# *tanggle*: an R package for visualization of phylogenetic networks in a *ggplot2* framework

Klaus Schliep[1], Marta Vidal-García[2], Leann Biancani, Francisco Henao Diaz, Eren Ada, and Claudia Solís-Lemus[3]

Last updated on 2020-11-05.

# 1. Introduction

Here we present a vignette for the R package *tanggle*, and provide an overview of its functions and their usage. *tanggle* extends the *ggtree* R package (Yu et al. 2017) to allow for the visualization of several types of phylogenetic networks using the *ggplot2* (Wickham 2016) syntax. More specifically, *tanggle* contains functions to allow the user to effectively plot: (1) split (i.e. implicit) networks (unrooted, undirected) and (2) explicit networks (rooted, directed) with reticulations. It offers an alternative to the plot functions already available in *ape* (Paradis and Schliep 2018) and *phangorn* (Schliep 2011).

# 2. List of functions

| Function name | Brief description |
| --- | --- |
| geom_splitnet | Adds a *splitnet* layer to a ggplot, to combine visualising data and the network |
| ggsevonet | Plots an explicit network from a *phylo* object |
| ggsplitnet | Plots an implicit network from a *phylo* object |
| minimize_overlap | Reduces the number of reticulation lines crossing over in the plot |
| node.depth.evonet | Returns the depths or heights of nodes and tips in the phylogenetic network |

# 3. Getting started

Load the necessary libraries:

```
packages_needed <- c("ggplot2", "ggtree", "phangorn")
packages_to_install <- packages_needed[!(packages_needed %in% installed.packages()[,"Package"])]
if(length(packages_to_install)) install.packages(packages_to_install)
```

If you need to install *ggtree* from github:

```
library(devtools)
install_github("YuLab-SMU/ggtree")
```

Install *tanggle* from the github repository:

```
library(devtools)
install_github("KlausVigo/tanggle")
```

And load all the libraries:

```
library(devtools)
library(tanggle, quietly=T)
library(phangorn, quietly=T)
library(ggtree, quietly=T)
```

# 4. Split Networks

Split networks are data-display objects which allow for the definition of 2 (or more) options for non-compatible splits. Split networks are most often used to visualize consensus networks (Holland et al. 2004) or neighbor-nets (Bryant and Moulton 2004). This can be done either by using the `consensusNet` or `neighbor-net` functions in *phangorn* (Schliep 2011) or by importing nexus files from SplitsTree (Huson and Bryant 2006).

## 4.1. Data Types

*tanggle* accepts three forms of input data for split networks. The following input options all generate a *networx* object for plotting.

- Nexus file created with SplitsTree (Huson and Bryant 2006) and read with the `read.nexus.network` function in *phangorn* (Schliep 2011).
- Read in a split network in nexus format:

```
fdir <- system.file("extdata/trees", package = "phangorn")
Nnet <- phangorn::read.nexus.networx(file.path(fdir,"woodmouse.nxs"))
```

2. A collection of gene trees (e.g.~from RAxML (Stamatakis 2014)) in one of the following formats:
    - Nexus file read with the function `read.nexus`
    - Text file in Newick format (one gene tree per line) read with the function `read.tree` A consensus split network is then computed using the function `consensusNet` in *phangorn* (Schliep 2011).

- Sequences in nexus, fasta or phylip format, read with the function `read.phyDat` in *phangorn* (Schliep 2011) or the function `read.dna` in *ape* (Paradis and Schliep 2018). Distances matrices are then computed for specific models of evolution using the function `dist.ml` in *phangorn* (Schliep 2011) or `dist.dna` in *ape* (Paradis and Schliep 2018). From the distance matrix, a split network is reconstructed using the function `neighborNet` in *phangorn* (Schliep 2011). **Optional**: branch lengths may be estimated using the function `splitsNetworks` in *phangorn* (Schliep 2011).

## 4.2. Plotting a Split Network:

We can plot the network with the default options:

```
ggsplitnet(Nnet) + geom_tiplab2()
```
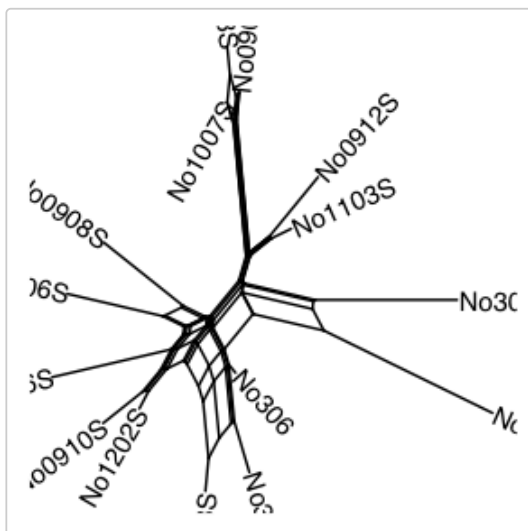


You can rename tip labels. Here we changed the names to species from 1 to 15:

```
Nnet$translate$label <- seq(1:length(Nnet$tip.label))
```

We can include the tip labels with `geom_tiplab2`, and customize some of the options. For example, here the tip labels are in blue and both in bold and italics, and we show the internal nodes in green:

```
ggsplitnet(Nnet) + geom_tiplab2(col = "blue", font = 4, hjust = -0.15) + geom_nodepoint(col = "green",
        size = 0.25)
```
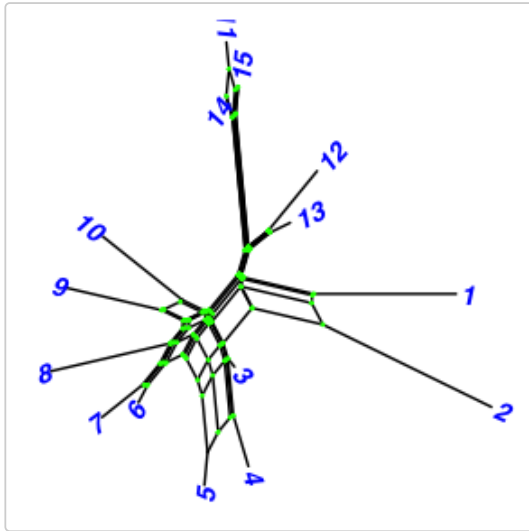


Nodes can also be annotated with `geom_point`.

```
ggsplitnet(Nnet) + geom_point(aes(shape=isTip, color=isTip), size=2)
```
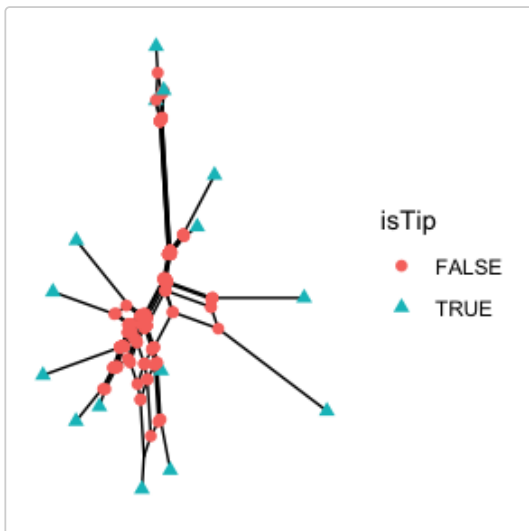


## 4.3. Plotting Explicit Networks

The function `ggevonet` plots explicit networks (phylogenetic trees with reticulations). A recent addition to *ape* (Paradis and Schliep 2018) made it possible to read in trees in extended newick format (Cardona, Rosselló, and Valiente 2008).

Read in an explicit network (example from Fig. 2 in Cardona et al. 2008):
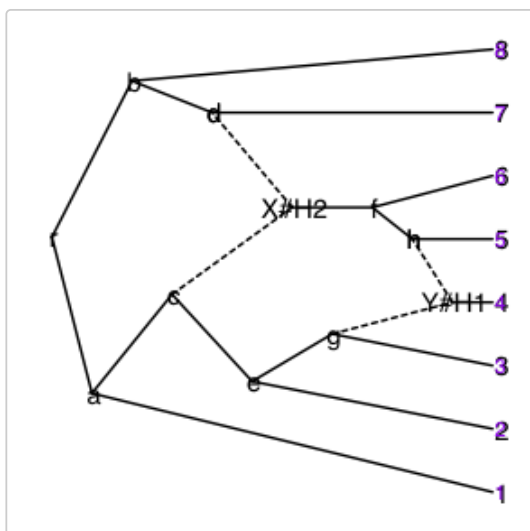
```
z <- read.evonet(text = "((1,((2,(3,(4)Y#H1)g)e,(((Y#H1,5)h,6)f)X#H2)c)a,
                ((X#H2,7)d,8)b)r;")
```
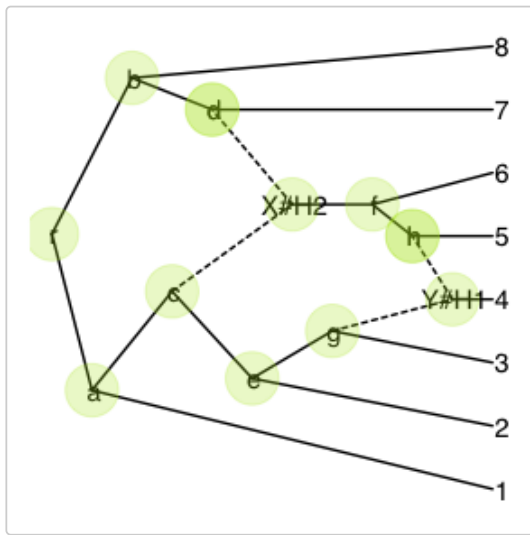
Plot an explicit network:

```
ggevonet(z) + geom_tiplab() + geom_nodelab()
```
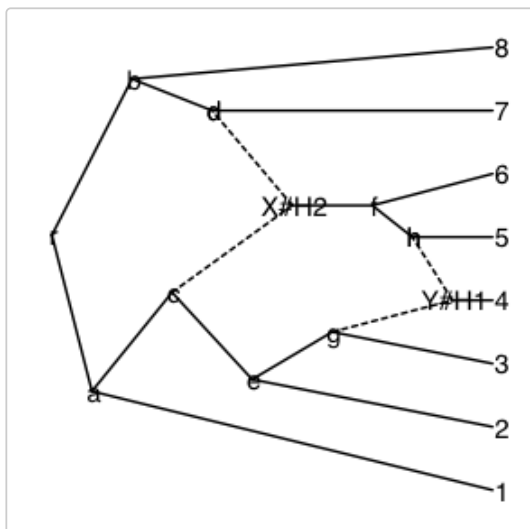


```
p <- ggevonet(z, layout = "slanted") + geom_tiplab() + geom_nodelab()
p + geom_tiplab(size=3, color="purple")
```



```
p + geom_nodepoint(color="#b5e521", alpha=1/4, size=10)
```

## 5. Summary

This vignette illustrates all the functions in the R package *tanggle*, and [rovide some examples on how to plot both explicit and implicit networks. The split network plots should take most of the functions compatible with unrooted trees in ggtree. The explicit network plots are phylogram or slanted.

## 6. Session info

```
sessionInfo()
#> R version 3.6.3 (2020-02-29)
#> Platform: x86_64-apple-darwin15.6.0 (64-bit)
#> Running under: macOS Catalina 10.15.7
#>
#> Matrix products: default
#> BLAS:
#>        /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versions/A/libBLAS.
#>
#> LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
#>
#> locale:
#> [1] en_AU.UTF-8/en_AU.UTF-8/en_AU.UTF-8/C/en_AU.UTF-8/en_AU.UTF-8
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#>  [1] phangorn_2.5.5   ape_5.4-1       tanggle_0.99.0  ggtree_2.3.5.993 ggplot2_3.3.2
#>      BiocStyle_2.14.4 devtools_2.3.2  usethis_1.6.3
#>  [9] tinytex_0.26     knitr_1.30
#>
#> loaded via a namespace (and not attached):
#>  [1] pkgload_1.1.0       tidyr_1.1.2         jsonlite_1.7.1      assertthat_0.2.1
#>      BiocManager_1.30.10 rvcheck_0.1.8
#>  [7] yaml_2.2.1          remotes_2.2.0       sessioninfo_1.1.1   pillar_1.4.6
#>      backports_1.1.10    lattice_0.20-41
#> [13] glue_1.4.2          quadprog_1.5-8      digest_0.6.25       promises_1.1.1      colorspace_1.4-
#>      1   websocket_1.3.1
#> [19] htmltools_0.5.0     httpuv_1.5.4        Matrix_1.2-18       pkgconfig_2.0.3     servr_0.18
#>      purrr_0.3.4
#> [25] patchwork_1.0.1     tidytree_0.3.3      scales_1.1.1        processx_3.4.4      later_1.1.0.1
#>      tibble_3.0.3
#> [31] generics_0.0.2      farver_2.0.3        ellipsis_0.3.1      withr_2.3.0         lazyeval_0.2.2
#>      cli_2.0.2
#> [37] mime_0.9            magrittr_1.5        crayon_1.3.4        pagedown_0.11       memoise_1.1.0
#>      evaluate_0.14
#> [43] ps_1.3.4            fs_1.5.0            fansi_0.4.1         nlme_3.1-149        pkgbuild_1.1.0
#>      tools_3.6.3
#> [49] prettyunits_1.1.1   lifecycle_0.2.0     stringr_1.4.0       aplot_0.0.6         munsell_0.5.0
#>      callr_3.4.4
#> [55] compiler_3.6.3      rlang_0.4.7         grid_3.6.3          rstudioapi_0.11     igraph_1.2.5
#>      labeling_0.3
#> [61] rmarkdown_2.3       testthat_2.3.2      gtable_0.3.0        R6_2.4.1            dplyr_1.0.2
#>      fastmatch_1.1-0
#> [67] rprojroot_1.3-2     treeio_1.13.1       desc_1.2.0          stringi_1.5.3       parallel_3.6.3
#>      Rcpp_1.0.5
#> [73] vctrs_0.3.4         tidyselect_1.1.0    xfun_0.17
```

## 7. References

Bryant, David, and Vincent Moulton. 2004. "Neighbor-Net: An Agglomerative Method for the Construction of Phylogenetic Networks." *Molecular Biology and Evolution* 21 (2): 255–65. https://doi.org/10.1093/molbev/msh018.

Cardona, Gabriel, Francesc Rosselló, and Gabriel Valiente. 2008. "Extended Newick: It Is Time for a Standard Representation of Phylogenetic Networks." *BMC Bioinformatics* 9 (1): 532. https://doi.org/10.1186/1471-2105-9-532.

Holland, Barbara R., Katharina T. Huber, Vincent Moulton, and Peter J. Lockhart. 2004. "Using Consensus Networks to Visualize Contradictory Evidence for Species Phylogeny." *Molecular Biology and Evolution* 21 (7): 1459–61. https://doi.org/10.1093/molbev/msh145.

Huson, D. H., and D. Bryant. 2006. "Application of Phylogenetic Networks in Evolutionary Studies." *Molecular Biology and Evolution* 23 (2): 254–67.

Paradis, Emmanuel, and Klaus Schliep. 2018. "Ape 5.0: An Environment for Modern Phylogenetics and Evolutionary Analyses in R." *Bioinformatics* 35 (3): 526–28.

Schliep, Klaus Peter. 2011. "Phangorn: Phylogenetic Analysis in R." *Bioinformatics* 27 (4): 592–93. https://doi.org/10.1093/bioinformatics/btq706.

Stamatakis, A. 2014. "raXml Version 8: A Tool for Phylogenetic Analysis and Post-Analysis of Large Phylogenies." *Bioinformatics* 30 (9): 1312–3.

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

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.

---

1. klaus.schliep@gmail.com↩
2. marta.vidalga@gmail.com↩
3. solislemus@wisc.edu↩