Supplementary Material for Scalable Inference on Kingman's Coalescent using Pair Similarity

Using `GreedyNN` to Search Over Parameters

Computation time and sample quality of the proposed algorithm `SMCnn` depends on the number of pairs R to do the costly computations for and the number of neighbors k to retrieve for the priority queue. The quality of the proposal distribution will improve with increasing R (and k) at the cost of increased computational time. It is important to use a parameter setting for which the proposal distribution is close to the true distribution, but the computational savings are still enough to allow using a large number of particles. The greedy counterpart of our proposed fast inference scheme can be utilized as a tool to guide the choice of parameters prior to sequential Monte Carlo sampling utilizing several particles. Figure 1 shows the change in log joint probability $\log P(X, \pi)$ as a function of R.[1] We see that the joint probability is low initially but it quickly increases with R. We choose to use the R value for which the performance of `GreedyNN` saturates.

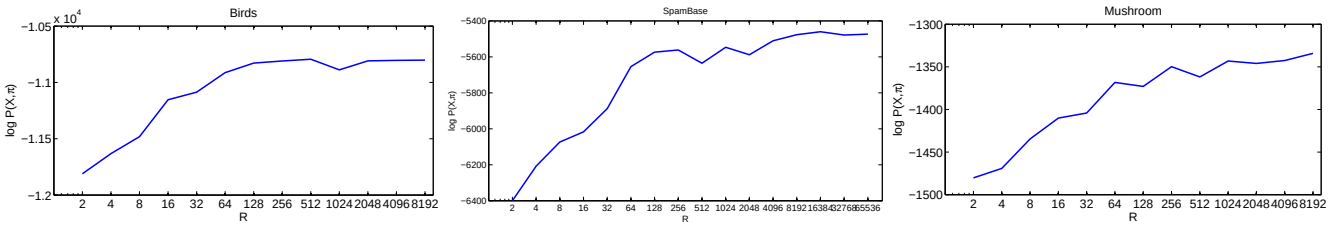

Figure 1: Change in joint probability as a function of R. **left:** Birds: 375 data points of 25 categories, **middle:** SpamBase: 256 data points, **right:** Mushroom: 128 data points.

Comparison of `GreedyNN` and `GreedyRate1` Trees

Different algorithms have different biases and therefore may result in samples with different characteristics in the finite sample or the greedy case. Figure 2 shows trees obtained by the two algorithms `GreedyNN` and `GreedyRate1` to point out this difference. The `GreedyNN` tree has a structure more reminiscent of samples from the coalescent with exponentially increasing branch lengths whereas `GreedyRate1` branching structure resembles a classical linkage algorithm output.

[1] We used $k = \min(5, R/20)$ for these experiments, but a grid search over k can also be done in a similar manner.

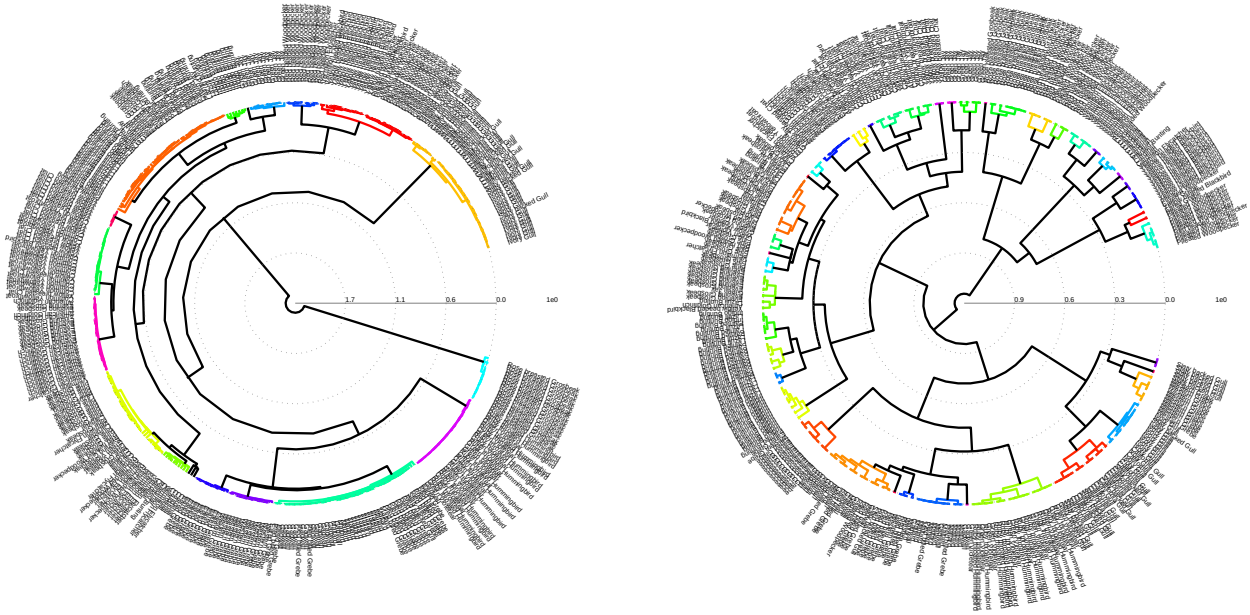

Figure 2: Trees obtained using `GreedyNN` (left) and `GreedyRate1` on the $n = 375$ subset of Birds200. This subset has 25 categories, with 15 data points in each category. The category label (names of the bird) is shown at the leaves.

Figures 3-5 show the cumulative coalescent time (branch length) information in a different representation than a dendrogram for the three datasets we consider in the paper. The numbers pointing to the curves are the log joint probability of the data and the tree for which the branch lengths are shown. It is clear from these figures that the different algorithms have different coalescent time sample characteristics. `GreedyRate1` always seems to disagree with the other algorithms. It is interesting to note that `GreedyNN SMCnn` and `PostPost` samples are quite similar, and `SMC1` agrees with these only on the Birds200 data where it achieved a similar level of performance (Figure 4 in the main paper). It would be informative to look into the difference in behavior of `SMC1` and find out how many particles would be necessary for it to achieve the same level of performance for SpamBase and Mushroom datasets. We believe its sample characteristics will be similar to the other algorithms in this regime.

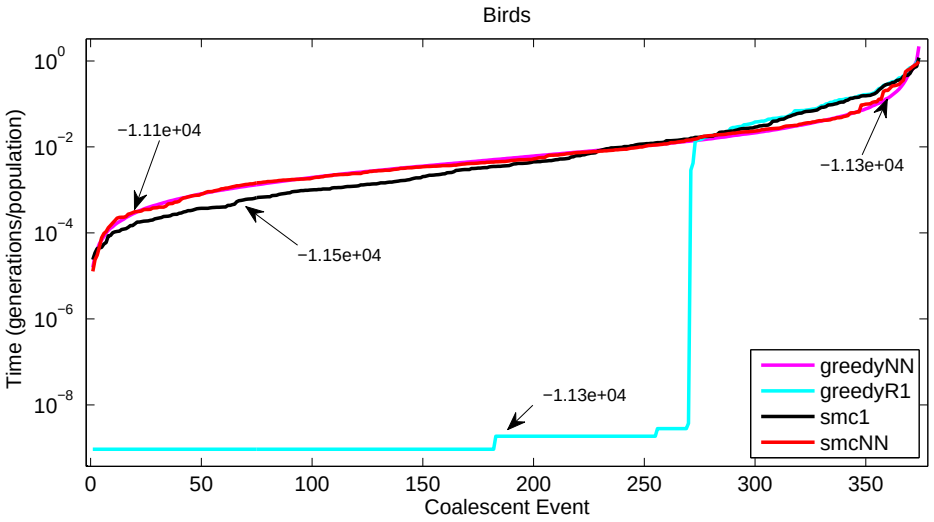

Figure 3: Coalescent times (cumulative branch lengths) of the different algorithms as a function of iterations for the $n = 375$ subset of Birds200 dataset.

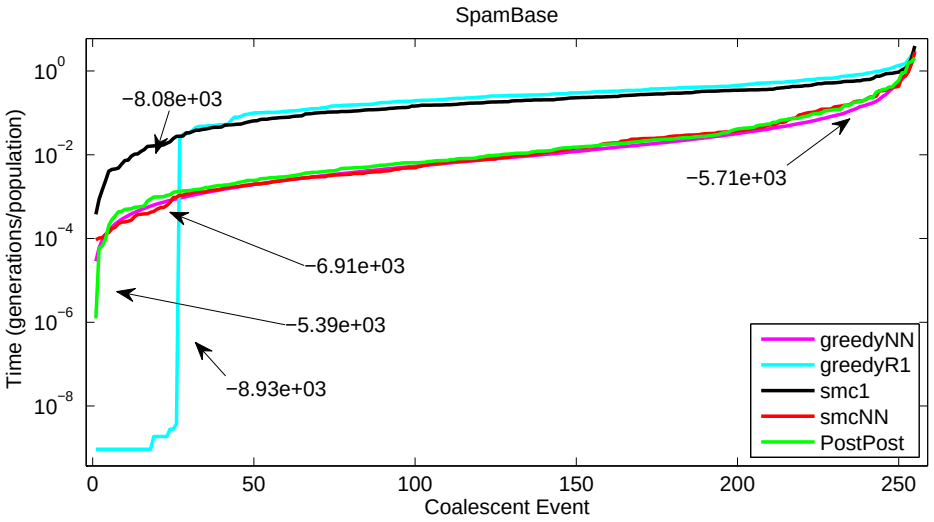

Figure 4: Coalescent times (cumulative branch lengths) of the different algorithms as a function of iterations for the $n = 256$ subset of SpamBase dataset.

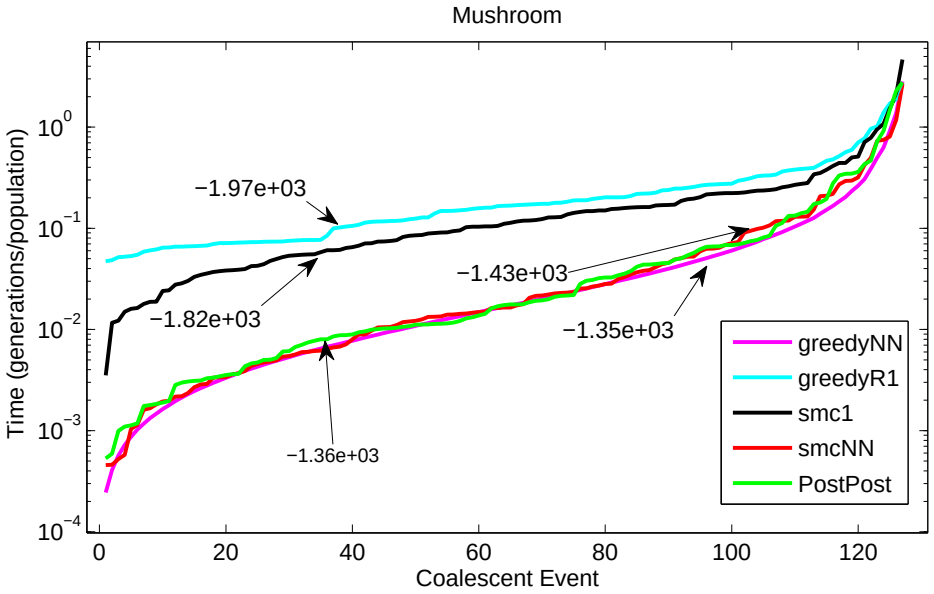

Figure 5: Coalescent times (cumulative branch lengths) of the different algorithms as a function of iterations for the $n = 128$ subset of Mushroom dataset.