEAN residuals). Once you know these two numbers, you can run the deconvolver. Notice that no CLEAN windows are needed here!

[2]See, e.g., Nityananda & Narayan (1982), JAA, 3, 419.