Computational Physics HW5

Ben Johnston

October 2023

1 Problem 1

This problem involved numerically calculating the gamma function $\Gamma(a)$, which is defined by the following integral:

$$\Gamma(a) = \int_0^\infty x^{a-1}e^{-x}dx \tag{1}$$

The first part of the problem was to write a program that plots the integrand $x^{a-1}e^{-x}$ as a function of x between $0 \leq x \leq 5$ for a = 2, 3, and 4. These curves can be seen below in *Figure 1*:

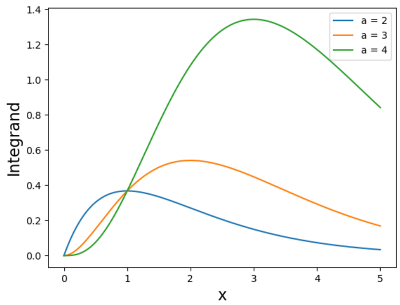

Figure 1: Numerical calculation of integrand

From this it can be seen that the integrands start at zero, rise to a maximum and then decay again.
The problem then asked to show analytically that the maximum of the integrand falls at $x = a - 1$. This is shown below:

$$f = x^{a-1}e^{-x}$$

$$\frac{df}{dx} = (a-1)x^{a-2}e^{-x} - x^{a-1}e^{-x}$$

$$0 = (a-1)x^{a-2}e^{-x} - x^{a-1}e^{-x}$$

$$\frac{x^{a-1}}{x^{a-2}} = a - 1$$

$$x = a - 1$$

As required.
Most of the area under the integrand falls near the maximum, so in order to get an accurate value of the gamma function we change the integral from [0,∞] to one over a finite range [0,1] using the following change of variables:

$$z = \frac{x}{c+x} \tag{2}$$

This change of variables puts the peak of the integrand in the middle of the integration range. In order to get the this peak at $z = \frac{1}{2}$, the appropriate choice of the parameter, c is worked out as follows (using $x_{max} = a - 1$:

$$\frac{1}{2} = \frac{a-1}{c+a-1}$$

$$\frac{1}{2}(c+a-1) = a - 1$$

$$c = a - 1 \tag{3}$$

Therefore, the appropriate choice of c is $c = a - 1$. The integrand $x^{a-1}e^{-x}$ can be hard to evaluate because the x^{a-1} can get very large and e^{-x} can get very small. This may then lead to large round off errors in the computation. If we write $x^{a-1} = e^{(a-1)\ln(x)}$, the integrand can be rewritten as:

$$e^{(a-1)\ln(x)-x} \tag{4}$$

This expression is better than the old one as round off errors due to the product of two terms with vastly different orders of magnitude are reduced. The final part of this problem involved numerically evaluating this integral for $a = \frac{3}{2}, 3, 6, 10$, which yielded values of:

$$\Gamma\left(\frac{3}{2}\right) = 0.8862269613087296$$

$$\Gamma(3) = 2.00000000000002$$

$$\Gamma(6) = 120.00000000000115$$

$$\Gamma(10) = 362880.00000000343$$

It can be seen here that for integer values of a, the gamma function is equal to the factorial of $a - 1$.

2 Problem 2

This problem demonstrates an application of linear algebra to signal analysis. Initially, a data file containing a signal was read into a Python script and plotted, as seen in *Figure 2*:

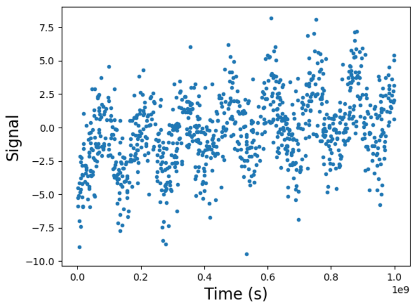

Figure 2: Plotted signal

The first part of the problem involved using the SVD technique to find the best third-order polynomial in time to fit the signal. This is shown below in *Figure 3*:

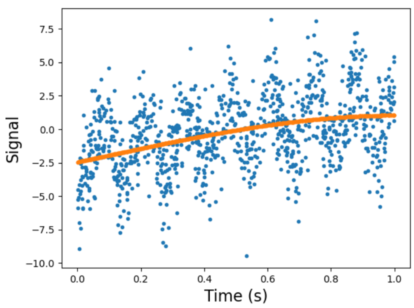

Figure 3: Signal plotted with third order polynomial

It is worth noting here that time had to be re-scaled here so that it was of order unity - this was done using the following change:

$$x' = \frac{x}{1 \times 10^9} \tag{5}$$

Following this, the residuals were calculated and plotted against the predicted data and time on two separate plots, as seen in *Figure 4*:

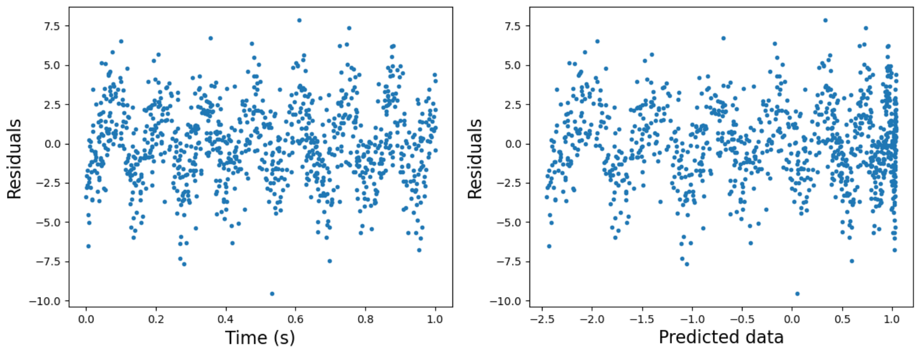

Figure 4: Residuals plot

From this it can be seen that the residuals are of the same order of magnitude as the signal itself and so therefore it can be reasoned that this model is not a good explanation for the data. In addition to this, we know that the uncertainties in the measurements are all the same, so therefore the residuals should be of a similar size but in this case they are oscillatory (like the signal) and so the model is not a good fit.

The next part of the problem involved using a much higher order polynomial to the data in order to see if a better model could be fitted. *Figure 5* shows a 30^{th} order polynomial model fit to the data, which seems much like a better fit:

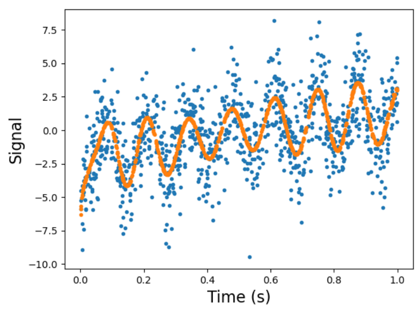

Figure 5: 30th order polynomial fit

However, the condition number of the design matrix used in this case was determined to be $1.447568649202812 \times 10^{16}$, which is in no way viable. For a condition number of a matrix to be viable it should be as close to 1 as possible, so therefore the aforementioned condition number is far too large. In order to see at what point the condition number gets too large for this particular calculation a plot of condition number as a function of polynomial order was plotted, as seen in *Figure 6*:

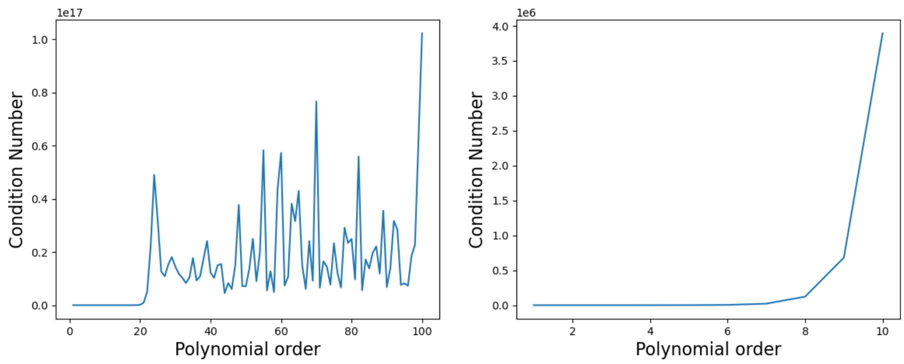

Figure 6: Left: Plot of condition number vs polynomial order, Right: Zoomed in graph of left

From the above graphs it is possible to see that the condition numbers stay relatively low until around a 5th order polynomial, following which they blow up very rapidly. From this is can be deduced that it is not particularly possible to have a reasonable polynomial to fit this signal data.

The final part of this question involved fitting a set of sin and cos functions plus a zero-point offset to the data. This was again done using the SVD technique, the implementation of which can be seen in the script *Problem2e.py*. This model fit can be seen in *Figure 7*:

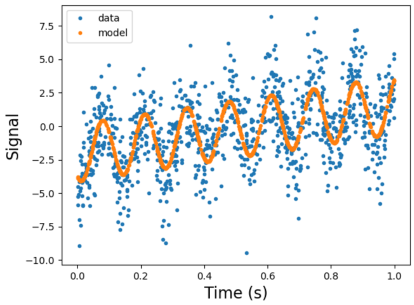

Figure 7: Combination of sine and cosine fit

This model visually looks like a good fit to the data, however in comparison to the previous fit the condition number of this design matrix is 4.346146446092807 which is much closer to 1 (of the same order of magnitude). Therefore it can be said this combination of sines and cosines to fit the data is a good model and explains the data well. The period of the signal was then determined from the fit to be 0.13307475366895194 in re-scaled time units.