Statistical Efficiency of Birth-Death Tree Parameter Estimation

Stat 700

Santosh Desai, Matt McAnear, Urvi Mehta

Department of Statistics, University of Michigan

December 4, 2025

The Phylodeep Package

Phylodeep Paper is byt J. Voznica et al [2]

- Parameter estimation is difficult and MLE method can be challenging for larger trees.
- Major components of the paper to address this
  - CBLV representation of trees
  - Pre-trained neural networks for pre-specified tree types
- The goal is to speed up inference

Usage

- Input: Phylogenetic tree in Newick format
- Output: Estimated parameters (e.g., birth rate (λ), death rate (μ), R_0)
- Given a tree, Phylodeep selects the pre-trained model with the highest probability
- Outputs actually come with uncertainty estimates from a parametric bootstrap

Maximum Likelihood Estimation: Overview

- **Goal**: Estimate birth rate (λ) and death rate (μ) from phylogenetic trees
- **Method**: Numerical optimization of Stadler (2010) [1] likelihood
- **Key Feature**: Works with **tree data only** - no nucleotide sequences required
- **Implementation**: Uses `scipy.optimize.minimize` with L-BFGS-B algorithm

Advantages:

- Statistically principled (maximum likelihood)
- Works on small trees (no minimum tip size requirement)
- Provides point estimates with optimization diagnostics

Stadler (2010) Likelihood Formulation

For a constant-rate birth-death process with sampling probability ρ:

$$\begin{aligned}\ell(\lambda, \mu | T, \rho) = &(n-1)\log(\lambda) \\ &- (\lambda + \mu) T_{\text{total}} \\ &+ n \log(\rho) \\ &- r \cdot T + \text{penalty terms}\end{aligned}$$

where:

- n = number of tips
- T_{total} = total branch length
- T = tree height (time from root to present)
- $r = \lambda - \mu$ = net diversification rate
- ρ = sampling probability

Key insight: Survival probability term ($-r \cdot T$) helps identify μ separately from λ

MLE Implementation: Tree Statistics Extraction

Extracted from each tree:

- Number of tips: n
- Total branch length: $T_{\text{total}} = \sum_{e \in E} \ell(e)$
- Tree height: $T = \max_{v \in V} d(\text{root}, v)$
- Branching times: All internal node ages
- Mean branch length: $\bar{\ell}$

Initial parameter estimates (method-of-moments):

$$\hat{r} = \frac{\log(n)}{T}$$

$$\hat{\lambda} = \frac{n-1}{T_{\text{total}}}$$

$$\hat{\mu} = \max(0.01, \hat{\lambda} - \hat{r})$$

These serve as starting values for numerical optimization.

MLE Implementation: Optimization

Numerical optimization:

- **Algorithm**: L-BFGS-B (bounded optimization)
- **Bounds**: $\lambda \in [0.01, 10.0]$, $\mu \in [0.01, 10.0]$
- **Constraint**: $\mu < \lambda$ (enforced via penalty)
- **Gradient**: Analytical gradient provided for faster convergence

Gradient components:

$$\frac{\partial \ell}{\partial \lambda} = -\frac{n-1}{\lambda} + T_{\text{total}} + T$$
$$\frac{\partial \ell}{\partial \mu} = T_{\text{total}} - T + \text{penalty term}$$

Regularization: Penalty term prevents μ from collapsing to lower bound in large trees

MLE Implementation: Key Features

1. **Survival Probability Term**
   - Includes $-r \cdot T$ term to help identify μ
   - Accounts for probability of observing n tips given λ and μ
   - Critical for separating μ from λ in the likelihood
2. **Regularization for Large Trees**
   - Penalty: $-0.1 \log(\mu)$ when $\mu < 0.1$ and $n > 10$
   - Prevents μ from getting stuck at lower bound (0.01)
   - Encourages realistic death rate estimates based on tree size
3. **Robust Error Handling**
   - Checks for invalid trees ($n \leq 1$, $T_{\text{total}} \leq 0$)
   - Handles optimization failures gracefully
   - Returns success status, log-likelihood, and iteration count
4. **Method-of-Moments Initialization**
   - Smart starting values: $\hat{r} = \log(n)/T$, $\hat{\lambda} = (n-1)/T_{\text{total}}$
   - Ensures faster convergence and avoids local minima

MLE: Advantages and Limitations

Advantages:

- ✓ Statistically principled (maximum likelihood)
- ✓ No minimum tree size requirement
- ✓ Works on small trees ($n < 50$)
- ✓ Provides optimization diagnostics
- ✓ Fast computation (seconds per tree)
- ✓ Based on well-established Stadler (2010) [1] theory

Limitations:

- × Higher RMSE than PhyloDeep on large trees
- × Point estimates only (no uncertainty quantification)
- × Requires careful initialization for convergence
- × Can struggle with very large trees ($n > 500$)

Best use case: Small to medium trees where statistical rigor is important

PhyloDeep: How We Used the Package

Pre-trained Models Used:

- **Model Type**: BD (Birth-Death) pre-trained models from PhyloDeep package
- **We ONLY estimated BD trees** - used PhyloDeep's BD models exclusively

Model Selection by Tree Size:

- **Medium trees** ($50 \leq n < 200$): PhyloDeep automatically uses BD models trained on medium-sized trees
- **Large trees** ($n \geq 200$): PhyloDeep automatically uses BD models trained on large trees
- **Automatic selection**: PhyloDeep's `paramdeep` function selects the appropriate model based on tree size

Implementation:

- Called `paramdeep(tree_file, sampling_prob=0.5, model='BD', vector_representation='FULL')`
- Used FULL (CNN-based CBLV) representation, not SUMSTATS

PhyloDeep: Key Features

1. **Pre-trained BD Models**
   - Used existing BD models trained on 3.9 million trees (Voznica et al.)
   - Fast inference (milliseconds per tree)
   - Automatic model selection by tree size
2. **FULL Tree Representation**
   - CNN-based CBLV representation
   - Converts tree to fixed-size vector for neural network processing
3. **Output**
   - Point estimates: λ, μ, R_0, infectious period

PhyloDeep: Advantages and Limitations

Advantages:

- ✓ Lower RMSE than MLE on large trees ($n \geq 50$)
- ✓ Extremely fast inference (milliseconds)
- ✓ Trained on massive dataset (3.9M trees)

Limitations:

- × Requires minimum 50 tips (cannot handle small trees)
- × Black-box model (less interpretable)
- × Point estimates only in our implementation

Best use case: Large trees ($n \geq 50$) where speed and accuracy are priorities

Method Comparison: PhyloDeep vs MLE

Feature	PhyloDeep	MLE
Approach	Deep learning	Classical stats
Training	Pre-trained (3.9M)	None
Optimization	Neural network	Single L-BFGS-B
Min Size	50 tips	None
Speed	Milliseconds	Seconds
Accuracy	Best (large trees)	Baseline
Interpretable	No	Yes

When to Use:

- **Small ($n < 50$)**: MLE only (PhyloDeep unavailable)
- **Medium ($50 \leq n < 200$)**: PhyloDeep (speed) or MLE (interpretability)
- **Large ($n \geq 200$)**: PhyloDeep (accuracy)

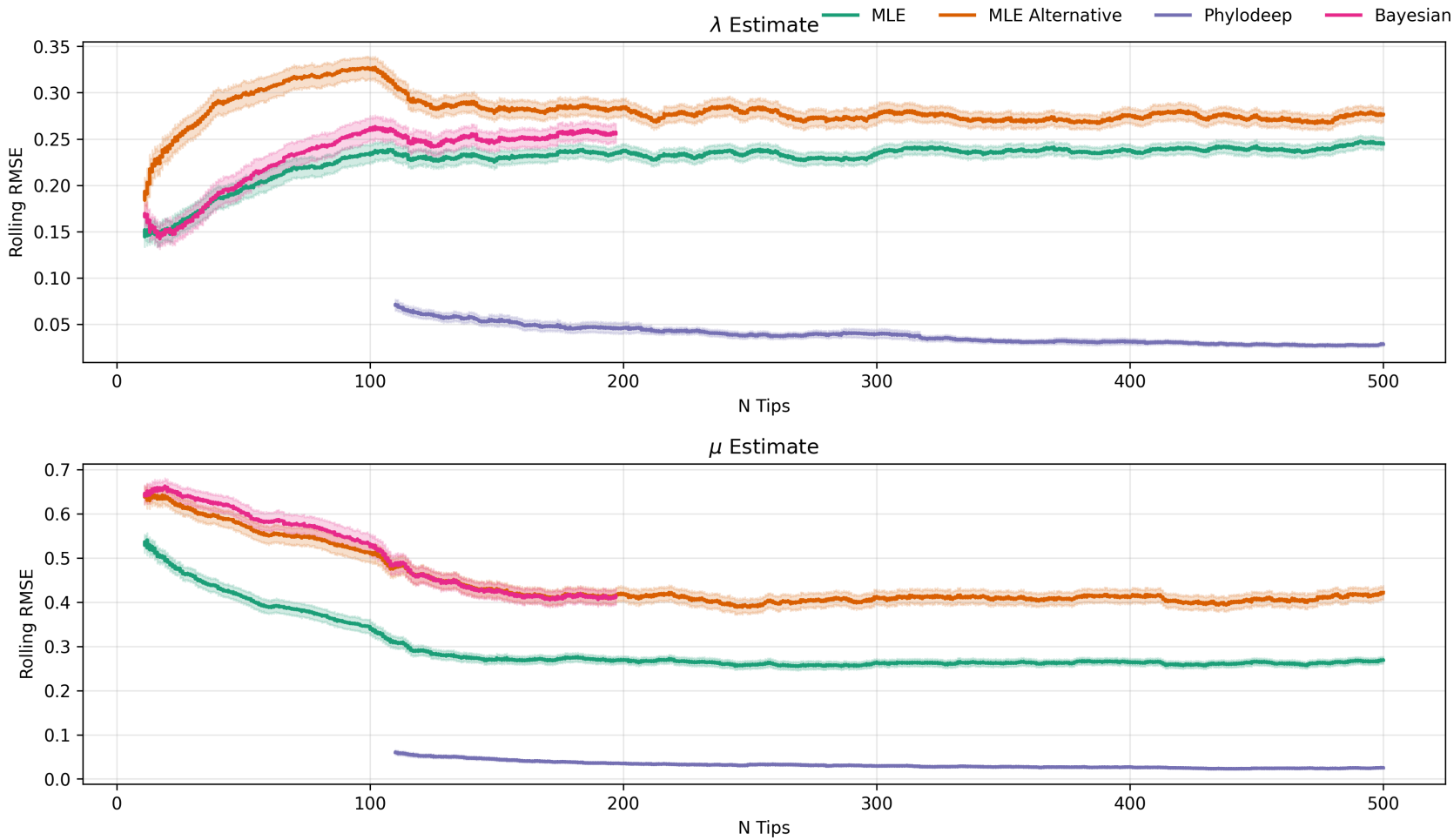

Figure: Rolling bootstrapped RMSE estimates of μ and λ by tree size and resulting 95% CI. Window size is 500.

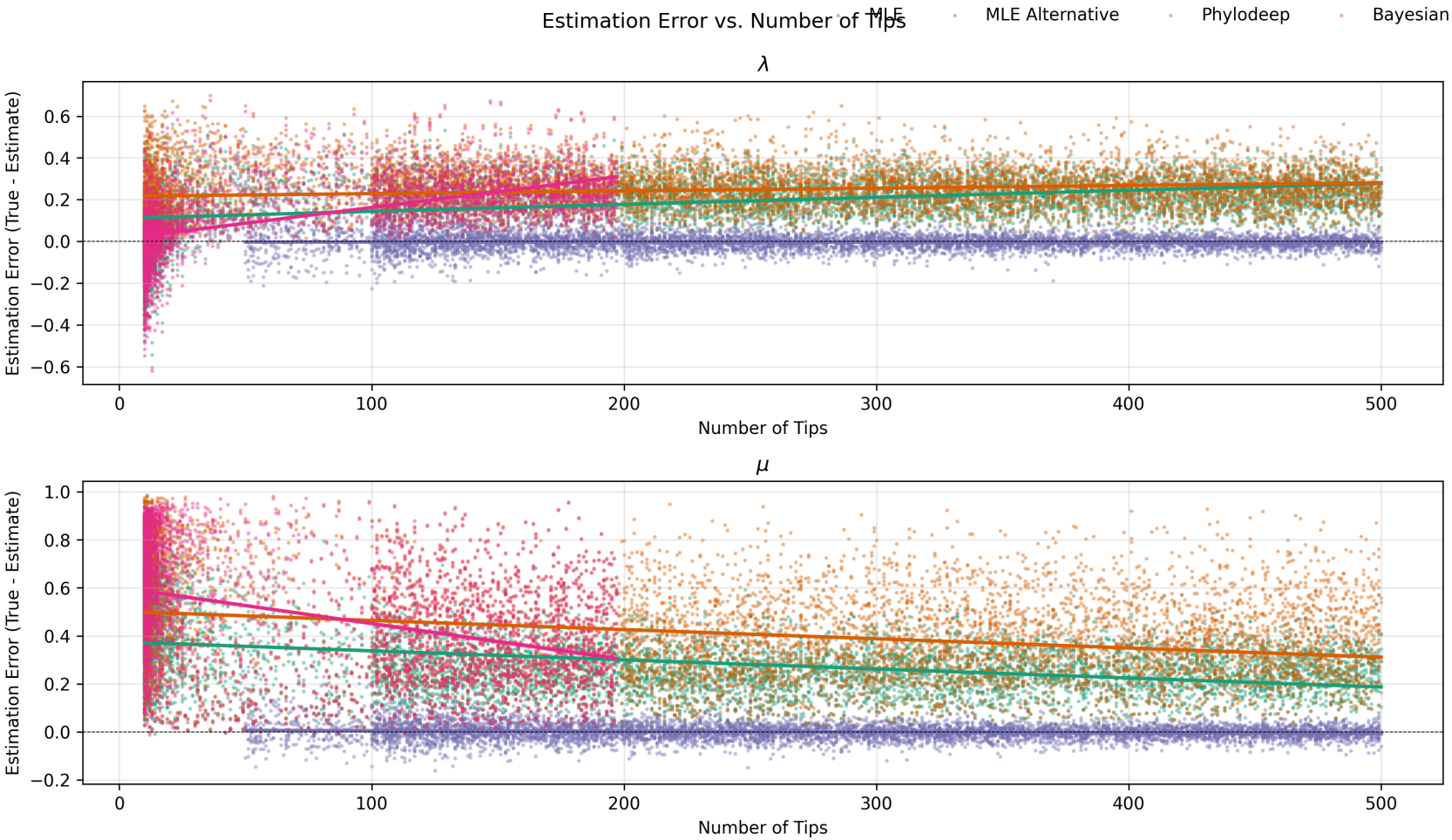

Figure: Raw plot of the absolute errors in λ, μ by tree size.

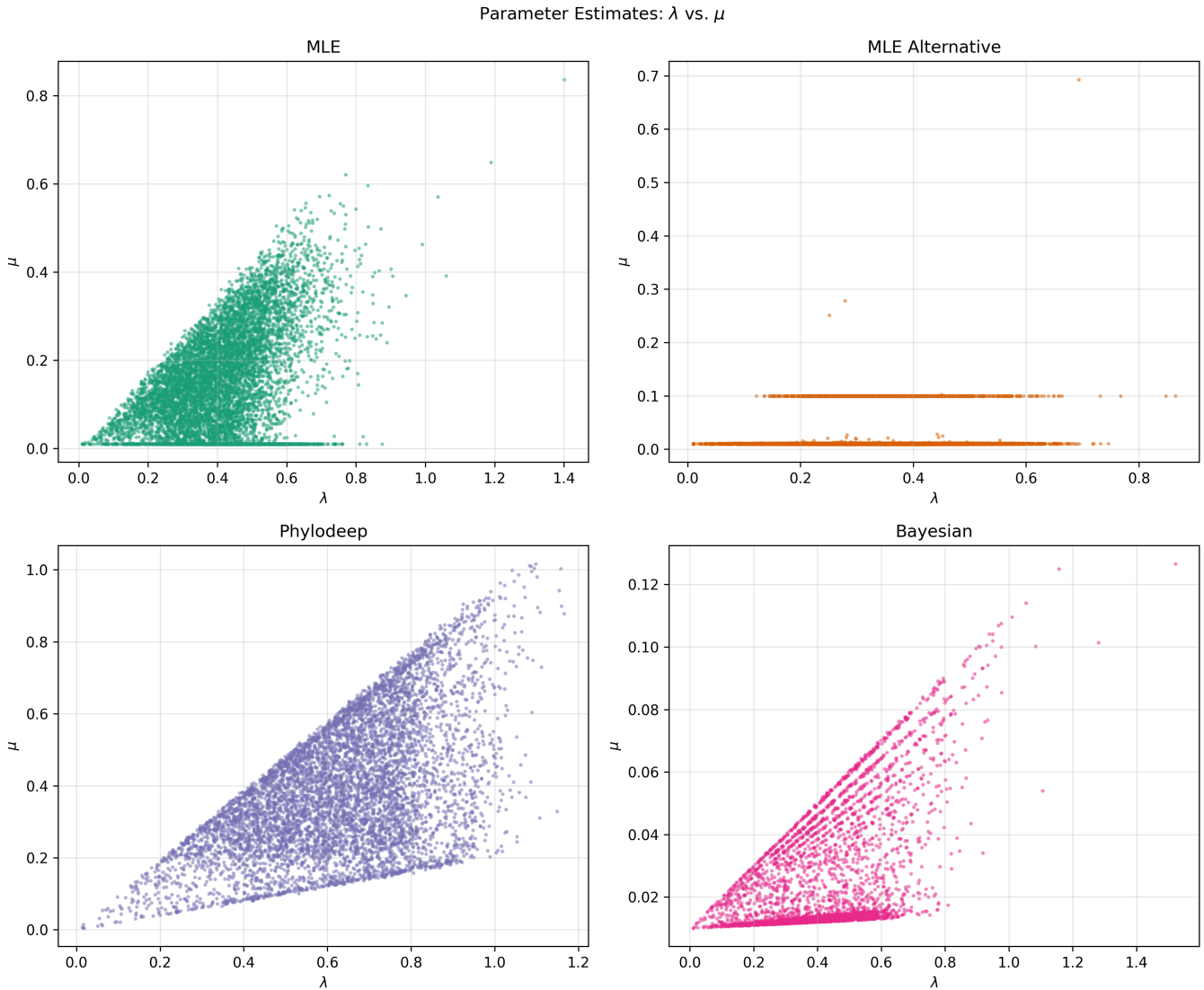

Figure: Plot of λ vs μ estimates to show distribution of estimates.

Bayesian MCMC: Attempted Analysis

What We Attempted:

- Implemented Bayesian MCMC using PyMC for birth-death parameter estimation
- Used same Stadler (2010) [1] likelihood as MLE
- Sequential processing to avoid PyTensor cache conflicts
- Goal: Full posterior distributions with uncertainty quantification

Initial Results:

- Successfully analyzed 3,550 trees (out of 8,655 total)
- Tree size range: 10-181 tips
- Success rate: 99.4% (3,530/3,550 trees)

Why We Did Not Pursue Further:

- × RMSE was not better than MLE or PhyloDeep
- × Computationally intensive (minutes per tree)
- × Sequential processing too slow for full dataset
- × No clear advantage to justify computational cost

Conclusion: Bayesian MCMC provides uncertainty quantification but does not improve RMSE, so we focused on MLE and PhyloDeep for the main analysis.

Results: Summary

Key Findings:

- RMSE decreases with increasing tree size for both methods
- **MLE**: Works on small trees ($n < 50$) where PhyloDeep fails
- **PhyloDeep**: Lower RMSE on average at all tree sizes where both methods apply ($n \geq 50$)
- **Bayesian MCMC**: Attempted but RMSE not better than MLE/PhyloDeep

Results: Method Comparison

By Tree Size:

- **Small trees ($n < 50$)**: MLE only (PhyloDeep unavailable)
- **Medium trees ($50 \leq n < 200$)**: PhyloDeep has lower RMSE on average
- **Large trees ($n \geq 200$)**: PhyloDeep has lower RMSE on average

Statistical Efficiency:

- PhyloDeep's advantage: training on 3.9M trees enables better accuracy on average at all tree sizes
- MLE: statistically principled baseline, works on all tree sizes including small trees
- Bayesian MCMC: Attempted but did not improve RMSE, so not pursued further

References

[1] Tanja Stadler. "Sampling-through-time in birth–death trees". In: *Journal of Theoretical Biology* 267.3 (Dec. 2010), pp. 396–404. ISSN: 0022-5193. DOI: 10.1016/j.jtbi.2010.09.010. URL: https://www.sciencedirect.com/science/article/pii/S0022519310004765 (visited on 09/18/2025).

[2] J. Voznica et al. "Deep learning from phylogenies to uncover the epidemiological dynamics of outbreaks". en. In: *Nature Communications* 13.1 (July 2022). Publisher: Nature Publishing Group, p. 3896. ISSN: 2041-1723. DOI: 10.1038/s41467-022-31511-0. URL: https://www.nature.com/articles/s41467-022-31511-0 (visited on 11/04/2025).