A Log Likelihood fit for extracting the D^0 lifetime

James Bremner
Department of Physics
Imperial College London
CID: 00943377

19th December 2016

Abstract: A Negative Log Likelihood (NLL) fit was applied first in one-dimensional parameter-space to determine a value for the lifetime of a D^0 meson using a set of measured decay times from a high energy particle physics experiment, this was found to be $\tau = (0.405 \pm 0.005)$ ps through implementation of a parabolic minimiser. The variation in the standard deviation estimate for τ with sample size (n) was examined for this one-dimensional minimisation, and was found to scale as $\sigma_\tau \sim \frac{1}{n^b}$ where $b = 0.473 \pm 0.002$. The number of samples at which the uncertainty in the τ estimate would drop below 10^{-15}s was found to be $n_{crit} = 276,000 \pm 26,000$. A parameter for the presence of background in the signal was then introduced and a two-dimensional NLL minimisation carried out on the same data, the revised decay time was found as: $\tau = (0.410 \pm 0.005)$ ps and the proportion of relevant decay events in the sample $a = 0.984 \pm 0.009$. This value of τ is consistent with the Particle Data Group's published value.

1 Introduction

High energy physics experiments produce large volumes of data (this particular type of experiment typically uses between 10,000-200,000 separate events [3]) which require sophisticated analysis in order to determine the true parameter values and associated uncertainties describing the particles in question. As well as the random statistical errors produced, the reality of experimental physics means that there are often also countless systematic errors to take into account, these can all be introduced as separate fitting parameters.

One then aims to find the most likely value for a set of parameters given a certain sample of data, this type of problem essentially reduces to an exercise in minimisation, for which there exist a number of alternative numerical methods that can be implemented. Here we apply such methods to extract the lifetime from a sample of D^0 meson ($c\bar{u}$) decay times.

2 Theoretical Background

2.1 Negative log likelihood fit

The data used in the analysis consisted of 10,000 pairs of (t, σ) values. If we ignore the uncertainty in the measurements, we'd expect such decay times to follow the exponential probability distribution:

$$f^t(t) = \begin{cases} 0 & \text{if } t < 0, \\ \frac{1}{\tau}\exp(\frac{-t}{\tau}) & \text{if } t \geq 0, \end{cases} \tag{1}$$

where τ is the average lifetime we wish to determine [1]. When we include the uncertainty, this fit function is convoluted with a Gaussian, after performing the convolution integral it becomes:

$$f^m_{sig}(t) = \frac{1}{2\tau}\exp\left(\frac{\sigma^2}{2\tau^2} - \frac{t}{\tau}\right)\text{erfc}\left(\frac{1}{\sqrt{2}}\left(\frac{\sigma}{\tau} - \frac{t}{\sigma}\right)\right). \tag{2}$$

To take into account the background in the signal we must add a further term which is a convolution of $\delta(0)$ and a Gaussian (since the background arises from random combinations all with zero lifetimes which are then also smeared by the detector resolution as before):

$$f^m_{bkg}(t) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\frac{t^2}{\sigma^2}\right). \tag{3}$$

The new fit function is then:

$$f(t) = a f^m_{sig}(t) + (1-a) f^m_{bkg}(t), \tag{4}$$

Where a is a parameter that controls what proportion of the data is from $f^m_{sig}(t)$, ie. the real D^0 meson decays. *Equation 4* is the fit function which can now be used to treat the data, this general form is shown graphically in the Appendix (*Section 7.1*). In order to find the most likely parameter given the data, we will maximise the Log Likelihood function, that is, minimise the Negative Log Likelihood (NLL) function:

$$\text{NLL}(\tau, a) = -\sum_{i=1}^{n} \log(f(\tau, \text{a}; t_i, \sigma_i)). \tag{5}$$

Here we sum over all of the data pairs for particular a and τ, the aim is to find the value of these parameters which minimise the NLL(τ, a).

2.2 Uncertainties

For τ close to the minimum, we can approximate the NLL by a parabola:

$$NLL(\tau) \approx \alpha\tau^2 + \beta\tau + \gamma, \tag{6}$$

where α, β and γ are all real constants. It can also be shown that the second-order accurate approximation for the standard deviation in the found parameter value, $\hat{\tau}$, is:

$$\sigma_{\hat{\tau}} = \frac{1}{\sqrt{\frac{d^2 NLL(\tau)}{d\tau^2}\Big|_{\hat{\tau}}}} \tag{7}$$

Differentiating *Equation 6* and substituting into *Equation 7*, we have:

$$\sigma_{\hat{\tau}} \approx \frac{1}{\sqrt{2\alpha^2}} \tag{8}$$

If we approximate the likelihood function by a purely Gaussian distribution, we can also define the standard deviation as the difference between the minimum and the point either side of the minimum at which the NLL has increased by $\frac{1}{2}$.

With this Gaussian approximation for the likelihood, we expect the standard deviation of τ in the case where a=1 to scale with sample size n according to:

$$\sigma_\tau(n) \approx \frac{k}{\sqrt{n}}, \tag{9}$$

where it follows that k represents the error in the parameter estimate if we were to take just one measurement. To derive errors in the two-dimensional case, one can start with a Hessian matrix for the NLL which is defined as:

$$H_{ij} = \frac{\partial^2 NLL(x_i, ..x_n)}{\partial x_i \partial x_j}. \tag{10}$$

In our case $x_i, ..x_n = \{\tau, a\}$. From this, a covariance matrix, **E**, can be produced:

$$\mathbf{E} = \mathbf{H^{-1}}. \tag{11}$$

The diagonal elements represent the variance of the parameters τ and a, whilst the off-diagonal elements can be used to find the correlation coefficients for the parameters [2]:

$$\rho_{\tau a} = \frac{E_{\tau a}}{\sigma_\tau \sigma_a} \tag{12}$$

3 Method

Python was used in conjunction with the numpy, scipy, and matplotlib libraries to program the NLL function in *Equation 5* and write methods to minimise it. First, $a = 1$ was fixed and the minimisation carried out in one-dimensional parameter-space using a parabolic minimiser. The minimiser was configured to stop searching once the change in the iteratively produced minimum τ parameter dropped below 10^{-11}. This particular threshold was chosen as it wasn't far from the 10^{-15} floating-point precision presented in the final τ value, and it was also far smaller than the precision of the experiment.

From this minimum, various approaches could be taken in order to find an estimate for the error in the τ parameter. Secant and bisection methods (for finding the roots of a function) were used to find the value of τ at which the NLL increased by $\frac{1}{2}$ (as described in *Section 2.2*), the parabolic fitting parameters produced by the minimisation were also used to provide an estimate for this value (as in *Equation 8*).

Separate functions were written to determine how the standard deviation changed with varying sample size. This was determined over a range of 100 to the full 10,000 data points and a regression fit (using scipy.optimize.curve_fit) was applied to this to test *Equation 9* and also to enable extrapolation. The fit form used here was:

$$\sigma_\tau(k, b; n) \approx \frac{k}{n^b}, \tag{13}$$

where k and b were passed into curve_fit as the fitting parameters. The covariance matrix produced by the fit was used to determine the errors in k and b and hence determine the error in the value of n_{crit} (the largest value of n at which $\sigma_\tau < 10^{-15}$ s) by using the combination of errors which produced the maximum and minimum values of n_{crit}.

To include the background, a (as in *Equation 4*) was now considered as a variable parameter and a new set of functions were written which implemented three different 2D minimisation techniques: Gradient, Newton and Quasi-Newton. These were chosen as they are considered to be relatively effective at locating minima in the case where the function is fairly "well-behaved": plotting the NLL (*Figure 3*) showed that it had only one minimum and no saddle-points therefore it was judged to fit this criterion. More details on each method can be found in the Appendix (*Section 7.2*).

These methods were all configured to stop searching for the minimum only once the difference in values for τ and a after each step dropped below 10^{-11}, as before. This was again excessively precise for the data given, however it was deemed advantageous to undertake subsequent calculations with the full precision before rounding the final answer. The speed of each method was also not hugely affected by this requirement. The initial points were chosen fairly close to the minimum, the rough location of which could initially be determined by examining the contour plot of the NLL.

Analytical derivatives were first used to compute the gradient (and the Hessian where it was required) of the NLL for these methods, however, certain small values of σ_t in the data sample produced overflow errors, the gradient and Hessian were therefore calculated using finite difference methods instead. This accuracy was deemed sufficient as the gradient was generally used only as a route to the minimum, so any discrepancies would only result in a *slower* method, not a less accurate one. However, the Hessian was used once to determine the uncertainties in two-dimensions and the correlation coefficient, here care was taken to ensure that the step size was sufficiently small.

A covariance matrix in τ and a was produced in order to find the standard deviation and correlation coefficients in both parameter values according to *Equations 11-12*. The stability of each method was then tested manually at a range of different initial parameter values.

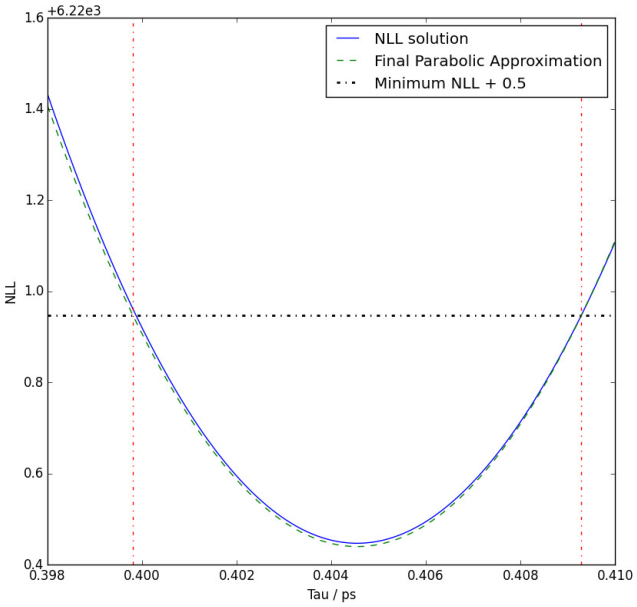

Figure 1: The parabolic approximation for the NLL. Vertical red dotted lines represent the uncertainties found from the curvature whilst the points of intersection of the black dotted line with the NLL solution represent where the errors were found with the Secant method.

4 Results & Discussion

4.1 One-dimensional minimisation

The parabolic minimisation applied to the data (ignoring the presence of a background signal) yielded: $\tau = (0.405 \pm 0.005)$ ps. Both methods for finding the uncertainty in this value were consistent to 2 s.f. but differed beyond this precision. Both are approximations: as can be seen in *Section 2.2*, the method using the curvature of the NLL near the minimum first uses a parabolic approximation and then a truncation error is introduced as the Taylor expansion of the NLL used in the derivation of *Equation 7* neglects $\mathcal{O}(\tau^3)$ terms. The Secant method approximates the likelihood function as a Gaussian so therefore also has inaccuracies, however it's hard to quantify which produces the most accurate uncertainty value(s). It is worth noting that the Secant method gives an asymmetric uncertainty interval about the minimum which may represent an advantage over the curvature method.

We can see graphically that the parabolic approximation used remains accurate for these small deviations from the minimum in *Figure 1*. However, this result for τ is only just within uncertainty of the generally accepted value from the Particle Data Group: $\tau = 0.4101 \pm 0.0015$ ps [3], this suggests that the initial assumption that we could neglect the background signal was invalid. We are therefore justified in applying a two-dimensional minimisation of the NLL as detailed in the following section.

The change in standard deviation against sample size was plotted for the Curvature and Secant methods as well as the fit applied (arbitrarily) to the Secant method, this is shown in *Figure 2*. The fitting parameters were obtained so that the standard deviation varied according to *Equation 13* with $k = (0.376 \pm 0.006)$ ps and $b = 0.473 \pm 0.002$. It is worth noting here that $b \approx 0.5$ as predicted by *Equation 9*. It is is harder to compare k, however it is of the same order of magnitude as the arithmetic mean of the uncertainties in the data so it seems consistent. Discrepancies in these

values are likely due to the fact that our likelihood function isn't perfectly Gaussian. Extrapolating with the fit, it was estimated that the number of measurements required to achieve an uncertainty of 10^{-15}s is $n_{crit} = 276,000 \pm 26,000$.

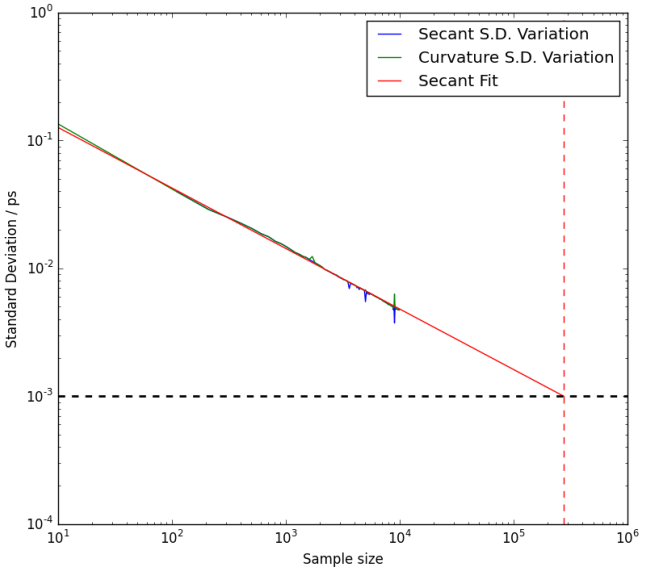

Figure 2: The Standard deviation in the one dimensional parameter estimate for τ against sample size on a log-log scale. Both methods for finding the standard deviation are plotted as well as a fit for the Secant method to extrapolate and determine the number of samples at which the error reaches 10^{-15}s (represented by the dashed lines).

4.2 Two-dimensional minimisation

When the two dimensional minimisation methods were applied to the NLL, all produced the same minimum as: $\tau = (0.410 \pm 0.005)$ ps and $a = 0.984 \pm 0.009$, ie. $(1.6 \pm 0.9)\%$ of the data in the signal is from background effects. This value for τ is in strong agreement with the lifetime published by the Particle Data Group (as quoted previously). It follows that this new τ value is larger (by $\sim 1.2\%$) than the first estimate as the background points (all with zero measured lifetime) previously acted to skew the results towards $\tau = 0$. The values for τ and a produced by both the one and two-dimensional minimisations are reproduced in *Table 1* for comparison.

	τ / ps	a
One-Dimensional	0.405 ± 0.005	(1.0)
Two-Dimensional	0.410 ± 0.005	0.984 ± 0.009
Particle Data Group [3]	0.4101 ± 0.0015	-

Table 1: Parameter values for τ and a as produced by first neglecting the background in the signal (by fixing $a = 1$ in the one-dimensional case) and then including it (two-dimensional). No a value was provided for the Particle Data Group value as it was produced from a systematic review.

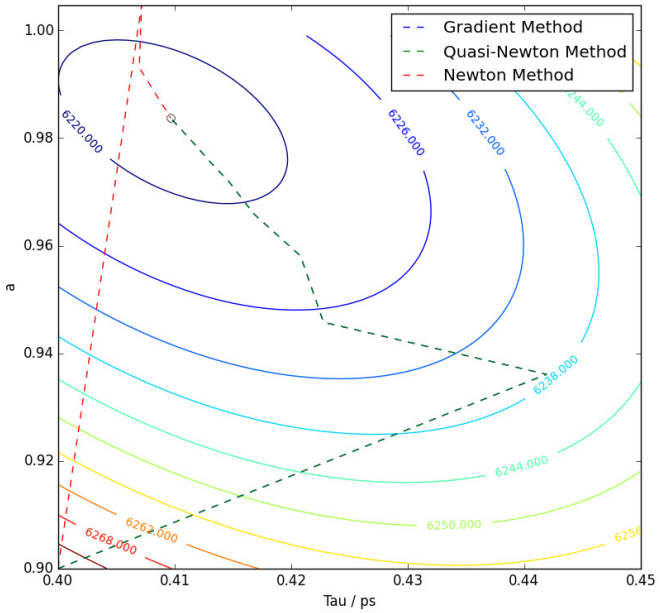

Figure 3: A contour-plot of the two-dimensional NLL in the $\tau - a$ parameter-space with minimisation trajectories for each of the three methods plotted all starting from the same point relatively close to the minimum (here represented by the small circle). It is hard to distinguish between the Quasi-Newton and Gradient methods as their trajectories were almost exactly the same.

Various examples of fit functions superimposed on the raw data can be found in the Appendix (*Section 7.1*), it can be seen that the final parameter estimates (visually) provide a good fit to the original sample data. From the off-diagonal entries in *Table 2*, the correlation coefficient between τ and a was found to be: $\rho_{\tau a} = -0.481$, therefore if τ fluctuates above its parameter estimate, a is likely to fluctuate *below* its respective estimate and vice versa. The trajectories of the three two-dimensional minimisation methods ap-

plied are shown in *Figure 3*, there is a more detailed comparison of their relative merits in the Appendix (*Section 7.3*).

	τ / ps^2	a
τ / ps^2	3.017×10^{-5}	-2.264×10^{-5}
a	-2.264×10^{-5}	7.335×10^{-5}

Table 2: Covariance matrix for τ and a parameter values, given here to an arbitrary precision.

5 Conclusions

A Negative Log Likelihood fit was applied to a sample of 10,000 data points from a high energy particle physics experiment to measure the lifetime of a D^0 meson, first in one-dimensional parameter space and secondly in two dimensions, this time taking into account the presence of background in the signal.

When neglecting background effects, the one-dimensional parabolic minimisation method produced: $\tau = (0.405 \pm 0.005)$ ps. The two-dimensional methods yielded: $\tau = (0.410 \pm 0.005)$ ps and the fraction of signal in the sample $a = 0.984 \pm 0.009$. The latter value for τ agrees strongly with the generally accepted quantity published by the Particle Data Group: $\tau = 0.4101 \pm 0.0015$ ps [3]. Ignoring the background was evidently an invalid approach as we can see the initial result for τ shows poor agreement with the published value.

The scaling of the standard deviation estimate for τ with sample size (n) was analysed for the one-dimensional minimisation, and was found to scale as $\sigma_\tau \sim \frac{1}{n^b}$ where $b = 0.473 \pm 0.002$. The number of samples at which the uncertainty in the τ estimate would drop below 10^{-15}s was found to be $n_{crit} = 276,000 \pm 26,000$.

If a larger data sample had been provided ($n \gtrsim 300,000$), as estimated, the uncertainty would have been of the same order of magnitude as that in the Particle Data Group's value for τ, and a closer comparison could have been made. The methods could have also easily been applied to the measured decay times of any other particle given a minimal alteration of initial parameter conditions.

6 References

[1] van Sebille E Uchida Y. Project b1: A log likelihood fit for extracting the d0 lifetime. Technical report, Imperial College London, Imperial College London, 2016.

[2] Paul Dauncey. Second year statistics of measurement. Technical report, Imperial College London, October 2015.

[3] C Patrignani, Particle Data Group, et al. Review of particle physics. *Chinese physics C*, 40(10):100001, 2016.

7 Appendix

7.1 Fit Function Examples

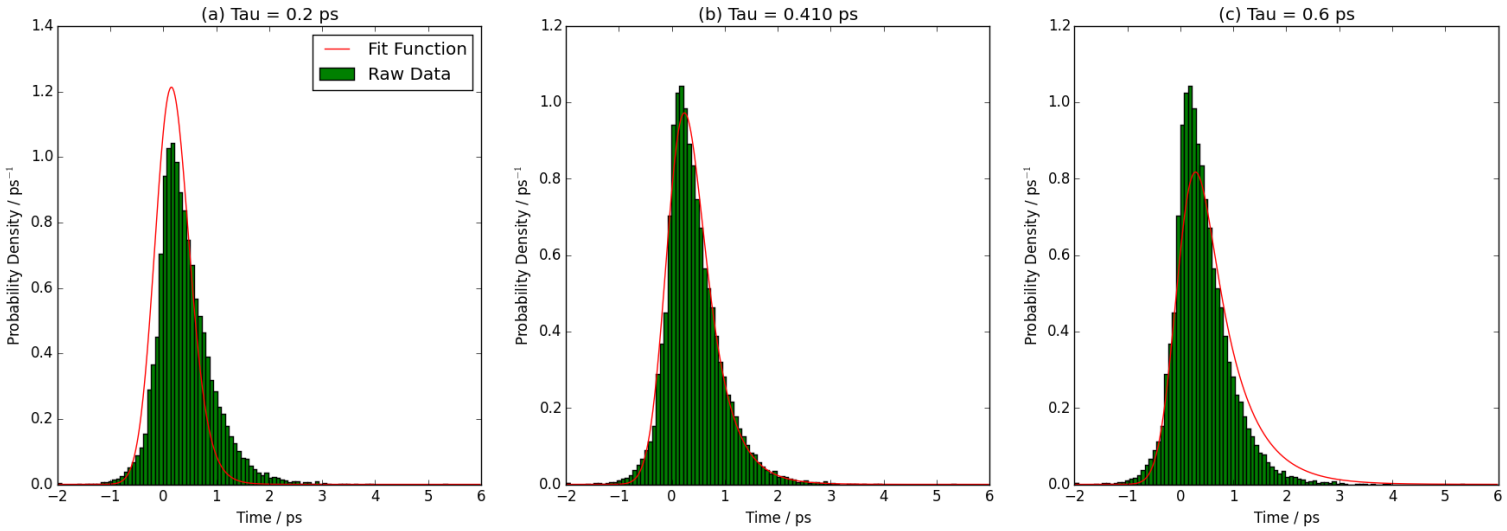

Figure 4: Histograms of the raw data with various fit functions using the optimised parameter for a found from the two-dimensional minimisation (see Table 1) but with varying τ. (b) shows the fit function with both parameters at their optimised values.

7.2 Two-dimensional minimisation methods

This section contains brief descriptions of the two-dimensional minimisation methods used. All rely on the concept of finding the gradient of the function at a particular point before then taking iterative steps in the negative direction of this gradient to travel closer to the minimum. They differ from each other in the additional techniques they use in order to make this process as efficient as possible.

7.2.1 Gradient

The Gradient method is the most straight-forward of the methods implemented here, each step requiring the least computation time. Its iterative formula is simply:

$$\vec{x_{n+1}} = \vec{x_n} - \alpha \vec{\nabla} f(\vec{x_n}). \tag{14}$$

Here $\vec{\nabla} f(\vec{x_n})$ is the gradient of $f(\vec{x_n})$ where in our case, $\vec{x_n} = \begin{bmatrix} \tau_n \\ a_n \end{bmatrix}$ and $f(\vec{x_n})$ is the NLL function evaluated at $\vec{x_n}$. In practice, this gradient can be approximated by a finite difference method. The α coefficient is a fixed parameter chosen by the user which determines the step-size, there often exists an optimum value to ensure both speed and stability depending on the particular conditions.

7.2.2 Newton

Although details won't be discussed here, the Newton method aims to make more accurate steps towards the minimum by using the Hessian (or curvature) matrix (as described in *Equation 10*):

$$\vec{x}_{n+1} = \vec{x_n} - [\mathbf{H}(\vec{x_n})]^{-1} \cdot \vec{\nabla} f(\vec{x_n}). \tag{15}$$

This approach requires many fewer steps (as evidenced in *Table 3*), however the calculation of the second derivatives for the Hessian is also costly so this method may not always prove ideal.

7.2.3 Quasi-Newton

In order to avoid the time-consuming calculation of the Hessian matrix in the Newton Method, one can instead use an approximation, this is the basis of the Quasi-Newton method. The formula is as follows:

$$\vec{x}_{n+1} = \vec{x_n} - \alpha \mathbf{G_n} \cdot \vec{\nabla} f(\vec{x_n}), \tag{16}$$

where $\mathbf{G_n}$ is the Hessian approximation and can be produced from a number of different update algorithms, the one used here was the DFP:

$$\vec{\delta_n} = \vec{x}_{n+1} - \vec{x}_n,$$

$$\vec{\gamma_n} = \vec{\nabla} f(\vec{x}_{n+1}) - \vec{\nabla} f(\vec{x}_n),$$

$$\mathbf{G_{n+1}} = \mathbf{G_n} + \frac{\vec{\delta_n} \otimes \vec{\delta_n}}{\vec{\gamma_n} \cdot \vec{\delta_n}} - \frac{\mathbf{G_n} \cdot (\vec{\delta_n} \otimes \vec{\delta_n}) \cdot \mathbf{G_n}}{\vec{\gamma_n} \cdot \mathbf{G_n} \cdot \vec{\gamma_n}} \tag{17}$$

7.3 Performance comparison of the two-dimensional minimisation methods

The Quasi-Newton and Gradient methods produced very similar results and initial parameter dependence. On closer inspection this was because the approximation of the inverse Hessian matrix ($\mathbf{G_n}$) used in the Quasi-Newton method was almost exactly equal to the identity matrix at each step. When $\mathbf{G_n} = \mathbf{I}$, the method reverts to the Gradient method, hence the resulting similarities between the two. For this reason, it may have been useful to use an alternative formula for the update of the $\mathbf{G_n}$ matrix (BFGS for example) which would have perhaps produced a better approximation, however for the purposes of the analysis done here it wasn't necessary.

It is understood that the speed of the methods is in many ways very subjective, but timed values are included in *Table 3* for a rough guide to their performance. Although as explained, the Gradient and Quasi-Newton methods' trajectories were very similar, the computation of the relatively complex update formula for $\mathbf{G_n}$ in the Quasi-Newton method proved burdensome on its run-time. The Newton Method proved to be the fastest minimisation method, also requiring the fewest number of steps, however it was found to be very unstable; one needed to start fairly close to the minimum for the result to converge. All three methods had upper and lower bounds in τ for stability as well as for the iteration step parameter α. There was also stability dependence on a in the Quasi-Newton and Gradient methods for low a, the Newton Method was unconditionally stable for all a.

	Run-time / ms	Steps
Newton	102	7
Gradient	147	46
Quasi-Newton	262	41

Table 3: Run-time values for each of the two-dimensional minimisation methods using the Python timeit() module. Initial conditions for each was: $\tau_{init} = 0.4$ ps and $a_{init} = 0.9$.