perfectphyloR: an R package for perfect phylogeny

Charith Bhagya Karunarathna[1] and Jinko Graham[1]

[1]Simon Fraser University

Abstract

A perfect phylogeny is a rooted binary tree that recursively partitions DNA sequences according to their ancestry. We present our progress on an R package perfectphyloR which reconstructs the local perfect phylogenies underlying a sample of sequences. We first partition the sequences using the algorithm of Gusfield and then further partition them using heuristics introduced by Mailund. The reconstructed perfect phylogenies for a sample of sequences can provide insight into their patterns of ancestry. For example, disease sequences may cluster together on a local ancestral tree indicating that they arise from a common ancestral mutation. The perfectphyloR package should therefore be useful to researchers seeking the ancestral structure of their sequence data in order to associate it with disease phenotypes. We will present examples of useful functions that we are developing for the package to explore the sequence data and associate its ancestry to phenotypes.

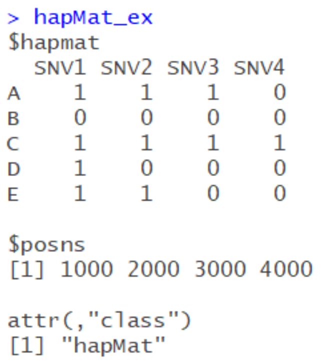
Background

A perfect phylogeny is a rooted binary tree that may be used to explain a set of compatible single-nucleotide variants (SNVs). A perfect phylogeny represents a recursive partitioning of a set of objects such as DNA sequences. Though they are not ancestral trees, their nested partition structures provide insight into the pattern of ancestry of DNA sequence data. Further, the perfect phylogeny near a disease-affecting mutation can provide useful information about the disease affected or unaffected classification of a sequence (Mailund et al., 2006). For example, in a case-control study design, case alleles may tend to cluster together on a partition for a variant that influences disease susceptibility. If a partition has a larger number of case sequences than other partitions, this suggests an association between the disease and the group of sequences belonging to that partition (Bardel et al., 2005). Also, the SNVs defining a partition that contains an excess of case sequences are good candidates to be involved in disease susceptibility. Thus, an R package to reconstruct perfect phylogenies from sequence data can be of use to researchers seeking insight into genetic causes of disease susceptibility.

We develop an R package perfectphyloR to reconstruct perfect phylogenies at a focal single nucleotide variant (SNV), underlying a sample of DNA sequences. We implement the partitioning of DNA sequences using the classic algorithm of Gusfield (1991), and then further partition them using heuristics introduced by Mailund et al. (2006). Starting from the focal SNV, we expand the neighborhood of compatible SNVs until we find an incompatible SNV. However, when the block of compatible SNVs is smaller than a user-defined minimum size, following Mailund et al. (2006), we expand the block to include incompatible SNVs in order of proximity to the focal SNV. Once we identify the neighborhood of SNVs, following Gusfield (1991), we order a set of compatible SNVS from the most ancient to the most recent. We then construct the perfect phylogeny by recursive partitioning on SNVs. The recursive partitioning algorithm first partitions on the most ancient SNV, and then recursively moves towards the present, partitioning at each SNV it encounters until either running of out of SNVs or until each partition consists of a single sequence.

Methods

Given a matrix of haplotypes, with haplotypes along rows and SNVs down the columns, we first construct a `hapMat` data object with the function `newHapMat()` and use this data object throughout our work. The convention is that at each SNV the ancestral and derived alleles are coded as 0 and 1, respectively. An example of a `hapMat` data object for five sequences and four SNVs is given in Figure 1.



```
> hapMat_ex
$hapmat
  SNV1 SNV2 SNV3 SNV4
A    1    1    1    0
B    0    0    0    0
C    1    1    1    1
D    1    0    0    0
E    1    1    0    0

$posns
[1] 1000 2000 3000 4000

attr(,"class")
[1] "hapMat"
```

Figure 1: An example of five sequences with four SNVs.

For the `hapMat` data object, users are required to specify;

- `hapmat`, a matrix of 0's and 1's, with rows representing haplotypes and columns representing SNVs.
- `snvNames`, a vector of names of SNVs for the columns of `hapmat`.
- `hapNames`, a vector of names of haplotypes for the rows of `hapmat`.
- `posns`, a numeric vector specifying the genomic positions (e.g. in base pairs) of SNVs in the columns of `hapmat`.

With the main function `MPPtree()`, we reconstruct the perfect phylogeny at a user-given focal SNV position using the algorithm of Gusfield (1991) to obtain the perfect phylogeny for compatible SNVs, and the modifications of Mailund et al. (2006) to include the incompatible SNVs that are nearby. The result is a `phylo` object, a data structure defined by the R package `ape` (Paradis et al., 2004) to represent binary trees. Storing the resulting partition of sequences as a `phylo` object enables the application of functions from `ape` to plot and summarize it. For example, the perfect phylogeny constructed from the `hapMat` object in Figure 1 is plotted in Figure 2.

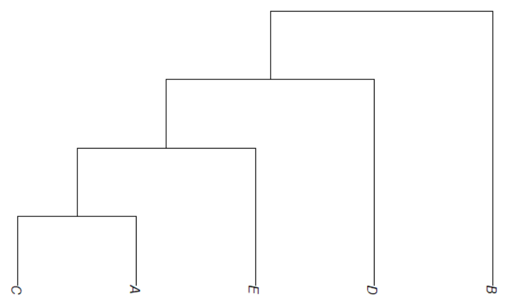


Figure 2: Figure showing the reconstructed perfect phylogeny at SNV3 for sequence data shown in Figure 1.

The main function `MPPtree()` of our implementation consists of the following major steps:

1. Select a window of SNVs at a given focal SNV with `selectWindow()`.
2. Build the partition of sequences for the window of SNVs with `buildTree()`
   (a) Order the SNVs with `orderSNVs()`.
   (b) Apply recursive partitioning on SNVs with `makeTree()`.

Starting at the given focal SNV, the function `selectWindow()` expands the neighborhood of compatible SNVs until it finds an incompatible SNV. Note that we judge our neighborhood blocks in terms of SNV compatibility according to the Four-Gamete Test (Hudson and Kaplan, 1985). If the incompatible SNV is to the left of the neighborhood, this function tries to extend the neighborhood to the right and vice versa. However, when the neighborhood of compatible SNVs about the focal SNV is too small for the user, we expand the neighborhood by including incompatible SNVs in order of proximity from the focal SNV. Once the window of SNVs about the focal SNV is selected, following Gusfield (1991), the function `orderSNVs()` orders the compatible SNVs by ancestry. Then following Mailund et al. (2006), it orders by incompatible SNVs according to their proximity to the focal SNV. With the ordered SNVs, the function `makeTree()` finally builds the perfect phylogeny at the focal SNV using recursive partitioning on the SNVs in the neighborhood.

We also include two extra functions `MPPdist()` and `Randindex()` in the package. `MPPdist()` returns a matrix of pairwise distances between haplotypes. We use the function `cophenetic.phylo` in the `ape` package to calculate pairwise distances between tips of the reconstructed partition of sequences obtained from `MPPtree()`. `Randindex()` computes the Rand index (Rand, 1971) which is a measure between 0 and 1 reflecting the similarity between two partitions.

Discussion

`perfectphyloR` is an R package that allows users to reconstruct local perfect phylogenies from DNA sequences. The main function `MPPtree()` reconstructs a perfect phylogeny at a given focal SNV, using the methods described in Gusfield (1991) and Mailund et al. (2006). Following the function `MPPregion()`, user can also reconstruct perfect phylogenies across a region of SNVs represented by the `hapMat` data object.

After reconstructing the perfect phylogeny at a given focal SNV, we offer functions for the user to compute the pairwise distances between haplotypes with the function `MPPdist()`. The pairwise distances between sequences are calculated based on the reconstructed partition. The reconstructed perfect phylogenies at different focal SNVs can be examined for clustering consistent with the case-control classification of sequences. Focal SNVs which display such clustering are candidates for disease-susceptibility variants. In this way, `perfectphyloR` is useful to explore the disease association with SNVs.

When the true partitions are known, for example from a simulated dataset, this package allows users to understand the accuracy of the reconstructions by computing a measure of similarity between the true and reconstructed partitions with the function `Randindex()`. The `Randindex()` function also allows users to measure similarity between two reconstructed partitions.

References

Claire Bardel, Vincent Danjean, Jean-Pierre Hugot, Pierre Darlu, and Emmanuelle Génin. On the use of haplotype phylogeny to detect disease susceptibility loci. *BMC Genetics*, 6(1):24, 2005. doi: 10.1186/1471-2156-6-24. URL `https://doi.org/10.1186%2F1471-2156-6-24`.

Dan Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21(1):19–28, jan 1991. doi: 10.1002/net.3230210104. URL `https://doi.org/10.1002%2Fnet.3230210104`.

RR Hudson and NL Kaplan. Statistical properties of the number of recombination events in the history of a sample of DNA sequences. *Genetics*, 111:147–64, Sep 1985.

Thomas Mailund, Søren Besenbacher, and Mikkel H Schierup. Whole genome association mapping by incompatibilities and local perfect phylogenies. *BMC Bioinformatics*, 7(1):454, 2006. doi: 10.1186/1471-2105-7-454. URL `https://doi.org/10.1186%2F1471-2105-7-454`.

E. Paradis, J. Claude, and K. Strimmer. APE: Analyses of Phylogenetics and Evolution in R language. *Bioinformatics*, 20(2):289–290, jan 2004. doi: 10.1093/bioinformatics/btg412. URL `https://doi.org/10.1093%2Fbioinformatics%2Fbtg412`.

William M. Rand. Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association*, 66(336):846–850, dec 1971. doi: 10.1080/01621459.1971.10482356. URL `https://doi.org/10.1080%2F01621459.1971.10482356`.