UNIVERSITY OF GOTHENBURG

Phylogenetic and Comparative Data in R

Emmanuel Paradis

1 June 2009

Three things to know about ape. . .

- close to R's logic ⇒ integration with standard statistical methods
- concerns a continuum of users from “low” to “high” levels
- “evolving”: new methods, bug fixing, web site

. . . and many others `http://ape.mpl.ird.fr/`

16 packages on CRAN “use” ape.

Main ideas on designing data structures in ape:

- Flat structures: elements are grouped according to their type (*vs.* recursive structures in other languages)

- Interaction with other R data structures (e.g., data frames)

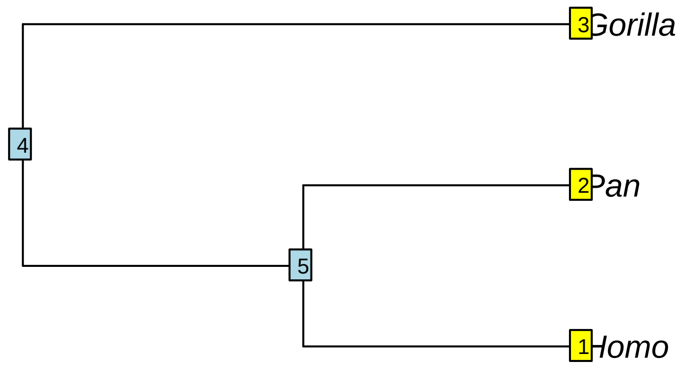
$\Rightarrow$ intensive use of numeric indexing

ape has three main data classes: `"phylo"`, `"multiPhylo"`, and `"DNAbin"`

Often a slight knowledge of the data structures is useful (e.g., when plotting trees and labels on nodes).

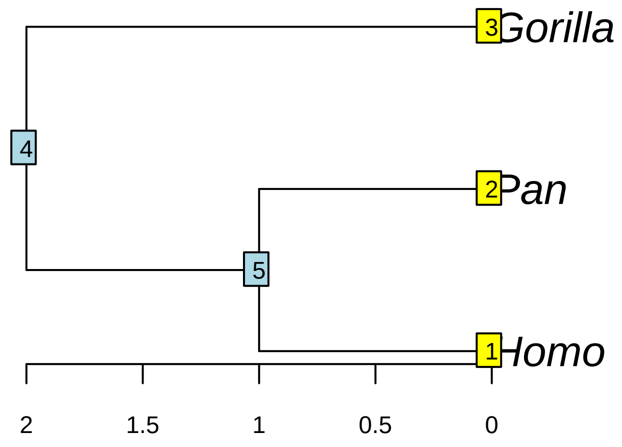
Trees

```
> x <- "((Homo,Pan),Gorilla);"
> tr <- read.tree(text = x)
> str(tr)
List of 3
 $ edge      : num [1:4, 1:2] 4 5 5 4 5 1 2 3
 $ tip.label: chr [1:3] "Homo" "Pan" "Gorilla"
 $ Nnode     : int 2
 - attr(*, "class")= chr "phylo"
> tr$edge
     [,1] [,2]
[1,]    4    5
[2,]    5    1
[3,]    5    2
[4,]    4    3
> plot(tr)
> nodelabels()
> tiplabels()
```



```
> x <- "((Homo:1,Pan:1):1,Gorilla:2);"
> tr2 <- read.tree(text = x)
> str(tr2)
List of 4
 $ edge       : int [1:4, 1:2] 4 5 5 4 5 1 2 3
 $ Nnode      : int 2
 $ tip.label  : chr [1:3] "Homo" "Pan" "Gorilla"
 $ edge.length: num [1:4] 1 1 1 2
 - attr(*, "class")= chr "phylo"

> plot(tr2)
> axisPhylo()
> nodelabels()
> tiplabels()
```



Trees of class `"phylo"` code for phylogenetic trees:

- tips (= leaves, terminal nodes) are observations (species, populations, sequences, . . .);
- no reticulation;
- (internal) nodes may be bi- or multifurcating (degree 3 or more, but not of degree 2 except possibly for the root).

Multichotomies ("hard") are taken into account explicitly:

```
> tr3 <- read.tree(text = "(a,b,c);")
> tr3$edge
     [,1] [,2]
[1,]    4    1
[2,]    4    2
[3,]    4    3
> str(tr3)
```

```
List of 3
 $ edge      : num [1:3, 1:2] 4 4 4 1 2 3
 $ tip.label: chr [1:3] "a" "b" "c"
 $ Nnode     : int 1
 - attr(*, "class")= chr "phylo"
```

Contrast with (“soft”):

```
> tr4 <- read.tree(text = "((a:1,b:1):0,c:1);")
> str(tr4)
List of 4
 $ edge       : int [1:4, 1:2] 4 5 5 4 5 1 2 3
 $ Nnode      : int 2
 $ tip.label  : chr [1:3] "a" "b" "c"
 $ edge.length: num [1:4] 0 1 1 1
 - attr(*, "class")= chr "phylo"
```

A tree with “soft” multichotomies requires branch lengths. By default, the plots will be the same (except for an expert’s eye), but not with:

```
> plot(tr3, use.edge.length = FALSE)
> plot(tr4, use.edge.length = FALSE)
```

Other tree structures in R:

`"hclust"`: ultrametric, binary trees only; `as.hclust` and `as.phylo` available

`"dendrogram"`: recursive structures with lists (very versatile)

`"mst"` (ape), `"haploNet"` (pegas): internal nodes are also observations

For all these classes: `plot` methods are available.

The `"multiPhylo"` class is a simple list of objects of class `"phylo"`: appropriate `plot` and `[` methods exist.

If all trees have the same tip labels, there can be a common `"TipLabel"` attribute.

Input/Output of Tree Files

ape has functions to read and write tree files in two widespread formats: Newick and NEXUS.

The Newick format is a parenthetic representation of the hierarchical relations in a tree. It has many variants (e.g., in its simplest form: `()` represents a single node). Examples of trees accepted by ape:

```
(a);
(a:1);
(a,b);
((a,b)c,d)e;
(a:1,b:1);
(a:1,b:1):1;
(a,b,c,d);
tree1(a,b);⋆
```

⋆ fixed in ape 2.3-1

- Comments (`[....]`) and spaces are ignored.
- Skeletons of trees, such as `((,),);`, are not accepted.

`read.tree` reads a Newick file and returns an object of class `"phylo"` or `"multiPhylo"`.

`write.tree` writes one or several trees in a Newick file.

Nexus is a file format designed specifically for phylogenetic data (trees, phenotypic data, molecular sequences, . . .).

Trees are coded in Newick strings with the distinction that labels can be substituted by integers (tokens). This is useful to store many trees with the same taxa:

```
> t1 <- read.tree(text = "((Pan,Homo),Gorilla);")
> t2 <- read.tree(text = "((Gorilla,Pan),Homo);")
> write.nexus(list(t1, t2))
```

```
#NEXUS
[R-package APE, Thu May 28 10:29:01 2009]

BEGIN TAXA;
        DIMENSIONS NTAX = 3;
        TAXLABELS
                Pan
                Homo
                Gorilla
        ;
END;
BEGIN TREES;
        TRANSLATE
                1       Pan,
                2       Homo,
                3       Gorilla
        ;
        TREE * UNTITLED = [&R] ((1,2),3);
        TREE * UNTITLED = [&R] ((3,1),2);
END;
```

`read.nexus` reads tree(s) from a Nexus file.
`write.nexus` writes tree(s) into a Nexus file.

DNA Sequences

The class `"DNAbin"` stores nucleotides in bytes (mode `"raw"`): they can be in a vector (single sequence), in a matrix (set of aligned sequences), or a list (set of any sequences).

The usual attributes (names of a list, rownames of a matrix) are used as labels of the individual sequences.

As these are standard R objects, usual manipulations can be done with `[`, `[[`, `$`, `cbind`, and `rbind` (and `c` in the last version of ape).

Phenotypic data

- Continuous traits: numeric vectors.
- Discrete traits: factors (= vectors of integers); the attributes `levels` give the names of the different categories.

Vectors and factors can have `names` used to match with the `tip.label` element of trees.

They can be grouped in a data frame in which case the `rownames` (mandatory for data frames) can be matched with `tip.label`. (Note that `names` applied on a data frame returns its `colnames`, because it is a list.)

Labels can be managed in R with several functions: `paste`, `gsub`, `substr`, `chartr`, `toupper`, `...`

ape has the function `makeLabel` that allows:

- truncation at a fixed length,
- substitution of blanks by a specified character (underscore by default),
- deletion of some characters (parentheses. . .),
- make unique labels.

Input/Output of Phenotypic data

R has a very flexible function to read data in tabular form in a text file:

```
> args(read.table)
function (file, header = FALSE, sep = "", quote = "\"'", dec=".",
row.names, col.names, as.is = !stringsAsFactors, na.strings="NA",
colClasses = NA, nrows = -1, skip = 0, check.names = TRUE,
fill = !blank.lines.skip, strip.white=FALSE, blank.lines.skip=TRUE,
comment.char = "#", allowEscapes = FALSE, flush = FALSE,
stringsAsFactors = default.stringsAsFactors(), encoding="unknown")
```

`read.csv` and `read.delim` have defaults well-suited to read files exported from Excel (e.g., `fill = TRUE`).

`read.csv2` and `read.delim2` have `dec = ","` (and thus `sep = ";"` for the former).

The package foreign can read files from DBF, Stata, S, SAS, SPSS, Epi-Info, Minitab and Systat.

> “The most common R data import/export question seems to be ‘how do I read an Excel spreadsheet’.” (R Data Import/Export manuel)

Three solutions:

Simple: save the spreadsheet in text and use `read.delim` or `read.csv`.

Practical: in Excel (or OpenOffice) select some cells with the mouse, copy and paste in a text file that will be with `read.delim`.

Efficient: if Perl is installed: `read.xls` in the package gdata (interesting under Linux).

Tree Manipulation in ape

Directory of functions:

```
root                compute.brlen     Ntip
unroot                                Nnode
drop.tip            makeNodeLabel     Nedge
extract.clade       makeLabel⋆
bind.tree                             is.rooted
ladderize           multi2di          is.binary.tree
rotate              di2multi          is.ultrametric

mrca                                  all.equal⋆
vcv.phylo                             unique⋆
branching.times
```

⋆generic function