Code and binaries available at `http://github.com/diyabc/abcranger`

AbcRanger

A fast and scalable random forest library for ABC model choice and parameter estimation

F.-D. Collin [2] A. Estoup [1] J.-M. Marin [2] L. Raynal [2]

[1]CBGP, INRA, CIRAD, IRD, Montpellier SupAgro, Univ. Montpellier [2]Université de Montpellier, CNRS, IMAG UMR 5149

First building block : ABC simulations

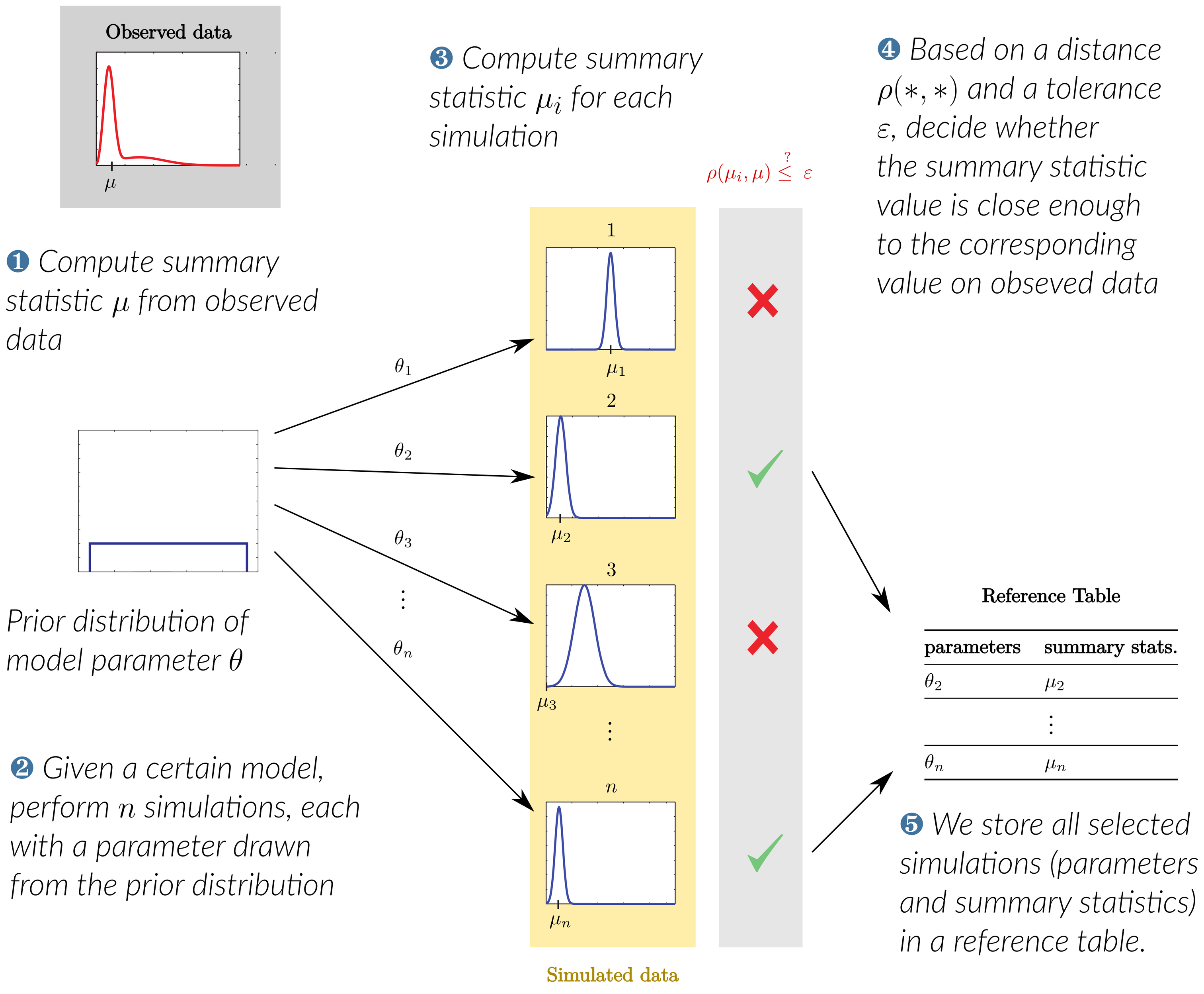

Figure 1. ABC details

Given an observed data, the basic idea of ABC, *Approximate Bayesian Computations* [1], is to approximate the likelihood of a parametrized model with selected simulations, by comparing the observed data and simulated ones via computed *summary statistics*. The table of summary statistics for simulated data is called *the reference table*.

ABC posterior methodologies

Model choice: Simulate data for several models and *choose the best model to fit our data*

Parameter estimation: Simulate data for one model and *infer one or several parameters for this model given the observed data*

A sensible workflow is to first choose a model and then infer its parameters.

❶ *Compute simulations with several models, and the reference table with model-indexed lines using a simulator (DIYAC, PyABC etc.)*

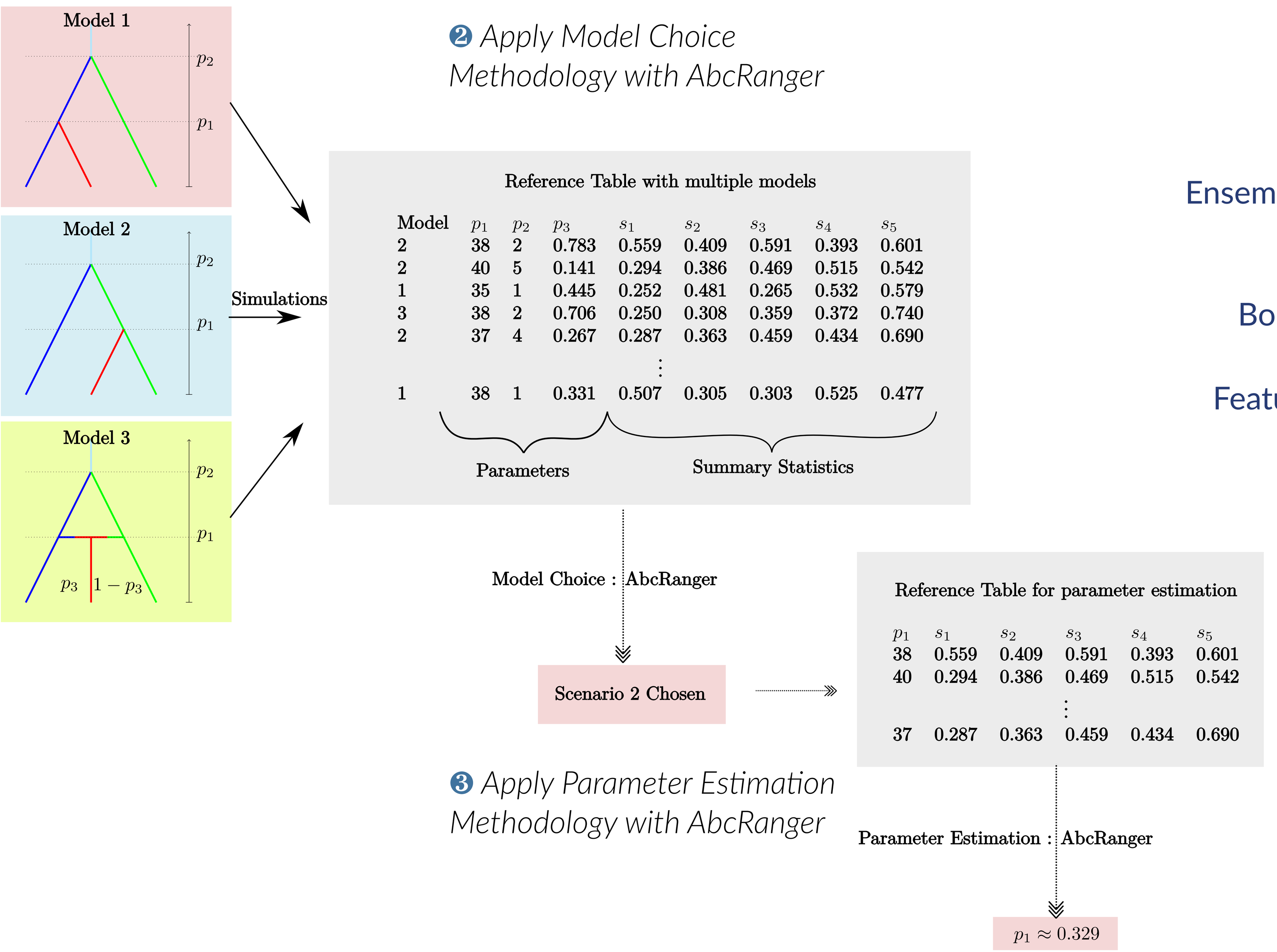

Figure 2. ABC workflow with AbcRanger

Challenges of ABC

in the context of population genetics recent advances

Number of simulated data : could be > 100 000

Number of summary statistics : could range from several hundred to tens of thousands (scenario with several populations and combinatory "explosion") : how to select the *meaningful* ones?

Classical Methods for ABC (k-nn and local methods) doesn't cope very well with this situation.

Our solution

[2] and [3] proposed a novel approach, relying on *Random Forests* to provide both model choice and parameter estimation methodologies

Second building block : Random Forests

CART

Random Forests are based on a CART, *Classification and Regression Trees*, algorithm [4].

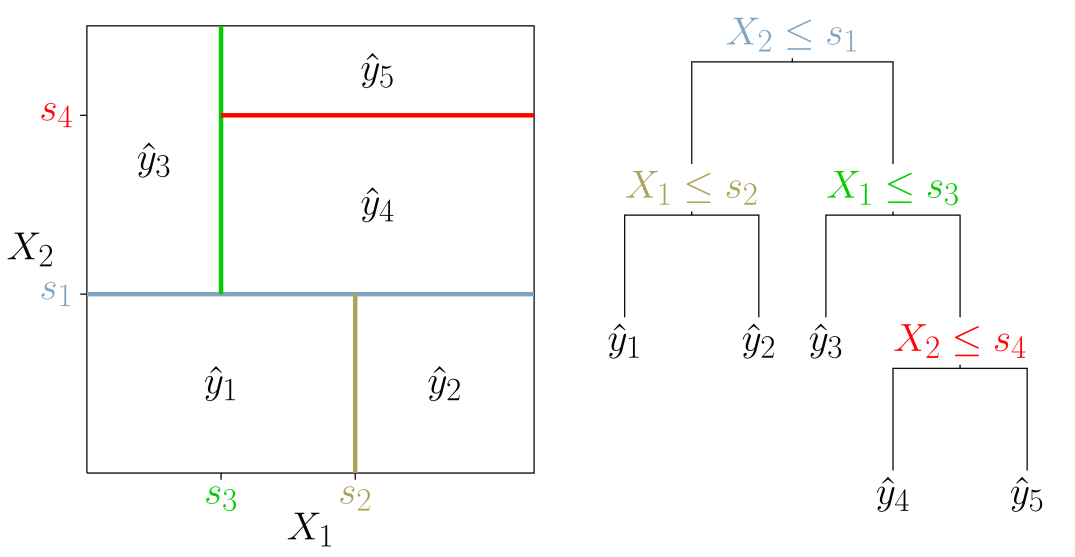

Figure 3. An example of CART and the associated partition of the two dimensional predictor space. Each splitting condition takes the form $X_j \leq s$ and the prediction at a leaf is denoted $\hat{y}_\ell$.

A CART is a *machine learning algorithm* whose principle is to partition the predictor space into disjoint subspaces, in an iterative manner, and each one is assigned a prediction value which will be used for test data falling in this subspace.
Once the partitioning is done, we have a binary tree structure which could predict outcomes from an input data, either classes or continuous values.

Random Forests

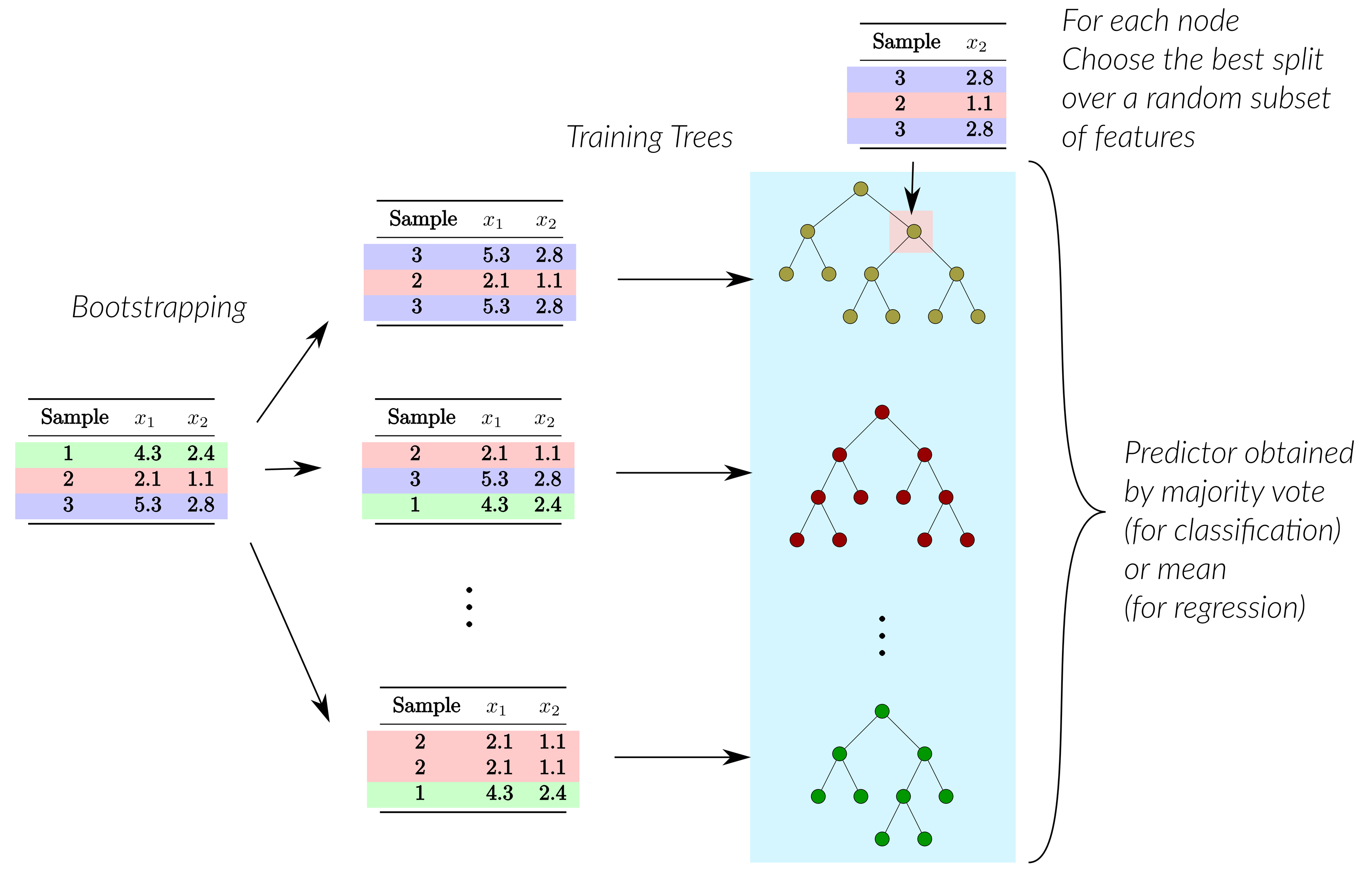

Figure 4. Random Forest

Random Forests [5] are a three pronged extension of CART:

Ensemble method Training a *set* of CART (not just one), and getting the majority vote (resp. mean) for classification (resp. regression)

Bootstrapping Training data is random sampled (with replacement) for *each* tree

Feature bagging At each node of a growing tree, find the best split on a random subset of the features

Advantages in an ABC setting :

- robust to noise
- (almost) free variable importance
- free (out-of-bag) cross-validation procedure
- easy parallelization
- good scaling properties (samples and features)
- classifier and regressor (both are used)

Computational challenges with ABC/Random Forests

With 100 000 lines and more than 10 000 summary statistics, each tree could reach over 1 gigabyte of memory size. Typically we need 500 or 1000 trees for good prediction performance, so, even with state of the art RF packages like [6], memory constraints are preventing completion of the training.

A new implementation of Random Forest for ABC

Since ABC procedures only use trained Random Forests on a known set of observations, we have altered the random forest training computation by using only a subset of in-memory trees at a time and accumulating the required outcomes (predictions and statistics). Memory footprint is vastly improved and there is no performance cost.

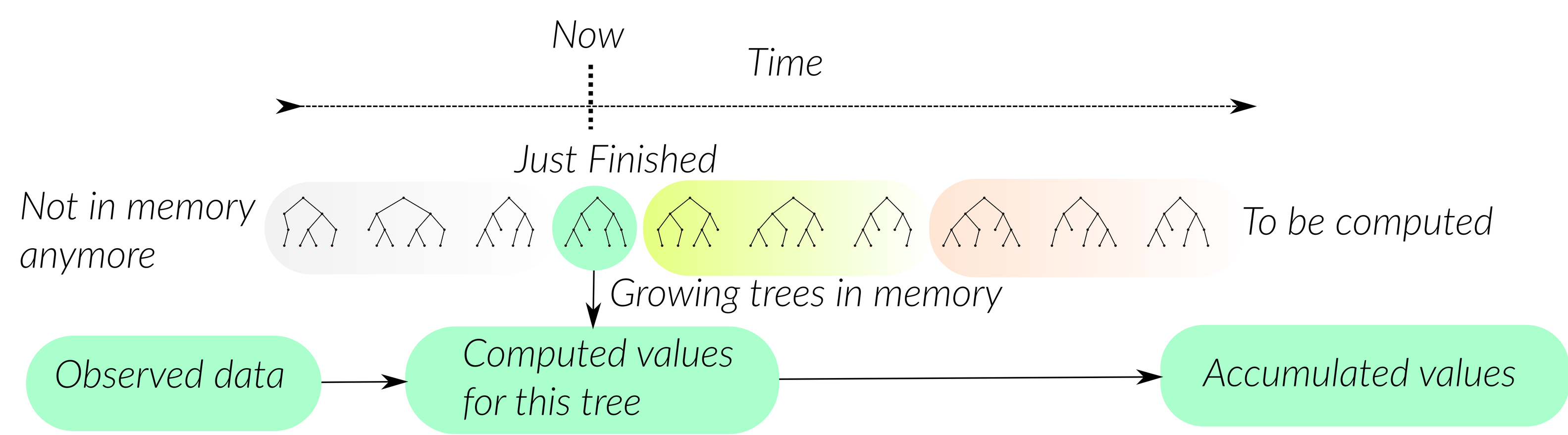

Figure 5. Window of growing trees

Ongoing project *LeafLitter* intends to pursue that line even further: for a growing tree, only encountered leaves are stored. Thus, the memory footprint of the trees becomes negligible, and their growing could finally be parallelized at full scale.

References

[1] Jean-Michel Marin, Pierre Pudlo, Christian P Robert, and Robin J Ryder. Approximate bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.

[2] Pierre Pudlo, Jean-Michel Marin, Arnaud Estoup, Jean-Marie Cornuet, Mathieu Gautier, and Christian P Robert. Reliable abc model choice via random forests. *Bioinformatics*, 32(6):859–866, 2015.

[3] Louis Raynal, Jean-Michel Marin, Pierre Pudlo, Mathieu Ribatet, Christian P Robert, and Arnaud Estoup. ABC random forests for Bayesian parameter inference. *Bioinformatics*, 35(10):1720–1728, 10 2018.

[4] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. The Wadsworth and Brooks-Cole statistics-probability series. Taylor & Francis, 1984.

[5] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] Marvin N Wright and Andreas Ziegler. Ranger: a fast implementation of random forests for high dimensional data in c++ and r. *arXiv preprint arXiv:1508.04409*, 2015.