How to draw a phylogenetic tree

Zuguang Gu <z.gu@dkfz.de>

October 1, 2013

In this short vignette, we would show you how to customize the circular style of phylogenetic tree through *circlize* package. Basically, a phylogenetic tree is a dendrogram which is a combination of lines. In R, there are several classes that describe such type of tree such as `hclust`, `dendrogram` and `phylo`. In this example, we will show how to draw the tree from the `dendrogram` class. Nevertheless, other classes can be converted to `dendrogram` with not too much difficulty.

The data we are using here is from `ape` package. The `bird.orders` data set is some kind related to species of birds. We also extract some other values such as the label for each bird (which is its name.). We split the tree into six categories by `cutree` and finally we convert the data into a `dendrogram` object.

```
> library(ape)
> data(bird.orders)
> hc = as.hclust(bird.orders)
> labels = hc$labels
> ct = cutree(hc, 6)  # cut tree into 6 pieces
> n = length(labels)
> hc = as.dendrogram(hc)
```

The `dendrogram` class stores data recursively which means, a `dendrogram` object contains two children nodes which are also `dendrogram` objects. Thus, we can implement drawing dendrogram tree in a recursive way as well.

For each node in the dendrogram, there are several attributes which are `members`, `midpoint`, `height`, and if the node is the leaf of the tree (i.e. the end of the tree), there would be an binary attribute called `leaf`. Here what shoud be noted is that `midpoint` is the distance to its two children nodes while not the coordinate of the point.

As we mentioned in the main vignette, the x-value for the phylogenetic tree is in fact index. Thus, the x-lim is just the minimum and maximum index of labels in the tree. Since there is only one phylogenetic tree, we only need one "big" sector. And in the first track, we draw the name of each bird, with different color to represent different category.

```
> library(circlize)
> par(mar = c(1, 1, 1, 1))
> circos.par(cell.padding = c(0, 0, 0, 0))
> circos.initialize("a", xlim = c(0, n))
> maxy = attr(hc, "height")
> circos.trackPlotRegion(ylim = c(0, 1), bg.border = NA, track.height = 0.3,
+     panel.fun = function(x, y) {
+         for(i in seq_len(n)) {
+             circos.text(i-0.5, 0, labels[i], adj = c(1, 0.5),
+                 direction = "vertical_left", col = ct[labels[i]], cex = 0.7)
+         }
+ })
```

In the second track, we are going to draw the circular dendrogram. Here we implement the code in a recursive way. The `draw.d` function add lines to its two children nodes, and once it reaches the leaf of the tree, the recursive executation will stop. You can see the advantage of *circlize* that if you replace `circos.lines` to `lines`, then you can almost use the function to draw dendrogram in the regular Cartesian coordinate system.

```
> circos.dendrogram = function(dend, maxy=attr(dend, "height")) {
+     labels = as.character(labels(dend))
+     x = seq_along(labels) - 0.5
+     names(x) = labels
+
+     is.leaf = function(object) (is.logical(L <- attr(object, "leaf"))) && L
+
+     draw.d = function(dend, maxy) {
+         leaf = attr(dend, "leaf")
+         d1 = dend[[1]]
+         d2 = dend[[2]]
+         height = attr(dend, 'height')
+         midpoint = attr(dend, 'midpoint')
+
+         if(is.leaf(d1)) {
+             x1 = x[as.character(attr(d1, "label"))]
+         } else {
+             x1 = attr(d1, "midpoint") + x[as.character(labels(d1))[1]]
+         }
+         y1 = attr(d1, "height")
+
+         if(is.leaf(d2)) {
+             x2 = x[as.character(attr(d2, "label"))]
+         } else {
+             x2 = attr(d2, "midpoint") + x[as.character(labels(d2))[1]]
+         }
+         y2 = attr(d2, "height")
+
+         circos.lines(c(x1, x1), maxy - c(y1, height), straight = TRUE)
+         circos.lines(c(x1, x2), maxy - c(height, height))
+         circos.lines(c(x2, x2), maxy - c(y2, height), straight = TRUE)
+
+         if(!is.leaf(d1)) {
+             draw.d(d1, maxy)
+         }
+         if(!is.leaf(d2)) {
+             draw.d(d2, maxy)
+         }
+     }
+
+     draw.d(dend, maxy)
+ }
```

And add the circular dendrogram to the second track:

```
> circos.trackPlotRegion(ylim = c(0, maxy), bg.border = NA,
+     track.height = 0.4, panel.fun = function(x, y) {
+         circos.dendrogram(hc, maxy)
+ })
> circos.clear()
```

Final figure is figure 1. We appologize that maybe the above script is not so well coded, but at least you can see the basic idea behind the code. And Once you know the basic rule for drawing such circular dendrogram, it would be more flexible to customize your figure such as adding bars, lines, points, or even highlighting some part of the tree.

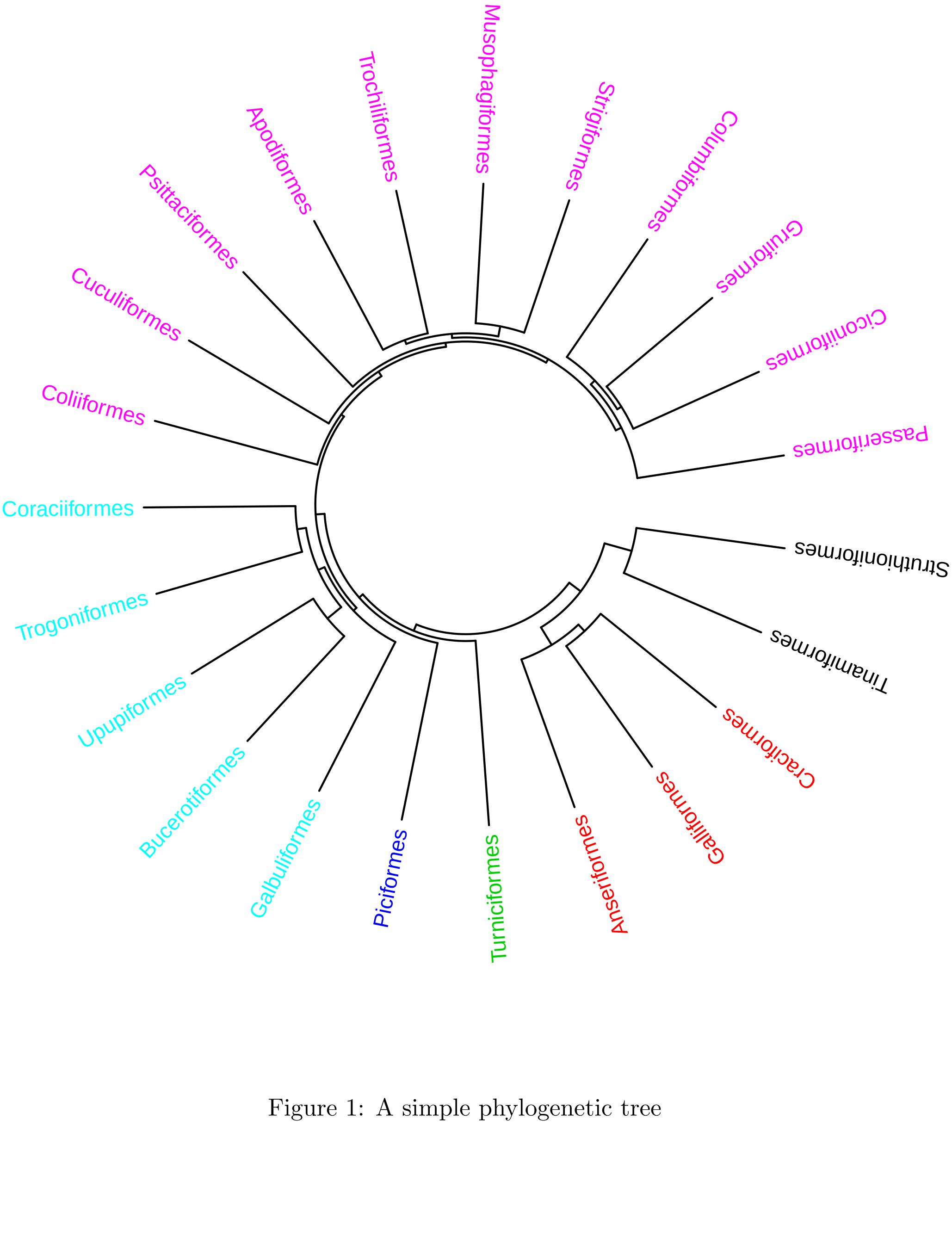

Figure 1: A simple phylogenetic tree