Statistical Methods Coursework: S1

Lucas Curtin

December 18, 2024

Abstract

This report analyses multi-dimensional probability density functions (PDFs) and their applications in statistical data modelling, focusing on signal and background distributions. Key PDFs, including the Truncated Crystal Ball and Truncated Exponential distributions, were derived, normalised, and implemented. A synthetic dataset was generated using vectorised rejection sampling, and an extended maximum likelihood (EML) fit was used to estimate parameters. Bias and uncertainty in parameter estimation were assessed using parametric bootstrapping and sWeights, with results benchmarked across sample sizes. The sWeights method reduced bias for smaller datasets, while bootstrapping provided robust validation for larger or complex datasets. These findings highlight efficient approaches to parameter estimation and bias reduction.

Table of Contents

1 Mathematical Derivation and Validation of PDFs

The probability density functions (PDFs) were implemented based on the provided coursework specifications. The following distributions were defined:

- $g_s(X)$: A Truncated Crystal Ball distribution, designed to capture a combination of Gaussian and power-law behaviour, with parameters including μ (mean), σ (standard deviation), β (transition point), and m (tail exponent). The truncation was applied to the domain $[0, 5]$.
- $h_s(Y)$: A Truncated Exponential decay function, expressed as $h_s(Y) = \lambda e^{-\lambda Y}$, normalised over the interval $[0, 10]$.
- $g_b(X)$: A Uniform distribution, constant over the domain $[0, 5]$, ensuring equal probability density within the specified range.
- $h_b(Y)$: A Truncated Normal distribution, centred at $\mu_b = 0$ with standard deviation $\sigma_b = 2.5$, truncated to the interval $[0, 10]$.

The Crystal Ball distribution is the most complicated PDF so it is analytically normalised below, which is then used to help normalise the truncated version.

1.1 Derivation of the Normalisation Constant

The Crystal Ball distribution combines a Gaussian core with a power-law tail. The normalisation constant ensures that the probability density function integrates to 1 over its domain. For a random variable X, the probability density function is defined as:

$$p(X; \mu, \sigma, \beta, m) = N \cdot \begin{cases} \exp\left(-\frac{Z^2}{2}\right) & \text{if } Z > -\beta, \\ \left(\frac{m}{\beta}\right)^m \exp\left(-\frac{\beta^2}{2}\right) \left(\frac{m}{\beta} - \beta - Z\right)^{-m} & \text{if } Z \leq -\beta, \end{cases}$$

where $Z = \frac{X-\mu}{\sigma}$ and N is the normalisation constant. N^{-1} can be determined by solving the below condition:

$$\int_{-\infty}^{\infty} p(X; \mu, \sigma, \beta, m)\, dX = 1.$$

1.1.1 Splitting the Integral

The integral is first split into two parts:

$$\int_{-\infty}^{\infty} p(X)\, dX = \int_{-\infty}^{\mu-\beta\sigma} p(X)\, dX + \int_{\mu-\beta\sigma}^{\infty} p(X)\, dX.$$

Substituting the definitions of $p(X)$ for each region:

$$I_1 = \int_{-\infty}^{\mu-\beta\sigma} \left(\frac{m}{\beta}\right)^m \exp\left(-\frac{\beta^2}{2}\right) \left(\frac{m}{\beta} - \beta - \frac{X-\mu}{\sigma}\right)^{-m} dX,$$

$$I_2 = \int_{\mu-\beta\sigma}^{\infty} \exp\left(-\frac{\left(\frac{X-\mu}{\sigma}\right)^2}{2}\right) dX.$$

Rewriting these integrals in terms of Z, where $Z = \frac{X-\mu}{\sigma}$ and $dX = \sigma\, dZ$:

$$I_1 = \sigma \int_{-\infty}^{-\beta} \left(\frac{m}{\beta}\right)^m \exp\left(-\frac{\beta^2}{2}\right) \left(\frac{m}{\beta} - \beta - Z\right)^{-m} dZ,$$
$$I_2 = \sigma \int_{-\beta}^{\infty} \exp\left(-\frac{Z^2}{2}\right) dZ.$$

1.1.2 Simplifying I_1

Starting with the integral:

$$I_1 = \sigma \int_{-\infty}^{-\beta} \left(\frac{m}{\beta}\right)^m \exp\left(-\frac{\beta^2}{2}\right) \left(\frac{m}{\beta} - \beta - Z\right)^{-m} dZ.$$

Let $u = \frac{m}{\beta} - \beta - Z$, so $du = -dZ$. The limits transform as follows:

- When $Z = -\infty$, $u \to \infty$.
- When $Z = -\beta$, $u = \frac{m}{\beta}$.

The integral becomes:

$$I_1 = \sigma \left(\frac{m}{\beta}\right)^m \exp\left(-\frac{\beta^2}{2}\right) \int_{\infty}^{\frac{m}{\beta}} u^{-m}(-du).$$

Simplifying:

$$I_1 = \sigma \left(\frac{m}{\beta}\right)^m \exp\left(-\frac{\beta^2}{2}\right) \int_{\frac{m}{\beta}}^{\infty} u^{-m}\, du.$$

The integral of u^{-m} is:

$$\int u^{-m}\, du = \frac{u^{1-m}}{1-m} \quad (\text{valid for } m > 1).$$

Evaluating the definite integral:

$$\int_{\frac{m}{\beta}}^{\infty} u^{-m}\, du = \left[\frac{u^{1-m}}{1-m}\right]_{\frac{m}{\beta}}^{\infty}.$$

Substituting the limits:

$$\int_{\frac{m}{\beta}}^{\infty} u^{-m}\, du = \lim_{u\to\infty} \frac{u^{1-m}}{1-m} - \frac{\left(\frac{m}{\beta}\right)^{1-m}}{1-m}.$$

Since $m > 1$, the term $\lim_{u\to\infty} u^{1-m}$ approaches 0, because $1 - m < 0$ makes the exponent negative. Thus:

$$\int_{\frac{m}{\beta}}^{\infty} u^{-m}\, du = 0 - \frac{\left(\frac{m}{\beta}\right)^{1-m}}{1-m}.$$

Simplifying:

$$\int_{\frac{m}{\beta}}^{\infty} u^{-m}\, du = \frac{\left(\frac{m}{\beta}\right)^{1-m}}{m-1}.$$

Substituting back:

$$I_1 = \sigma \left(\frac{m}{\beta}\right)^m \exp\left(-\frac{\beta^2}{2}\right) \cdot \frac{\left(\frac{m}{\beta}\right)^{1-m}}{m-1}.$$

Simplifying:

$$I_1 = \sigma \frac{m \exp\left(-\frac{\beta^2}{2}\right)}{\beta(m-1)}.$$

1.1.3 Simplifying I_2

Let $Z = \frac{X-\mu}{\sigma}$, so $dX = \sigma\, dZ$. Then:

$$I_2 = \sigma \int_{-\beta}^{\infty} \exp\left(-\frac{Z^2}{2}\right) dZ = \sigma\sqrt{2\pi}\Phi(\beta),$$

where $\Phi(\beta)$ is the cumulative density function of the standard normal distribution.

1.1.4 Combining Results

Combining I_1 and I_2:

$$N^{-1} = \sigma \left(\frac{m}{\beta(m-1)} e^{-\beta^2/2} + \sqrt{2\pi}\Phi(\beta)\right)$$

1.2 Implementation and Validation of PDFs

To ensure correctness, each PDF was implemented as a separate function, parameterised by its defining constants and domain bounds. The truncation process involved setting probability density values to zero outside the specified bounds, and renormalising the distributions to ensure they integrate to unity over their respective domains. This was done analytically, and then confirmed numerically. For example, the Truncated Crystal Ball distribution $g_s(X)$, the normalisation constant was calculated as the sum of contributions from the Gaussian and tail components. The Gaussian contribution was integrated as:

$$C_{\text{gauss}} = \sqrt{2\pi}\sigma \left[\Phi\left(\frac{x_{\max}-\mu}{\sigma}\right) - \Phi\left(\max\left(\frac{x_{\min}-\mu}{\sigma}, -\beta\right)\right)\right],$$

where $\Phi(z)$ is the cumulative distribution function (CDF) of the standard normal distribution. The tail contribution was evaluated as:

$$C_{\text{tail}} = \frac{(m/\beta)^m e^{-\beta^2/2}}{m-1} \left[\left(\frac{m}{\beta} - \beta - \frac{x_{\min}-\mu}{\sigma}\right)^{1-m} - \left(\frac{m}{\beta} - \beta - \frac{x_{\max}-\mu}{\sigma}\right)^{1-m}\right],$$

for $x_{\min} \leq -\beta\sigma + \mu$. The total normalisation constant C was then:

$$C = \sigma(C_{\text{gauss}} + C_{\text{tail}}).$$

As for the Truncated Exponential distribution $h_s(Y)$, the integral was computed as:

$$\int_{y_{\min}}^{y_{\max}} \lambda e^{-\lambda y}\, dy = \left[e^{-\lambda y_{\min}} - e^{-\lambda y_{\max}}\right],$$

The implementation of these pdfs utilised `Numba JIT` (just-in-time)[1] compilation to optimise performance, particularly for computationally intensive operations such as evaluating the truncated normal and Crystal Ball distributions. Visualisations of the individual distributions, as shown in Figure 1, provide a clear depiction of the behaviour of each PDF across their respective domains:

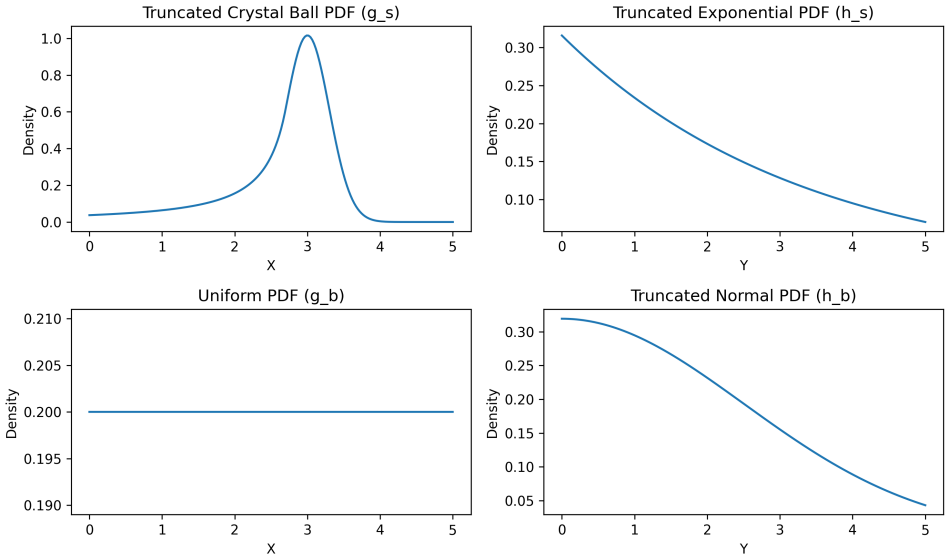

Figure 1: Individual probability density functions for X and Y. Top row: $g_s(X)$ and $h_s(Y)$. Bottom row: $g_b(X)$ and $h_b(Y)$.

1.3 PDF Projections

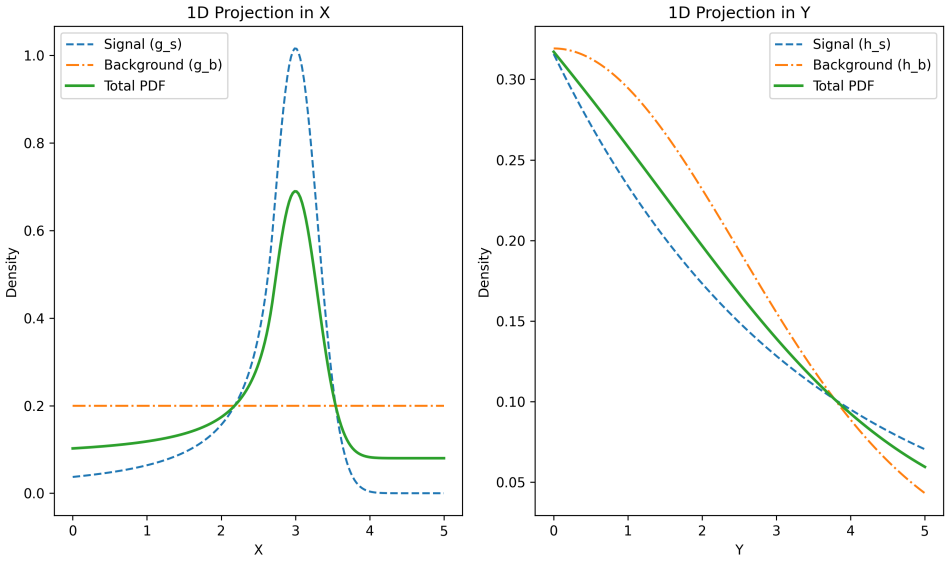

Figure 2: These plots overlay the signal, background, and total distributions, clearly showing the relative contributions of the components. The scaling of each component by f ensures that the combined distribution accurately reflects the overall mixture of signal and background events.

The total probability density function (PDF) combines contributions from both signal and background components. It is expressed as:

$$f(X,Y) = f \cdot s(X,Y) + (1-f) \cdot b(X,Y),$$

where f is the signal fraction, $s(X,Y)$ is the signal PDF, and $b(X,Y)$ is the background PDF. Each of these components can be further factorised into their respective marginal distributions and are illustrated in Figure 3.

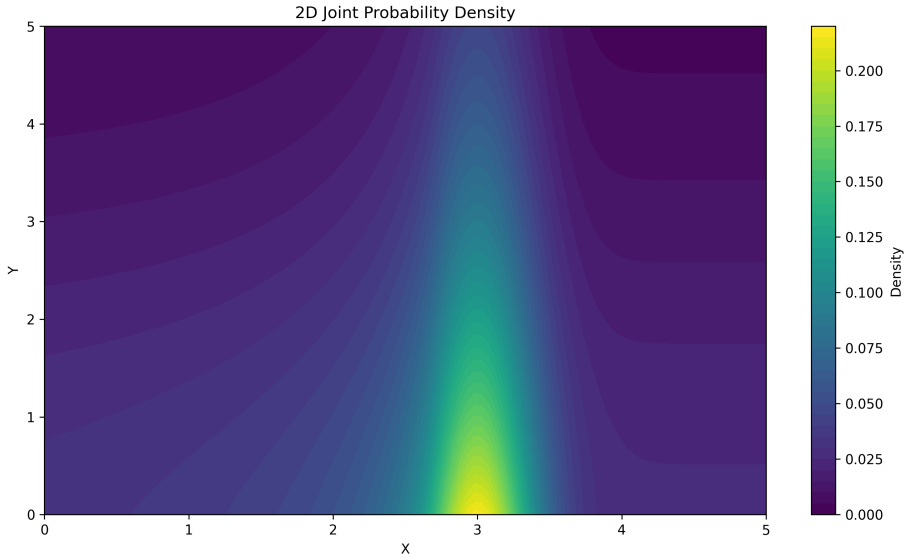

Figure 3: Two-dimensional joint PDF, highlighting contributions from signal and background.

$$f(X,Y) = f \cdot g_s(X) \cdot h_s(Y) + (1-f) \cdot g_b(X) \cdot h_b(Y).$$

- $g_s(X)$ and $g_b(X)$ are the marginal distributions of X for the signal and background, respectively.
- $h_s(Y)$ and $h_b(Y)$ are the marginal distributions of Y for the signal and background, respectively.

2 Data Generation and Parameter Estimation

The parameter estimation process begins by generating a synthetic dataset using vectorised rejection sampling. This dataset is then used to estimate the parameters of the total probability density function (PDF) via an extended maximum likelihood (EML) fit. The key steps involved are outlined below.

2.1 Data Generation via Vectorised Rejection Sampling

The first step in the analysis involves generating a synthetic dataset through vectorised rejection sampling. The process is as follows:

- Random samples for the variables x and y are generated within the specified bounds $\text{BOUNDS}_X = [0, 5]$ and $\text{BOUNDS}_Y = [0, 10]$. These samples are drawn from uniform distributions over the corresponding intervals.
- To ensure a sufficient number of candidate events for acceptance, the number of generated samples is set to ten times the desired number of events, i.e., $10 \times n_{\text{events}}$.
- For each generated sample, a uniform random variable u is drawn. This variable is used to determine whether the sample should be accepted, based on the rejection criterion.

- The total PDF, which combines contributions from both signal and background PDFs, is calculated for each pair of x and y. The fraction f represents the proportion of signal events in the total distribution, while $1 - f$ accounts for the background events.

- Each sample is subjected to the rejection criterion:

$$u \cdot P_{\max} \leq P(x, y; \theta),$$

  where $P_{\max}$ is the maximum value of the PDF in the sampling region, and $P(x, y; \theta)$ is the value of the total PDF for the current values of x and y.

- Accepted samples are collected, and the process continues until the desired number of events is reached. The final arrays of accepted x and y values are returned once the required sample size is achieved.

2.2 Extended Maximum Likelihood Fit

Once the synthetic dataset has been generated, the next step is to perform parameter estimation using an extended maximum likelihood (EML) fit.

- The negative log-likelihood (NLL) function is defined for the observed x and y values, based on the total PDF evaluated at each data point:

$$-\ln \mathcal{L}(\theta) = -\sum_{i=1}^{N} \ln P(x_i, y_i; \theta),$$

  where N represents the total number of events in the dataset.

- The extended NLL function is constructed to incorporate the total number of events, n_{total}, directly as a parameter to be estimated. The function is minimised using the `Minuit`[2] optimisation package.

- Constraints are applied to ensure that parameters remain within physically meaningful ranges, such as $\beta \geq 0$ and $m \geq 1$.

- After minimisation, the Hessian matrix is used to estimate the covariance and uncertainties of the fitted parameters.

This procedure allows for the simultaneous estimation of the shape parameters and the total number of events, n_{total}, with associated uncertainties derived from the fit.

2.3 Results and Comparison of Actual and Estimated Parameters

The actual values used to generate the synthetic dataset are compared with the estimates obtained from the extended maximum likelihood fit. To aid comparison, the value of n_{total} is normalised by the total number of events, n_{events}, since it represents the overall scale of the dataset. The other parameters (μ, σ, β, m, f, λ, μ_b, and σ_b) are presented as-is all in Figure 4.

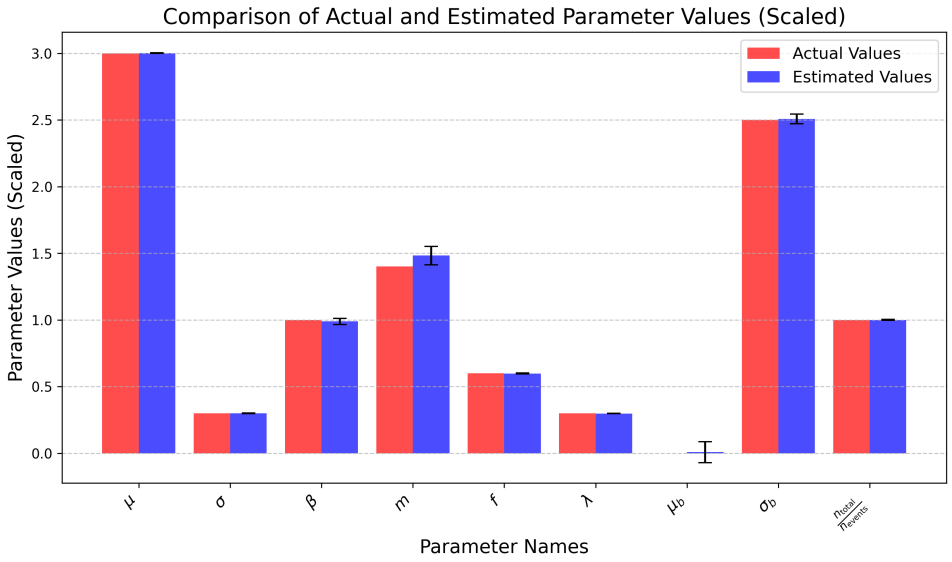

Figure 4: Comparison of actual and estimated parameter values. Error bars represent the uncertainties in the parameter estimates obtained from the fit. The value of n_{total} is scaled by the total number of events for normalisation.

2.4 Benchmarking of Function Execution Times

To evaluate the performance of the key functions involved in this analysis, a series of timing benchmarks were conducted. These benchmarks focused on three primary operations: generating random numbers from a normal distribution, generating synthetic data according to the total probability distribution, and performing the fitting procedure. The benchmarks were repeated over 100 runs, divided into 10 batches, to provide a reliable estimate of uncertainty.

The parameters used for the benchmarking are as follows:

- **Total number of runs**: 100 iterations.
- **Number of batches for uncertainty estimation**: 10 batches.
- **Number of runs per batch**: 10 runs (i.e., 100 total runs divided into 10 batches).

For each of the three operations, the time taken to complete the task was measured over the specified number of runs. The results were then averaged, and the standard deviation was computed for each task. The uncertainty in the timings was estimated by calculating the relative uncertainties, which provide insight into the variability of each operations execution time across different batches. The operations benchmarked include:

1. **Generation of random numbers (`np.random.normal`)**: The time taken to generate random numbers from a normal distribution, a fundamental operation in the analysis.
2. **Sample generation using vectorised rejection sampling (`total_pdf_sampler`)**: The time required to generate synthetic data based on the total probability distribution, which includes both signal and background components.
3. **Fit execution (`perform_fit`)**: The time taken to perform the maximum likelihood fit on the generated data, including parameter estimation and uncertainty propagation.

The results of the benchmarking, averaged over the 100 runs and presented with their respective uncertainties, are summarised below:

- **(i) np.random.normal**: 0.001496 ± 0.000549 seconds.
- **(ii) Sample generation**: 0.176858 ± 0.010790 seconds (relative: 118.24 ± 44.00).
- **(iii) Fit execution**: 7.513818 ± 2.436609 seconds (relative: 5023.42 ± 2460.51).

3 Bias and Uncertainty in λ

This section investigates the bias and uncertainty associated with the decay constant λ, which governs the signal behaviour in the Y variable. Two complementary approaches were employed: a parametric bootstrapping study and an analysis using sWeights. These methods were used to estimate λ across various sample sizes, quantify potential biases, and compare the accuracy of weighted and unweighted fits.

3.1 Parametric Bootstrapping Study

A parametric bootstrapping study was conducted to examine the relationship between the bias and uncertainty of λ and the sample size. Five sample sizes were considered: 500, 1000, 2500, 5000, and 10,000. For each sample size:

1. A dataset was generated using the true total probability density function (PDF). The number of events in each dataset was drawn from a Poisson distribution centred on the specified sample size.
2. The dataset was fitted using an extended maximum likelihood (EML) approach, minimising the negative log-likelihood function to estimate λ and other parameters.
3. The fitted parameters were then used to generate an ensemble of 250 toy datasets, with event counts drawn from the estimated total number of events. Each toy dataset was subsequently fitted to evaluate the stability of the original model.
4. For each sample size, the mean and standard deviation of the fitted λ values from the toy datasets were computed. The mean quantified potential bias, while the standard deviation reflected variability. Additionally, the uncertainties on λ, derived from the optimisation process, were analysed.

3.2 sWeights Analysis

The sWeights method was applied as an alternative to estimate λ with reduced bias. For each sample size:

1. The initial dataset was used to perform an extended maximum likelihood fit in the X dimension, estimating the relative contributions of signal and background components.
2. The sWeights were calculated using the fitted signal and background PDFs and their respective yields. These weights isolated the signal contribution in the orthogonal Y dimension.

3. The signal weights were used in a weighted likelihood fit to the Y variable to estimate λ. This approach accounted for background contamination while using the same original dataset.

4. The weighted estimate of λ and its uncertainty were recorded for comparison against the unweighted estimates.

This process allowed for a direct comparison between the unweighted and weighted estimates of λ across varying sample sizes, providing insights into the performance of both methods under different conditions.

3.3 Results and Visualisations

The results of the parametric bootstrapping study and sWeights analysis are summarised in Figure 5. For each of the five sample sizes considered, the following distributions are presented:

- **Fitted λ Values:** The distribution of λ estimates across the ensemble of bootstrap samples is shown, with vertical lines indicating the true λ value, the mean fitted value, and the weighted λ obtained from the sWeights method. These plots highlight the overall bias and variability of the estimator.

- **Bias Distribution:** The bias, defined as the difference between the fitted and true λ values, is visualised. The mean and standard deviation of the bias distribution are provided for comparison with the weighted bias obtained from the sWeights approach.

- **Uncertainty Distribution:** The distribution of uncertainties on λ, as estimated from the fits, is plotted. The weighted uncertainty from the sWeights fit and the true uncertainty from the bootstrapping process are included for reference.

3.4 Comparison of Methods

The comparison of the two methods (parametric bootstrapping and sWeights)reveals important insights regarding their performance in terms of bias, uncertainty, and sensitivity to sample size.

Bias Reduction The sWeights method demonstrates a consistent improvement in bias reduction across all sample sizes compared to the unweighted maximum likelihood fits used in parametric bootstrapping. For smaller sample sizes (e.g., 500 and 1000), the bias observed in the bootstrapping approach is larger, as indicated by the mean of the bias distributions. In contrast, the sWeights method effectively reduces this bias, aligning the weighted λ estimates closer to the true value. As the sample size increases, both methods show a significant reduction in bias. For example, at a sample size of 10,000, the bias is minimal for both approaches, with the parametric bootstrapping and sWeights yielding nearly identical estimates. This indicates that while the sWeights approach is particularly advantageous for smaller datasets, its relative benefit diminishes as the sample size grows.

Uncertainty Estimates The uncertainty distributions highlight another key distinction between the methods. The parametric bootstrapping method produces a broader spread of uncertainties for small sample sizes, reflecting the natural variability introduced by Poisson fluctuations and the limited statistical power of smaller datasets. Conversely, the sWeights method

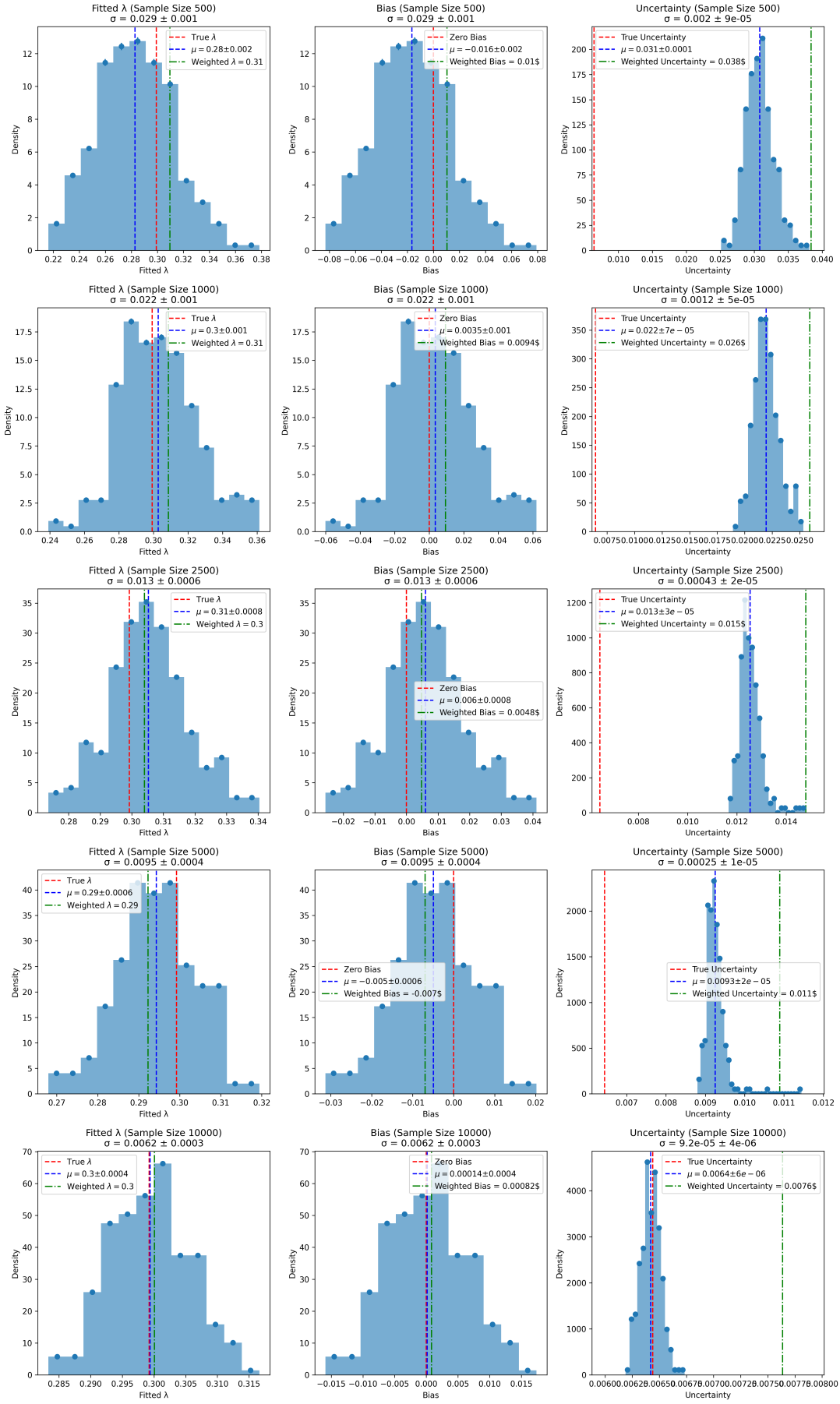

Figure 5: Comparison of fitted λ values, biases, and uncertainties across different sample sizes.

consistently produced larger uncertainties, for each sample size. Due to the nature of the implementation of the sWeights, only one value for the uncertainty is returned, whilst the parametric bootstrapping method returns an associated uncertainty for each 'toy'. Thus with larger and larger sample sizes, the bootstrapping method gradually achieved very good alignment with the original 'true' uncertainty.

Computational Efficiency The sWeights method is computationally more efficient than parametric bootstrapping as it eliminates the need to generate and fit multiple bootstrap samples. Instead, it determines signal and background contributions in a single fit to the X variable and applies the resulting weights to the Y variable. In contrast, the bootstrapping approach requires generating multiple datasets, performing fits on each, and analysing the resulting distributions, which can be computationally expensive, particularly for large ensembles.

Applicability and Scenarios The choice between the sWeights method and parametric bootstrapping depends on the specific analysis goals, sample size, and computational constraints. Both methods have distinct advantages and limitations:

- **Small Sample Sizes:** The sWeights method is particularly advantageous for smaller datasets, as it significantly reduces bias compared to the unweighted maximum likelihood fits used in parametric bootstrapping. This is demonstrated by the improved alignment of the weighted λ estimates with the true values, even for small sample sizes such as 500 or 1000. However, the sWeights method returns a single uncertainty value, which may not fully capture variability in these cases. In contrast, the bootstrapping approach can yield broader uncertainty estimates due to the inherent variability of the resampled datasets.
- **Large Sample Sizes:** For larger datasets (e.g., 10,000 or more), both sWeights and parametric bootstrapping show minimal bias, with nearly identical estimates for key parameters. However, the bootstrapping method provides additional robustness by not relying on the specific assumptions underlying sWeights. In these scenarios, bootstrapping achieves better alignment with the true uncertainty due to its ability to generate a distribution of uncertainties from multiple resamples, making it a valuable tool for high-precision studies.
- **Complex Background Models:** When dealing with complex or poorly understood background distributions, parametric bootstrapping is a more reliable choice. By directly sampling from the true probability density function (PDF), it avoids potential biases introduced by an incorrect model used to estimate the weights in the sWeights method. However, this approach requires significantly more computational resources, particularly for large ensembles.

Due to the nature of the parametric bootstrapping approach, for this analysis it is the preferred method due to the ease of sampling more and more data. Furthermore, given the scope of the problem, computational times were not a large consideration so sWeights are again less favourable.

Conclusion

This study applied statistical techniques to multi-dimensional data analysis, with a focus on the derivation, normalisation, and implementation of probability density functions (PDFs) for signal and background distributions. The synthetic dataset generated using vectorised rejection sampling enabled parameter estimation through an extended maximum likelihood fit. Bias and

uncertainty in these estimates were further investigated using parametric bootstrapping and sWeights methods.

The parametric bootstrapping approach, while computationally intensive, provided robust and unbiased estimates, particularly valuable for large datasets. Conversely, the sWeights method offered computational efficiency and improved bias reduction for smaller datasets but returned a single uncertainty value, limiting its representation of variability in such cases. Future extensions of this work could include:

1. **Complex Model Integration**: Expanding the analysis to more intricate signal and background models to understand overlapping distributions more effectively.
2. **Scalability to Higher Dimensions**: Investigating the scalability and performance of these methods in higher-dimensional datasets to handle complex real-world scenarios.
3. **Algorithmic Efficiency**: Leveraging parallel processing or GPU-based computations to optimise the performance of bootstrapping methods for extensive datasets.

References

[1] S. K. Lam, A. Pitrou, and S. Seibert, "Numba: A llvm-based python jit compiler," in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 2015, pp. 1–6.

[2] H. Dembinski and P. O. et al., "scikit-hep/iminuit," Dec 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3949207

[3] H. Dembinski, M. Kenzie, C. Langenbruch, and M. Schmelling, "Custom orthogonal weight functions (cows) for event classification," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 1040, p. 167270, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168900222006076

Appendix A: Declaration of AI Assistance

This report was prepared independently; however, the AI tool ChatGPT (version GPT-4, developed by OpenAI) was used to assist with specific tasks. The primary areas where AI assistance was utilised include:

- **Drafting and Proofreading:** Suggestions were provided to refine grammar, sentence structure, and word choice to improve the clarity and flow of this report.
- **Formatting Guidance:** Assistance was given for structuring the LaTeX document, including consistent formatting of equations, figures, and references.
- **Coding:** AI support was used primarily to aid the plotting process for this report, and aid in explanation especially for the final lambda task and the implementation of sWeights. It was also used extensively to help find a good manner in which to plot the results. No supplemented code has gone unrevised or thoroughly checked.