ASTR4004/ASTR8004
Astronomical Computing
Assignment 4 (exam assignment)

Christoph Federrath, Mark Krumholz, Philip Taylor, Trevor Mendel

due Friday, 29 October 2021
(no extensions possible – exam assignment)

1 Project 1 – Fourier transforms and parallel computing (multi-threaded FFT)

Here you will make a python program that reads a column density map of a molecular cloud called 'The Brick' near the Galactic Centre (you can read more about this cloud in Federrath et al., 2016), apply mirroring and zero-padding to the image, compute the Fast Fourier Transform (FFT) with the `pyFFTW` library (https://pypi.python.org/pypi/pyFFTW), make a Fourier image and compute the power spectrum of the column density map.

1. Download the observational column density map from http://www.mso.anu.edu.au/~chfeder/teaching/astr_4004_8004/material/brick.fits. Use the astropy lib to read the data map in the fits file (http://docs.astropy.org/en/stable/io/fits/) into a numpy array.

2. Make a python function to produce an image of the map with a colour bar and write the image to a pdf file named 'brick.pdf'. See the left-hand panel of Figure 1 for an example thumbnail image of how this should look like.

3. Use the numpy functions `np.fliplr` and `np.flipud` to produce a mirrored array and image. Write the image to a pdf file called 'brick_mirrored.pdf' (see the middle panel of Figure 1 for a thumbnail).

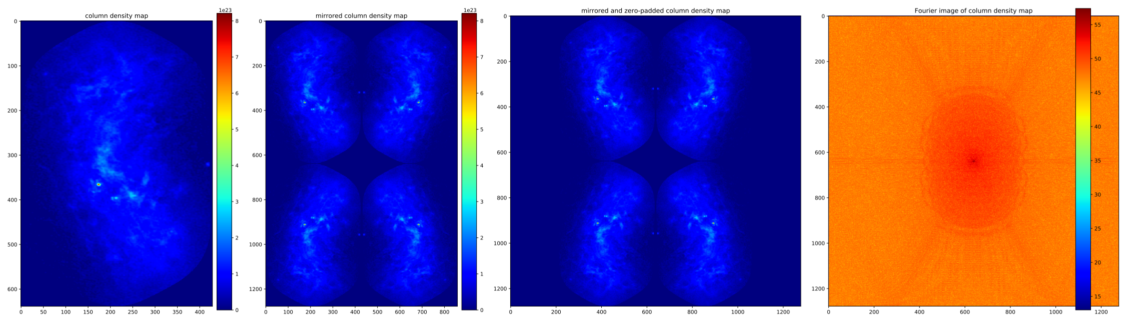

Figure 1: Left to right: original column density map, mirrored, zero-padded, and $\log_{10}$ Fourier image. Make sure to reproduce these not so small as in this assignment, but with readable font sizes; these are just meant as thumbnails to give you some idea of what the output of your script should look like.

4. Now use the numpy function `np.pad` to pad zeros symmetrically to the left and right of the image, such that the total dimensions become (1278, 1278). Make an image of this called 'brick_mirrored_zped.pdf' (see Fig. 1 for how this should look.)

5. Install `pyfftw`. Make a 2D threaded FFTW (use 1, 2 or 4 threads) of the mirrored-and-zero-padded column density map. Shift the $\mathbf{k} = (0, 0)$ position to the centre of the Fourier image and write out an image called 'brick_fourier_image.pdf'. The result of this should look like the last panel of Figure 1.

6. Compute the 1D power spectrum $P(k)$ of the mirrored and zero-padded column density, where $k = \sqrt{k_x^2 + k_y^2}$. Make a log-log plot of the power spectrum, $P(k)$, and write this out as an image called 'brick_power_spectrum.pdf'.

7. (Optional) scaling test: replicate the mirrored and zero-padded image $N\times$ in the x and y direction. Try $N = 10$ to produce a very big array with (12780, 12780) points. Test the multi-threaded FFTW with 1, 2, 4, and 8 threads for the parallelised FFT and produce a plot of speedup versus number of threads for the FFTW part of your script.

Put everything into a single Bash-shell-executable python script that runs the entire analysis with the input file (the column-density fits file) sitting in the same folder. The script should automatically produce the images with the requested filenames above (original column density image, mirrored, zero-padded, and Fourier image), as well as the final plot of the column-density power spectrum.

(10 points)

2 Project 2 – numerical solution of ordinary differential equations

Consider a simple harmonic oscillator: a spring with spring constant k is attached to a mass m, and the displacement of the mass from its rest position is x. The mass experiences a restoring force,

$$F = -kx. \tag{1}$$

At time $t = 0$, the mass is released at rest at the initial position $x(0)$, and is allowed to oscillate.

2.1 Part 1

Write a python function that takes as inputs the value of the spring constant k, the mass m, the initial displacement $x(0)$, the amount of time for which to integrate T, and the number of times N at which the position should be recorded, and returns the position and velocity of the mass at times 0, $T/(N-1)$, $2T/(N-1)$, $\ldots$, T. Verify that your code matches the analytic solution for $x(0) = 0.1\,\mathrm{m}$, $k = 50\,\mathrm{N/m}$, and $m = 1\,\mathrm{kg}$.

2.2 Part 2

Modify your routine to that it works for a nonlinear spring; one with a restoring force

$$F = -k_1 x - k_2 x^3. \tag{2}$$

Make a plot comparing the solution for a simple harmonic oscillator with the parameters given in Section 2.1 and the solution for a non-linear oscillator with the same values of $x(0)$, k_1, and m, and a nonlinear coefficient $k_2 = 10^3\,\mathrm{N/m^3}$.

(5 points)

3 Project 3 – Friends of Friends Program

In this question you will write a serial Friends of Friends program and use it to analyse data from a cosmological simulation. The outline of such a code is provided on Wattle (fof.c) along with a file containing the data (cosmo.txt). You should submit two files for this question: a PDF with your responses & figures for each part of the question; and the .c file containing your code.

1. Examine cosmo.txt. You'll see that the first line contains an integer (number of particles) and a float (simulation box size), and subsequent lines contain data for each particle of the form [x, y, z, mass, type]. The positions have units of kpc, and masses are in solar masses. Write a function that reads the data from cosmo.txt and populates an array of structures in C. What is the total number of particles?

2. The 'type' of each particle tells you if it is a gas (type=0), dark matter (1), star (4), or black hole (5) particle. Define a linking length of 5 kpc and compare the separation of all pairs of dark matter particles to this value; add any pair closer than this distance to the same group. How many particles of each type are there?

3. For each baryonic particle (i.e., the gas, stars, and black holes), find the closest dark matter particle to it, and add it to that dark matter particle's group. How many groups (of any size) have you found?

4. Find the total mass as well as the gas, dark matter, stellar, and black hole masses of each group. For groups with a stellar mass greater than $10^9\,\mathrm{M_\odot}$, write these quantities to a file. How many of these massive groups are there?

5. Compile and run your program.

6. In a programming language of your choice (python, IDL, etc.) read in the group mass data from the file you just produced. Plot histograms of $\log M_{\mathrm{DM}}$ and $\log M_*$, and scatter plots of $\log M_{\mathrm{gas}}$ and $\log M_{\mathrm{BH}}$ against $\log M_*$. Write a few sentences interpreting these plots physically.

(10 points)

4 Project 4 - model selection

Here you will explore fitting multiple functional forms to observational data of the blackhole mass, stellar velocity dispersion, and bulge mass.

1. Download the MBHdata.cat file from Wattle and examine its contents. A description of the individual columns is provided at the start of the file. Use python to read in the file and store information on the black hole mass, black hole mass uncertainty, and stellar velocity dispersion into three separate numpy arrays.

2. Use the `scipy.optimize.minimize` routine to fit a simple linear model to these data of the form

$$y = a_1 x + a_2 \tag{3}$$

   where $y = \log$(black hole mass) and $x = \log$(stellar velocity dispersion). Note that the quantities in MBHdata.txt are already in log! Your fit should incorporate the uncertainties on black hole mass.

3. Re-run your fit, but this time using a higher-order model of the form

$$y = a_1 + a_2 x + a_3 x^2 + a_4 x^4. \tag{4}$$

   Make a plot showing the input data, with the best-fit curves from Equations 3 and 4 overlayed.

4. Use leave-one-out cross validation to report the average mean-squared error for each model. Based on this, which model should you adopt? Note that you should write your own code to run the cross validation, rather than using the methods included as part of, for example, `scikit-learn`.

(5 points)

(Total 30 points)

Please submit your solutions via Turnitin by the assignment deadline. To upload the files for each section, please name them to indicte the section number, and make a tarball named <Uni-ID>.tar.gz, containing all submission files. **Please note that this is the exam assignment, so absolutely no extensions are possible!**

References

Federrath, C., Rathborne, J. M., Longmore, S. N., et al. 2016, *Astrophys. J.*, 832, 143