Simulate your own interferometer(s)

Getting started:

- Input data: Latitude, longitude (or X,Y,Z) coordinates of the telescopes (either from your own imagination or based on some existing examples). Have fun with this, and also try out some crazy configurations if you like!

- Objectives: Determine the uv-distribution of the telescope baselines and the resulting shape of the synthesized beam (i.e. response to a point source) for a variety of different telescope configurations (also for both instantaneous snapshots and longer observations). Remember that the visibilities and source brightness are related by:

  $$I(l,m) = \iint V(u,v) e^{2\pi i(ul+vm)} du dv ,$$

  which is a 2D Fourier transform. So once we have the uv-distribution of the various baselines, we will do a Fourier transform to get the synthesized beam shape.
- Consult Lecture 6 slides for info on how to turn telescopes positions (X,Y,Z) in an Earth-based frame into baseline vectors (Bx,By,Bz). See Slide 63.
- The matrix equation on Slide 64 allows you to map the various (Bx,By,Bz) onto (u,v,w).
- To start, assume that the source is at Hour Angle H = 0 deg and declination delta = 90 deg.
- These assumptions will greatly simplify the matrix equation, such that: u = By (E-W), v = -Bx (N-S) and w = Bz (Up-Down).
- To start you'll get the instantaneous synthesized beam shape. As a next step, you can calculate the uv tracks over the course of several hours (use different Hour Angles as the sky rotates w.r.t. the telescope) and see how this changes the beam shape resulting from a longer synthesis observation (cf. Slide 65).

We use some approximations in order to obtain the so-called synthesized beam: we suppose a constant and uniform response of the antennas (and give them equal weighting) and a constant, uniform source illuminating all the antennas equally. We also implicitly assume some other approximations, e.g. all the antennas are at the same altitude, the source is monochromatic, etc.

Practical tips (read carefully):

- By default, we'll use python to create the necessary scripts and matplotlib to do the plotting. (unless you're an expert in some other computing language)

- Import the module called numpy in python to create 2D arrays [a matrix of (n,m) zero-elements can be created with the command numpy.zeros([n,m])].

- Import the module matplotlib.pyplot to plot the data. In particular, the command matplotlib.pyplot.imshow(matrix) will produce an image of the matrix, and matplotlib.pyplot.colorbar() will plot the legend (don't worry if the resulting image is rotated).

- Use the string matplotlib.pyplot.show() to show the image on the screen and matplotlib.pyplot.savefig('*name_of_img*.png') to save it on file; note that these commands need to be used for every single image you want to display or to save.
  Note that you can either display or save plots, you can't do both without re-plot.
  Always check that the saved plots aren't just a white image.

- You can write your answers and comments to a normal text file.

- You can put your answers, plots and results in a folder named "Results" inside the "Practicum1" directory

- *Please use self-explanatory names for the files*! (Ex. "Westerbork_beam_plot.png")

Part 0 - Set up the telescope positions and uv coords

1. Access your “playground” on CEP3 (see notes from first practicum session).

2. Create a “Practicum1” directory in your home directory.

3. Create a grid representing the positions of the antennas of the telescope on ground ((X,Y,Z) coordinates, see Slide 63).
   To start just consider a 2D grid, representing the telescope on a plane. This is a good approximation for all compact telescopes.

   Give to each pixel of the grid a value of 1 if an antenna is present, 0 otherwise. As a starting point, create an empty field with no antennas (each element of the grid is 0).

   Tip: A good technique is to set the coordinates of the antennas in two different arrays for each x and y coordinates. Arrays can be obtained as numpy.array([list of elements]).
   Then, you can select and set to one the pixels of the ground matrix corresponding to those coordinates. To select the element (i,j) of a numpy matrix just type matrix[i,j].

4. Calculate the instantaneous uv-coverage of the antenna baselines (see Slide 64). Remember that the number of baselines is N (N-1) / 2 (see Slide 9). Some of these will potentially be redundant (they will have the same orientation and length in number of wavelengths).

   a. You can represent the uv points in a second 2D grid (i.e. make a 2D array with these points). Most array grid points will be zero, except for where there is a u,v point, where you can use the value 1.
   b. Remember: a baseline is just the vector connecting two antennas, reported in the uv coordinate system and scaled for the wavelength number (which is a constant since we are considering a monochromatic source, and we can also assume that the wavelength is 1m).
      This means that you can obtain one u coordinate by subtracting the x-coordinate of two antennas and the v coordinate by subtracting the y-coordinate of same antennas; then you can repeat this for all the possible pair combinations of antennas.
   c. In the uv plane, all the possible baselines need to be represented (i.e. from tel_a to tel_b and vice-versa, this creates a symmetry).

5. Create a grid on the sky with the same pixels as the uv one. This second grid represents the synthesized beam. Its coordinates are the (l,m) values in the formula.

Calculate the value of the sky brightness for the pixels of the sky grid.

a. Tip: notice that the formula represents a Fourier transform, which can be easily calculated through the numpy function numpy.fft.fft(array) or numpy.fft.fft2(matrix). The absolute value of an imaginary number can be calculated with numpy.absolute(immaginary_value)
b. You may need to rotate the array to get the main beam in the center of the image. This can be done with the command *numpy.roll*(matrix,matrix.size/2)

Part I - The Westerbork Telescope

The first telescope we are going to reproduce is the Westerbork telescope, in the Netherlands. This is a simple example because it is a purely E-W array.

0. Create a “Westerbork” folder under “Practicum1” in your home dir.
   Create as many python files as you think are necessary to complete the Part 1 of the practicum (could be just one script).

1. Take a look on the ASTRON page dedicated to the telescope (www.astron.nl/radio-observatory/public/public-0)
   How many antennas form the telescope? What is their configuration?

2. Take a look at the real coordinates of the antennas at www.astron.nl/radio-observatory/astronomers/wsrt-guide-observations/3-telescope-parameters-and-array-configuration.
   First use the positions of the nearest 10 antennas. Plot the ground grid to verify that the antennas are well placed. Produce the synthesized beam with the algorithm written in Part 0 and plot it as well.

3. Do you think the instantaneous uv-coverage of the Westerbork telescope is adequate? How would you image a source with this coverage?

   The Westerbork telescope has produced many good-quality images (e.g. www.astron.nl/sites/astron.nl/files/cms/hydrogen.jpg).
   How can we improve the uv-coverage? Simulate the uv-coverage achievable by the telescope in a 12-h observation and plot the associated synthesized beam.

   Tips: Consult Slide 64 to see how to rotate the uv points as a function of HA. For a 12-hr synthesis the HA will go from -180 deg to +180 deg.
   If you consider a declination of 90 deg you can use the following code to rotate the matrix (change the commented lines). You can briefly explain why it is.

```
ang = np.pi/2  #The angle you want to rotate the system (90 deg in this case)
coord_a =  #coord_a and _b are the arrays containing the coordinates
coord_b =  #before the rotation
matrix =  #matrix is the square grid representing the coordinate system

#The resulting matrix is called rotated_matrix


 shift = matrix.shape[0]
coord_a -= shift/2
coord_b -= shift/2
new_dim = int(np.sqrt(2)*shift+3)
rotated_matrix = np.zeros([new_dim,new_dim])
steps = 1000

for i in range (0,steps):
   angle = ang*i/steps
   a = coord_a * np.cos(angle) - coord_b * np.sin(angle) + shift
   b = coord_a * np.sin(angle) + coord_b * np.cos(angle) + shift
   rotated_matrix[a.astype(int),b.astype(int)] = 1
plt.imshow(rotated_matrix)
```

4. Westerbork has 14 antennas in total. Now do the same simulation with all of them added in (add in telescopes A,B,C,D to the array).

   How does the uv-coverage and synthesized beam compare between the 10-telescope and 14-telescope array?

Part 2 - The Very Large Array (VLA) Telescope

Our second telescope is the Very Large Array (VLA), which is located in the USA.

0. Create a “VLA” folder under “Practicum1” in your home dir.
   Create as many python files as you think are necessary to complete the Part 2 of the practicum (could be just one script).

1. Read the main characteristics on the relative Wikipedia page.
   Why do you think this particular design (Y shape) has been chosen?
   In what way is this design better with respect to the in-line design of the Westerbork array?

2. In order to simulate the telescope, use 27 equally-spaced antennas divided in three equal arms. The arms are separated by an angle of 120 deg, and each of them is composed of 9 antennas.
   Simulate only the instantaneous uv-coverage and synthesized beam.

   Note: the real telescope is more complicated because the antennas are not equally spaced, but they are more dense in the center of the array. You could try to show why using a simulation... (not compulsory)

3. One of the peculiar characteristics of VLA is that its antennas are mobile, and can move closer or more distant from the center of the telescope along tracks. Why do you think so much money and efforts have been spent to enable this feature?

   In the A configuration, the dimension of the telescope is 35.5 times larger than in the D configuration. Simulate the A and D configurations.

   Tips: You can continue to use equally-spaced antennas.

   Note: The exact coordinates of the antennas in all the four configurations is reported in science.nrao.edu/facilities/vla/docs/manuals/oss2015A/ant_positions.ps (just for reference)

Part 3 - LOFAR

Our last telescope is the LOw-Frequency ARray (LOFAR) telescope. It has been built in the Netherlands during the last decade.

0. Create a “LOFAR” folder under “Practicum1” in your home dir.
   Create as many python files as you think are necessary to complete the Part 3 of the practicum (could be just one file).

1. New-generation, low-frequency arrays are composed of thousands of antennas, which give much more uniform uv-coverage.

2. Let's simulate one of the LOFAR low-band stations. It is composed by 96 antennas roughly randomly distributed in a circle with a higher density in the center. Simulate an instantaneous observation and compare it with the previous cases. What has changed and why?

   Tip: To create the coordinates of the antennas, you can use numpy.random.standard_normal(number_of_antennas).