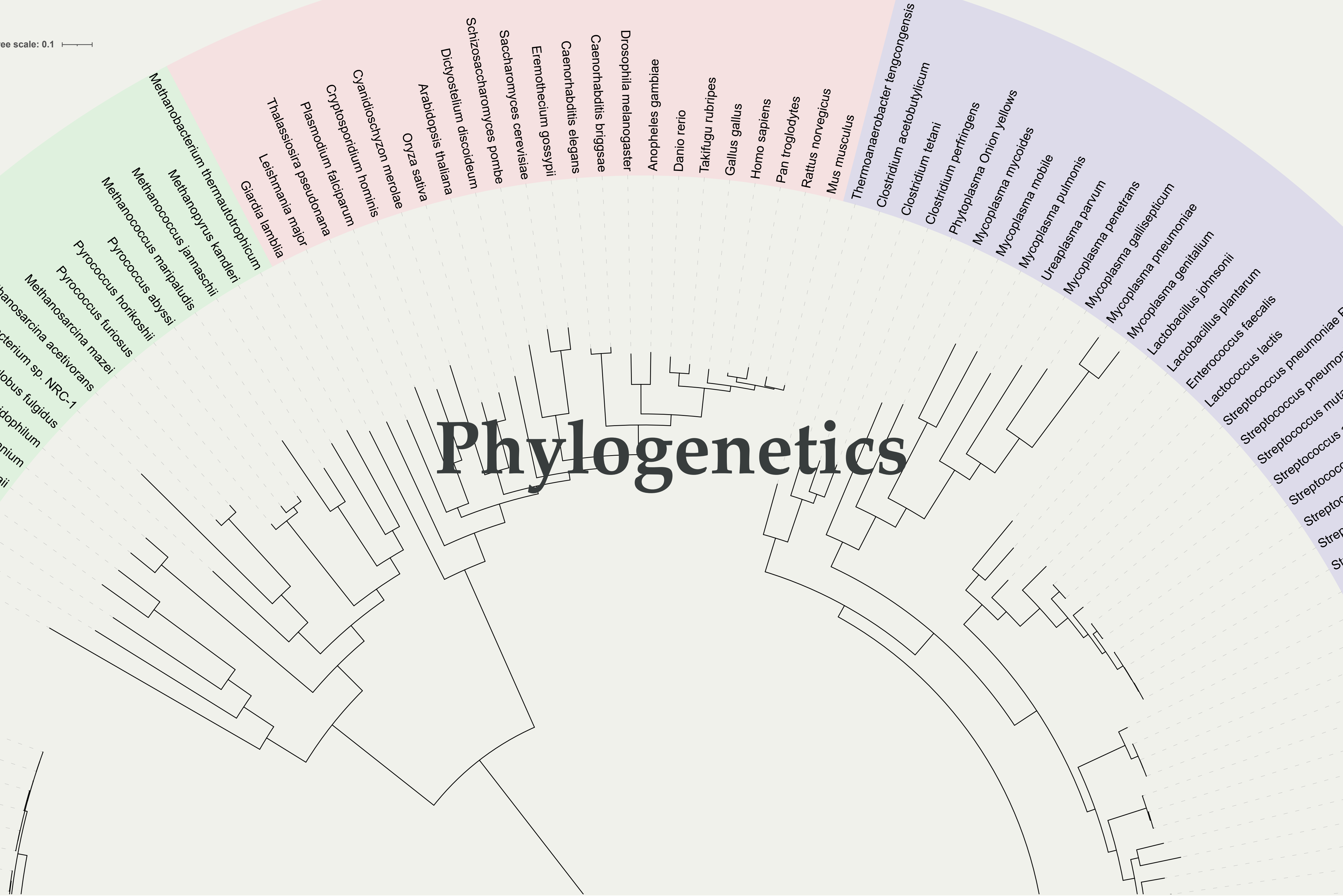
TreePPL—a new DSL in Miking for Phylogenetics

Viktor Senderov
École normale supérieure - PSL

2023-11-22

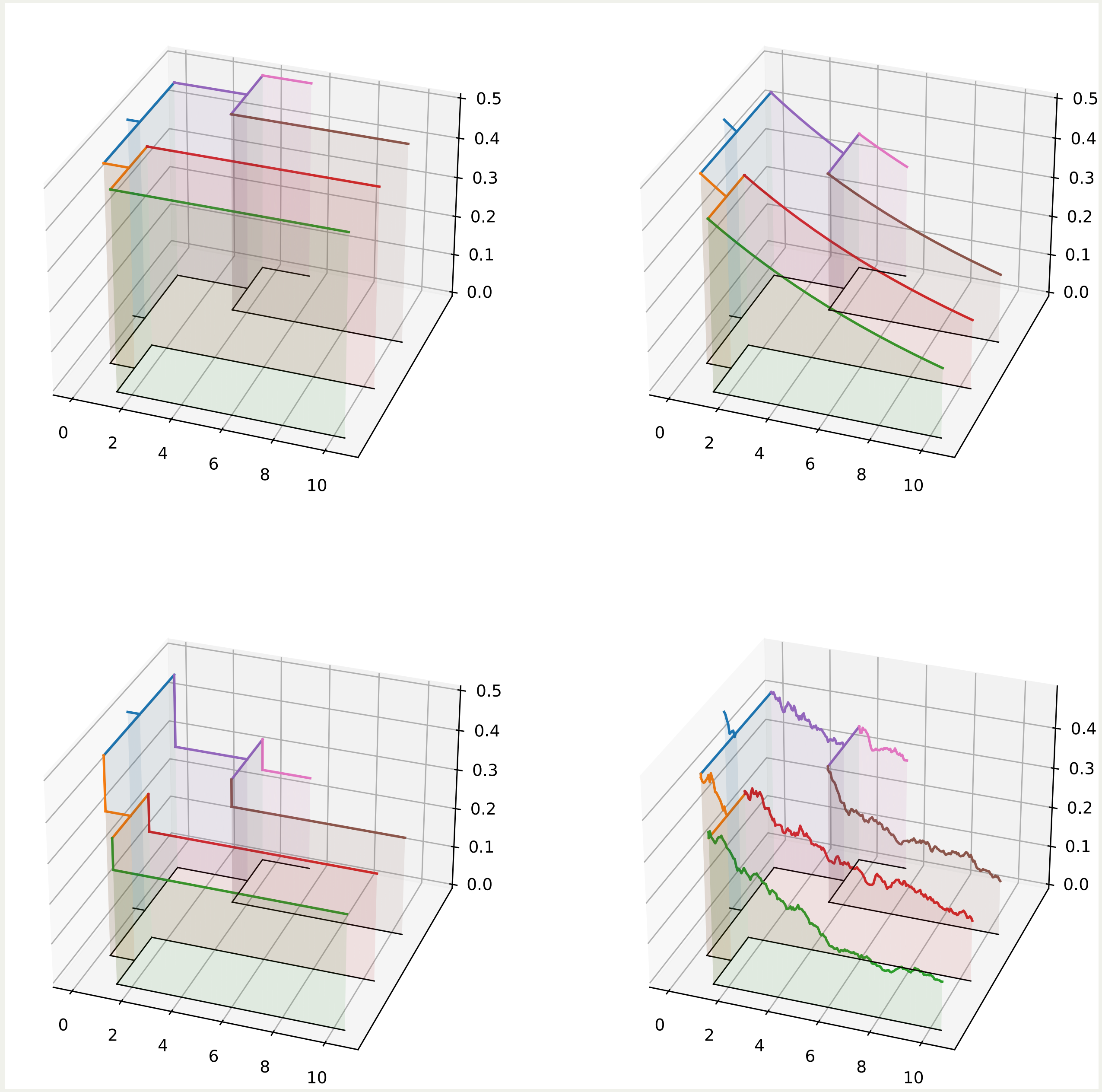
Contents

Phylogenetics



Phylogenetics discussion

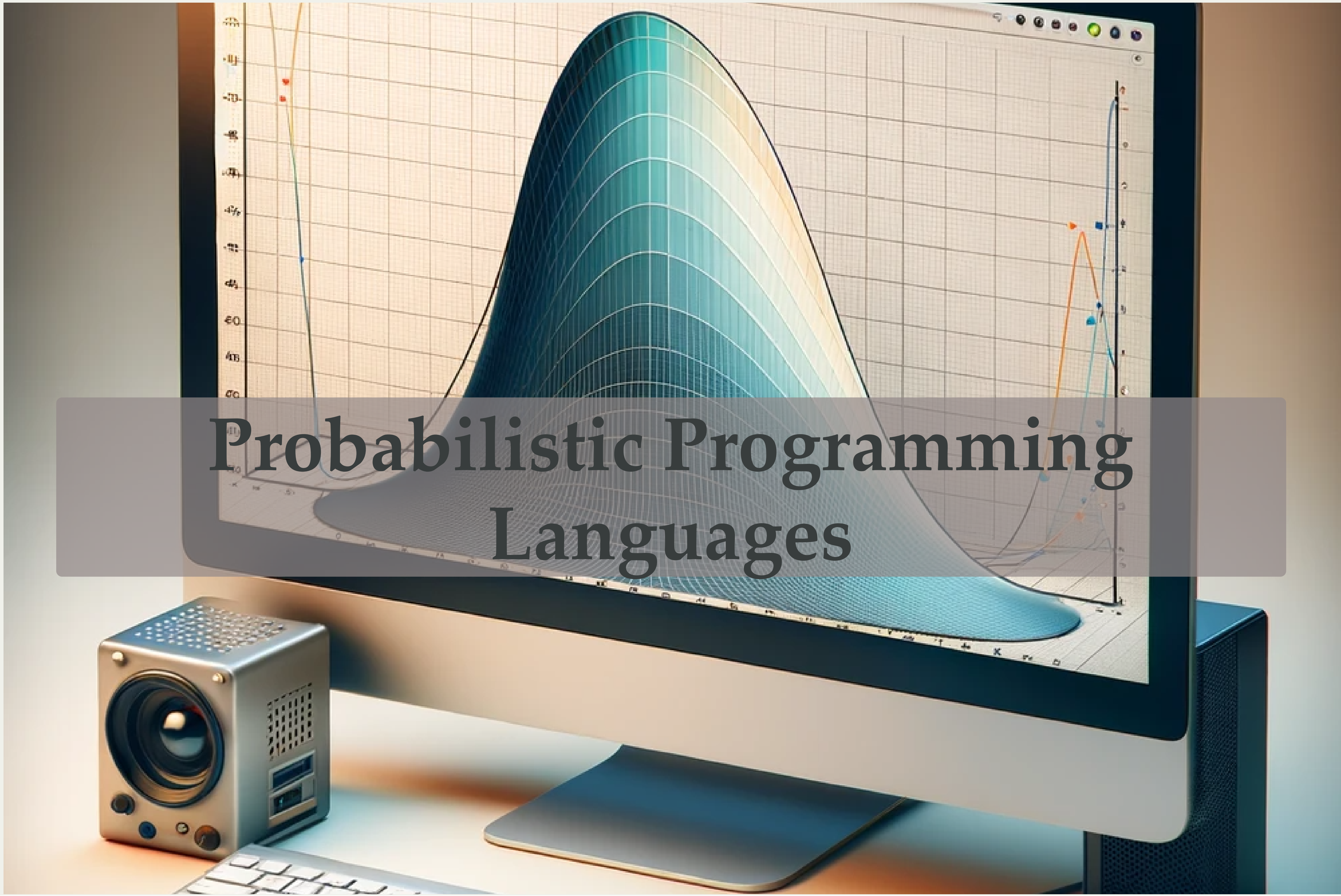
- Problems that phylogenetics deals with
  - Reconstructing phylogenetic trees
  - Using phylogenetic trees to understand evolution and ecology



Diversification models on a phylogenetic tree

What can the users do?

- Methods of phylogenetics
  - Parsimony
  - Statistical methods: ML and Bayesian
- Software for phylogenetics
  - monolithic software: e.g. MrBayes, RAxML
  - extensible software: e.g. BEAST, BEAST2, RevBayes
- Write the model yourself
  - implement in a general-purpose language: e.g. C, Java, Python, or R
  - implement in a probabilistic programming language: e.g. WebPPL, Birch
  - implement in a phylogenic PPL: TreePPL



Probabilistic Programming Languages

Bayes theorem

Let x be some observed data and θ be an unobserved parameter in whose value we are interested in in.

$$\overbrace{p(\theta|x)}^{\text{posterior distribution}} = \frac{\overbrace{p(x|\theta)}^{\substack{\text{data distribution/}\\ \text{likelihood}}} \quad \overbrace{p(\theta)}^{\text{prior distribution}}}{\underbrace{p(x)}_{\text{normalizing constant}}}$$

Bayes theorem

Let x be some observed data and θ be an unobserved parameter in whose value we are interested in in.

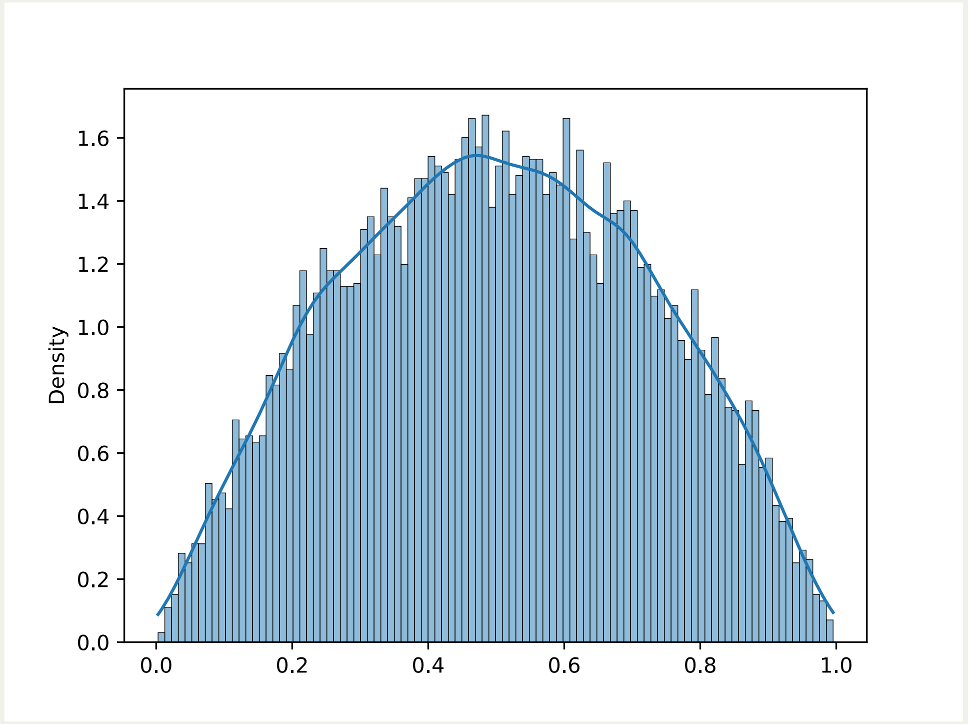
$$\overbrace{p(\theta|x)}^{\text{posterior distribution}} = \frac{\overbrace{p(x|\theta)}^{\substack{\color{red}{\text{observe/factor}} \\ \text{data distribution/} \\ \text{likelihood}}} \quad \overbrace{p(\theta)}^{\substack{\color{red}{\text{assume/sample}} \\ \text{prior distribution}}}}{\underbrace{p(x)}_{\text{normalizing constant}}}$$

Coin example

```python
outcomes=[True, True, True, False, True, False, True, True, True, False,
    False, True, True, False, True, False, False, True, False, False]
```

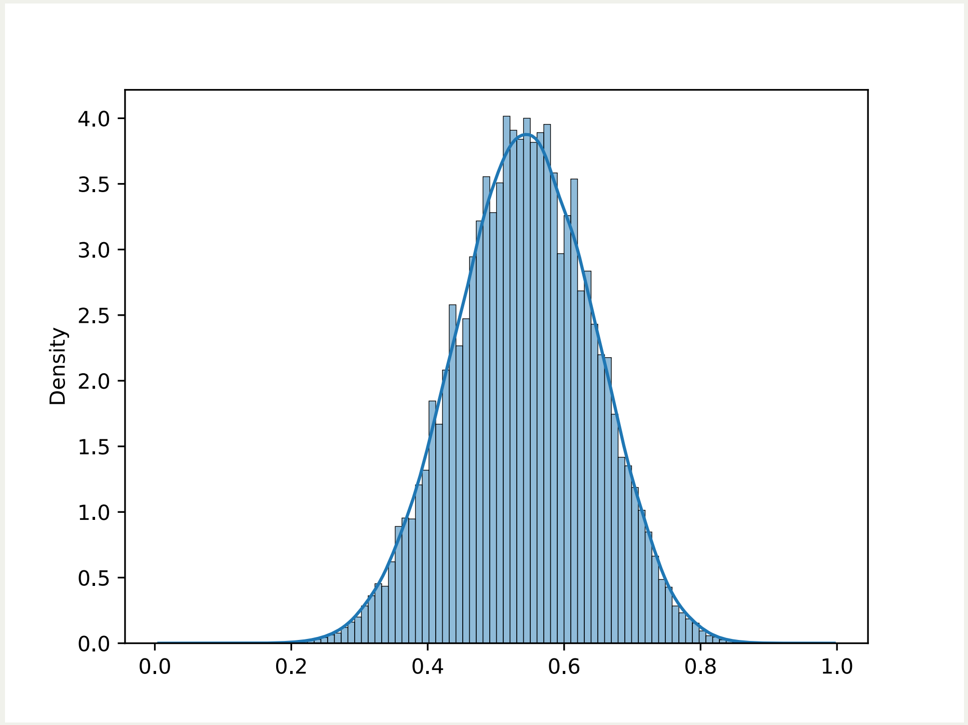
Define prior

```
assume p ~ Beta(2.0, 2.0);
```



Condition on data

```
for i in 1 to (length(outcomes)) {
    observe outcomes[i] ~
Bernoulli(p);
}
```



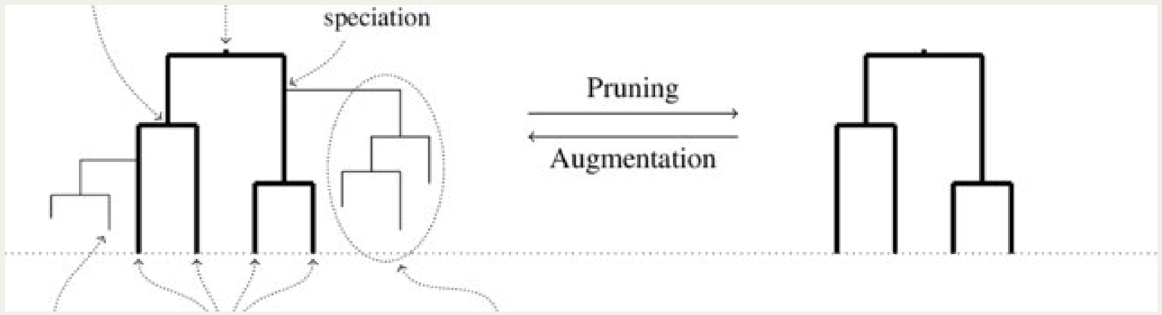
The complete coin example in TreePPL using Python as a wrapper.

```
source = """\
model function coin(outcomes: Bool[]): Real {
  assume p ~ Beta(2.0, 2.0);
  for i in 1 to (length(outcomes)) {
    observe outcomes[i] ~ Bernoulli(p);
  }
  return(p);
}
"""

with treeppl.Model(source=source, samples=10_000, method="is-lw") as coin:
    res = coin(
      outcomes=outcomes
    )
    sns.histplot(
        x=res.samples, weights=res.nweights, bins=100, stat="density", kde=
    )
    plt.show()
```

Universal PPLs

```
function simulateSubtree(time: Real, lambda: Real, mu: Real) {
  assume waitingTime ~ Exponential(lambda + mu);
  if waitingTime > time {
    weight 0.0;
    resample;
  } else {
    assume isSpeciation ~ Bernoulli(lambda / (lambda + mu));
    if isSpeciation {
      simulateSubtree(time - waitingTime, lambda, mu);
      simulateSubtree(time - waitingTime, lambda, mu);
    }
  }
}
```



Simulating unobserved evolutionary lineages

TreePPL

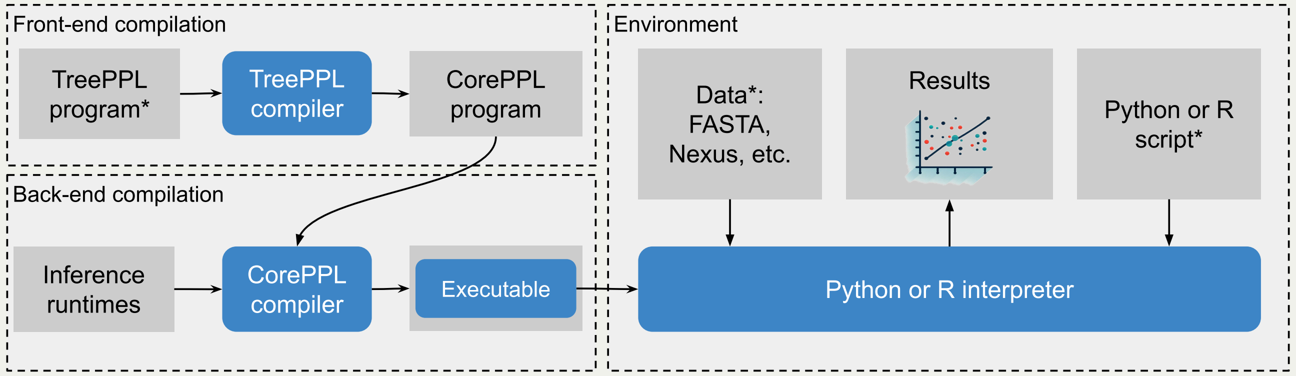
Phylogenetic Data
Supports natively the PhyJSON format for evolutoinary trees

Simplicity
Designed to meet the needs of computational biologists

Rich Model Library
Offers state-of-the art diversificatio
models as templates

Powerful Statistical Inference
Sequential Monte-Carlo (SMC) and Markov-chain Monte-Carlo (MCMC) inference

Architecture



Optimizations

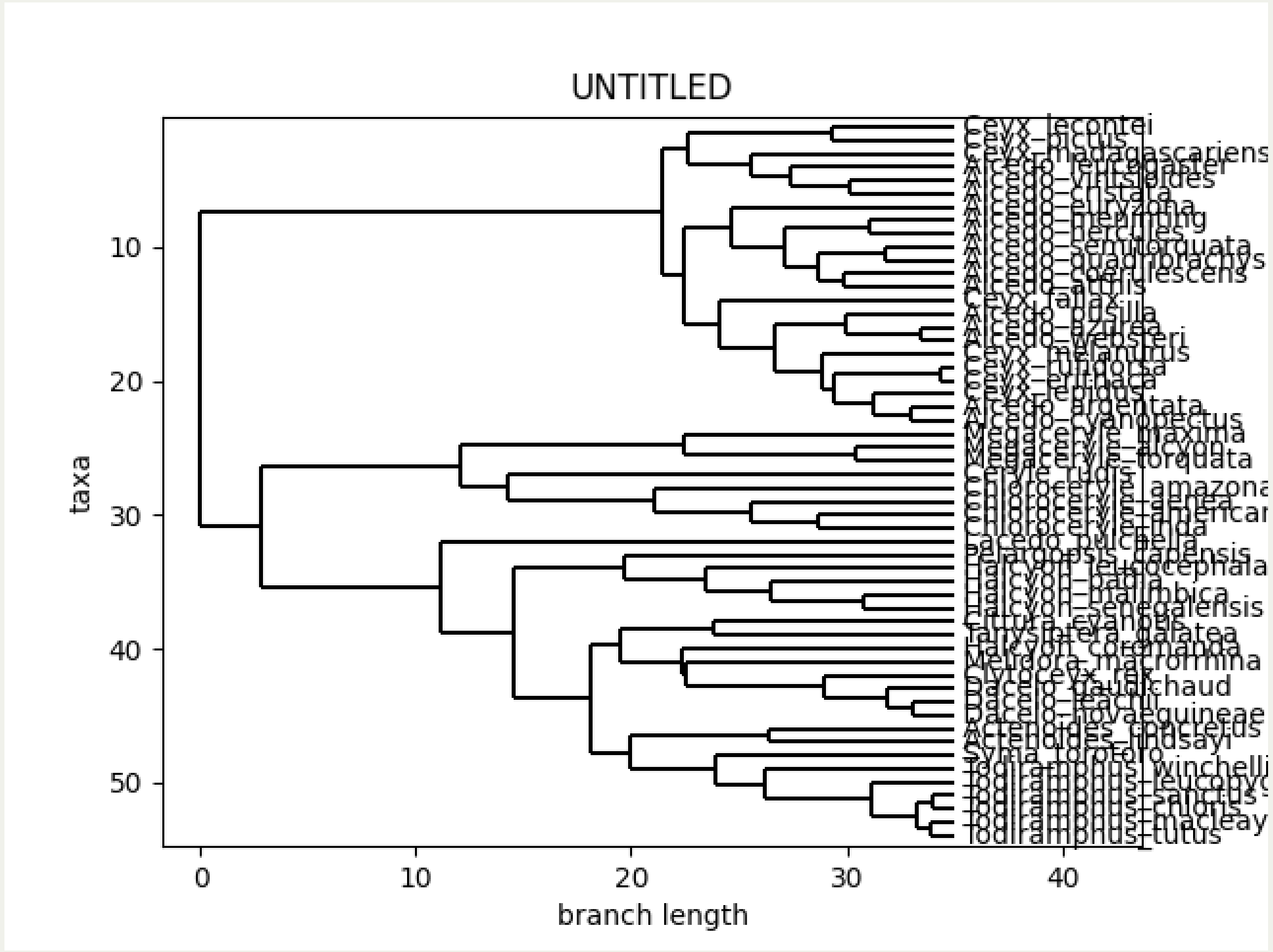
- Partial CPS transformation
- Automatic alignment
- (Delayed sampling/ conjugacy analysis)

Inference strategies

- Lightweight importance sampling
- Sequential Monte Carlo: Bootstrap Particle Filter,
- Sequential Monte Carlo: Alive particle filter
- Lightweight MCMC
- Trace MCMC
- Particle MCMC

Computer demo Challenges

Write a program that infers the bifurcation rate λ and the extinction rate μ under the assumption that they are constant for a given binary evolutionary tree.

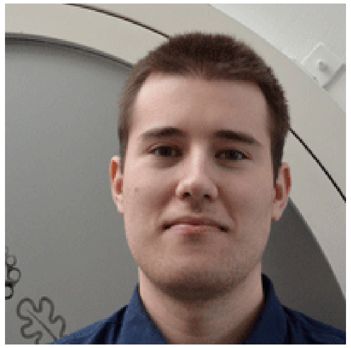

UNTITLED
taxa
branch length
0
10
20
30
40
50
Ceyx_lecontei
Ceyx_pictus
Ceyx_madagascariens
Alcedo_leucogaster
Alcedo_vintsioides
Alcedo_cristata
Alcedo_euryzona
Alcedo_meninting
Alcedo_hercules
Alcedo_semitorquata
Alcedo_quadribrachys
Alcedo_coerulescens
Alcedo_atthis
Ceyx_fallax
Alcedo_pusilla
Alcedo_azurea
Alcedo_websteri
Ceyx_melanurus
Ceyx_rufidorsa
Ceyx_erithaca
Ceyx_lepidus
Alcedo_argentata
Alcedo_cyanopectus
Megaceryle_maxima
Megaceryle_alcyon
Megaceryle_torquata
Ceryle_rudis
Chloroceryle_amazona
Chloroceryle_aenea
Chloroceryle_americar
Chloroceryle_inda
Lacedo_pulchella
Pelargopsis_capensis
Halcyon_leucocephala
Halcyon_badia
Halcyon_malimbica
Halcyon_senegalensis
Cittura_cyanotis
Tanysiptera_galatea
Halcyon_coromanda
Melidora_macrorrhina
Clytoceyx_rex
Dacelo_gaudichaud
Dacelo_leachii
Dacelo_novaeguineae
Actenoides_concretus
Actenoides_lindsayi
Syma_torotoro
Todiramphus_winchelli
Todiramphus_leucopyg
Todiramphus_sanctus
Todiramphus_chloris
Todiramphus_macleay
Todiramphus_tutus

Viktor Senderov

Postdoctoral Researcher at L'École normale supérieure

Jan Kudlicka

Associate Professor of Data Science at BI Norwegian Business School



Daniel Lundén

Senior Member of Technical Staff at Oracle

Viktor Palmkvist

Ph.D. Candidate at KTH Royal Institute of Technology

Thank you

Mariana P. Braga

Postdoctoral Researcher at the Swedish University of Agricultural Sciences

Emma Granqvist

Postdoctoral Researcher at Department of Bioinformatics and Genetics, Swedish Museum of Natural History

Fredrik Ronquist

PI together with Broman (eq. contribution) Department of Bioinformatics and Genetics, Swedish Museum of Natural History

David Broman

PI together with Ronquist (eq. contribution) EECS and Digital Futures, KTH Royal Institute of Technology and Computer Science Department, Stanford University

Acknowledgements

- Funded by the EU Horizon 2020 Marie Skłodowska-Curie grant agreement PhyPPL No. 898120, and by the Swedish Foundation for Strategic Research, as well as Vetenskapsrådet (VR).
- Slides have been designed using images from Flaticon.com as well DALL-E

Downloading

Project homepage: https://treeppl.org

Main repo: https://github.com/treeppl/treeppl

Python library: https://github.com/treeppl/treeppl-python

Paper:
https://www.biorxiv.org/content/10.1101/2023.10.10.561673v1

Citation:

```
TreePPL: A Universal Probabilistic Programming Language for Phylogenetics
Viktor Senderov, Jan Kudlicka, Daniel Lundén, Viktor Palmkvist, Mariana P.
Braga, Emma Granqvist, David Broman, Fredrik Ronquist
bioRxiv 2023.10.10.561673; doi: https://doi.org/10.1101/2023.10.10.561673
```

Bonus material

Discussion of PPLs

- PPL semantics
  - Usual language: *input* $\rightarrow$ *output* + *side effects*
  - PPL: *each program trace describes a sample from a probability distribution*
- Advantages of PPLs
  - natural and generative way of encoding a posterior distribution
  - no need to implement inference
  - enables model comparison as estimation of normalizing constant possible