S2: Statistics for Data Science
The Lighthouse Problem

Max Talberg

March 28, 2024
Word count: 2241

Contents

0.1 (i) Trigonometric Analysis of the Lighthouse Problem

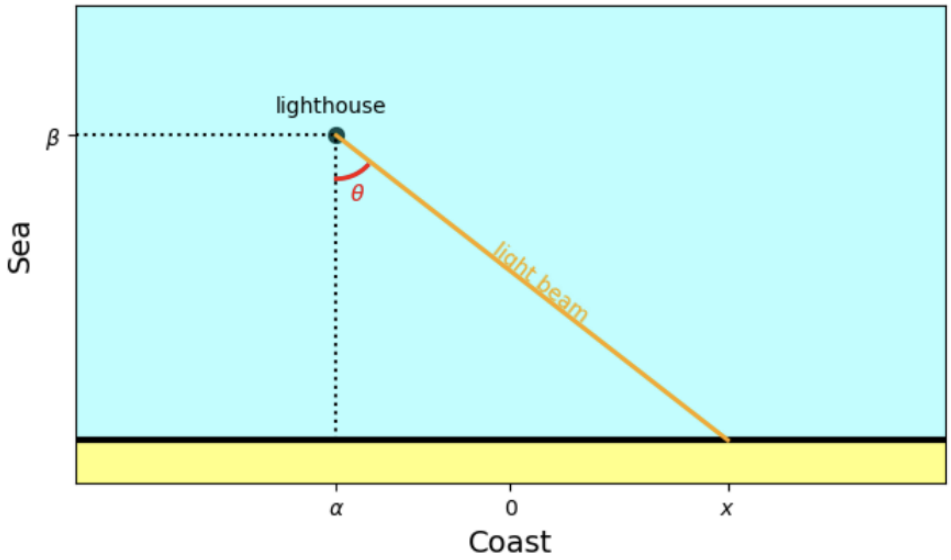

Figure 1: Diagram of the setup of the lighthouse problem.

Using the geometry of the problem illustrated in Figure 1, the trigonometric relationship between the lighthouse's position at (α, β), the angle of the light beam θ, and the point on the coastline x can be established.

From trigonometric principles the tangent of angle θ is the ratio of the opposite side (horizontal distance to x) to the adjacent side (lighthouse's height), yielding:

$$\tan(\theta) = \frac{\text{opposite}}{\text{adjacent}} = \frac{x-\alpha}{\beta} \tag{1}$$

$$\beta \tan(\theta) = x - \alpha \tag{2}$$

$$x = \beta \tan(\theta) + \alpha \tag{3}$$

0.2 (ii) Derivation of the Likelihood Function for Flash Location

The angle of a flash is denoted by θ and is uniformly distributed in the range $-\pi/2 < \theta < \pi/2$.

Uniform Distribution of θ: The probability density function (PDF) for θ is given by,

$$P(\theta) = \mathbb{1}_{(-\pi/2,\pi/2)}(\theta)\frac{1}{\pi}, \tag{4}$$

where $\mathbb{1}_{(-\pi/2,\pi/2)}(\theta)$ is an indicator function ensuring that $P(\theta)$ is defined only within the specified range. This reflects the assumption that flashes are equally likely to occur in any direction within this range.

Transformation to x: The likelihood of observing a flash at location x, given α and β, involves transforming the PDF from θ to x. This is based on the transformation law,

$$P(x|\alpha,\beta)dx = P(\theta|\alpha,\beta)d\theta, \tag{5}$$

$$P(x|\alpha,\beta)dx = P(\theta|\alpha,\beta)\frac{d\theta}{dx}dx. \tag{6}$$

Equation 3 relates θ and x, which rearranges to give $\theta = \arctan\left(\frac{x-\alpha}{\beta}\right)$. Differentiating this with respect to x gives,

$$\frac{d\theta}{dx} = \frac{\beta}{\beta^2 + (x-\alpha)^2}. \tag{7}$$

Substituting this into the transformation law yields the PDF of x,

$$P(x|\alpha,\beta)dx = P(\theta|\alpha,\beta)\frac{\beta}{\beta^2 + (x-\alpha)^2}dx, \tag{8}$$

$$P(x|\alpha,\beta) = \frac{1}{\pi}\frac{\beta}{\beta^2 + (x-\alpha)^2}. \tag{9}$$

Conclusion: The derived PDF $P(x|\alpha,\beta)$ represents the likelihood $\mathcal{L}_x(x|\alpha,\beta)$ of observing a flash at location x, given the parameters α and β. This likelihood is given by,

$$\mathcal{L}_x(x|\alpha,\beta) = \frac{1}{\pi}\frac{\beta}{\beta^2 + (x-\alpha)^2}. \tag{10}$$

This represents the PDF of the Cauchy distribution with location parameter α and scale parameter β. The Cauchy distribution is a pathological function as both its mean and variance are undefined.

0.3 (iii) Most Likely Flash Location

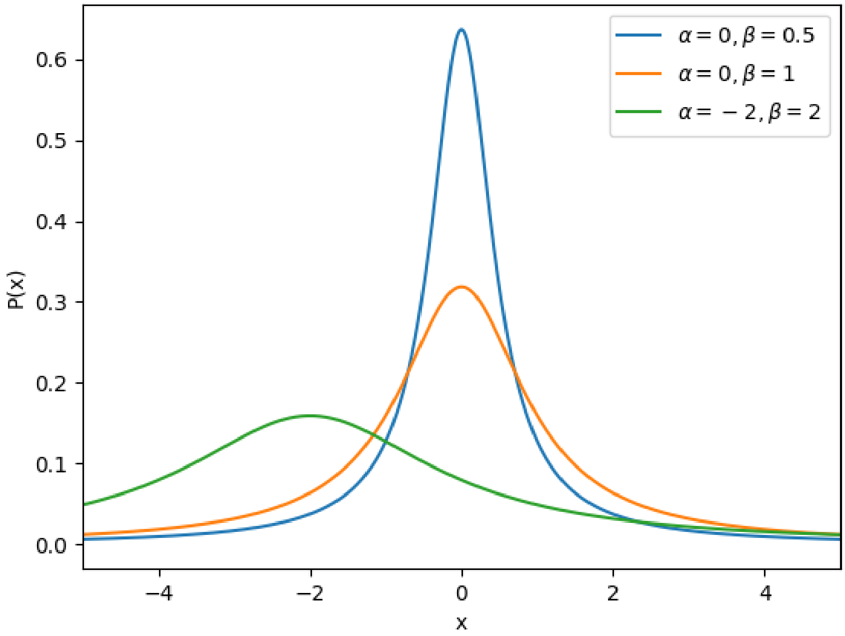

Figure 2: The graph displays three Cauchy distributions with varying location (α) and scale (β) parameters. The location parameter α shifts the peak of the distribution along the x-axis, representing the most frequent flash coordinates of the lighthouse. The scale parameter β influences the spread of the distribution, with higher values indicating a larger spread and suggesting a greater distance from the shore to the lighthouse. The blue line (α = 0, β = 0.5) shows a narrow spread centred at zero, the orange line (α = 0, β = 1) represents the standard Cauchy distribution with a wider spread, and the green line (α = -2, β = 2) shows the widest spread, shifted to the left.

Cauchy distribution: The colleague is correct that the most likely location for any flash to be received is α. This is due to the symmetric properties of the Cauchy distribution, which can be seen in Figure 2, around its location parameter α. At this median and modal point, the PDF reaches its maximum making α the most likely value for any single observation. Using the sample mean, $(1/N)\sum_k x_k$, isn't a good estimator for α because the heavy tails of the Cauchy distribution lead to an undefined mean, as the integral meant to calculate the mean fails to converge. An improved estimator for α would be the Maximum Likelihood Estimate (MLE). The MLE of α is derived by setting up and maximising the likelihood function based on the Cauchy distribution's PDF. This process leads to the sample median being the MLE for α, due to the symmetric nature of the Cauchy distribution. The median is more robust in the presence of outliers and extreme values, which are characteristic of the Cauchy distribution.

Analytical comparison: Investigating the expectation value of a simplified expression of the likelihood function, equation 10, proves the undefined nature of the sample mean,

$$\mathbb{E}_x[x] = \int_0^L x\frac{1}{(a-x^2)}dx, \tag{11}$$

$$\mathbb{E}_x[x] = \left[ln(x-a) - \tfrac{a}{x-a}\right]_0^L. \tag{12}$$

It is clear the sample mean tends towards lnL as L becomes large. This property from the continuous distribution carries over to the behaviour of the sample mean when computed from discrete samples drawn from a Cauchy distribution. It is clear this function does not possess a well-defined mean and does not converge as more data is collected. Therefore, the sample mean is not a good estimator of the most likely location for a flash to be received.

Due to the symmetric property of the Cauchy function, the most likely location for a flash to be received is better found using the median or mode represented by the MLE. The total likelihood of all observations is,

$$\mathcal{L}_x(\{x_k\}|\alpha,\beta) = \prod_k^n \frac{1}{\pi}\frac{\beta}{\beta^2+(x-\alpha)^2}. \tag{13}$$

Taking the natural logarithm of the total likelihood gives the log-likelihood function,

$$\log\mathcal{L}_x(\{x_k\}|\alpha,\beta) = \sum_k^n \log\left(\frac{1}{\pi}\frac{\beta}{\beta^2+(x-\alpha)^2}\right). \tag{14}$$

This simplifies to,

$$\log\mathcal{L}_x(\{x_k\}|\alpha,\beta) = \sum_k^n \left(\log(\beta) - \log(\pi) - \log[\beta^2+(x_k-\alpha)^2]\right), \tag{15}$$

$$\log\mathcal{L}_x(\{x_k\}|\alpha,\beta) = n\log(\beta) - n\log(\pi) - \sum_k^n \log[\beta^2+(x_k-\alpha)^2]. \tag{16}$$

To find the MLE for α the derivative of the log-likelihood function is taken with respect to α and set equal to zero,

$$\frac{\partial}{\partial\alpha}\log\mathcal{L}_x(\{x_k\}|\alpha,\beta) = 2\sum_k^n \frac{x_k-\alpha}{\beta^2+(x_k-\alpha)^2} = 0. \tag{17}$$

It is clear to see for a single observation x the MLE for α is itself x. The case for multiple observations is significantly more complex and is not a single observed value x_k but a value that accommodates the distribution of values, which for the Cauchy distribution tends to the median.

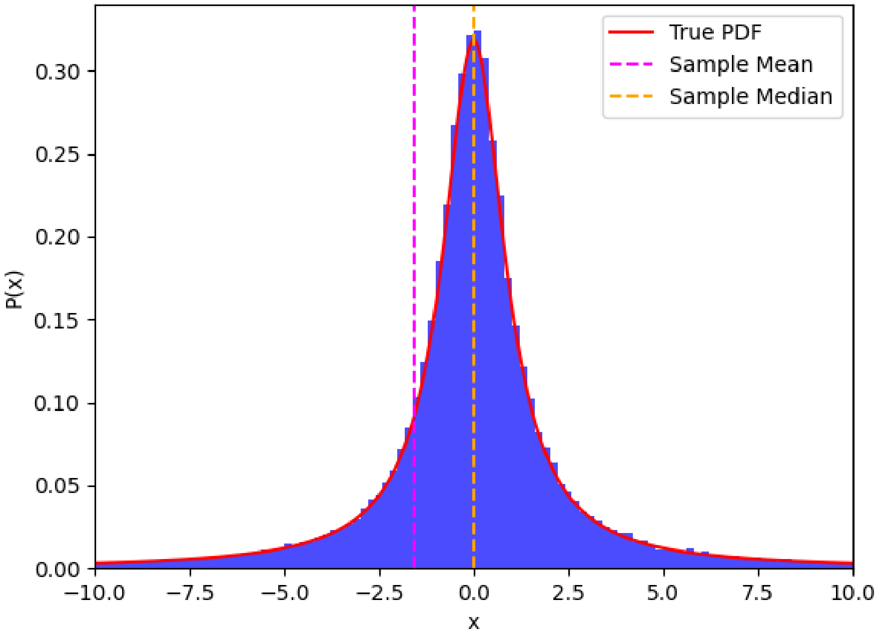

Figure 3: The histogram represents the simulated distribution of 100,000 random data points (flashes) generated from a Cauchy distribution with location parameter $\alpha = 0$ and scale parameter $\beta = 1$. Overlaid on the histogram is the true PDF of the Cauchy distribution, depicted by the red curve. The vertical dashed lines indicate the calculated sample mean (magenta) and sample median (orange) of the data set. These lines illustrate the concept that for distributions with heavy tails like the Cauchy distribution, the median can be a more robust measure of the most likely location than the mean.

Empirical comparison: Evidenced by Figure 3 the median's closer alignment with the peak of the true PDF compared to that of the mean supports the analytical and theoretical evidence that the MLE is a better estimator for α than the sample mean.

0.4 (iv) Selecting a Suitable Prior for α and β

The Bayesian prior represents the state of knowledge before any data. There is no information about the location of the lighthouse, hence a non-informative uniform distribution over the rectangular region spanning horizontally from a to b and vertically from c to d satisfies this ignorance,

$$\pi_x(\alpha, \beta) = \begin{cases} \frac{1}{(b-a)(d-c)} & \text{for } a < \alpha < b \text{ and } c < \beta < d \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

0.5 (v) Lighthouse Location Using Flash Locations

Stochastic sampling algorithm: The Markov chain Monte Carlo (MCMC) algorithm used here is the Hamiltonian Monte Carlo (HMC) sometimes known as the hybrid Monte Carlo [1]. The HMC utilises Hamiltonian dynamics to explore high-dimensional spaces and is utilised here to draw samples from a target posterior distribution[2].

The HMC begins by augmenting the state space with pairs of position and momentum variables. A canonical distribution is composed of the target distribution (representing the position) and an easy-to-sample distribution (representing momentum), typically a multivariate Gaussian. These terms correspond to the potential and kinetic energy, together forming the Hamiltonian. The trajectory of the system is explored using leapfrog integration, Algorithm 1, which alternates between updating the position and momentum. This integration proposes a new state which is subject to a Metropolis-Hastings acceptance step. If the proposed step is rejected, the algorithm retains the current state. Momentum terms are discarded and resampled from their distribution. This process of drawing momentum, simulating Hamiltonian dynamics and performing the Metropolis-Hastings acceptance step is iteratively repeated in Algorithm 2.

Algorithm 1 Leapfrog step

```
procedure LEAPFROG($x, p, \Delta t, M$)
    $p \leftarrow p - \frac{\Delta t}{2} \nabla V_{\mathrm{E}}(x)$        ▷ half step for momentum
    $x \leftarrow x + \Delta t M^{-1} p$        ▷ full step for position
    $p \leftarrow p - \frac{\Delta t}{2} \nabla V_{\mathrm{E}}(x)$        ▷ half step for momentum
    return $x, p$
end procedure
```

```
$x_0 \sim \alpha$        ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ do        ▷ Iterate $i = 0, 1, 2, \ldots$
    $p \sim Q$        ▷ Draw random momentum
    $x \leftarrow x_i$        ▷ Integrate from current chain position
    $H_{\text{initial}} \leftarrow \mathcal{H}(x, p)$        ▷ Initial Energy
    for $l = 0, 1, \ldots, L - 1$ do
        $x, p \leftarrow$ LEAPFROG($x, p, \Delta t, M$)        ▷ Integrate
    end for
    $H_{\text{final}} \leftarrow \mathcal{H}(x, p)$        ▷ Final Energy
    $a \leftarrow \exp(H_{\text{initial}} - H_{\text{final}})$        ▷ MH acceptance probability
    $u \sim U(0, 1)$
    if $u < a$ then
        $x_{i+1} \leftarrow y$        ▷ Markov transition (accept)
    else
        $x_{i+1} \leftarrow x_i$        ▷ Markov transition (reject)
    end if
    $i \leftarrow i + 1$
end while
```

HMC is advantageous because it does not require a proposal distribution and the small time steps in the leapfrog integration conserves the Hamiltonian well, resulting in a high acceptance rate. The HMC uses gradient information and Hamiltonian dynamics to explore parameter space, compared to undesirable random walk behaviour from other sampling algorithms.

The HMC however can only be applied to smooth target distributions because the algorithm requires derivatives of the target distribution. Despite the heavy tails and the undefined mean the Cauchy function is a smooth distribution, which means derivatives are well defined.

During the implementation of the algorithm, there are many input parameters to tune, which is a drawback for HMC. A variation of the HMC is the No U-Turn Sampler (NUTS). NUTS provides an automated way to determine the step size L in the leapfrog integration. This results in fewer parameters to tune when implementing HMC[3].

Software implementation: The NUTS algorithm, an extension of the HMC algorithm, was implemented using PyMC3 [4]. PyMC3 uses Theano to perform automatic differentiation of the model log-posterior, which is then used by the NUTS sampler to automatically adjust the leapfrog integration step. NUTS uses a dual-averaging algorithm, to tune the leapfrog step size, finding an optimal step size that balances exploring the parameter space and computational efficiency. This can be tuned by the target acceptance probability, this hyper-parameter determines what percentage of proposed points are accepted. A high acceptance rate suggests small conservative steps are being made leading to inefficient random walk behaviour, a low acceptance rate means many proposed moves are being rejected, maybe due to ambitious steps. The acceptance rate is tuned to optimise the exploration of the parameter space and computational efficiency. An outcome from a construction of a universal optimisation criteria for tuning the step size shows the acceptance probability can be relaxed to $0.6 \leq a \geq 0.9$ [5]. As such the acceptance probability used is 0.8. During the tuning phase the mass matrix, used to define the kinetic energy part of the Hamiltonian, is adapted based on the sample covariance. This matrix aids scaling ensuring sensible steps are being taken and that the correlation of parameters is considered leading to faster convergence. PyMC3 simplifies the MCMC process by automatically initialising and tuning the NUTS sampler. The prior, likelihood and range of parameters is initialised. As well as the following hyper-parameters: samples, tune-in period, number of chains and target acceptance probability. The tune-in period acts as a period for PyMC3 to adjust the leapfrog step and mass matrix to achieve the target acceptance rate, the tune-in period is discarded and helps the chain start sampling effectively from the posterior removing the need for a burn-in period here. The PyMC3 model is defined and then simulated for multiple chains in parallel and then later thinned and assessed for convergence.

Posterior distribution: The posterior distribution in the range $a < \alpha < b$ and $c < \beta < d$ is given by,

$$P(\alpha, \beta|\{x_k\}) = \frac{\mathcal{L}_x(\{x_k\}|\alpha, \beta)\pi(\alpha, \beta)}{Z}, \tag{19}$$

$$P(\alpha, \beta|\{x_k\}) = \frac{1}{Z}\prod_k^n \frac{1}{\pi}\frac{\beta}{\beta^2 + (x-\alpha)^2}\frac{1}{(b-a)(d-c)}. \tag{20}$$

The HMC method works with the unnormalised log-posterior to ensure numerical stability, the evidence term can be omitted as this disappears in the Metropolis-Hastings acceptance (ratio) step,

$$P(\alpha, \beta|\{x_k\}) \propto \log\left(\prod_k^n \frac{1}{\pi}\frac{\beta}{\beta^2 + (x_k-\alpha)^2}\frac{1}{(b-a)(d-c)}\right), \tag{21}$$

$$P(\alpha, \beta|\{x_k\}) \propto n\log(\beta) - n\log(\pi) - \log((b-a)(d-c))) - \sum_k^n \log[\beta^2 + (x_k-\alpha)^2]. \tag{22}$$

Sampling: The posterior distribution was sampled in Python using NUTS provided by PyMC3 using a range of $-5 < \alpha < 5$ and $0 < \beta < 8$. With the following hyper-parameters: $8,000$ tune-in steps, $30,000$ steps for 8 chains run in parallel with a target probability of 80%. Based on the minimum ESS the $\tau = 1.47$, to be conservative the samples were thinned with $\tau = 2$. Leaving a thinned trace of $120,000$ samples in total, with $15,000$ samples per chain across 8 chains. The results below, excluding the trace plot, were plotted using the thinned data.

0.5.1 (i) Joint posterior of α and β:

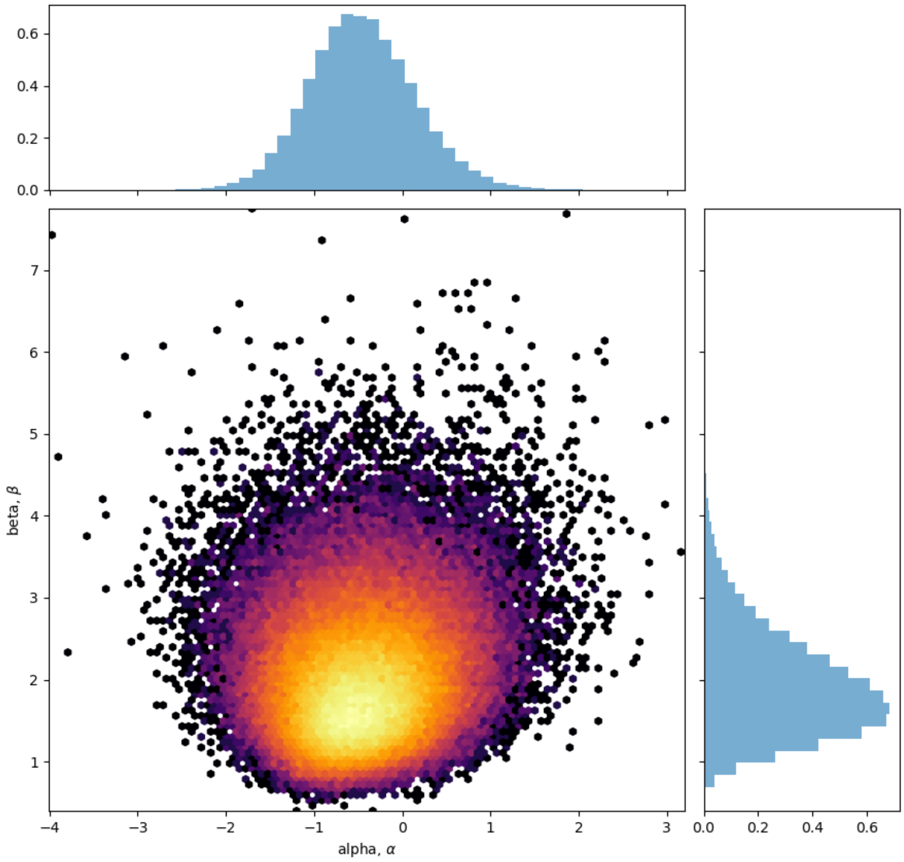

Figure 4: Visualisation of the joint posterior distribution for parameters α and β generated from a NUTS simulation with 8 independent chains, running for 15,000 steps each. The contour plot highlights the density of the samples and the marginal histograms show the distribution of each parameter.

0.5.2 (ii) Marginalised posterior of α and β:

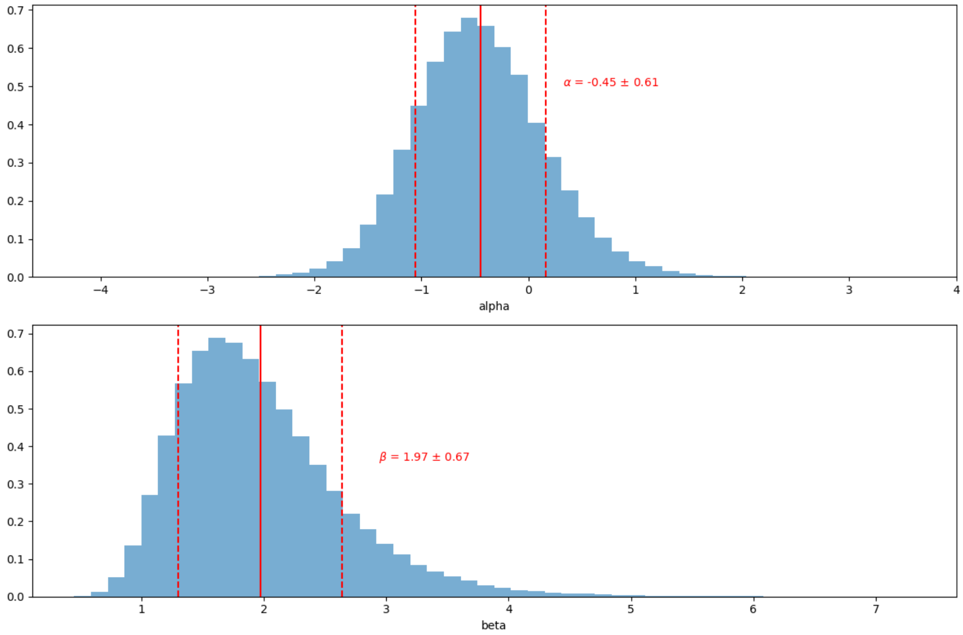

Figure 5: Marginal posterior distributions for parameters α and β from a NUTS algorithm with 8 independent chains, running for 15,000 steps each. The histograms represent the probability density of samples obtained for each parameter, with the solid line showing the mean μ and the dashed lines the standard deviations σ. The flat line represents the uniform prior. The plots reflect the degree of uncertainty in the estimation of each parameter, with α centred around -0.45 ± 0.61 and β around 1.97 ± 0.67.

0.5.3 (iii) Measurements of α and β:

Parameter	Mean	SE of Mean	Std. Dev.	SE of Std. Dev.
α	-0.45	0.00176	0.61	0.00125
β	1.97	0.00193	0.67	0.00137

Table 1: Summary of measurements for μ_α and μ_β from the posterior distribution of lighthouse locations.

0.5.4 (iv) Convergence diagnostic for Markov chain:

Convergence was assessed using trace plots, the autocorrelation rate (τ), the effective sample size (ESS), the Geweke z-score and the Gelmen-Rubin diagnostic ($\hat{R}$).

Trace plots: Trace plots depict the sampled values against the iteration number. Traces plots are a qualitative way of assessing for chain stability. Displays of asymptotic behaviour can be evidence for convergence and indicate the chains are mixing well and sampling from the entire posterior distribution.

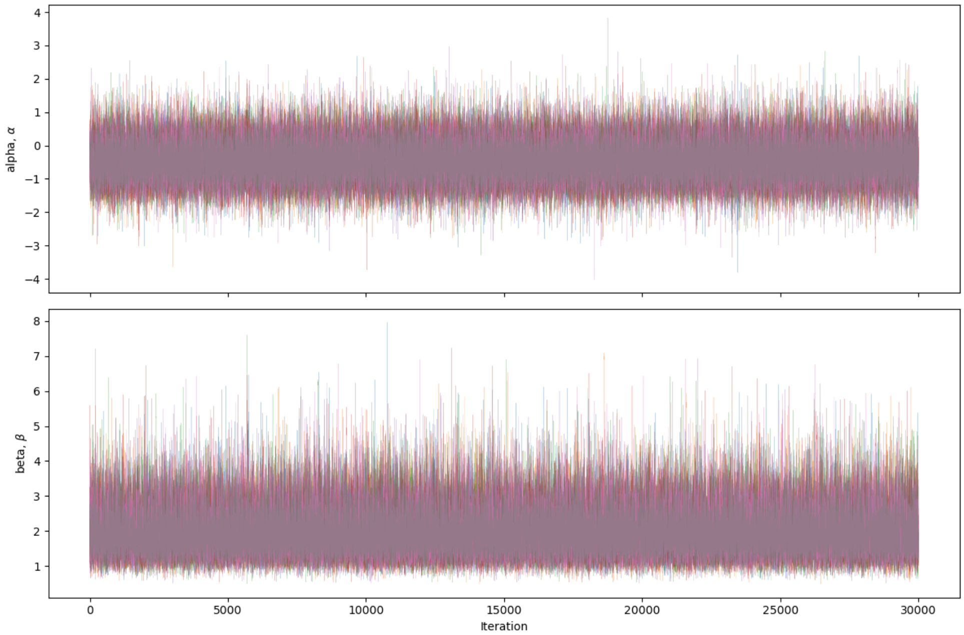

Figure 6: Trace plot displaying the sampling paths of 8 NUTS chains for α and β across 30,000 iterations each, 240,000 in total. Each coloured line represents one of the multiple chains used in the analysis, illustrating the convergence behaviour and mixing of the chains over the iterations. The stable and overlapping traces in the plots suggest good convergence and mixing for both parameters.

Autocorrelation and ESS: Autocorrelation in MCMC sampling is the similarity between successive samples within a chain. Depending on the context obtaining i.i.d samples can be important. The autocorrelation for this chain, calculated from the ESS value was $\tau = 1.47$. To be conservative a value of $\tau = 2$ was used, meaning every other sample is independent this value was used to thin the chain. For this model that meant $200\mu s$ per i.i.d sample.

Geweke: The trace plot is a visual measure, the autocorrelation provides information about the relationship of points in the chain. The Geweke z-score is a more formal measure of convergence. The Geweke z-score compares the mean and variance of segments of the chain from the beginning to the end [6].

$$z = \frac{\bar{\theta}_a - \bar{\theta}_b}{\sqrt{\mathrm{Var}(\bar{\theta}_a) + \mathrm{Var}(\bar{\theta}_b)}}, \tag{23}$$

where a is the early interval and b the late interval. The z-scores can provide evidence for convergence, if. the majority of points fall within 2 standard deviations of zero.

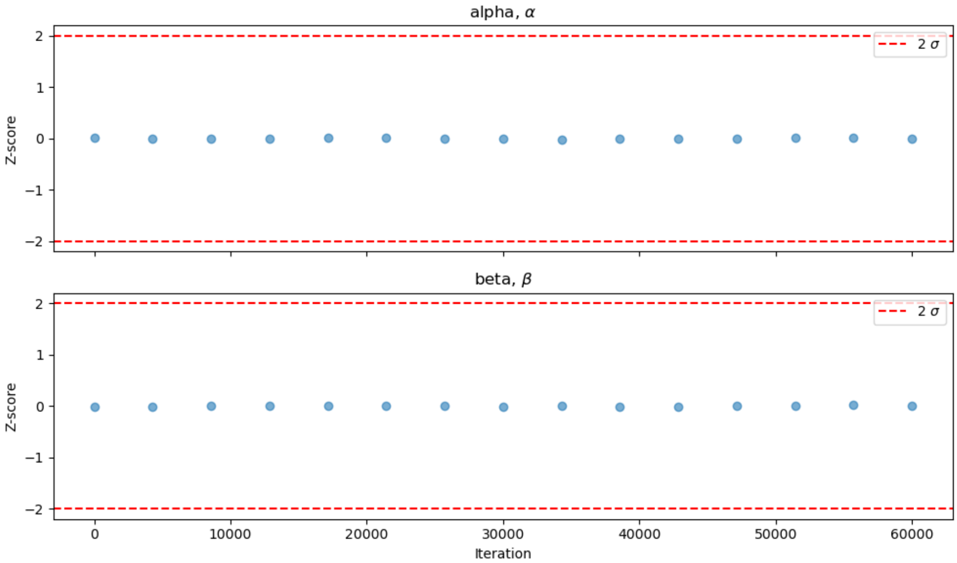

Figure 7: Geweke diagnostic plots for MCMC convergence of parameters. Z-scores for α and β are well-contained within the ±2 standard deviation bounds, suggesting that the chains have converged and that the sampling process is reliable

The Gelmen-Rubin diagnostic: The Gelmen-Rubin method analyses the between-chain variance and within-chain variance to assess convergence. It measures whether the chains have converged to a common distribution. An $\hat{R}$ value close to 1 suggests convergence. Both α and β had $\hat{R}$ values of 1.0.

The trace plot provided qualitative evidence for mixing and asymptotic behaviour, τ gives insight into the amount of information the chain contains. The Geweke z-score and $\hat{R}$ were formal methods to assess convergence, this analysis suggests this chain has converged.

0.6 (vi) Selecting a Suitable Prior for I_0

The Pareto distribution was selected as it aligns with the positive nature of I_0 and its heavy tails serve as an ignorance prior for the parameter space. This distribution is also consistent with the inverse-square law behaviour of I_0, providing a broad and flexible examination of the sample space.

$$f(x|\alpha_p, m) = \begin{cases} \frac{\alpha_p m^{\alpha_p}}{x^{\alpha_p+1}} & \text{for } x \geq m \\ 0 & \text{for } x < m. \end{cases} \tag{24}$$

0.7 (vii) Lighthouse Location Using Flash Locations and Intensities

The new intensity likelihood term ($\mathcal{L}_I$) follows the log-normal distribution with uncertainty $\sigma = 1$. The expectation of the log-intensity is $\mu = log(I_0/d^2)$ where I_0 is the absolute intensity of the lighthouse and $d^2 = \beta^2 + (x - \alpha)^2$,

$$\mathcal{L}_I = \frac{\exp\left(\frac{-(\log I - \mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi\sigma^2}}. \tag{25}$$

Posterior distribution: Due to the Independence of flash locations (x) and intensity (I) the new posterior becomes,

$$P(\alpha, \beta, I_0|\{x_k\}, \{I_k\}) = \frac{\mathcal{L}_x(\{x_k\}|\alpha, \beta)\mathcal{L}_I(\{I_k\}|I_0)\pi_x(\alpha, \beta)\pi_I(I_0)}{Z_x Z_I}, \tag{26}$$

Sampling: This log posterior distribution was sampled in Python using NUTS provided by PyMC3 using a range of $-5 < \alpha < 5$ and $0 < \beta < 8$ for the uniform priors and $\alpha_p = 2$ and $m = 0.01$ for the Pareto distribution. The parameters were to define the I_0 as positive and to strike a balance between the tail being too heavy and too light. The following hyper-parameters: $8,000$ tune-in steps, $30,000$ steps for 8 chains run in parallel with a target probability of 80%. Based on the minimum ESS the $\tau = 1.88$, to be conservative the samples were thinned with $\tau = 2$. Leaving a thinned trace of $120,000$ samples in total, with $15,000$ samples per chain across 8 chains. The results below, excluding the trace plot, were plotted using the thinned data.

0.7.1 (i) Joint posterior of α, β and I_0:

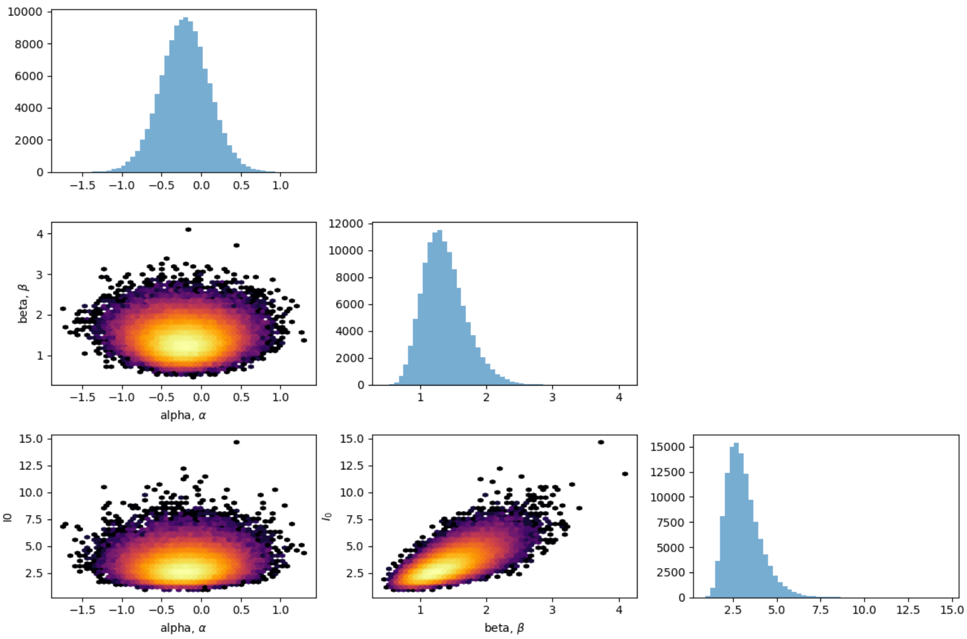

Figure 8: Visualisation of the joint posterior distribution for parameters α, β and I_0 generated from NUTS simulation with 8 independent chains, running for 15,000 steps each. The contour plots highlight the density of the with respect to different parameters, the marginal histograms show the individual distribution of each parameter.

0.7.2 (ii) Marginalised posterior of α, β and I_0:

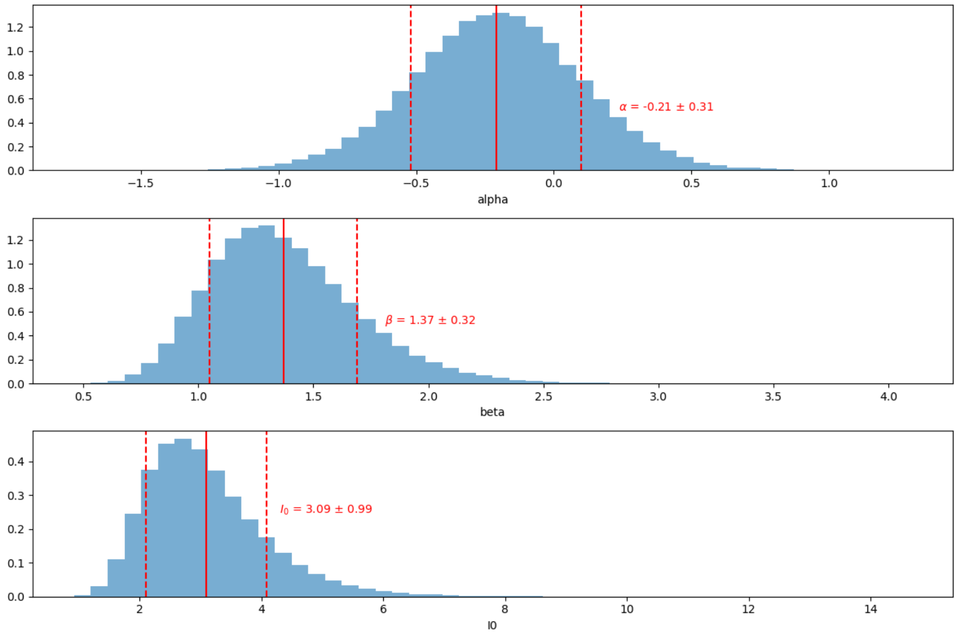

Figure 9: Marginal posterior distributions for parameters α, β and I_0 generated from NUTS simulation with 8 independent chains, running for 15,000 steps each. The histograms represent the probability density of samples obtained for each parameter, with the solid line showing the mean μ and the dashed lines the standard deviations σ. With α centered around -0.21 ± 0.31, β around 1.37 ± 0.32 and I_0 at 3.09 ± 0.99.

0.7.3 (iii) Measurements of α, β and I_0:

Parameter	Mean	SE of Mean	Std. Dev.	SE of Std. Dev.
α	-0.21	0.000895	0.31	0.000633
β	1.37	0.000924	0.32	0.000653
I_0	3.09	0.002858	0.99	0.002021

Table 2: Summary of measurements for parameters from the posterior distribution.

0.7.4 (iv) Convergence diagnostic for Markov chain:

Convergence was assessed using the trace plots, Gelmen-Rubin diagnostic, acceptance rate, the autocorrelation rate (τ) and the effective sample size (ESS).

Trace plots: The trace plot displays asymptotic behaviour and is evidence that the chains are mixing well and sampling from the entire posterior distribution.

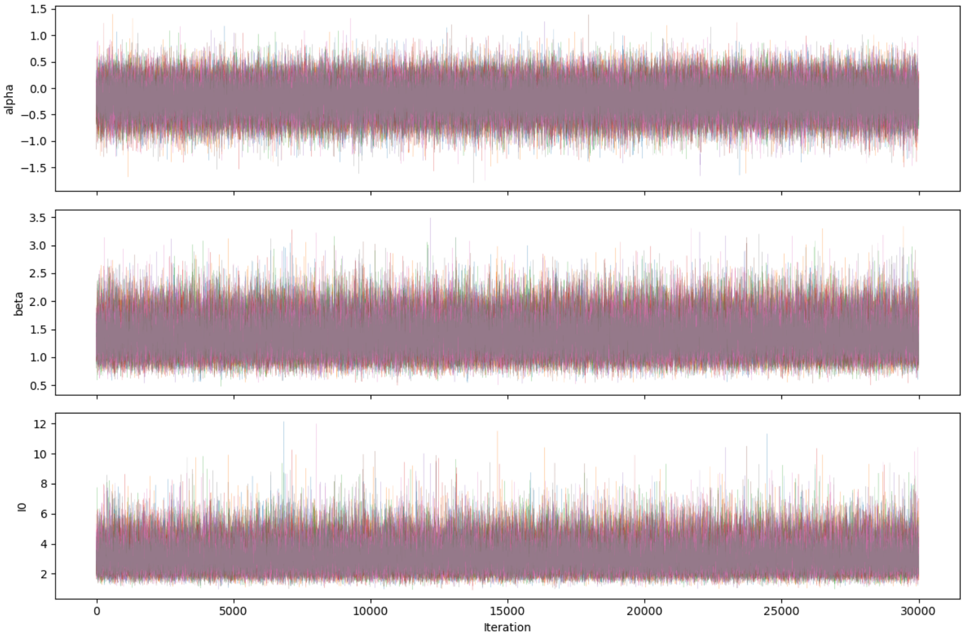

Figure 10: Trace plot displaying the sampling paths of 8 HMC chains for α, β and I_0 across 30,000 iterations each, 240,000 in total. Each coloured line represents one of the multiple chains used in the analysis, illustrating the convergence behaviour and mixing of the chains over the iterations. The stable and overlapping traces in the plots suggest good convergence and mixing for both parameters."

Autocorrelation and ESS: The autocorrelation for this chain, calculated from the ESS value was $\tau = 1.88$. To be conservative a value of $\tau = 2$ was used This value was used to thin the chain. This model took $200\mu s$ per i.i.d sample.

The Geweke and Gelmen-Rubin diagnostic: The $\hat{R}$ values of α, β and I_0 all had values of 1.0 implying convergence. Similarly, the Geweke diagnosis suggests the samples have converged to a stationary distribution.

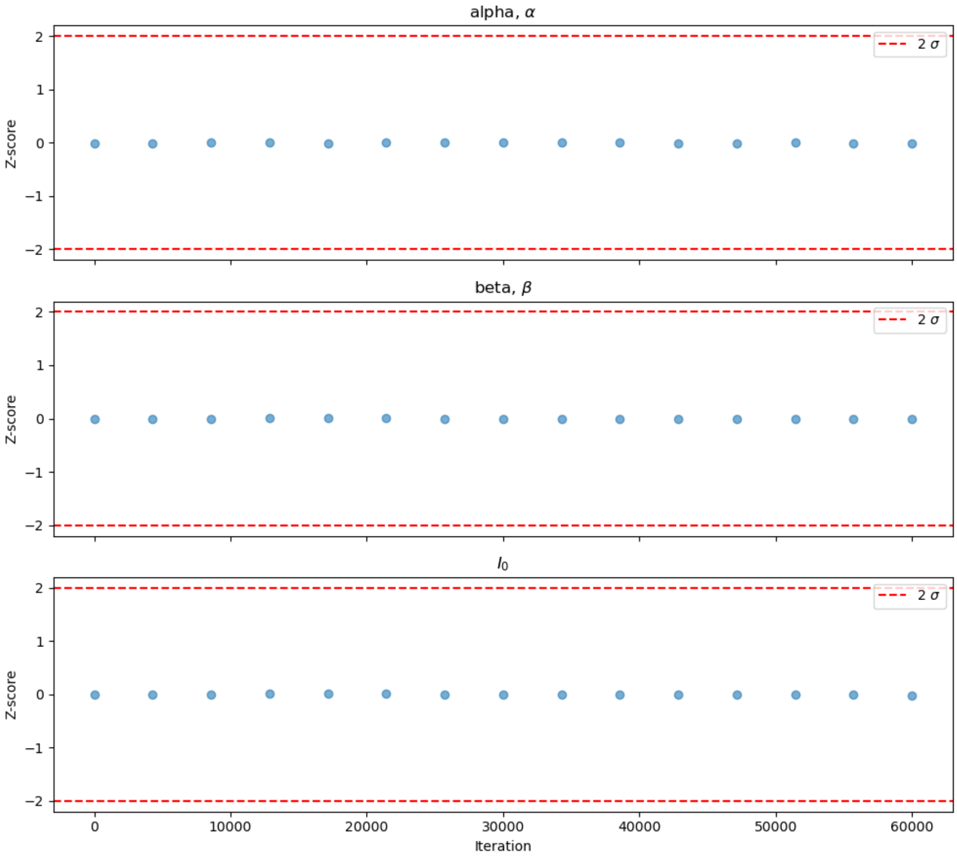

Figure 11: Geweke diagnostic plots for MCMC convergence of parameters. Z-scores for α, β, and I_0 are well-contained within the ± 2 standard deviation bounds, suggesting that the chains have converged and that the sampling process is reliable.

The trace plot suggests the samples mixed well, τ gives insight into the amount of information the chain contains. The Geweke z-score and $\hat{R}$ provided evidence that this chain has converged.

0.8 Comparison of models

Flash locations model The flash locations model used a uniform prior and a Cauchy distributed likelihood. Following NUTS sampling and chain thinning, 120,000 i.i.d samples gave a mean alpha value of $\mu_\alpha = -0.45$ and standard deviation of $\sigma_\alpha = 0.61$.

Flash locations and intensities model The flash locations and intensities model used a uniform prior and a Cauchy distributed likelihood to represent the flash locations and a Pareto prior and log-normal likelihood to represent the intensities. Following NUTS sampling and chain thinning, 120,000 i.i.d samples gave a mean alpha value of $\mu_\alpha = -0.21$ and standard deviation of $\sigma_\alpha = 0.31$.

Comparison of α: The samples from the second model are grouped closer together, therefore providing a more precise estimate of α. This estimate of α is roughly twice as close as the first model. The added complexity of the Monte Carlo simulation resulted in an improved measurement of double precision.

Bibliography

[1] Duane, Kennedy, Pendleton & Roweth (1987) “Hybrid Monte Carlo”, Physics Letters B, 195 (2) 216–222 doi:10.1016/0370-2693(87)91197-X.

[2] Radford Neal. MCMC Using Hamiltonian Dynamics. Handbook of Markov Chain Monte Carlo, 2011. https://arxiv.org/abs/1206.1901

[3] Hoffman & Gelman (2014) “The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo”, Journal of Machine Learning Research, 15 1593-1623,

[4] Salvatier J., Wiecki T.V., Fonnesbeck C. (2016) Probabilistic programming in Python using PyMC3. PeerJ Computer Science 2:e55 DOI: 10.7717/peerj-cs.55.

[5] Betancourt, M., Byrne, S., & Girolami, M. (2014). Optimizing The Integrator Step Size for Hamiltonian Monte Carlo. arXiv preprint arXiv:1411.6669. Retrieved from https://arxiv.org/pdf/1411.6669.pdf

[6] J. Geweke. Evaluating the accuracy of sampling-based approaches to calculating posterior moments. In Bernardo et al. 1992, pages 169–193.

0.9 Appendix

0.9.1 Posterior Distribution of Flash Locations

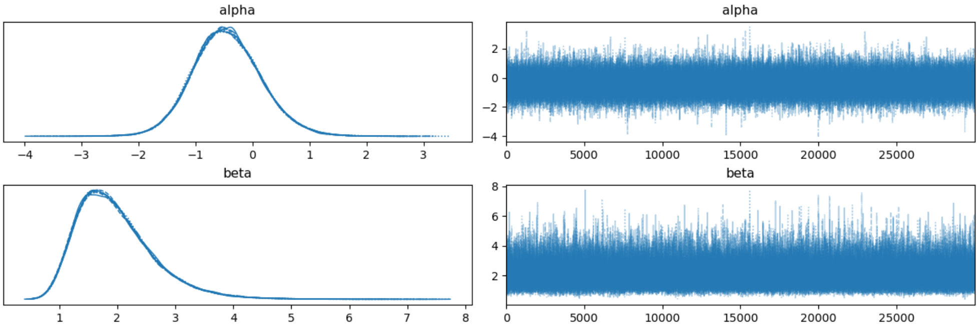

Figure 12: Posterior distribution trace plot before thinning.

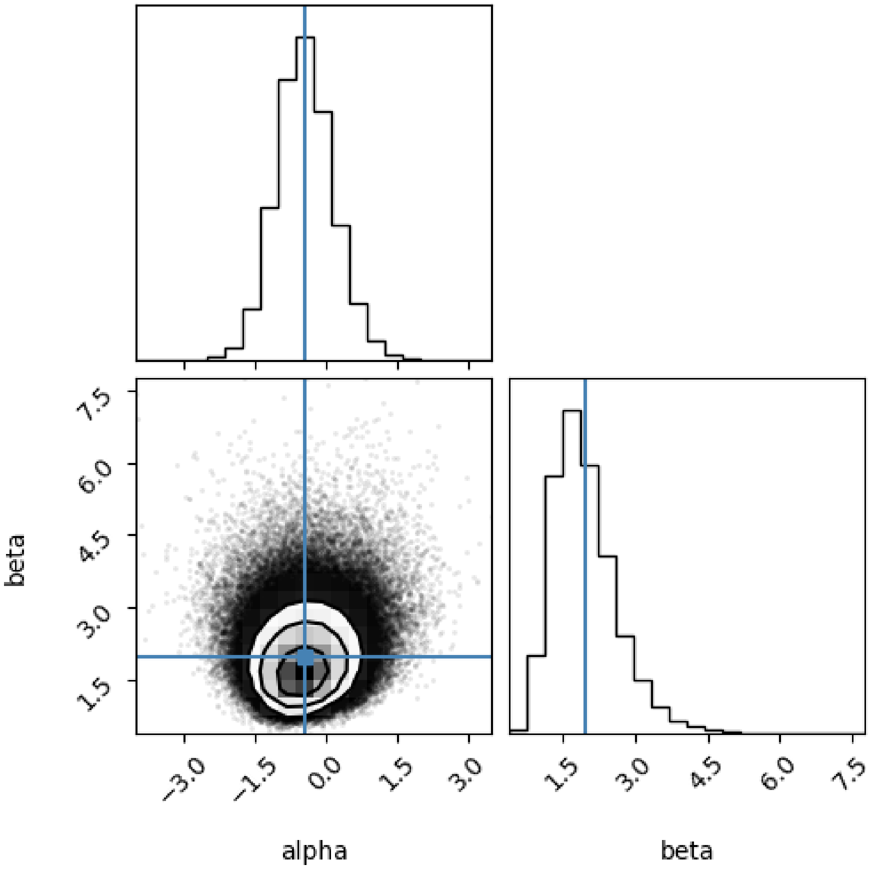

Figure 13: Posterior distribution corner plot before thinning.

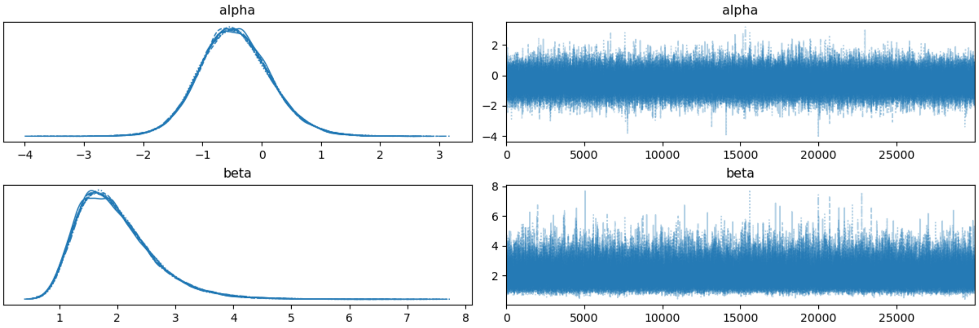

Figure 14: Posterior distribution trace plot after thinning.

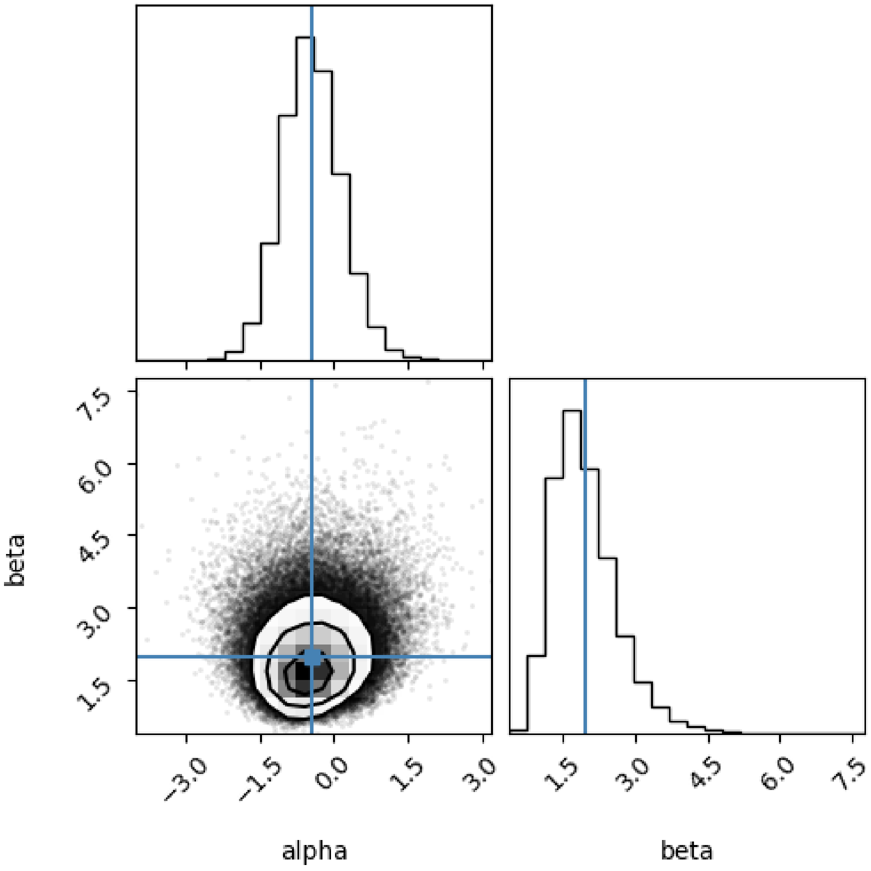

Figure 15: Posterior distribution corner plot after thinning.

Param.	Mean	SD	HDI 3%	HDI 97%	ESS Mean	ESS SD	ESS Bulk	ESS Tail	$\hat{R}$
α	-0.45	0.61	-1.58	0.70	166788.85	124861.09	171269.59	134273.38	1.0
β	1.97	0.67	0.87	3.21	148982.67	130062.48	162991.04	141076.53	1.0

Table 3: Summary of posterior distributions for parameters α and β before thinning.

Param.	Mean	SD	HDI 3%	HDI 97%	ESS Mean	ESS SD	ESS Bulk	ESS Tail	$\hat{R}$
α	-0.45	0.61	-1.59	0.69	110291.22	95001.43	111345.45	99807.36	1.0
β	1.97	0.67	0.87	3.21	100655.84	94755.39	105097.43	98949.41	1.0

Table 4: Summary of posterior distributions for parameters α and β after thinning.

0.9.2 Posterior Distribution of Flash Locations and Intensities

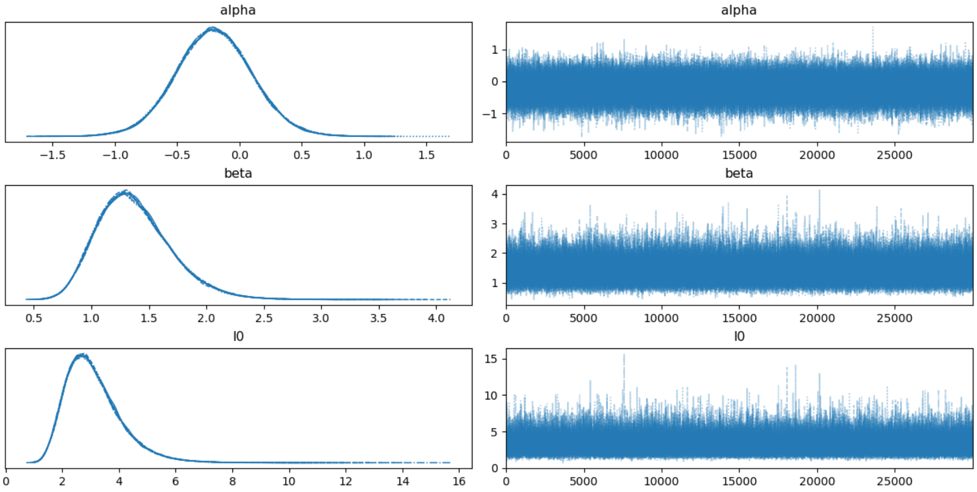

Figure 16: Posterior distribution trace plot before thinning.

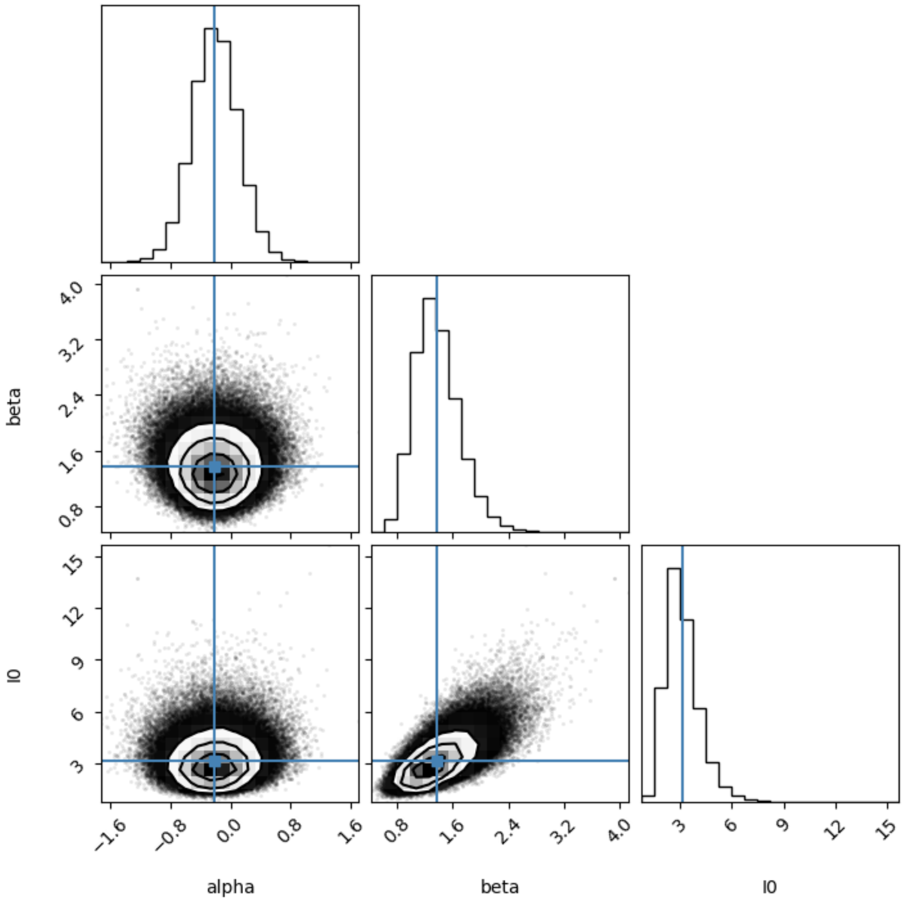

Figure 17: Posterior distribution corner plot before thinning.

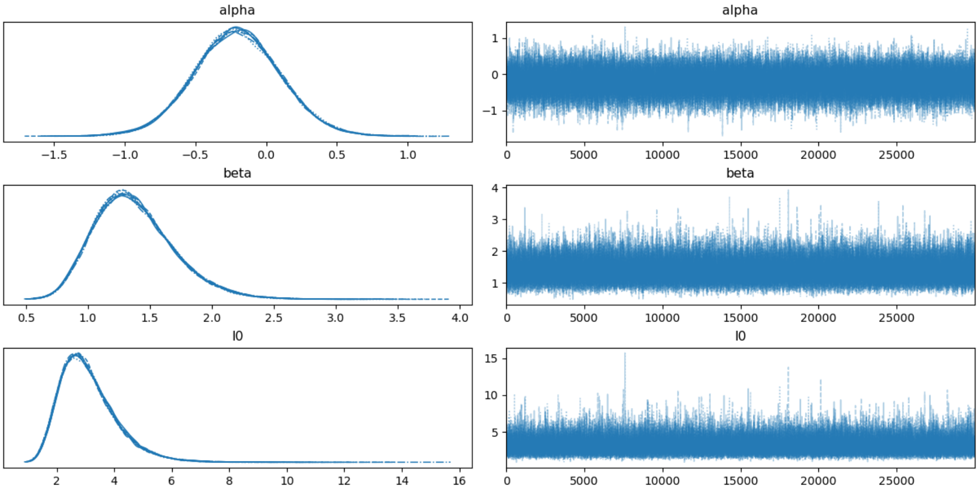

Figure 18: Posterior distribution trace plot after thinning.

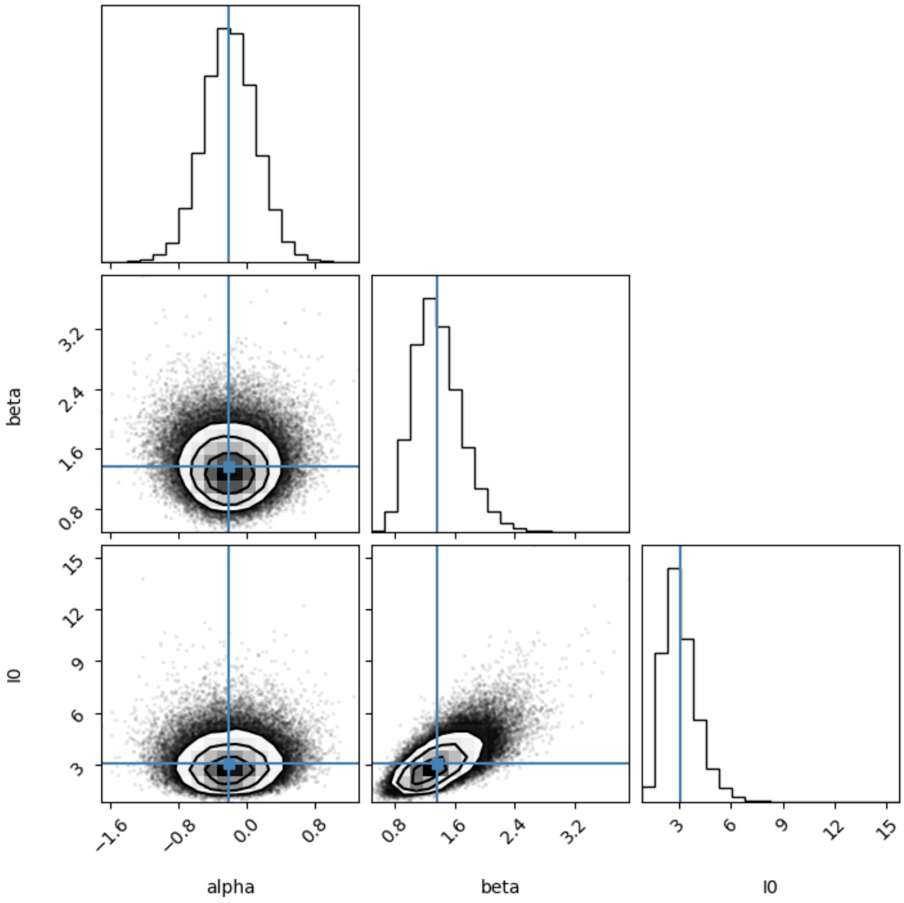

Figure 19: Posterior distribution corner plot after thinning.

Param.	Mean	SD	HDI 3%	HDI 97%	ESS Mean	ESS SD	ESS Bulk	ESS Tail	$\hat{R}$
α	-0.21	0.31	-0.78	0.38	175092.96	132197.10	176398.99	151413.73	1.0
β	1.37	0.32	0.80	1.98	117491.24	114748.52	120743.15	141039.56	1.0
I_0	3.11	1.00	1.48	4.94	116208.35	115189.52	119653.19	143876.74	1.0

Table 5: Summary of posterior distributions for parameters α, β, and I_0 before thinning.

Param.	Mean	SD	HDI 3%	HDI 97%	ESS Mean	ESS SD	ESS Bulk	ESS Tail	$\hat{R}$
α	-0.21	0.31	-0.79	0.37	78367.60	74452.13	78434.51	75323.16	1.0
β	1.37	0.32	0.80	1.98	72035.82	71327.14	72716.15	74571.56	1.0
I_0	3.10	1.00	1.44	4.91	71126.40	71126.40	72148.63	75808.02	1.0

Table 6: Summary of posterior distributions for parameters α, β, and I_0 after thinning.

0.9.3 Auto-generative tools

This project has utilised auto-generative tools in the development of documentation that is compatible with auto-documentation tools, latex formatting and the development of plotting functions.

Example prompts used for this project:

- Generate doc-strings in NumPy format for this function.
- Generate this equation in Latex
- Generate Latex code for a subplot.
- Generate Latex code for a table.
- Generate Python code for a 3 by 1 subplot.