Supplemental File of

ggtreeExtra: A universal package to visualize compact circular layers of phylogenetic tree

Shuangbin Xu, Zehan Dai, Pingfan Guo, Xiaocong Fu, Shanshan Liu, Lang Zhou, Wenli Tang, Tingze Feng, Meijun Chen, Li Zhan and Guangchuang Yu*

*correspondence: guangchuangyu@gmail.com, gcyu1@smu.edu.cn

1 The purpose of development and overview

Integrating and visualizing associated data to the phylogenetic tree can help to find biological patterns and generate new hypotheses. The associated data type of phylogenetic tree can be roughly divided into continuous data and categorical data (discrete data). The continuous data sets represent measurements, they can be measured but not be counted, such as the height, weight, abundance of species, gene expression and the number of target genes etc. The categorical data sets represent characteristics, they can not be measured but they can be counted, such as endemic region information of virus, taxonomy information of species, type of target gene and sampling location information etc. Certainly, categorical data can also take on numerical values (for example, 1 for target gene A, 2 for target gene B). The associated data sets are also often multi-dimensional. Several tools have been developed to integrate and display associated data to phylogenetic tree. However, they still have some shortcomings, such as don't support annotation circular layout (tree and graphic alignment), provide few geometric layers, need predefined input etc, which make them not universal. Here, we developed *ggtreeExtra* to annotate multi-dimensional data to the outer of phylogenetic tree (Fig. S1). It can link *ggtree*(Yu et al. 2017) and geometric layers function defined in *ggplot2*(Wickham 2016) or other ggplot2-based package. And it supports not only circular layout, but also other layouts defined in *ggtree*(Yu et al. 2017). The tree can be annotated by geometric function defined in ggtree(Yu et al. 2017) (Fig. S1). In addition, it was developed based on the grammar of graphics(Wilkinson 2012). So user can easily map the variables (abundance of species, length of genome, sampling location) of associated data to aesthetic attributes (size, color, shape) of outer geometric objects (bar, point, box plot) of circular phylogenetic tree using *ggtreeExtra*. The details (such as legends and theme) of figure can be adjusted by corresponding *scale* function and theme function defined in ggplot2(Wickham 2016) (Fig. S1). Compared other tools, *ggtreeExtra* supports annotation of multiple layout of phylogenetic tree, it can integrate more geometric layers and don't need predefined input, which make it more universal (Tab. S1).

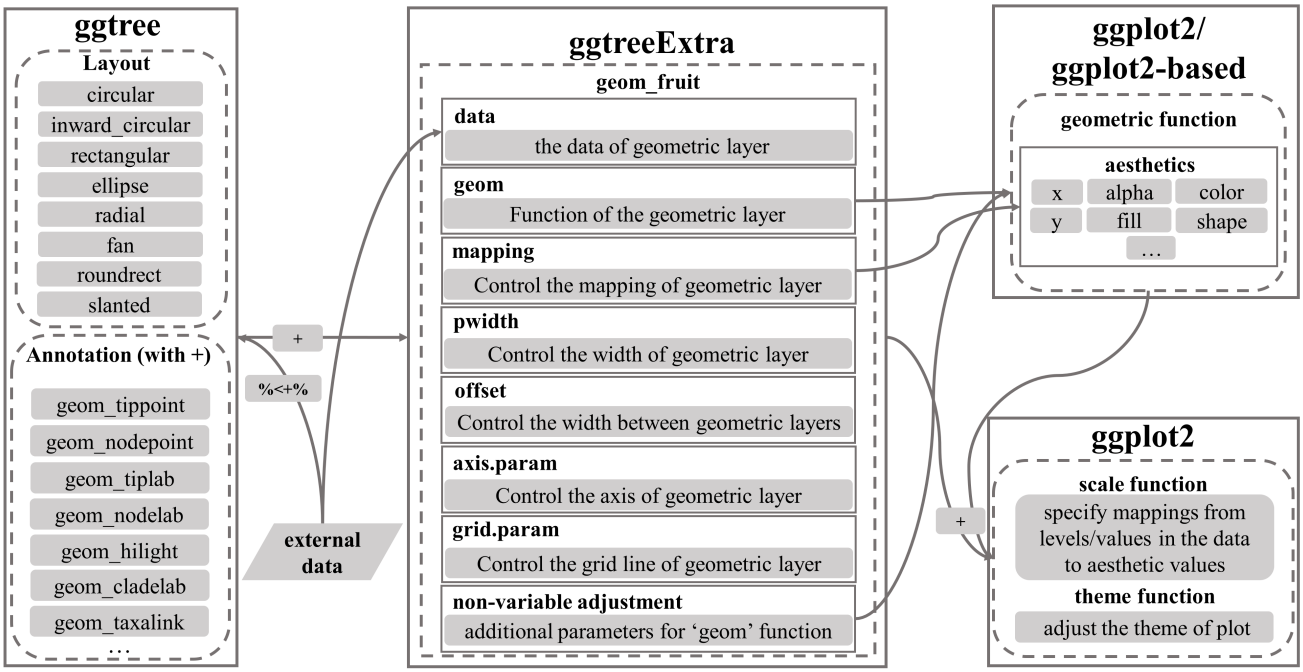

Fig. S1: overview of ggtreeExtra.

Table S1: Comparison list of ggtreeExtra and other tools

Tools	**Platform**	**Supported layouts for tree annotation**[a]	**Annotation layers**	**Supported grammar of graphic**	**Combined freely**[b]	**Methods of figure out**	**Layer operations**
ggtreeExtra	R package	circular, inward circular, rectangular, slanted, ellipse, round rectangular, radial	Heat map, scatter, simple bar, patter bar, stacked or dodged bar, box, pattern box, violin, dot intervals plot, density plot, pie, image plot	Yes	Yes	programming	add, modify, delete
GraPhlAn	Python package	circular	Heat map, scatter, simple bar	No	Yes	command line (configure file)	add
ETE3	Python package	rectangular	Heat map, scatter, simple bar, stacked bar, box, pie, image plot	No	Yes	programming and interaction (mouse click)	add
iTOL	Web tool	circular, rectangular	Heat map, scatter, simple bar, stacked bar, box, pie, image plot	No	Yes	interaction (mouse click configure file)	add
Microreact	Web tool	circular, rectangular	Heat map, scatter	No	Yes	interaction (mouse click and command line configure file)	add
Evolview	Web tool	circular, rectangular, slanted, round rectangular	Heat map, scatter, simple bar, stacked bar, box	No	Yes	interaction (mouse click configure file)	add

[a] tree annotation: tree and geometric alignment;
[b] Combined freely: layers can be combined freely.

2 Geometric layers that supported by *geom_fruit* of *ggtreeExtra*

ggtreeExtra is designed to link ggtree(Yu et al. 2017) and some **geom** functions defined in *ggplot2*(Wickham 2016) and other *ggplot2* extension packages (Fig. S1). Here is the list of the geometric layer functions which work seamlessly with *geom_fruit* of *ggtreeExtra* (Tab. S2). Each geometric layer function has own unique geometric attributes (Tab. S2). User can choose appropriate geometric layer functions according the type of associated data. And the variables of associated data can be mapped to the attributes of corresponding geometric layer function. For example, when user want to view the distribution and uncertainty (continuous data) of associated data in different groups (such as the gene expression or species abundance in different samples), Box, violin or dot interval plot etc can be used to display them (Fig. S2). For the simple numeric data (length of genome, abundance of species), they can be visualized with simple (pattern) bar plot or grouped (pattern) bar plot (pattern bar plot is efficient to the visualization that don't want to use color) (Fig. S3). Certainly, the aesthetics parameters of **geom** not only can be mapped by variables, but also can be used to adjust the attributes of geometric layers directly (such as *pattern_fill* in Fig. S3). Since *ggtreeExtra* can work seamlessly with the *geom* functions, it can integrate more geometric layers compare other tools (Tab. S1). Until now, *ggtreeExtra* can integrate heat map, scatter plot, simple (grouped) (pattern) bar plot, simple (grouped) (pattern) box plot, violin, dot intervals plot, density plot, image plot, pie (Tab. S1). As the *ggplot2*(Wickham 2016) community keeps expanding and more *geom* functions will be implemented in either *ggplot2*(Wickham 2016) or other extensions, *geom_fruit* will gain more power to present data in future.

Table S2: List of geometric layers supported by 'geom_fruit()'

Package	Geom layer	Visual characteristic	Description
ggdist	geom_dots	alpha, color, fill, size, shape	creates dotplots that automatically determines a bin width that ensures the plot fits within the available space
	geom_dotsinterval	alpha, color, fill, size, shape	creates dots, intervals, and quantile dotplots
	geom_pointinterval	alpha, color, fill, size, shape	creates point and multiple uncertainty interval
	geom_slab	alpha, color, fill	creats slab geom
	geom_slabinterval	alpha, color, fill	creates slab, point and interval meta-geom
ggimage	geom_image	alpha, color, size	visualizes image files
	geom_phylopic	alpha, color, size	queries image files from phylopic database and visualizes them
ggpattern	geom_bar_pattern	pattern_alpha, pattern_color, pattern_fill pattern_angle, alpha, color, fill	draws bar charts with support for pattern fills
	geom_boxplot_pattern	pattern_alpha, pattern_color, pattern_fill pattern_angle, alpha, color, fill	draws box and whiskers plot with support for pattern fills
	geom_col_pattern	pattern_alpha, pattern_color, pattern_fill pattern_angle, alpha, color, fill	draws bar charts using 'stat_identity()' with support for pattern fills
	geom_tile_pattern	pattern_alpha, pattern_color, pattern_fill pattern_angle, alpha, color, fill	draws rectangle by using the center of the tile and its size with support for pattern fills
ggplot2	geom_bar	alpha, color, fill	draws bar charts
	geom_boxplot	alpha, color, fill	draws box and whiskers plot
	geom_col	alpha, color, fill	draws bar charts using 'stat_identity()'
	geom_label	alpha, color, fill, size	draws a rectangle behind the text
	geom_point	alpha, color, fill, shape, size	creats scatterplots
	geom_raster	alpha, fill	a high performance special case for all the tiles are the same size
	geom_text	color, size	adds text to the plot
	geom_tile	alpha, color, fill	draws rectangle by using the center of the tile and its size
ggpmisc	geom_plot	vp.width,vp.height	ggplot objects as insets to the base ggplot, using syntax similar to that of 'geom_label'
	geom_table	size	adds a textual table directly to the ggplot using syntax similar to that of 'geom_label'
ggrepel	geom_text_repel	color, size	adds text to the plot. The text labels repel away from each other and away from the data points
	geom_label_repel	alpha, color, fill, size	draws a rectangle underneath the text. The text labels repel away from each other and away from the data ponts
ggridges	geom_density_ridges	alpha,fill	arranges multiple density plots in a staggered fashion
	geom_density_ridges2	alpha,fill	arranges multiple density plots in a staggered fashion
	geom_ridgeline	alpha,color,fill	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	color,fill	works just like 'geom_ridgeline' except that the 'fill' aesthetic can vary along the x axis
ggstance	geom_barh	alpha, color, fill	horizontal version of 'geom_bar()'
	geom_boxploth	alpha, color, fill	horizontal version of 'geom_boxplot()'
	geom_colh	alpha, color, fill	horizontal version of 'geom_col()'
ggstar	geom_star	alpha, color, fill, size, starshape	creates scatterplots
ggsymbol	geom_symbol	alpha, color, fill, size, symbolshape	creates scatterplots
scatterpie	geom_scatterpie	alpha, color, fill	creates scatter pie plot

```
library(tibble)
library(tidyr)
library(ggdist)
library(ggtree)
```

```r
library(ggplot2)
library(ggtreeExtra)
library(patchwork)
set.seed(1024)
# To generate associated data, which has a column contained tip labels.
df = tribble( ~id, ~class, ~value,
    "t1",  "phy1", rnorm(100, mean = 5),
    "t2",  "phy1", rnorm(100, mean = 6, sd = 1.5),
    "t3",  "phy2", rnorm(100, mean = 8),
    "t4",  "phy2", rnorm(100, mean = 9),
    "t5",  "phy1", rnorm(100, mean = 5.5),
    "t6",  "phy2", rnorm(100, mean = 8.5),
    "t7",  "phy3", rnorm(100, mean = 3),
    "t8",  "phy2", rnorm(100, mean = 6.8),
    "t9",  "phy3", rnorm(100, mean = 3.5),
    "t10", "phy3", rnorm(100, mean = 4)
  ) %>% unnest(value)
tr <- rtree(10)
p1 <- ggtree(tr, layout="roundrect", size=0.3) + geom_tiplab(size=3)
p2 <- ggtree(tr, layout="ellipse", size=0.3) + geom_tiplab(size=3)
# The associate data also contains the value of different tip labels in different class.
p1 <- p1 +
      geom_fruit(data=df, geom=geom_dots,
                 mapping=aes(y=id, x=value, fill=class),
                 pwidth=1.8, offset=0.1,
                 position=position_identityx(),
                 color=NA, dotsize=3,
                 orientation="y", side="right",
                 grid.params=list(),
                 axis.params=list(axis="x",vjust=1, text.size=2, nbreak=6)
      ) + theme(legend.key.size = unit(0.3, 'cm'))
p2 <- p2 +
      geom_fruit(data=df, geom=geom_slabinterval,
                 mapping=aes(y=id, x=value, fill=class),
                 position=position_identityx(),
                 pwidth=1.8, offset=0.1,
                 orientation="y", side="right",
                 stat=StatSampleSlabinterval,
                 interval_size_range=c(0.2, 1),
                 grid.params=list(),
                 axis.params=list(axis="x", vjust=1, text.size=2, nbreak=6)
      ) + theme(legend.key.size = unit(0.3, 'cm'))
p1 / p2
```

```r
library(ggtree)
library(ggtreeExtra)
library(ggpattern)
library(ggplot2)

set.seed(1024)
tr <- rtree(20)
dat <- data.frame(id=tr$tip.label, value=abs(rnorm(20, 3)),
                  group=c(rep("A", 5), rep("B", 5), rep("C", 5), rep("D", 5)))
dt <- data.frame(id=rep(tr$tip.label, 8), value=abs(rnorm(20 * 8, 8, 1.5)),
                 class=rep(rep(c("A","B","C","D"), 5), 8))
p1 <- ggtree(tr, size=0.2, branch.length="none")
p2 <- ggtree(tr, size=0.2, layout="slanted", branch.length="none")
p3 <- ggtree(tr, size=0.2, layout="fan", open.angle=180, branch.length="none")
p4 <- ggtree(tr, size=0.2, layout="fan", open.angle=180, branch.length="none")
```

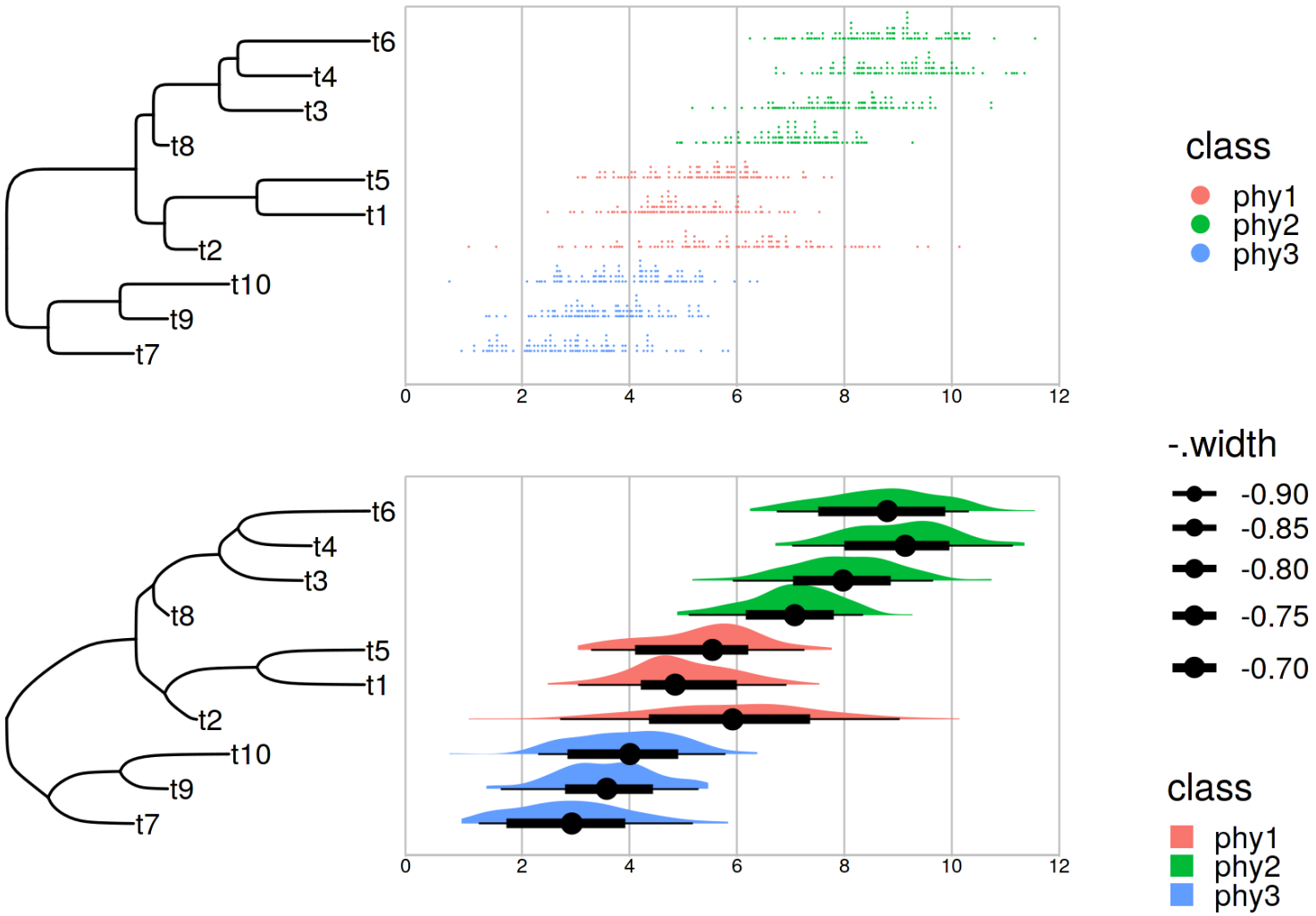

Fig. S2: This example shows *ggtreeExtra* can work with *geom_dots* and *geom_slabinterval* of *ggdist*(Kay 2020), the associated data was import with *data* of *geom_fruit*, it has a column contained tip labels. Then the tip labels column was assigned to *y*, the *value* is continuous data, which was mapped to the *x*, and *class* is categorical data, which was mapped to the *fill* (color).

```
p1 <- p1 +
      geom_fruit(
          data=dat,
          geom=geom_bar_pattern,
          mapping=aes(y=id, x=value, pattern=group, pattern_angle=group),
          width=0.6, stat="identity",
          pwidth = 0.6, pattern_spacing =0.01,
          pattern_size = 0.1, pattern_density = 0.4,
          fill = "grey", pattern_fill="grey35",
          position=position_identityx(),
          axis.params=list(axis="x", text.size=1.5, text.angle=-45, hjust=0)
      ) + theme(legend.key.size = unit(0.3, 'cm'))

p2 <- p2 +
      geom_fruit(
          data=dat,
          geom=geom_bar_pattern,
          mapping=aes(y=id, x=value, pattern=group, pattern_fill=group),
          width=0.6, stat="identity",
          pwidth = 0.6, pattern_spacing =0.01,
          pattern_size = 0.1, pattern_density = 0.4,
          fill = "grey",
```

```
          position=position_identityx(),
          axis.params=list(axis="x",text.size=1.5, text.angle=-45, hjust=0)
      ) + theme(legend.key.size = unit(0.3, 'cm'))

p3 <- p3 +
      geom_fruit(
          data=dt,
          geom=geom_boxplot_pattern,
          mapping=aes(y=id, x=value, pattern=class, pattern_angle = class),
          size=0.1, outlier.size=0.5,
          pwidth=0.5, pattern_size = 0.1,
          pattern_density = 0.4, pattern_spacing =0.01,
          fill = "grey", pattern_fill="grey35",
          position=position_dodgex(),
          grid.params=list(),
          axis.params=list(axis="x", text.size=1.5, text.angle=-45, hjust=0)
      ) +
      theme(legend.key.size = unit(0.35, 'cm'))

p4 <- p4 +
      geom_fruit(
          data=dt,
          geom=geom_boxplot_pattern,
          mapping=aes(y=id, x=value, pattern=class, pattern_fill = class),
          size=0.1, outlier.size=0.5,
          pwidth = 0.5, pattern_size = 0.1,
          pattern_density = 0.4, pattern_spacing = 0.01,
          fill = "grey",
          position=position_dodgex(),
          grid.params=list(),
          axis.params=list(axis="x", text.size=1.5, text.angle=-45, hjust=0)
      ) +
      theme(legend.key.size = unit(0.35, 'cm'))

(p1 + p2)/(p3 + p4)
```

3 Examples of mapping and visualizing associated data on circular layout tree.

ggtreeExtra can integrate many geometric layers by linking *geom* function defined in *ggplot2*(Wickham 2016) or ggplot2-extension packages (Tab. S2 and S1). This is the basis for *ggtreeExtra* to be used for the annotation of phylogenetic tree. We presented two examples (Fig. S2 and S3) to show how to use *ggtreeExtra* to map simple associated data to rectangular phylogenetic tree by linking *geom* funtions. For the multiple associated data sets, circular layout is an efficient way to visualize multi-dimensional data sets(Gu et al. 2014), since it can reduce space and make the graph more compact. Fortunately, *ggtreeExtra* support annotation of multiple layouts of tree (Fig. S1 and S3 and Tab. S1), including circular layout. Furthermore, *ggtreeExtra* is developed based on the grammar of graphics(Wilkinson 2012). The associated data can be imported with the *data* parameter of *geom_fruit*, it can also be integrated to tree data (ggtree graphic object) with %<+% of *ggtree*(Yu et al. 2018), before passing it to *geom_fruit* (Fig. S1). Then the variables (abundance of species, length of genome, sampling location) of associated data can be mapped to the attributes of outer geometric objects (bar, point, boxplot) of circular phylogenetic tree (Fig. S1 and S2). Here, we present several examples to elucidate how to map and display the associated data on the outer rings of circular phylogenetic trees using *ggtreeExtra*. More examples can be found on the *chapter10* of online book[1].

[1] http://yulab-smu.top/treedata-book

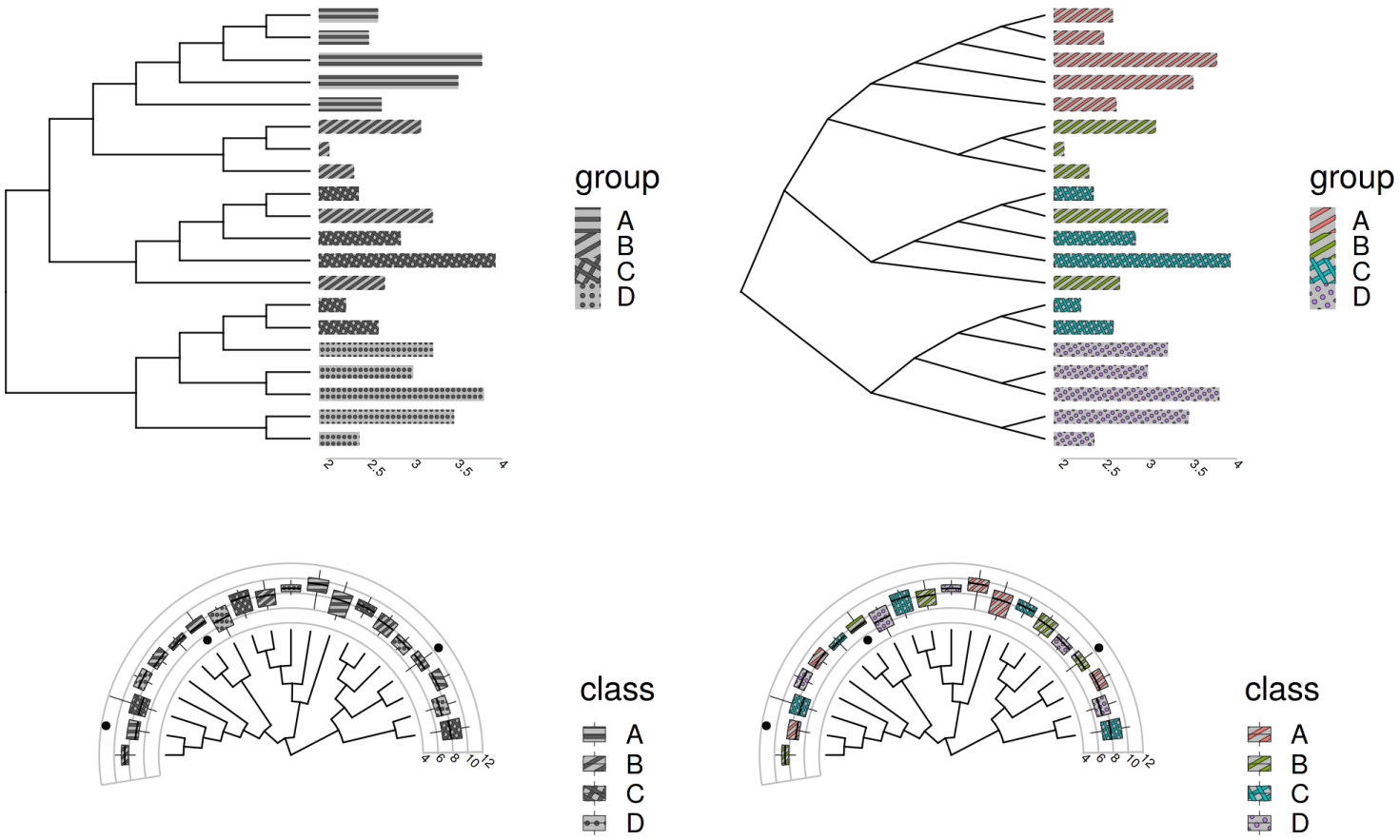

Fig. S3: This example shows *ggtreeExtra* can work with *geom_bar_pattern* and *geom_boxplot_pattern* of *ggpattern*(FC 2020). This example also shows *ggtreeExtra* can work with multiple layouts of tree.

3.1 Data supported by *ggtreeExtra*

3.2 Displaying multiple associated data to circular phylogenetic tree using grammar of graphic

3.3 Annotating circular phylogenetic tree with the associated data contained in ggtree graphic object

3.4 Annotating large phylogenetic tree with multiple associated data

3.5 Annotating associated data to the phylogenetic tree combined relationship data

4 Summary

References

FC, Mike. 2020. *Ggpattern: Geoms with Patterns.*

Gu, Zuguang, Lei Gu, Roland Eils, Matthias Schlesner, and Benedikt Brors. 2014. "Circlize Implements and Enhances Circular Visualization in R." *Bioinformatics* 30 (19): 2811–2. https://doi.org/10.1093/bioinformatics/btu393.

Kay, Matthew. 2020. *ggdist: Visualizations of Distributions and Uncertainty.* https://doi.org/10.5281/zenodo.3879620.

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

Wilkinson, Leland. 2012. "The Grammar of Graphics." In *Handbook of Computational Statistics: Concepts and Methods*, edited by James E. Gentle, Wolfgang Karl Härdle, and Yuichi Mori, 375–414. Berlin, Heidelberg: Springer Berlin Heidelberg.

https://doi.org/10.1007/978-3-642-21551-3_13.

Yu, Guangchuang, Tommy Tsan-Yuk Lam, Huachen Zhu, and Yi Guan. 2018. "Two Methods for Mapping and Visualizing Associated Data on Phylogeny Using Ggtree." *Molecular Biology and Evolution* 35 (2): 3041–3. https://doi.org/10.1093/molbev/msy194.

Yu, Guangchuang, David 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.