

1. The Theoretical Framework

Heavy-Tailed distributions

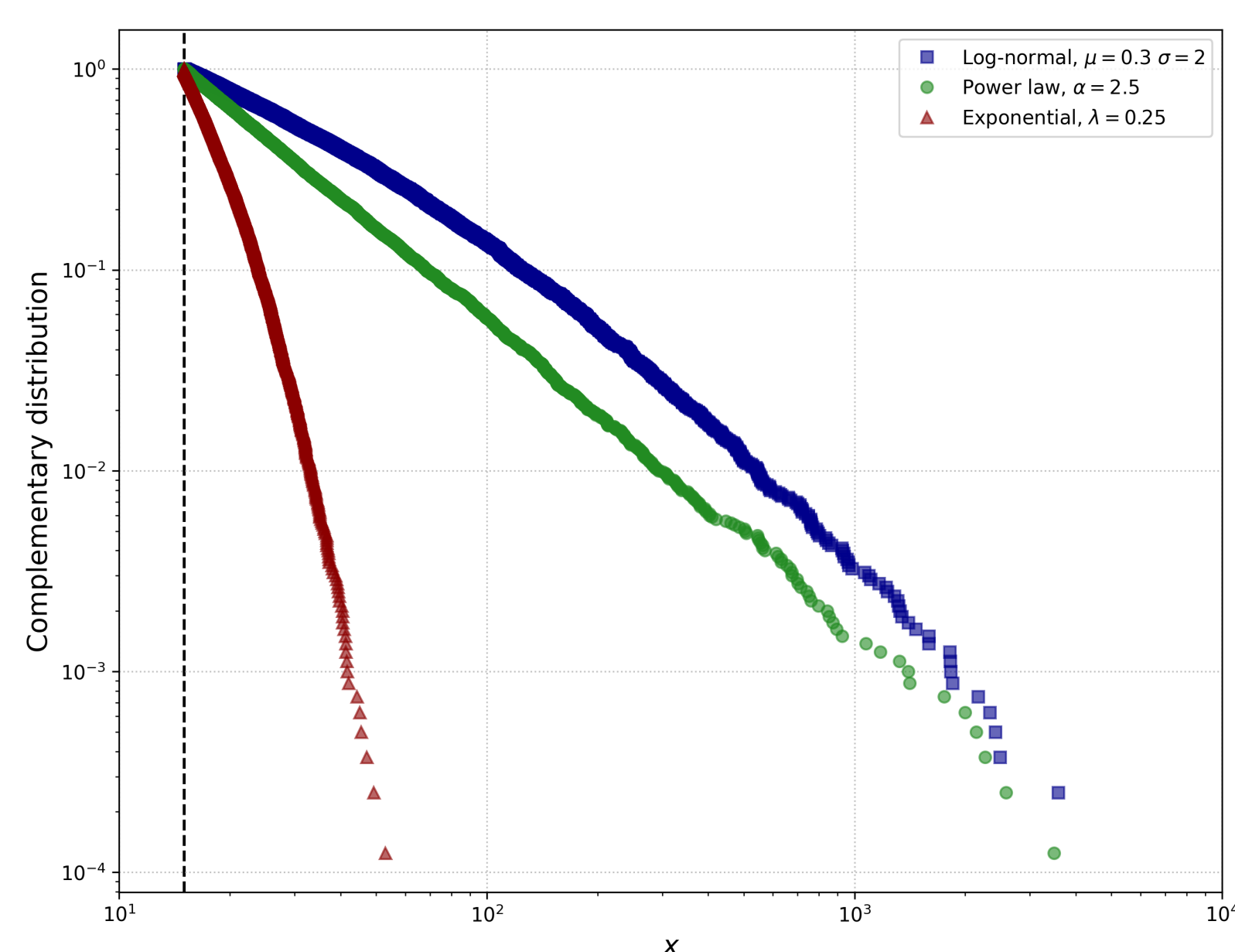
We call $f(x; \theta)$ a heavy-tailed distribution if, for a given m , its moments diverge:

$$\langle x^m \rangle \rightarrow \infty, \quad (1)$$

meaning that rare, extreme events always have an appreciable probability of occurrence.

- They appear everywhere in Complex Systems, from natural languages to network science.
- Empirically, this is an **asymptotic behavior**, i.e., it holds only for values $x \geq x_{\min}$.

The *challenge* is to correctly distinguish their behavior from other distributions and accurately identify where this tail begins without subjective visual bias.



The standard method for fitting power laws

Given a dataset of random variables, $D = \{x_j\}$, we estimate the lower bound by finding the $\hat{x}_{\min}$ which minimizes the **Kolmogorov-Smirnov statistic**:

$$\mathcal{K}[F] = \max_{x \in D} |S(x) - F(x)|, \quad (2)$$

while the exponent α is found by maximizing the log-likelihood:

$$\mathcal{L} = \sum_{j=1}^n \log [f(x_j; \theta)]. \quad (3)$$

This standard framework, proposed by Clauset et al. [3], is a powerful tool, but has severe limitations:

- It usually *overestimates* the lower bound, meaning a significant amount of valid data is lost in the process;
- Naive implementations scale poorly in complexity, creating bottlenecks for large datasets.

What is new in mle_fit?

The main goal of our library is to address these issues and provide a state-of-the-art, user-friendly, and fast-as-it-can-be tool for fitting power laws.

1. The Regularized Loss Functional:

We extend the Clauset method by introducing a loss functional that penalizes the algorithm for discarding too much data:

$$\Lambda[F, x_{\min}] = \mathcal{K}[F] - \lambda \left(\frac{n_{\text{tail}}(x_{\min})}{N} \right), \quad (4)$$

where n_{tail}/N is the fraction of data kept in the tail and λ is the regularization hyperparameter (≈ 0.01).

2. Greedy Search Algorithm:

The algorithm scans the KS-landscape for local minima and applies dynamic thresholds, find the first lower bound with statistical significance.

2. Benchmarking

The Local Minima Problem: Moby Dick

To illustrate our method's advantage, we analyze the word frequencies in *Moby Dick* [2, 3].

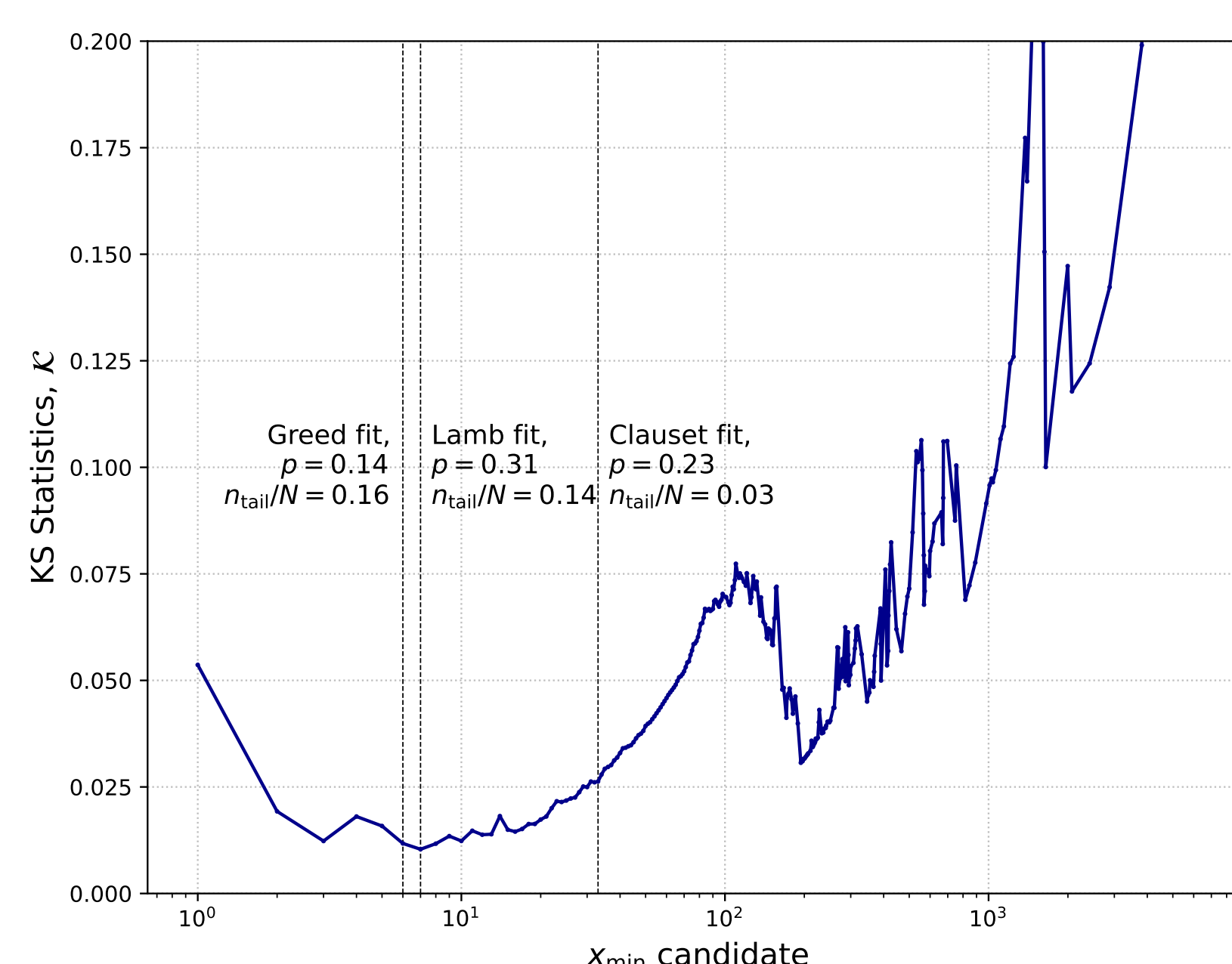


Figure 1: The KS-statistic landscape.

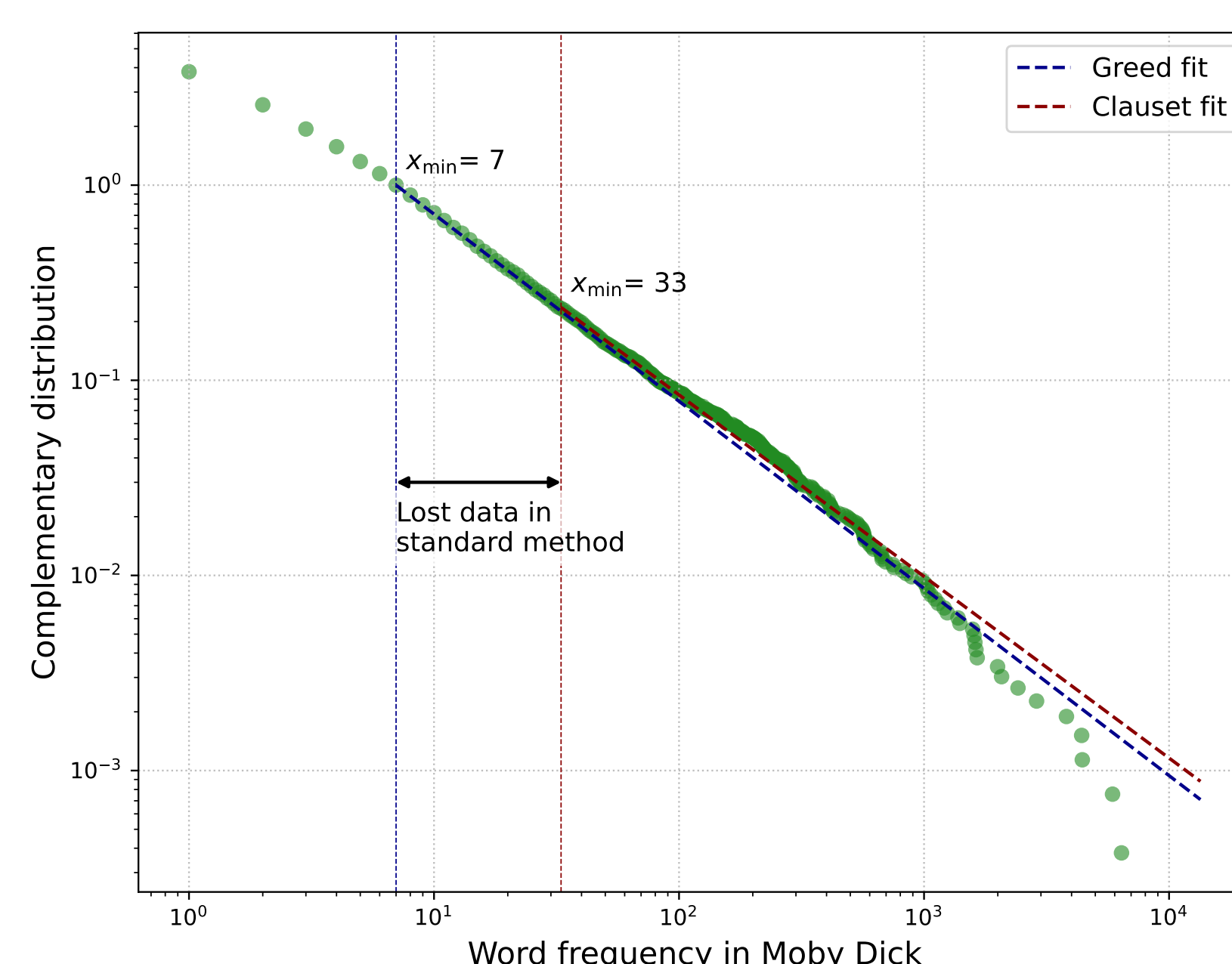


Figure 2: Complementary Cumulative Distribution Function (CCDF) of word frequencies.

Robustness in empirical datasets

We benchmarked *mle_fit* in 147 empirical datasets, including real networks, population and word frequency.

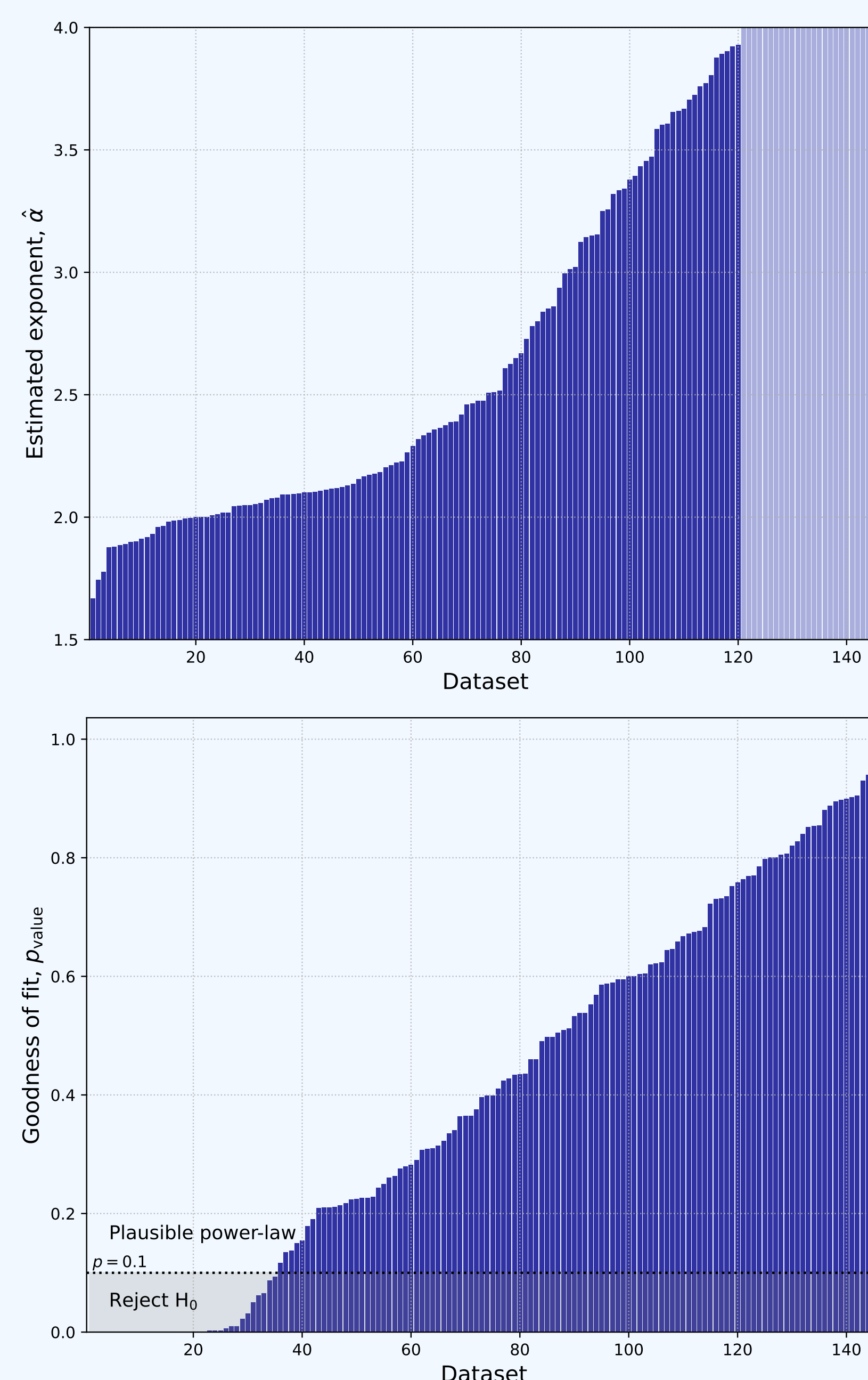
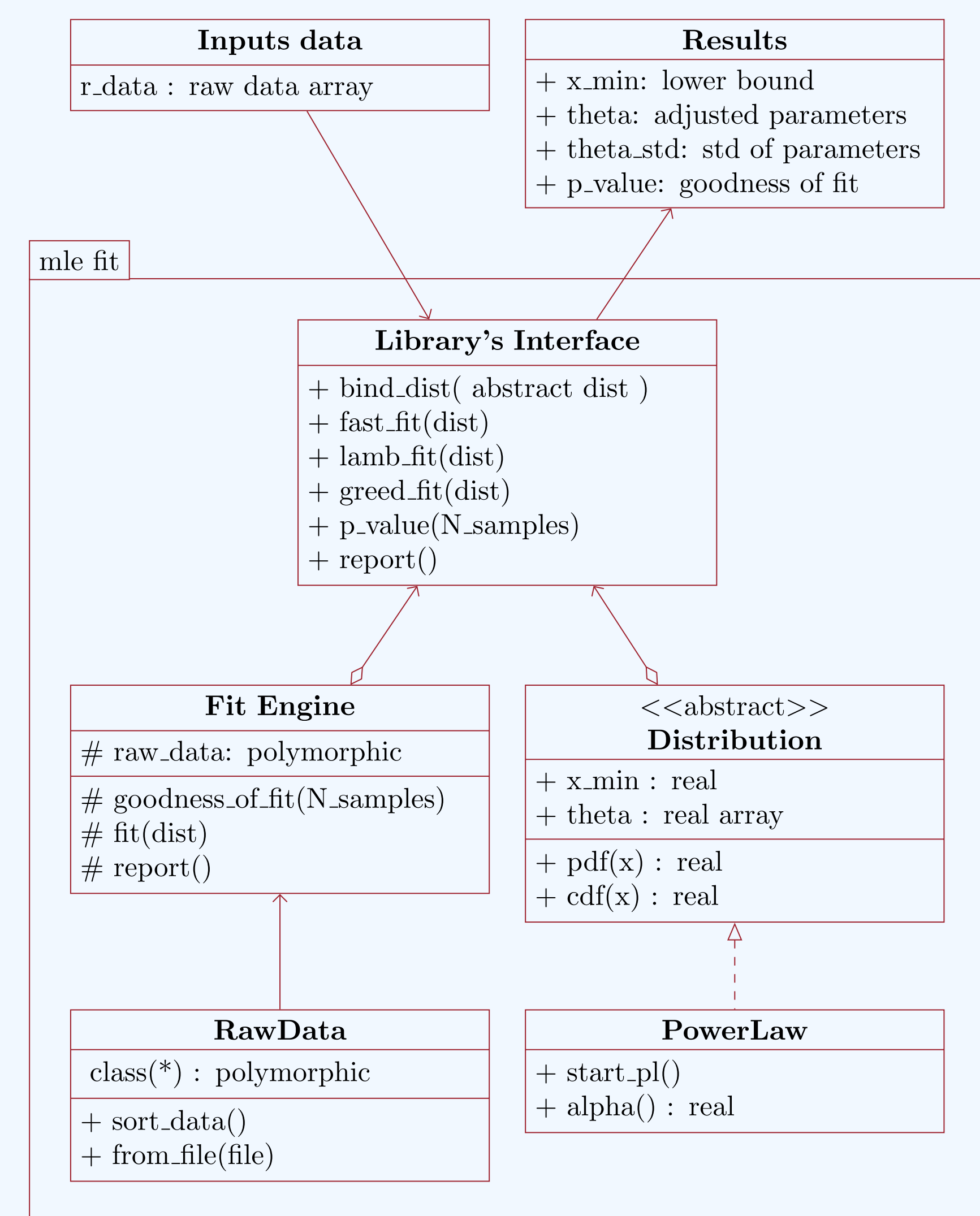


Figure 3: (Top) Estimated power-law exponents ($\hat{\alpha}$) sorted by magnitude. (Bottom) Testing null hypothesis, H_0 , with *mle_fit*. The vast majority of datasets yield $p \geq 0.1$.

3. Software Architecture & Performance

Library's Architecture



Speedup vs. Python

We compare the performance against the widely used Python package *powerlaw* Alstott et al. [1].

- On average, our library is about 300× faster than the Python equivalent for integer data and 10000× for real data;
- We reduce the bootstrap test time from hours to seconds by using parallelization with OpenMP.

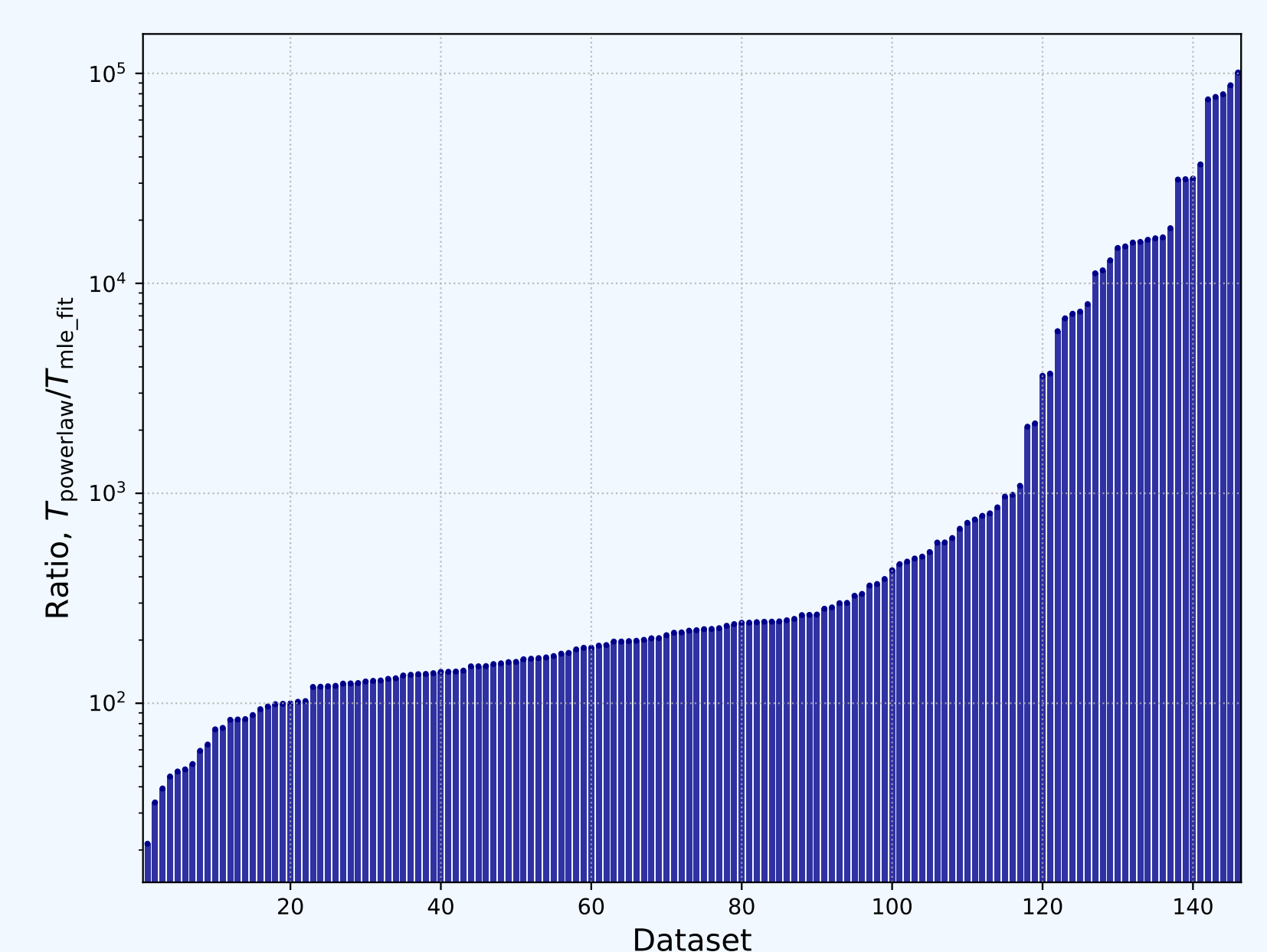


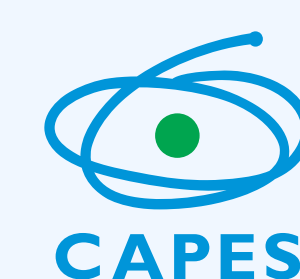
Figure 4: Execution time ratio ($T_{\text{powerlaw}}/T_{\text{mle_fit}}$).

Check out the Source Code



Available via Fortran Package Manager (fpm):
@timotheosf/mle_fit

Acknowledgments



References

- [1] Jeff Alstott et al. "powerlaw: A Python Package for Analysis of Heavy-Tailed Distributions". In: *PLOS ONE* 9.1 (Jan. 2014), pp. 1–11.
- [2] Eduardo G. Altmann. *Statistical Laws in Complex Systems: Combining Mechanistic Models and Data Analysis*. Springer Nature Switzerland, 2024.
- [3] Aaron Clauset et al. "Power-Law Distributions in Empirical Data". In: *SIAM Review* 51 (June 2007).