Supplemental File of

Two methods for mapping and visualizing associated data on phylogeny using ggtree

Guangchuang Yu, Tommy Tsan-Yuk Lam, Huachen Zhu, Yi Guan

*correspondence: guangchuangyu@gmail.com, yguan@hku.hk

1. Examples of mapping and visualizing associated data on phylogenetic trees

`ggtree` (Yu et al. 2017) supports tree with data parsed by `treeio` package and thus supports evolutionary statistics inferred by commonly used software packages including `ASTRAL` (Mirarab and Warnow 2015), `BEAST` (Bouckaert et al. 2014), `EPA` (Berger, Krompass, and Stamatakis 2011), `HyPhy` (Pond, Frost, and Muse 2005), `IQ-TREE` (Nguyen et al. 2015), `MrBayes` (Huelsenbeck and Ronquist 2001), `PAML` (Yang 2007), `PHYLODOG` (Boussau et al. 2013), `pplacer` (Matsen, Kodner, and Armbrust 2010), `r8s` (Sanderson 2003), `RAxML` (Stamatakis 2014) and `RevBayes` (Höhna et al. 2014) to be used to annotate the tree. In addition, `ggtree` supports tree objects defined by other R packages, including `obkData`, `phyloseq`, `phylo4` and `phylo4d`, so that associated data stored in these objects (*e.g.* outbreak data in `obkData`, microbiome data in `phyloseq` and matrix in `phylo4d`) can be used directly to annotate the tree (see Appendix of (Yu et al. 2017), Fig. S3 and S8A). The methods introduced in this paper emphasize on integrating **external data**. Here, we present several examples to demonstrate how data can be mapped and visualized to annotate phylogenetic trees. More examples can be found on published figures that collected on `ggtree` project website[1].

1.1. Integrating node/edge data

This example uses method 1, the `%<+%` operator, to integrate taxon (`tip_data.csv`) and internal node (`inode_data.csv`) information and map the data to different colors or shapes of symbolic points and labels. The tip data contains `imageURL` that links to online figures of the species, which can be parsed and used as tip labels in `ggtree`. `ggtree` supports labeling both internal or external nodes using local or online image files such as those deposited on phylopic database[2]. This also provides a solution of using subplots to annotate the tree. Online tools such as `iTOL` (Letunic and Bork 2007) and `EvolView` (He et al. 2016) support displaying subplots on phylogenetic tree. However only bar and pie charts are supported by these tools. Users may want to visualize node-associated data with other visualization methods, such as violin plot (Grubaugh et al. 2017), venn diagram (Lott et al. 2015), sequence logo *etc*, and display them on the tree. In `ggtree`, all kinds of subplots are supported as we can export all subplots to image files and use them to label corresponding nodes on the tree.

Although the data integrated by `%<+%` operator in `ggtree` is for tree visualization, the data attached to the `ggtree` graphic object can be converted to `treedata` object that contains the tree and the attached data. The data stored in the `treedata` object can be used directly to annotate the tree in `ggtree`.

```
## convert ggtree object to treedata object
y <- as.treedata(p3)

## all the data in 'tip_data.csv' and 'inode_data.csv' was incorporated
print(y)
```

```
## 'treedata' S4 object'.
##
```

[1] https://guangchuangyu.github.io/software/ggtree/gallery/
[2] http://phylopic.org/

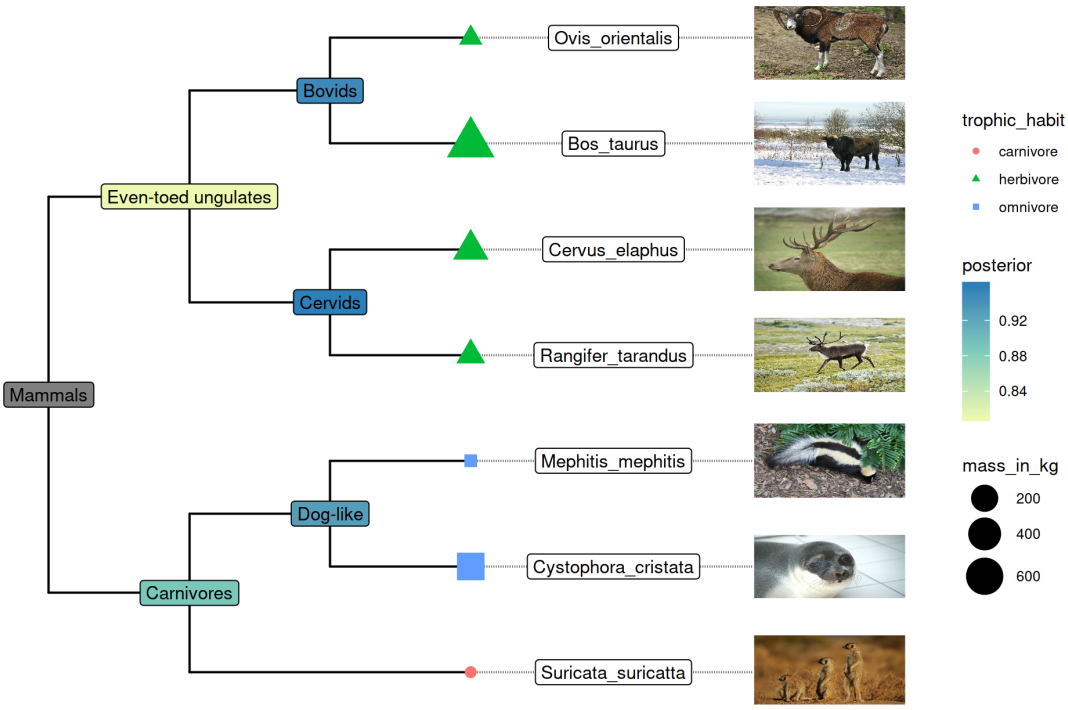

Fig. S1: Example of attaching multiple datasets and labelling taxa with image files

```
## ...@ phylo:
## Phylogenetic tree with 7 tips and 6 internal nodes.
##
## Tip labels:
##  Rangifer_tarandus, Cervus_elaphus, Bos_taurus, Ovis_orientalis, Suricata_suricatta, Cystophora_cris
## Node labels:
## [1] "Mammalia"     "Artiodactyla" "Cervidae"     "Bovidae"
## [5] "Carnivora"    "Caniformia"
##
## Rooted; includes branch lengths.
##
## with the following features available:
##  'vernacularName.x', 'imageURL', 'imageLicense', 'imageAuthor',
##  'infoURL.x',    'mass_in_kg',   'trophic_habit',    'ncbi_taxid',   'rank.x',
##  'vernacularName.y', 'infoURL.y',    'rank.y',   'bootstrap',    'posterior'.
```

```
## Now all the features (from external data attahced to the tree), that stored in
## tree object can be used to annotate the tree.
ggtree(y) + geom_tiplab(aes(label = vernacularName.x, color= trophic_habit)) +
    geom_label(aes(x=branch, label = vernacularName.y, fill=bootstrap)) +
    scale_fill_gradientn(colors=RColorBrewer::brewer.pal(3, "YlGnBu")) +
    theme(legend.position="right") + xlim(-.1, 3.5)
```

The `treedata` object can be exported to `BEAST` compatible `NEXUS` file[3], which can be parsed by `treeio` or `FigTree`[4]. This creates the possibility to integrate external data with the tree-associated data into a single file and visualize the data not only with `ggtree`, but also `FigTree`, *etc.*

[3]https://bioconductor.org/packages/release/bioc/vignettes/treeio/inst/doc/Exporter.html
[4]http://tree.bio.ed.ac.uk/software/figtree/

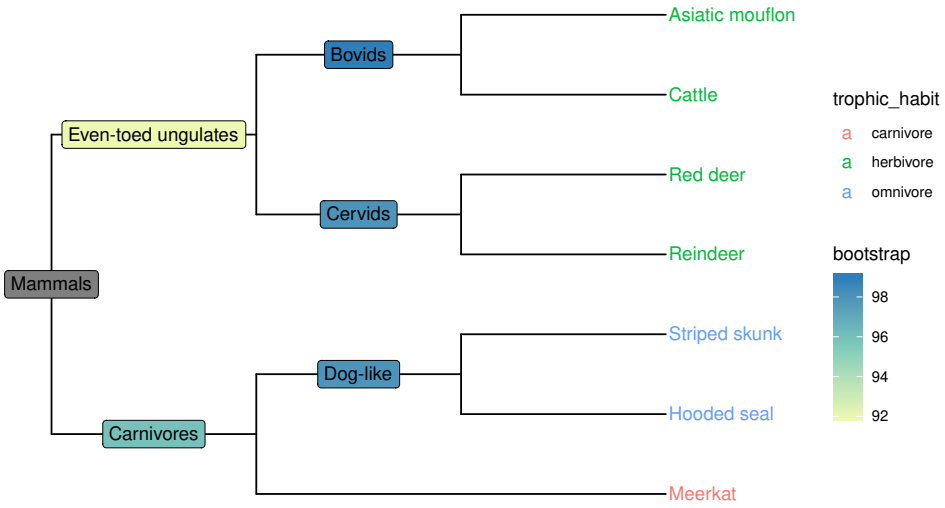

Fig. S2: Example of visualizing treedata object that contains tree with associated data

1.2. Visualizing species abundance distribution with phylogenetic tree

Species abundance is continuous numerical data and usually summarized as boxplot, violinplot or density curve, which are all suported by `facet_plot` (Table S1). This example uses microbome data that provided in `phyloseq` package and density ridgeline is employed to visualize the abundence data. `facet_plot` automatically re-arranges the abundance data according to the tree structure, visualizes the data using the specify `geom` function, *i.e.* `geom_density_ridges`, and align the density curves with the tree as demonstrated in Fig. S3. Note that data stored in the `phyloseq` object is visible to `ggtree` and can be used directly in tree visualization (`Phylum` was used to color tips and density ridgelines in this example).

```
library(phyloseq)
library(ggridges)
library(dplyr)
library(ggtree)

data("GlobalPatterns")
GP <- GlobalPatterns
GP <- prune_taxa(taxa_sums(GP) > 1000, GP)
sample_data(GP)$human <- get_variable(GP, "SampleType") %in%  c("Feces", "Skin")
mergedGP <- merge_samples(GP, "SampleType")
mergedGP <- rarefy_even_depth(mergedGP,rngseed=394582)
mergedGP <- tax_glom(mergedGP,"Order")

melt_simple <- psmelt(mergedGP) %>%
  filter(Abundance < 120) %>%
  select(OTU, val=Abundance)

p <- ggtree(mergedGP) +
  geom_tippoint(aes(color=Phylum), size=1.5)

facet_plot(p, panel="Abundance", data=melt_simple,
           geom_density_ridges, mapping = aes(x=val,group=label,
                                              fill=Phylum),
           color='grey80', lwd=.3)
```

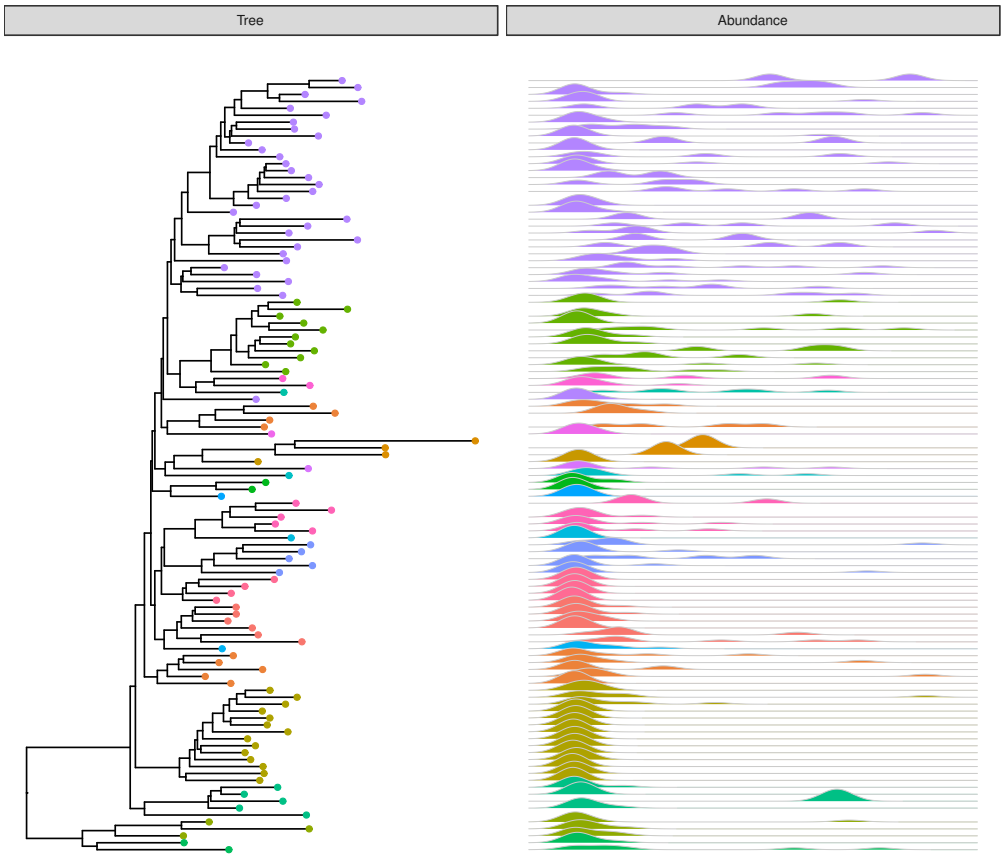

Fig. S3: Phylogenetic tree with OTU abundance densities

1.3 Visualizing pairwise nucleotide sequence distance with phylogenetic tree

This example reproduces Fig. 1 of (Chen et al. 2017). It extracts accession numbers from tip labels of the HPV58 tree and calculated pairwise nucleotide sequence distances. The distance matrix is visualized as dot and line plots. This example demonstrating the abilities of adding multiple layers to a specific panel. As illustrated on Fig. S4, the `facet_plot` function displays sequence distances as a dot plot and then adds a layer of line plot to the same panel, *i.e.* Sequence Distance. In addition, the tree in `facet_plot` can be fully annotated with multiple layers (clade labels, bootstrap support values, *etc*).

```
require(tibble)
require(tidyr)
require(Biostrings)
require(treeio)
require(ggplot2)
require(ggtree)

tree = read.tree("HPV58.tree")
clade = c(A3=92, A1=94, A2=108, B1=156, B2=159, C=163, D1=173, D2=176)
tree = groupClade(tree, clade)

cols <- c(A1="#EC762F", A2="#CA6629", A3="#894418", B1="#0923FA",
          B2="#020D87", C="#000000", D1="#9ACD32",D2="#08630A")

## visualize the tree with tip labels and tree scale
```

```
p <- ggtree(tree, aes(color=group), ladderize=FALSE) %>% rotate(rootnode(tree)) +
    geom_tiplab(aes(label=paste0("italic('", label, "')")), parse = TRUE, size=2.5) +
    geom_treescale(x = 0, y = 1, width=0.002) +
    scale_color_manual(values=c(cols, "black"), na.value="black", name= "Lineage",
                       breaks=c("A1", "A2", "A3", "B1", "B2", "C", "D1", "D2")) +
    guides(color = guide_legend(override.aes = list(size=5, shape=15))) +
    theme_tree2(legend.position = c(.1, .88))

## Optional
## add labels for monophyletic groups using geom_cladelabel
## and paraphyletic group (B) using geom_strip
p <- p + geom_cladelabel(94, "italic(A1)", color=cols[["A1"]], offset = .003, align=TRUE,
                         offset.text=-.001, barsize=1.2, extend = c(0, 0.5), parse=TRUE) +
    geom_cladelabel(108, "italic(A2)", color = cols[["A2"]], offset = .003, align=TRUE,
                         offset.text=-.001, barsize=1.2, extend = 0.5, parse=TRUE) +
    geom_cladelabel(131, "italic(A3)", color = cols[["A3"]], offset = .003, align=TRUE,
                         offset.text=-.001, barsize=1.2, extend = c(0.5, 0), parse=TRUE) +
    geom_cladelabel(92, "italic(A)", color = "darkgrey", offset = .00315, align=TRUE,
                         offset.text = 0.0002, barsize=2, fontsize=5, parse=TRUE) +
    geom_cladelabel(156, "italic(B1)", color = cols[["B1"]], offset = .003, align=TRUE,
                         offset.text=-.001, barsize=1.2, extend = c(0, 0.5), parse=TRUE) +
    geom_cladelabel(159, "italic(B2)", color = cols[["B2"]], offset = .003, align=TRUE,
                         offset.text=-.001, barsize=1.2, extend = c(0.5, 0), parse=TRUE) +
    geom_strip(65, 71, "italic(B)", color="darkgrey", offset=0.00315, align=TRUE,
                         offset.text=0.0002, barsize=2, fontsize=5, parse=TRUE) +
    geom_cladelabel(163, "italic(C)", color = "darkgrey", offset = .0031, align=TRUE,
                         offset.text=0.0002, barsize=3.2, fontsize=5, parse=TRUE) +
    geom_cladelabel(173, "italic(D1)", color = cols[["D1"]], offset = .003, align=TRUE,
                         offset.text=-.001, barsize=1.2, extend = c(0, 0.5), parse=TRUE) +
    geom_cladelabel(176, "italic(D2)", color = cols[["D2"]], offset = .003, align=TRUE,
                         offset.text=-.001, barsize=1.2, extend = c(0.5, 0), parse=TRUE) +
    geom_cladelabel(172, "italic(D)", color = "darkgrey", offset = .00315, align=TRUE,
                         offset.text = 0.0002, barsize=2, fontsize=5, parse=TRUE)
## Optional
## display support values
p <- p + geom_nodelab(aes(subset = (node == 92), label="*"),
                      color="black", nudge_x=-.001, nudge_y=1) +
    geom_nodelab(aes(subset = (node == 155), label="*"),
                      color="black", nudge_x=-.0003, nudge_y=-1) +
    geom_nodelab(aes(subset = (node == 158), label="95/92/1.00"),
                      color="black", nudge_x=-0.0001, nudge_y=-1, hjust=1) +
    geom_nodelab(aes(subset = (node == 162), label="98/97/1.00"),
                      color="black", nudge_x=-0.0001, nudge_y=-1, hjust=1) +
    geom_nodelab(aes(subset = (node == 172), label="*"),
                      color="black", nudge_x=-.0003, nudge_y=-1)
```

```
## extract accession numbers from tip labels
tl <- tree$tip.label
acc <- sub("\\w+\\|", "", tl)
names(tl) <- acc

## read sequences from GenBank directly into R
## and convert the object to DNAStringSet
tipseq <- ape::read.GenBank(acc) %>% as.character %>%
```

```
    lapply(.,paste0,collapse="") %>% unlist %>%
    DNAStringSet
## align the sequences using muscle
tipseq_aln = muscle::muscle(tipseq)
tipseq_aln <- DNAStringSet(tipseq_aln)
```

```
## calculate pairwise hamming distances among sequences
tipseq_dist <- stringDist(tipseq_aln, method="hamming")

## calculate percentage of differences
tipseq_d <- as.matrix(tipseq_dist)/width(tipseq_aln[1]) * 100

## convert the matrix to tidy data frame for facet_plot
dd <- as_data_frame(tipseq_d)
dd$seq1 <- rownames(tipseq_d)
td <- gather(dd,seq2, dist, -seq1)
td$seq1 <- tl[td$seq1]
td$seq2 <- tl[td$seq2]

g <- p$data$group
names(g) <- p$data$label
td$clade <- g[td$seq2]

## visualize the sequence differences using dot plot and line plot
## and align the sequence difference plot to the tree using facet_plot
p2 <- facet_plot(p, panel="Sequence Distance", data=td, geom_point,
            mapping = aes(x=dist, color=clade, shape=clade), alpha=.6) %>%
    facet_plot(panel = "Sequence Distance", data=td, geom=geom_path,
            mapping=aes(x=dist, group=seq2, color=clade), alpha=.6) +
    scale_shape_manual(values=1:8, guide=FALSE)

print(p2)
```

2. Geometric layers that supported by `facet_plot`

`facet_plot` is a general solution for linking graphic layer to a tree. The function internally re-order the input data based on the tree structure and visualize the data at the specific panel by the geometric function. Users are freely to visualize several panels to plot different types of data as demonstrated in Fig. S6 and use different geometric functions to plot the same dataset (Fig. S4) or different datasets (Fig. S8) on the same panel.

`facet_plot` is designed to work with most of the `geom` functions defined in `ggplot2` (Wickham 2016) and other `ggplot2`-based packages. Here is the list of the geometric functions that works seamlessly with `facet_plot`. As the `ggplot2` community keeps expanding and more `geom` functions will be implemented in either `ggplot2` or other extensions, `facet_plot` will gains more power to present data in future. Note that different `geom` functions can be combined to present data on the same panel (Fig. S4 and S8) and the combinations of different `geom` functions create the possibility to present more complex data with phylogeny.

3. Comparing ggtree with other `R` packages

We have presented detail comparison of `ggtree` with `ape` (Paradis, Claude, and Strimmer 2004), `phytools` (Revell 2012), `phyloseq` (McMurdie and Holmes 2013) and `OutbreakTools` (Jombart et al. 2014) on Ap-

Table S1: Geometric layers that supported by 'facet_plot()'

Package	Geom Layer	Description
ggalt	geom_dumbbell	create dumbbell charts
ggbio	geom_alignment	show interval data as alignmen
ggfittext	geom_fit_text	shrinks, grows or wraps text to fit inside a defined rectangular area
gggenes	geom_gene_arrow	draws genes as arrows
ggimage	geom_image	visualizes image files
	geom_phylopic	queries image files from phylopic database and visualizes them
ggplot2	geom_hline	adds horizontal lines
	geom_jitter	adds a small amount of random variation to the location of each point
	geom_label	draws a rectangle behind the text
	geom_point	creats scatterplots
	geom_raster	a high performance special case for all the tiles are the same size
	geom_rect	draw rectangle by using the locations of the four coners
	geom_segment	draws a straight line between points
	geom_spoke	a polar parameterisation of 'geom_segment()'
	geom_text	adds text to the plot
	geom_tile	draw rectangle by using the center of the tile and its size
	geom_vline	adds vertical lines
ggrepel	geom_text_repel	adds text to the plot. The text labels repel away from each other and away from the data points
	geom_label_repel	draws a rectangle underneath the text. The text labels repel away from each other and away from the data ponts
ggridges	geom_density_ridges	arranges multiple density plots in a staggered fashion
	geom_density_ridges_gradient	works just like 'geom_density_ridges' except that the 'fill' aesthetic can vary along the x axis
	geom_ridgeline	plots the sum of the 'y' and 'height' aesthetics versus 'x', filling the area between 'y' and 'y + height' with a color
	geom_ridgeline_gradient	works just like 'geom_ridgeline' except that the 'fill' aesthetic can vary along the x axis
ggstance	geom_barh	horizontal version of 'geom_bar()'
	geom_boxploth	horizontal version of 'geom_boxplot()'
	geom_crossbarh	horizontal version of 'geom_crossbar()'
	geom_errorbarh	horizontal version of 'geom_errorbarh()'
	geom_histogramh	horizontal version of 'geom_histogram()'
	geom_linerangeh	horizontal version of 'geom_linerange()'
	geom_pointrangeh	horizontal version of 'geom_pointrange()'
	geom_violinh	horizontal version of 'geom_violin()'
ggtree	geom_motif	draws aligned motifs

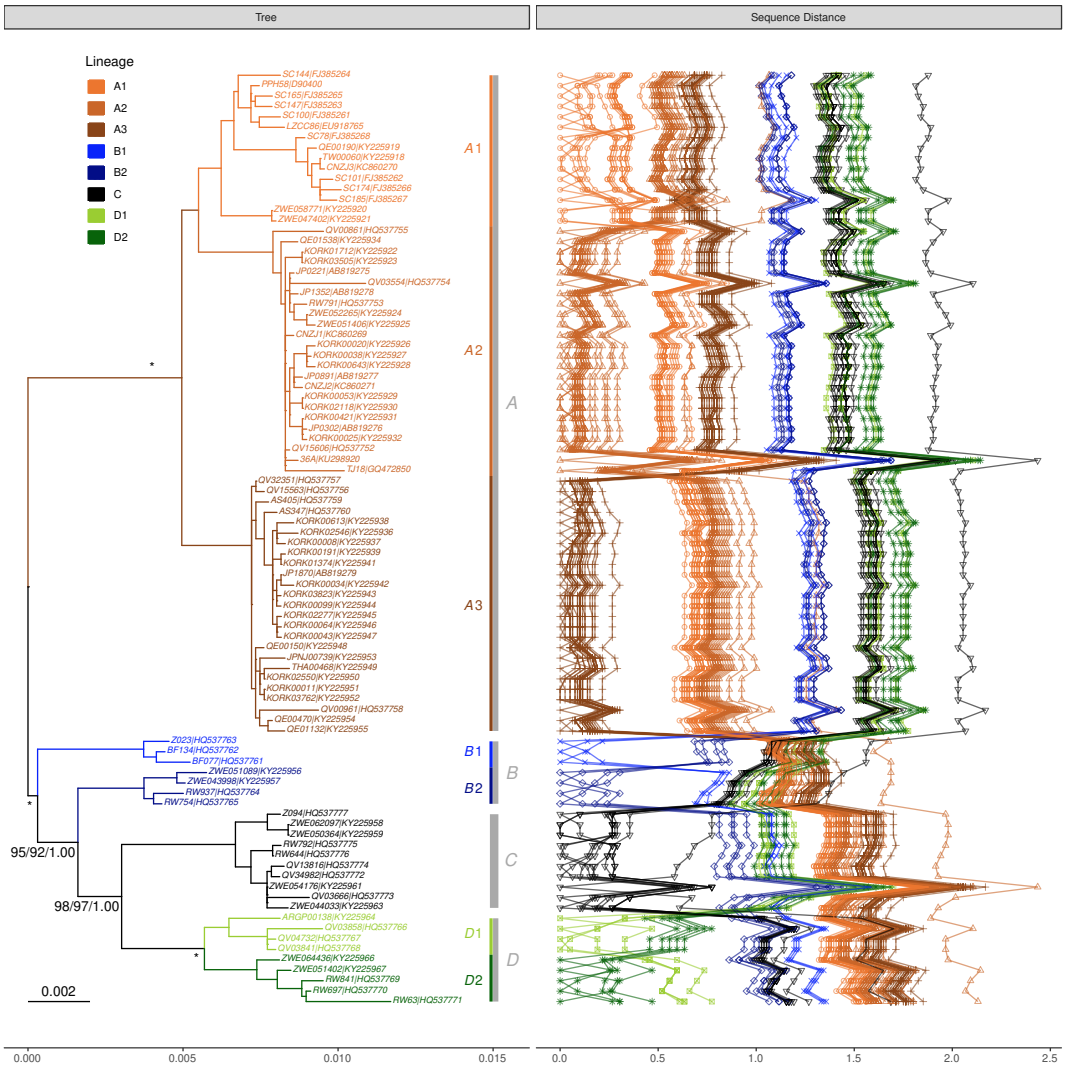

Fig. S4: Phylogeny of HPV58 complete genomes with dot and line plots of pairwise nucleotide sequence distances

pendix S1 of (Yu et al. 2017). Here we extend the comparison with `plotTree` scripts[5], `metacoder` (Foster, Sharpton, and Grünwald 2017) and `phylobase`[6], which have the ability to integrate external data.

3.1 plotTree scripts

Although `plotTree` is not an `R` package, we include the scripts for comparision as the scripts has the abilities of plotting tree with basic information, heatmap and bar plots. Phylogenetic tree is visualized by `plot.phylo` provided by `ape` (Paradis, Claude, and Strimmer 2004). The tip information used in `plotTree` is not really integrated, as it cannot be mapped to visual characteristics, such as color, shape and size that are supported in `ggtree`. The information is hard-coded to only color tip with circle symbols and print the text next to the tips. In addition, there is no solution for mapping data to internal nodes. For plotting tree with data, `plotTree` only supports heatmap for matrix, barplot for numerical data, dotplot for allele data and line segments for genome blocks. The capabilities is restricted to specific needs and only applied to specific

[5]https://github.com/katholt/plotTree
[6]https://CRAN.R-project.org/package=phylobase

data. An example of plotting tree with data using `plotTree` is demonstrated in Fig. S5 and a corresponding `ggtree` version is illustrated in Fig. S6.

```
## need to clone the repo before running the scrtip
## access date: 2018-07-23
## git clone https://github.com/katholt/plotTree.git
setwd("plotTree/tree_example_april2015")
source("../plotTree.R")

## the 'location' that used to color the nodes, cannot be used to color bar and snp data.
## the positions of data panels are hard-coded.
##
## issues:
## 1. bar data can not be displayed properly
## 2. legend not shown properly,
##    there is not taxa sampled from locations label as 'other' or 'VN'
plotTree(tree="tree.nwk",
         infoFile="info.csv", infoCols=NA,
         colourNodesBy="location", legend.pos="topleft",
         barData="bar.csv",snpFile="alleles.csv")
```

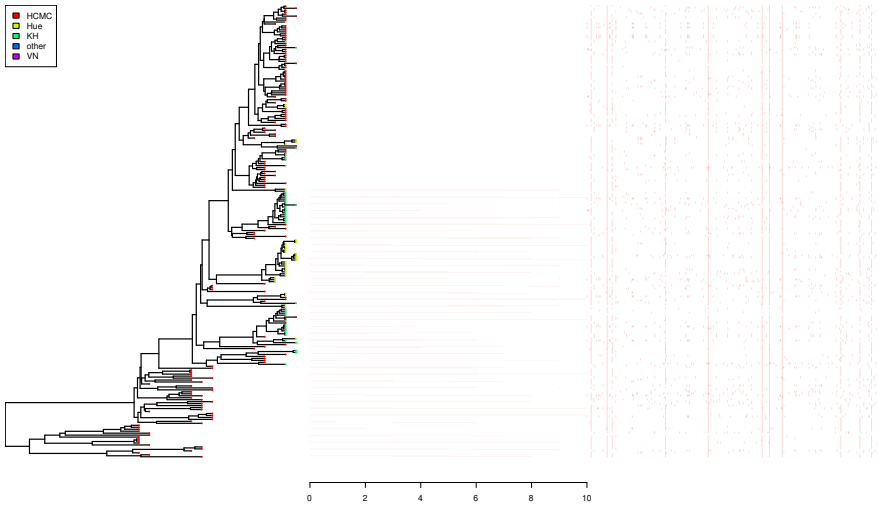

Fig. S5: Example of plotTree

`ggtree` provide general solutions for integrating data. Method 1, the `%<+%` operator, can integrating external and internal node data and mapping the data as visual characteristics to visualize the tree (Fig. S1) and other datasets used in `facet_plot` (in Fig. S6, SNP and Trait data were colored by isolation location that was attached by `%<+%`). Method 2, the `facet_plot` function, has no restriction of input data as long as there is a `geom` function available to plot the data (*e.g.* species abundance displayed by `geom_density_ridges` that demonstrated in Fig. S3). Users is free to combine different panels (Fig. S6) and combine different `geom` layers in the same panel (Fig. S4 and S8). The `plotTree` scritps can be easily reproduced using `ggtree`[7], while `ggtree` offers more flexibility and can do much more. As `ggtree` is more powerful, the authors of `plotTree` also use `ggtree` to present tree with associated data including genotypes, isolation locations, patient status, mutations *etc*, in their recent paper (Britto et al. 2018).

```
library(ggtree)
remote_folder <- paste0("https://raw.githubusercontent.com/katholt/",
                        "plotTree/master/tree_example_april2015/")
```

[7]https://github.com/GuangchuangYu/plotTree-ggtree

```r
## read the phylogenetic tree
tree <- read.tree(paste0(remote_folder, "tree.nwk"))

## read the sampling information data set
info <- read.csv(paste0(remote_folder,"info.csv"))

## read and process the allele table
snps<-read.csv(paste0(remote_folder, "alleles.csv"), header=F,
               row.names=1, stringsAsFactor=F)
snps_strainCols <- snps[1,]
snps<-snps[-1,] # drop strain names
colnames(snps) <- snps_strainCols

gapChar <- "?"
snp <- t(snps)
lsnp <- apply(snp, 1, function(x) {
        x != snp[1,] & x != gapChar & snp[1,] != gapChar
    })
lsnp <- as.data.frame(lsnp)
lsnp$pos <- as.numeric(rownames(lsnp))
lsnp <- tidyr::gather(lsnp, name, value, -pos)
snp_data <- lsnp[lsnp$value, c("name", "pos")]

## read the trait data
bar_data <- read.csv(paste0(remote_folder, "bar.csv"))

## visualize the tree
p <- ggtree(tree)

## attach the sampling information data set
## and add symbols colored by location
p <- p %<+% info + geom_tippoint(aes(color=location))

## visualize SNP and Trait data using dot and bar charts,
## and align them based on tree structure
p2 <- facet_plot(p, panel="SNP", data=snp_data, geom=geom_point,
               mapping=aes(x=pos, color=location), shape='|') %>%
    facet_plot("Trait", bar_data, ggstance::geom_barh,
               aes(x=dummy_bar_value, color=location, fill=location),
               stat="identity", width=.6) +
    theme_tree2(legend.position=c(.05, .85))
print(p2)
```

3.2 metacoder

`metacoder` (Foster, Sharpton, and Grünwald 2017) is designed for visualizing herachical data, mainly for community taxonomic diversity data. It produces heat tree that uses *e.g.* taxa abundance to scale color and size of nodes and edges. This is similar to method 1 presented in our manuscript. `ggtree` (Yu et al. 2017) is more flexible and can do more, such as coloring text labels and plotting variety of node shapes using categorical data.

The heat tree implemented in `metacoder` is designed for herachical data, but not for phylogeny. `metacoder` internally use graph layout algorithms implemented in `igraph` (Csardi and Nepusz 2006) and doesn't support

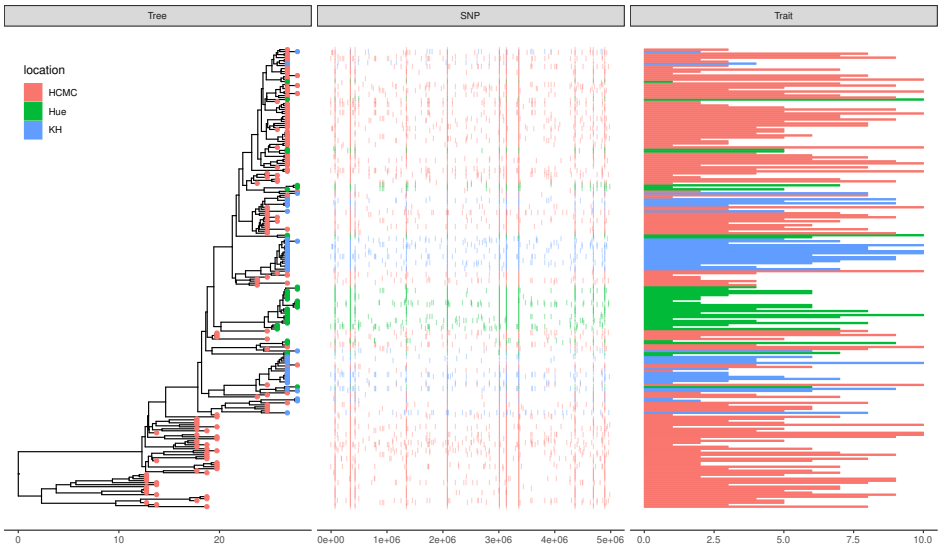

Fig. S6: Example of plotting SNP and trait data using ggtree

widely used phylogenetic tree layouts (*e.g.* rectangular and slanted layouts). Edge length for phylogenetic tree is also not supported.

`ggtree` is designed for general purpose and can be easily extended for specific research applications. For example, the R package, `microbiomeViz`[8] for visualizing microbiome data, is developed based on `ggtree`. `microbiomeViz` is similar to the python library, `GraPhlAn`, which supports phylogeny and has better annotation abilities than `metacoder` (according to (Foster, Sharpton, and Grünwald 2017)).

In addition, `ggtree` supports visualizing microbiome data stored in the `phyloseq` object (McMurdie and Holmes 2013). Examples can be found on Appendix S1 of (Yu et al. 2017) and Fig. S3.

3.3 phylobase

The `phylobase`[9] package defines `phylo4d` class that combines a tree with a data frame, and provides `plot` method, which internally call the `treePlot` function, to display the tree with the data. However there are some restrictions of the `plot` method, it can only plot numeric values for tree-associated data as bubbles and cannot generate figure legend. `Phylobase` doesn't implement visualization method to display categorical values. Using associated data as visual characteristics such as color, size and shape, is also not supported. Although it is possible to color the tree using associated data, it requires users to extract the data and map them to color vector manually follow by passing the color vector to the `plot` method. This is tedious and error-prone since the order of the color vector needs to be consistent with the edge list stored in the object.

Here is the example of plotting associated data with `phylobase`:

```
library(phylobase)
data(geospiza_raw)
g1 <- as(geospiza_raw$tree, "phylo4")
g2 <- phylo4d(g1, geospiza_raw$data, missing.data="warn")
```

```
## Warning in formatData(phy = x, dt = tip.data, type = "tip", ...): The
## following nodes are not found in the dataset: olivacea
```

[8] https://github.com/lch14forever/microbiomeViz
[9] https://CRAN.R-project.org/package=phylobase

```
plot(g2)
```

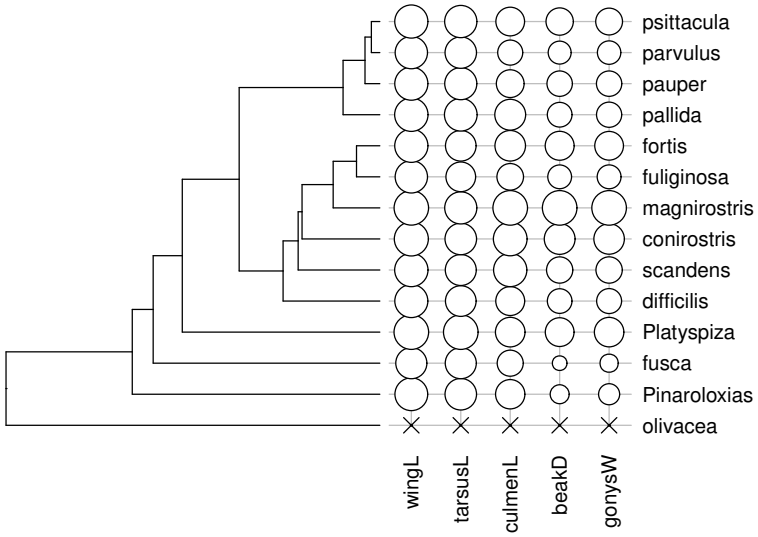

Fig. S7: Example of phylobase

The `phylo4d` object is supported directly by `ggtree`, so that we can use `ggtree(g2)` to directly visualize the tree. All the associated data stored in the `phylo4d` object can be used directly to annotate the tree (Fig. S8A). In addition, users can use `facet_plot` to visualize the associated data (Fig. S8B).

```
d1 <- data.frame(x = seq(0.93, 1.15, length.out=5),
                 lab = names(geospiza_raw$data))

## plot bubbles directly using data stored in 'g2'
p1 <- ggtree(g2) + geom_tippoint(aes(size = wingL), x=d1$x[1], shape=1) +
    geom_tippoint(aes(size = tarsusL), x=d1$x[2], shape=1) +
    geom_tippoint(aes(size=culmenL), x=d1$x[3], shape=1) +
    geom_tippoint(aes(size=beakD), x=d1$x[4], shape=1) +
    geom_tippoint(aes(size=gonysW), x=d1$x[5], shape=1) +
    scale_size_continuous(range=c(3,12), name="") +
    geom_text(aes(x=x, y=0, label=lab), data=d1, angle=90) +
    geom_tiplab(offset=.3) + xlim(0, 1.3) +
    theme(legend.position=c(.1, .75)) + labs(tag = "A")

library(dplyr)
library(tidyr)

## extract tip data from 'g2' and use 'facet_plot' to visualze the data
d <- tipData(g2)
d$tip <- rownames(d)
dd <- gather(d, feature, value, -tip)
cat = seq(ncol(d))
names(cat) = names(d)
dd$cat = cat[dd$feature]
d2 <- select(dd, -value) %>% filter(tip ==  'fuliginosa')
p <- ggtree(g2) + geom_tiplab() + xlim_tree(c(0, 1.2))
p2 <- facet_plot(p, "Morphometric data", dd, geom_point, aes(x=cat, size=value), shape=1) %>%
    facet_plot("Morphometric data", d2, geom_text, aes(x=cat, y=0, label=feature), angle=90) +
```

```
    scale_size_continuous(range=c(3, 12)) +
    theme(legend.position="right") +
    coord_cartesian(clip="off") + labs(tag = "B")
cowplot::plot_grid(p1, p2)
```

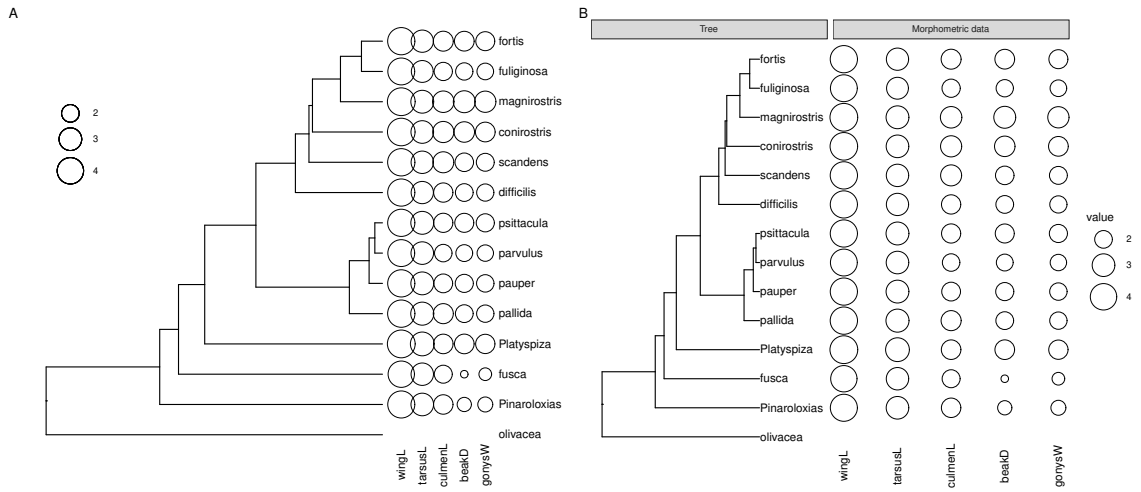

Fig. S8: Visualizing phylo4d data using ggtree

`phylobase` only provides `plot` method to visualize numerical values, and will ignore categorical values if there are any. In `ggtree`, we are able to attach additional information, either numerical or categorical data, and are able to display these information in our favorite way. Here as an example, the `diet` information was attached to the tree and used to color symbolic points, tip labels (Fig. S9A) and phenotypic data (Fig. S9B). The numerical values can be visualized not only as bubble plot (Fig. S8) but also heatmap (Fig. S9A) or other types (*e.g.* stacked bars). Heatmap is commonly used in phylogeny for comparative study. Before developing `facet_plot`, `ggtree` implemented a `gheatmap` function (Yu et al. 2017) to visualize phylogenetic tree with heatmap of numerical or categorical values. `facet_plot` is a general solution for plotting data with the tree, including heatmap (Fig. S9A). `gheatmap` is specifically designed for plotting heatmap with tree and provides shortcut for handling column labels and color palette. Another difference is that `facet_plot` only supports rectangular and slanted tree layouts while `gheatmap` supports rectangular, slanted and circular layouts. `gheatmap` works seemlessly with `facet_plot` as illustrated in Fig. S9B.

```
diet = data.frame(species=tipLabels(g2),
    Diet=c("Seeds","Seeds", "Seeds", "Cacti", "Cacti", "Seeds",
        "Insects", "Insects", "Insects", "Insects", "Fruits",
        "Insects", "Insects", "Insects"))

p <- ggtree(g2) %<+% diet + geom_tiplab(aes(color = Diet), offset=.05) +
    geom_tippoint(aes(color = Diet), size=5, alpha=.5) +
    xlim_tree(c(0, 1.2))

p2 <- gheatmap(ggtree(g2), tipData(g2), colnames_angle = 90) %<+% diet

p3 <- facet_plot(p, "Morphometric data", dd, geom_tile, aes(x=cat, fill = value)) %>%
    facet_plot("Morphometric data", d2, geom_text, aes(x=cat, y=0, label=feature), angle=90) +
    scale_fill_viridis_c(na.value = "white") + labs(tag = "A") +
    theme(legend.position="right")

p4 <- facet_plot(p2, "Morphometric data", dd, geom_point, aes(x=cat, size=value, color=Diet)) %>%
    facet_plot("Morphometric data", d2, geom_text, aes(x=cat, y=0, label=feature), angle=90) +
```

```
    scale_size_continuous(range=c(3, 12)) + labs(tag = "B") +
    theme(legend.position="right") + coord_cartesian(clip="off")

cowplot::plot_grid(p3, p4)
```

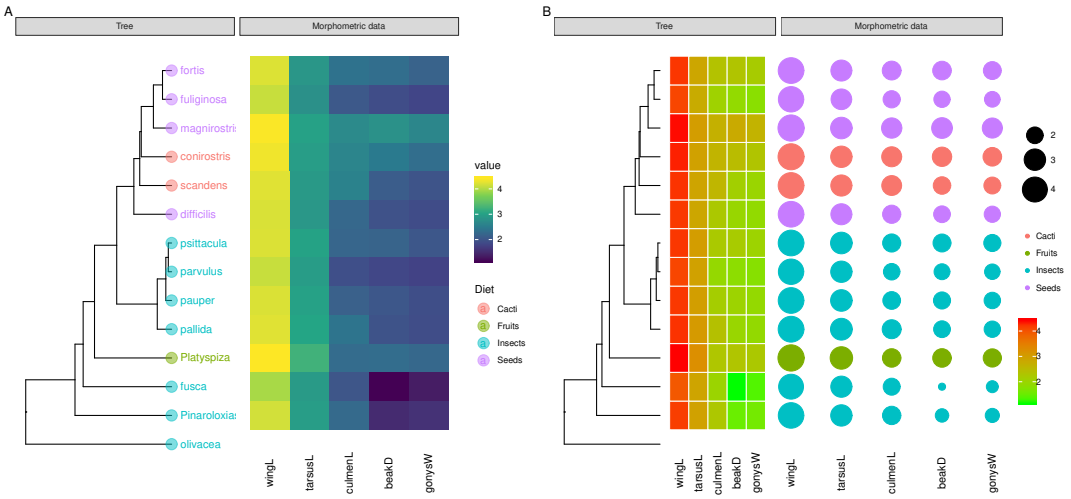

Fig. S9: Visualizing phylo4d data as heatmap using 'facet_plot()' with additional data to color taxa (A), 'facet_plot()' works with 'gheatmap' (B)

3.4 Summary

Although there are many software packages support visualizing phylogenetic tree, plotting tree with data is often missing or with only limited supports. Some of the packages defines `S4` classes to store phylogenetic tree with domain specific data, such `OutbreakTools` (Jombart et al. 2014) defined `obkData` for storing tree with epidemiology data and `phyloseq` (McMurdie and Holmes 2013) defines `phyloseq` for storing tree with microbiome data. These packages are capable to present some of the data stored in the object on the tree. However, not all the associated data are supported. For example, species abundance stored in `phyloseq` object is not supported to be visualized using `phyloseq` package. These packages did not provide any utilities to integrate external data for tree visualization. `metacoder` is able to integrate external data to be used as visual characteristics (limited to size and color) to produce heat tree. However, phylogenetic tree layouts are not supported and branch lengthes will be ignored in `metacoder`. None of these packages support visualizing external data and align the plot to tree based on the tree structure.

The `phylobase` package support integrating external data (especially numerical matrix) to phylogenetic tree and visualize the data as bubble plot with the tree side by side. The `plotTree` scripts supports visualizing predefined datasets as heatmap or bar plots and aligns the plot to the tree. They only provide limited supports to visualize specific data types. The funtions provided by packages or scripts are hard-coded to plot simple tree with specific panel (bubble plot or heatmap, *etc*) and there is no utilities provided to add further layer of annotations.

In contrast, `ggtree` has many unique features that cannot be found in all these implementations:

1. Integrating node/edge data to the tree can be mapped to visual characteristics of the tree or other datasets (Fig. S1).
2. Capable of parsing expression (math symbols or text formatting), emoji and image files (Fig. S1).
3. No predefined of input data types or how the data should be plotted in `facet_plot` (Table S1).
4. Combining different `geom` functions to visualize associated data is supported (Fig. S4).
5. Visualizing different datasets on the same panel is supported (Fig. S8B and S9).

6. Data integrated by `%<+%` can be used in `facet_plot` (Fig. S6 and S9B).
7. Able to add further annotation to specific layers (Fig. S8B and S9).
8. Modular design by separating tree visualization, data integration (method 1) and graph aligning (method 2).

Modular design is a unique feature for `ggtree` to stand out from other packages. The tree can be visualized with data stored in tree object or external data linked by `%<+%` operator, and fully annotated with multiple layer of annotations (Fig. S1 and S4), before passing it to `facet_plot`. And `facet_plot` can be called progresively to add multiple panels (Fig. S6) or multiple layers on the same panels (Fig. S4). This creates the possibility of plotting fully annotated tree with complex data panels that contains multiple graphic layers.

`ggtree` fits the `R` ecosystem and extends the abilities of integrating and presenting data with trees to existing phylogenetic packages. As demonstrated in this paper, we are now able to plot species abundance distribution with `phyloseq` object (Fig. S3), visualize numerical matrix stored in `phylo4d` object using other methods instead of just bubble plot (Fig. S9A) and integrate categorical values to color other data (Fig. S9), *etc.* All these cannot be easily done without `ggtree`. With `ggtree`, we are able to attach additional data to these tree objects using `%<+%` and align graph to tree using `facet_plot`. Integrating `ggtree` to existing workflows will definitely extends the abilities and broaden the applications to present phylogeny-associated data, especially for comparative studies.

NOTE: source code that produces this file can be obtained online[10].

References

Berger, Simon A., Denis Krompass, and Alexandros Stamatakis. 2011. "Performance, Accuracy, and Web Server for Evolutionary Placement of Short Sequence Reads Under Maximum Likelihood." *Systematic Biology* 60 (3): 291–302. https://doi.org/10.1093/sysbio/syr010.

Bouckaert, Remco, Joseph Heled, Denise Kühnert, Tim Vaughan, Chieh-Hsi Wu, Dong Xie, Marc A. Suchard, Andrew Rambaut, and Alexei J. Drummond. 2014. "BEAST 2: A Software Platform for Bayesian Evolutionary Analysis." *PLoS Comput Biol* 10 (4): e1003537. https://doi.org/10.1371/journal.pcbi.1003537.

Boussau, Bastien, Gergely J. Szöllősi, Laurent Duret, Manolo Gouy, Eric Tannier, and Vincent Daubin. 2013. "Genome-Scale Coestimation of Species and Gene Trees." *Genome Research* 23 (2): 323–30. https://doi.org/10.1101/gr.141978.112.

Britto, Carl D., Zoe A. Dyson, Sebastian Duchene, Michael J. Carter, Meeru Gurung, Dominic F. Kelly, David R. Murdoch, et al. 2018. "Laboratory and Molecular Surveillance of Paediatric Typhoidal Salmonella in Nepal: Antimicrobial Resistance and Implications for Vaccine Policy." *PLOS Neglected Tropical Diseases* 12 (4): e0006408. https://doi.org/10.1371/journal.pntd.0006408.

Chen, Zigui, Wendy C. S. Ho, Siaw Shi Boon, Priscilla T. Y. Law, Martin C. W. Chan, Rob DeSalle, Robert D. Burk, and Paul K. S. Chan. 2017. "Ancient Evolution and Dispersion of Human Papillomavirus 58 Variants." *Journal of Virology* 91 (21): e01285–17. https://doi.org/10.1128/JVI.01285-17.

Csardi, Gabor, and Tamas Nepusz. 2006. "The Igraph Software Package for Complex Network Research." *InterJournal* Complex Systems: 1695. http://igraph.org.

Foster, Zachary S. L., Thomas J. Sharpton, and Niklaus J. Grünwald. 2017. "Metacoder: An R Package for Visualization and Manipulation of Community Taxonomic Diversity Data." *PLoS Computational Biology* 13 (2): e1005404. https://doi.org/10.1371/journal.pcbi.1005404.

Grubaugh, Nathan D., Jason T. Ladner, Moritz U. G. Kraemer, Gytis Dudas, Amanda L. Tan, Karthik Gangavarapu, Michael R. Wiley, et al. 2017. "Genomic Epidemiology Reveals Multiple Introductions of Zika Virus into the United States." *Nature* 546 (7658): 401–5. https://doi.org/10.1038/nature22400.

[10] https://github.com/GuangchuangYu/plotting_tree_with_data

He, Zilong, Huangkai Zhang, Shenghan Gao, Martin J. Lercher, Wei-Hua Chen, and Songnian Hu. 2016. "Evolview V2: An Online Visualization and Management Tool for Customized and Annotated Phylogenetic Trees." *Nucleic Acids Research* 44 (W1): W236–W241. https://doi.org/10.1093/nar/gkw370.

Höhna, Sebastian, Tracy A. Heath, Bastien Boussau, Michael J. Landis, Fredrik Ronquist, and John P. Huelsenbeck. 2014. "Probabilistic Graphical Model Representation in Phylogenetics." *Systematic Biology* 63 (5): 753–71. https://doi.org/10.1093/sysbio/syu039.

Huelsenbeck, J. P., and F. Ronquist. 2001. "MRBAYES: Bayesian Inference of Phylogenetic Trees." *Bioinformatics (Oxford, England)* 17 (8): 754–55.

Jombart, Thibaut, David M. Aanensen, Marc Baguelin, Paul Birrell, Simon Cauchemez, Anton Camacho, Caroline Colijn, et al. 2014. "OutbreakTools: A New Platform for Disease Outbreak Analysis Using the R Software." *Epidemics* 7 (June): 28–34. https://doi.org/10.1016/j.epidem.2014.04.003.

Letunic, Ivica, and Peer Bork. 2007. "Interactive Tree of Life (iTOL): An Online Tool for Phylogenetic Tree Display and Annotation." *Bioinformatics* 23 (1): 127–28. https://doi.org/10.1093/bioinformatics/btl529.

Lott, Steffen C., Björn Voß, Wolfgang R. Hess, and Claudia Steglich. 2015. "CoVennTree: A New Method for the Comparative Analysis of Large Datasets." *Frontiers in Genetics* 6: 43. https://doi.org/10.3389/fgene.2015.00043.

Matsen, Frederick A, Robin B Kodner, and E Virginia Armbrust. 2010. "Pplacer: Linear Time Maximum-Likelihood and Bayesian Phylogenetic Placement of Sequences onto a Fixed Reference Tree." *BMC Bioinformatics* 11 (1): 538. https://doi.org/10.1186/1471-2105-11-538.

McMurdie, Paul J., and Susan Holmes. 2013. "Phyloseq: An R Package for Reproducible Interactive Analysis and Graphics of Microbiome Census Data." *PloS One* 8 (4): e61217. https://doi.org/10.1371/journal.pone.0061217.

Mirarab, Siavash, and Tandy Warnow. 2015. "ASTRAL-II: Coalescent-Based Species Tree Estimation with Many Hundreds of Taxa and Thousands of Genes." *Bioinformatics (Oxford, England)* 31 (12): i44–52. https://doi.org/10.1093/bioinformatics/btv234.

Nguyen, Lam-Tung, Heiko A. Schmidt, Arndt von Haeseler, and Bui Quang Minh. 2015. "IQ-TREE: A Fast and Effective Stochastic Algorithm for Estimating Maximum-Likelihood Phylogenies." *Molecular Biology and Evolution* 32 (1): 268–74. https://doi.org/10.1093/molbev/msu300.

Paradis, Emmanuel, Julien Claude, and Korbinian Strimmer. 2004. "APE: Analyses of Phylogenetics and Evolution in R Language." *Bioinformatics* 20 (2): 289–90. https://doi.org/10.1093/bioinformatics/btg412.

Pond, Sergei L. Kosakovsky, Simon D. W. Frost, and Spencer V. Muse. 2005. "HyPhy: Hypothesis Testing Using Phylogenies." *Bioinformatics* 21 (5): 676–79. https://doi.org/10.1093/bioinformatics/bti079.

Revell, Liam J. 2012. "Phytools: An R Package for Phylogenetic Comparative Biology (and Other Things)." *Methods in Ecology and Evolution* 3 (2): 217–23. https://doi.org/10.1111/j.2041-210X.2011.00169.x.

Sanderson, Michael J. 2003. "R8s: Inferring Absolute Rates of Molecular Evolution and Divergence Times in the Absence of a Molecular Clock." *Bioinformatics* 19 (2): 301–2. https://doi.org/10.1093/bioinformatics/19.2.301.

Stamatakis, Alexandros. 2014. "RAxML Version 8: A Tool for Phylogenetic Analysis and Post-Analysis of Large Phylogenies." *Bioinformatics*, January, btu033. https://doi.org/10.1093/bioinformatics/btu033.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. http://ggplot2.org.

Yang, Ziheng. 2007. "PAML 4: Phylogenetic Analysis by Maximum Likelihood." *Molecular Biology and Evolution* 24 (8): 1586–91. https://doi.org/10.1093/molbev/msm088.

Yu, Guangchuang, David K. Smith, Huachen Zhu, Yi Guan, and Tommy Tsan-Yuk Lam. 2017. "Ggtree: An R Package for Visualization and Annotation of Phylogenetic Trees with Their Covariates and Other

Associated Data." *Methods in Ecology and Evolution* 8 (1): 28–36. https://doi.org/10.1111/2041-210X.12628.