Statistical Methods 2024: Assignment 1 – Exploratory analysis of Gaia cluster candidate stars

This is the first of three assignments, for which extensive help is available during the tutorials. It is worth 15% of the final course grade.

The deadline for this assignment is Monday 15 Jan at 12:00 noon. Late work which is submitted up to 24 hours after the deadline will receive a penalty of -20% of the awarded grade, while work submitted 24-48 hours after the deadline will receive a penalty of -40% of the awarded grade. Work submitted later than this will not be assessed. The rubric for the assessment of all the assignments, listing the categories assessed and the requirements for each of them, will be provided separately on Canvas.

What you should submit
You should submit your work via Canvas. ***It must be in the form of a Jupyter notebook.*** Make sure that you upload the correct file, and check that all the cells run successfully (***and in the correct order***, from start to finish) before you submit!

Before you start it is essential that you read through the assignment grading explanation document on Canvas, since this explains what we expect from you in your answers. When answering each question, use markdown cells for explanations, assumptions and comments on your results: do not include these as comments in code cells, which are reserved only for comments about the code itself!
Remember that the usual plagiarism rules apply to your work: if you cut-and-paste code from somewhere/someone else (including code generated by an AI) you must cite the source (simply replacing variable names is not sufficient to make it your own!). See the grading explanation for more details. We also expect you to help each other, at least early on, and/or be inspired by methods you see online, so programming ***your own version*** (i.e. not cut and pasted) of someone else's method is fine and does not require citation.

The Assignment

For this assignment you will be using a dataset of astrometric and photometric data for nearly 1.3 million stars observed by the Gaia mission, which have been identified as belonging to more than 7000 star clusters and looser associations in our galaxy. Gaia, launched in 2013, is ESA's successor to the Hipparcos astrometry satellite, data for which is used in the 'Extras' episode to introduce Pandas, which will also be useful for this assignment. Gaia continuously scans the sky using two telescopes set at a precisely known (via an on-board laser interferometer) angle to each other. The relative positions of many stars, focused on to two CCDs allows a precise astrometric solution that can measure the angles between the stars down to a precision of tens of microarcseconds – note that one microarcsec is $\approx 5 \times 10^{-12}$ radian! Besides measuring extremely accurate positions on the sky, Gaia provides measurement of the star's proper motion – its movement on the sky – and parallax – the annual angular motion against distant background objects as the Earth (and satellite) moves in its orbit. The parallax can be used to directly estimate the distance to the star.

The data you will be using is taken from a recent paper by Emily L. Hunt & Sabine Reffert[1] which identifies and analyses star clusters found in the data from Gaia's Data Release 3 (DR3), a database of 1.46 billion sources (c.f. 2.5 million for Hipparcos!) which includes 5D astrometric data (RA and Dec positions, proper motion on the sky and parallaxes to measure the distances), as well as photometry in 3 optical bands[2]. Hunt & Reffert use a sophisticated clustering analysis of the astrometric information to identify which objects are associated with one another, e.g. in globular and open clusters or looser associations of stars.

[1] https://ui.adsabs.harvard.edu/abs/2023A%26A...673A.114H/abstract - it is not necessary for you to read this paper in order to complete the assignment!
[2] https://www.cosmos.esa.int/web/gaia/iow_20180316

Initial setup

The data for each star which is a candidate cluster member is contained in the FITS file `gaiadr3_cluster_stars.fits`. Besides the descriptions in the FITS header, the data columns are described in the additional file `data_description.txt`.

To load in the FITS file you need to have installed the astropy package. If you do not have it yet, you can install it using conda or pip depending on whether you use Anaconda python distribution or not (see here). Then in your first cell add:

```
from astropy.io import fits
```

and to load in the data, you can use e.g.:

```
dr3stars = fits.open('gaiadr3_cluster_stars.fits')
dr3stars.info()
print(dr3stars[1].columns)

stars = pd.DataFrame(dr3stars[1].data)
stars['Name'] = stars['Name'].str.strip()
```

which will open the FITS file, provide information about the file structure and columns in the data table, and then (if you use Pandas, which we recommend) assigns the table to the Pandas dataframe `stars` (you can choose your own dataframe name if you wish). The last line of code strips the trailing white spaces from the original cluster names given in the `Name` column (which are 20 characters long including trailing white spaces). This fix makes it easier to obtain data for a given cluster name, without having to pad the string with white spaces to obtain a match.

This should be all you need to start working with the data using Pandas, scipy, numpy etc., e.g. using the commands described in the Extras episode on Working with and plotting large multivariate data sets. However, for more discussion on working with FITS files in Python, see this Episode from the Programming for A&A course.

For the following tasks it will be useful to have information on the number of stars associated with a given cluster name in a pandas dataframe, and information on the extent of the cluster in the RA and Dec directions. You can obtain this information quickly using the following code:

```
clcounts = stars.groupby(['Name']).size().reset_index(name='count')
```

which creates a dataframe `clcounts`, where each column shows the number of counts which is identical to the number of stars associated with that cluster name in the stars dataframe. You can use this information to find clusters with a certain minimum number of stars.

Assignment tasks

Each task contributes an equal weight to the assignment total grade:

1. The parameter `Prob` gives a conservative estimate of the probability that the star is associated with the cluster, by doing a 'clustering'[3] analysis of the stars in the 5-dimensional astrometric parameter space, i.e. using `RAdeg`, `DEdeg`, `Plx`, `pmRA` and `pmDE`. Use the Pandas `sample` function on your cluster star counts dataframe, to randomly select 4 clusters, only from clusters with >1000 candidate stars. Split each cluster into two subsamples corresponding to stars with `Prob` ≤ 0.8 and stars with `Prob` > 0.8 and for each cluster, make a scatter plot matrix (see the extras episode) to show **on the same figure** the data points for both subsamples on this 5-D parameter space. Note that it is fine to

[3] Of data points in the parameter space, not stars!

make a new sample of the clusters you want to plot if you wish, **but be sure to write in your notebook the names of the clusters you plotted and commented on.**
In your brief comment on your results, explain: what is the likely reason for the differences between the distributions of points shown by each subsample?

2. For the remaining analysis in this assignment we will only limit ourselves to stars with `Prob` > 0.8 and stars that are more tightly clustered together, to rule out looser associations which will have a wide spatial variance in their parameters due to the range of distances from us. To select such clusters you can use the following lines of code to first create a new stars dataframe containing only the higher-probability stars, then create a new clusters dataframe with the star counts and standard deviations in RA and Dec for each cluster:

```
stars_hiprob = stars[stars.Prob > 0.8]
clusters_hiprob = stars_hiprob.groupby(['Name']).size().reset_index(name='count')
clusters_sd_hiprob = stars_hiprob.groupby(['Name']).std(numeric_only=True).reset_index()
clusters_hiprob['sd_RAdeg'] = clusters_sd_hiprob['RAdeg']
clusters_hiprob['sd_DEdeg'] = clusters_sd_hiprob['DEdeg']
```

   Now use the new `clusters_hiprob` dataframe to make a new cluster sample containing only clusters with > 200 stars and standard deviations of RA and Dec < 0.1° (to constrain the cluster size). **You should work with these clusters (and `Prob` > 0.8 stars) for the remainder of the assignment.**

   An interesting question is whether there is any spatial (RA and Dec) variation of the other astrometric and photometric parameters in each cluster Now:
   a. Select a cluster from your sample (your choice but you can randomly select from the clusters if you wish) and split it into 2 subsamples in RA, corresponding to stars with RA: i) greater than the mean RA, ii) less than the mean RA. Do the same for Dec, to create 2 subsamples selected on Dec. Then for the RA-selected subsamples, plot a figure with 5 separate subplots (e.g. side-by-side) which show the histograms of the following parameters for each of the 2 subsamples: `Plx`, `pmRA`, `pmDE`, `Gmag` and `BP-RP`. I.e. each subplot will show two histograms, one for each subsample, so you can compare the distributions for stars on one side of the cluster vs the other. Repeat this for the 2 subsamples selected on Dec.
   b. Use t-tests to compare the 2 subsamples in RA and then the 2 subsamples in Dec for the following parameters: `Plx`, `pmRA`, `pmDE`, `Gmag` and `BP-RP`. For the t-test you can assume populations with the same variance.
      I.e. you will do 5 t-tests for the subsamples selected on RA and 5 for the subsamples selected on Dec, to see if there is any evidence that the populations of stars which each subsample is drawn from is different from the other subsample, i.e. does it change with position in the cluster?
   c. Comment on the implications of your test results, and repeat the same procedure for two other clusters. Based on the parameter distributions, is the t-test an appropriate test in all cases?

3. Now you will analyse all the clusters in your new sample (i.e. star counts > 200, constrained in RA and Dec standard deviation). For all these clusters, repeat the 10 t-tests carried out in task 2a and 2b. For each parameter and coordinate (RA or Dec) being tested, plot the resulting p-values in a histogram (so you have 10 histograms in total, each calculated from n p-values where n is the number of clusters analysed).

4. In a large sample of objects and different statistical tests, we always expect some outliers that appear significant in a single test, even if the null hypothesis is true. But if the null hypothesis for a given statistical test is actually true, the p-values from the test should be drawn from a uniform distribution between 0 and 1. The K-S (Kolmogorov-Smirnov) test can be used to compare two distributions to see if their CDFs are consistent. Look up this test and its scipy implementation and apply it to each of the 10 histograms obtained in Task 3 and comment on your results. Now consider what happens if you repeat the K-S-tests but only include p-values > 0.01 in the test. Does this make a difference to which parameters show a systematic dependence on position in the cluster? What does this tell you about whether an effect is due to a few outliers or is present in the whole population?