Problem set 2
The Exoplanet 51 Peg b

Anthony Brown

2024

1 Introduction

Humans have long looked into space and wondered whether there is life elsewhere but for most of that time we did not even know there were planets outside our solar system. In the last 30 years, this has changed dramatically with the discovery of a planet orbiting a main sequence star (why this qualification?) in 1995[1].

The most direct way to observe these planets, direct imaging, is very challenging since many of these planets are very close to their parent stars and many orders of magnitude fainter than them. Instead, the main progress in observing extra-solar planets (exo-planets in the following) was through indirect techniques.

For a number of years, the main technique was the radial velocity detection technique — where the reflex motion of a star in response to the gravitational attraction from an orbiting planet can be observed. This is also the way the first 'normal' exo-planet was detected — 51 Pegasus b, normally known as 51 Peg b (51 Peg a is the star). Although we cannot see this planet directly, we can measure the motion of the star around the planet-star barycenter and hence the radial velocity v_{rad}.

From Newton's law of gravity, we can then derive that the variation in radial velocity, v_{rad}, of the star caused by a planet in a circular orbit with period P, and inclination i, has an amplitude k given by

$$K = \left(\frac{2\pi G}{P}\right)^{1/3} \frac{M_{\mathrm{P}} \sin i}{M_\star^{2/3}}, \tag{1}$$

where M_{P} is the mass of the planet, $M_\star$ the mass of the star, and G is as usual the gravitational constant.

Problem 1

Derive equation 1. *Hint: Make use of Kepler's third law.* Write equation 1 with P in units of days, $M_\star$ in units of solar masses ($\mathrm{M}_\odot$), and M_{P} in units of Jupiter masses (M_{J}). (NB: $M_{\mathrm{P}} \ll M_\star$). What is the amplitude of the variation in v_{rad} of the sun caused by the Earth? And by Jupiter?

[1]Problem Set 2 based on previous versions by Paul van der Werf, Henk Hoekstra, Jarle Brinchmann, and Reinout van Weeren.

2 Measuring the minimal mass of 51 Peg b

The first exoplanet around 51 Peg was discovered in 1995 through the radial velocity method. In this problem set, we will carry out this analysis and measure the period of the v_{rad} variations and the amplitude of the variations in order to understand the properties of the system. The radial velocity data are provided on the course website.

The file provides the original dataset from 1995 which was used to find the exoplanet. The file consists of three columns of data. The first column gives the Julian date of the observations relative to some reference date. The second column gives v_{rad} measured in meters per second and the third column gives the estimated uncertainty on v_{rad}.

In this problem set we will calculate the period and amplitude using Python. To do this we will adopt the following model for v_{rad}:

$$v_{\mathrm{rad}} = K \sin\left(\frac{2\pi(t - t_0)}{P}\right) + v_0 = K \sin\left(2\pi(f + f_0)\right) + v_0, \tag{2}$$

where we want to estimate K and P because they are required to estimate $M_{\mathrm{P}} \sin i$ (c.f. equation 1).

We will fit this model by minimising χ^2, where χ^2 is given by

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{v_{\mathrm{rad},i} - \mathrm{model}_i}{\sigma_{v_{\mathrm{rad},i}}}\right)^2. \tag{3}$$

In this equation N is the number of measurements, $v_{\mathrm{rad},i}$ is the v_{rad} measurement and $\sigma_{v_{\mathrm{rad},i}} = \sigma_i$ gives the uncertainty estimate for $v_{\mathrm{rad},i}$.

A χ^2 fit is quite simply a *weighted* least-squares fit. If the uncertainty on each data point were the same, the χ^2 expression reduces to a standard least-squares fit. In general the lower that χ^2 is the better the fit. The most common approach, and the one we will take, is to take the model with the lowest χ^2 as providing the best estimate of the period, P. To get this estimate we also need to estimate the other parameters of the model, K, v_0, and f_0. Think carefully about how you can get a good initial estimate of v_0.

Problem 2

Read the file into Python and plot v_{rad} against time. Give an approximate estimate of the period and amplitude by eye.

Use the χ^2 formulation to estimate P and K. Let the period vary over ± 1 day from the period you estimated by eye, use steps of 0.01 day. Make sure you translate these values into phase, f. Vary now for each P also K by $\pm 10\,\mathrm{m\,s^{-1}}$ from the value estimated by eye in steps of $1\,\mathrm{m\,s^{-1}}$. Let f_0 vary between 0 and 1 in steps 0.01 and vary v_0 using your own estimate of range and steps. For each P calculate the lowest χ^2 and put this in an array.

Important: Later you will need a function to calculate χ^2. To save yourself grief later, write this already now as a function that takes K, P, f_0, and v_0 as input and returns χ^2.

Plot χ^2 as a function of P and indicate the minimum value of χ^2. Which period gives the best fit? Improve this fit further by using smaller steps around the best-fit values. Use this to derive the best fitting values of P, K, and f_0. Show this solution in a diagram of v_{rad} against phase, f, with the best-fit indicated.

What is the minimal mass of the planet? *Hint: You need to find the mass of the main star yourself.*

3 Is the fit statistically acceptable?

In the preceding problem, we fitted the radial velocity variation of the star caused by a planet. From this fit, we derived an amplitude and a period and together these give us an estimate of the minimal mass of the planet. But how much should you trust this calculation?

This is a very common question in astronomy and a key aspect to realize is that we can never be completely sure that we have the right model. There are many possible reasons for why the model perhaps is not the correct one in this case:

- The planet might move on an elliptical orbit.
- Perhaps are there multiple planets in the planetary system.
- Perhaps there are systematic errors in the observations.
- ...

A similar situation can occur in many investigations and one of the ways to say more about this is to check whether the fit of the model to the data is a ‘good’ fit. To be more precise, this means that we will check whether the deviations between the model and the data are within what is expected from a statistical analysis.

3.1 Statistical tests

Imagine that you have a set of measurements, $y_{\mathrm{obs},i}$, with uncertainty estimates σ_i, and that we want to test whether a model $y_{\mathrm{mod},i}$ is a good fit to the measurements. To do this there are a series of approaches in the literature — here will use a fairly simple approach for which some cautionary remarks are needed and we will touch on these later.

The approach we take is known as hypothesis testing. We can get a feeling of how this works by simply looking at possible outcomes. We will imagine that there are only two possibilities for our fitting:

1. model gives a good fit to the data: model is accepted = 'positive'
2. model gives a bad fit to the data: model is rejected = 'negative'

In other words: Either the model gives a good fit to the data, or it does not and we want to distinguish between these two options.

To check whether this is the case we now need to use a statistical test. There are no tests that always give the correct answers, so again we need to consider various possible outcomes. For either of the two answers the test can either give a true or false outcome so we have in total four possible outcomes:

1. The model is good + the test accepts the model = 'true positive'
2. The model is good + the test rejects the model = 'false negative'
3. The model is wrong + the test accepts the model = 'false positive'
4. The model is wrong + the test rejects the model = 'true negative'

So clearly the cases 2 and 3 are problematic. In the statistics literature, it is common to refer to case 2 as a Type II error and case 3 as a Type I error. In any statistical test, there is a balance between these two types of error and we must choose which one we want to minimize. In some situations this is clear — imagine devising a test for whether someone has a serious disease — and we typically try to minimize the case 2, the Type II errors. A typical value is that we only want a 5% chance to have a false negative.

This kind of significance test can be very helpful but also has a potential for being misused, and indeed it frequently is. The two most important points to keep in mind are:

- If the test finds that the model is a good fit, we say that *the model provides a good description of the data*. It does **not** say and should not be taken to mean that the model is *true*. That is a very different and much harder question to answer.
- When the test rejects the model, we usually say that the model is rejected with 95% significance, or with a 95% confidence limit. That 95% comes from 100%-5%, where the 5% is the chance that the model was incorrectly rejected.

3.2 The χ^2 test

Above we used the χ^2 as a way to find the best-fitting model for the data. However, the χ^2 has another important function as well and we use it to test for the quality of a fit/model. If we go back to the equation for χ^2, we can write this as

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{y_{\mathrm{obs},i} - y_{\mathrm{mod},i}}{\sigma_i} \right)^2 , \tag{4}$$

where $y_{\mathrm{obs},i}$ are the observables with associated uncertainty estimate σ_i and $y_{\mathrm{mod},i}$ are the model estimates. Now in this equation, the observables are random variables due to the noise and if the uncertainty estimates, σ_i are known, then χ^2 is a random number that is distributed according to the χ^2 distribution. This analytic form for this distribution is a Gamma distribution

$$f_{\chi^2}(\nu) = \frac{1}{2^{n/2}\Gamma(n/2)} \nu^{(n/2)-1} e^{-\nu/2}, \tag{5}$$

where n is the number of degrees of freedom of the distribution, a number we'll get back to below. We do of course need to adjust for summing over more data when calculating χ^2. If you want to read more about the χ^2 test, then chapter 15 in *Numerical recipes* or section 7.4 in 'Modern Statistical Methods for Astronomy' by Feigelson & Babu, are good places to go. The Wikipedia article on Pearson's chi-square test is also okay.

We use the χ^2 test to assess whether the fit of the data to the model is acceptable or not. The idea is that if χ^2 is very large, then the fit of models to the data is not very good, while a very small value might also indicate a possible problem. What then is the approach? It turns out that there is a particular value for χ^2, χ^2_{lim}, above which the fit is sufficiently bad that we should reject our model. Thus the approach becomes:

- Hypothesis: The model provides a good fit to the data.
- If $\chi^2 < \chi^2_{\text{lim}}$: Accept the hypothesis, or better: you cannot reject the hypothesis.
- If $\chi^2 > \chi^2_{\text{lim}}$: Reject the hypothesis.

What then is χ^2_{lim}? In our example, it is the value so that the chance of finding a χ^2 value larger than χ^2_{lim}, is less than 5%. Mathematically we can express this as:

$$P\left(\chi^2 > \chi^2_{\text{lim}}\right) = 0.05. \tag{6}$$

To calculate this number, we need to know the number of degrees of freedom, n in equation 5. Since we will assume that our measurements are Gaussian random variables and that each measurement is independent, we have a total of N measurements. However we also need the χ^2 value that we calculated, and that depends on the measurements, this therefore reduces the number of free parameters. You should therefore not use N as the number of degrees of freedom, but rather $N - m$ where m is the number of free parameters in your model (the number of fit parameters).

In Python, you can find various routines related to χ^2 in the `scipy.stats` package. The likelihood distribution of χ^2 from equation 4 above, is provided by the `chi2.pdf` function and the cumulative distribution is given by `chi2.cdf`.

Problem 3

Adapt the χ^2-test for the model you have fitted to the observations. Give the limit χ^2_{lim} for a chance of false negative of 5%, and indicate the value of n, the number of degrees of freedom, and how you decided on this value. What are your conclusions?

4 Uncertainties in the results

In the preceding, we tested how good the fit was that we obtained for the radial velocity data of 51 Peg. What is still missing is an estimate of the uncertainty on the estimated parameters.

There are several ways to obtain these uncertainties. We can do this from the χ^2 values, which is a classical approach, but we can also use a more general and powerful approach, known as Markov Chain Monte Carlo (MCMC) and we will use this now in the following.

MCMC models are used in many parts of science (and beyond) today. There is a lot of literature and sophisticated theory behind them but for our needs here we can view an MCMC model as a machine that efficiently can explore multi-dimensional parameter space and calculate the likelihood of each corresponding model given the observed data. To repeat the notation used in the lectures, I will call the model we consider $M_{\boldsymbol{\theta}}(x)$ — in the present problem, $\boldsymbol{\theta}$ is a four-element vector $\boldsymbol{\theta} = (P, K, f_0, v_0)$, and x is the time variable.

We will use a Python implementation of MCMC called `emcee`. This is an easy-to-use package that works well in many practical situations although it is not a solution for every problem you may encounter. `emcee` should be installed at the STRW, if you want to install it on your own laptop you can install it most easily with `pip` or `easy_install`. For instance:

```
> pip install emcee
```

You can also download it directly from `https://emcee.readthedocs.io/en/stable/`, where you can also find the manual and pointers to documentation on MCMC and the paper describing `emcee`. See also the `jupyter` notebook on Brightspace which introduces MCMC for the simple problem of fitting a straight line (it is overkill to use MCMC for this particular problem — although more complex line-fits might require MCMC).

To use MCMC we need to consider Bayes' formula

$$P\left(\text{model} \mid \text{data}\right) \propto P\left(\text{data} \mid \text{model}\right) P\left(\text{model}\right), \tag{7}$$

which allows you to calculate the likelihood of each model, which can be written in words as

$$\text{Posterior probability} \propto \text{Likelihood} \times \text{Prior}. \tag{8}$$

In your code, you need to provide the likelihood and the prior, in fact, the normal approach is to give the natural log of these, so the log-likelihood and the log prior. This is all detailed in the `jupyter` notebook. In particular, you have that the log-likelihood, $\ln L = -\chi^2/2$, where χ^2 is given by equation 3. The prior is more complex — you can make a simple box prior as used in the notebook, or you can give a Gaussian prior on each parameter so that the prior is

$$\text{Prior} = \prod_i e^{-\left(\theta_i - \hat{\theta}_i\right)^2/2\sigma_i^2}, \tag{9}$$

where $\hat{\theta}_i$ is a good guess for the correct value of the parameter, and σ_i is the width of the prior. σ_i should be large enough to allow the model to move away from $\hat{\theta}$, but not so large as to jump into regions that you think are unrealistic.

In order to run `emcee` on the Peg 51b problem, you should extend the code that is given in the `jupyter` notebook and do the following steps:

- Write a function that can calculate $\ln L$. If you followed my advice above, you will already have this.
- Write a function that can calculate the prior likelihood of the model.
- Estimate a good starting point — a typical way to do this is to run a χ^2 fit as you did above.

- Decide on the number of *walkers* you need and give a random starting point for each.
- Initialise the sampler, the piece of code that will pop around and sample your posterior probability.
- Run the MCMC sampler.
- Determine the burn-in time and only use samples from after this.
- Calculate statistics and visualize the final samples.

While the list might look long you can cannibalize much of the code in the notebook to do the work.

Problem 4

Write a log prior function that uses a Gaussian prior, and justify your choice of $\hat{\theta}_i$ and σ_i. Run `emcee` for your model for 51 Peg b, using this prior and the log-likelihood for this model fitting; use your previous best-fit values as starting points. Produce a corner plot of the final result (see notebook for instructions).

The samples for each parameter contain the various values the MCMC sampler tries with a frequency proportional to the final likelihood of the parameter value. Thus if you now want to calculate the median value of, say, v_0, you could do

```
median_v0 = np.percentile(samples[:, 3], 50.0).
```

This is commonly used as the estimate of v_0, say, when you need to quote a number.

What then about uncertainties? Here we look back at the Gaussian function: when you do normal propagation of errors, you quote the standard deviation of a Gaussian. Thus in an expression $x = 4 \pm \sigma_x$, what we mean is that there is

$$\int_{-\sigma_x}^{\sigma_x} \frac{1}{\sigma_x\sqrt{2\pi}} e^{-(x-4)^2/(2\sigma_x^2)}\, dx = 0.68, \tag{10}$$

chance that the true value is in the range $[4-\sigma_x, 4+\sigma_x]$. Using this we say that $4-\sigma$ corresponds to the lower limit or the 0.16 quantile or 16th percentile, while $4+\sigma_x$ corresponds to the 0.84 quantile or 84th percentile. Transferring this over onto our values here, we can calculate these percentiles through

```
p16_v0 = np.percentile(samples[:, 3], 16),
```

for instance, and the uncertainty on our parameter would be

```
sigma_v0 = 0.5*(p84_v0-p16_v0).
```

Problem 5

To close off, please calculate the uncertainty on the estimate of the minimal mass of the planet 51 Peg b.