Center for Tropical Forest Science R Package Manual

Pamela Hall, Suzanne Lao, Ellen Connell and Marie Massa

Version 1.00 March 29, 2006

3.0 Using the R Help Facility

The purpose of this chapter is to orient the user to the format and terminology used in the R help facility so that the help pages for R functions become a substantial resource for the user. The R help pages are very similar in form and terminology to Unix man pages.

3.1 Help Pages

The R help facility contains a collection of help pages each of which provides a description and explanation of the use and arguments of a particular function. Help pages are an invaluable resource in R. The CTFS R package help pages that follow the R format. However, these pages tend to be a bit more chatty as they are written for CTFS users specifically. There are overview help pages for many of the collections of CTFS functions as well as help pages for each function (as required for R packages). The CTFS help pages can be accessed within an R.

Help pages appear as simple text "man" pages by default. If *help.start()* is run (often included in the R preferences file or a startup file), then help pages in HTML format appear. From here it is assumed that the HTML format is available.

3.2 Accessing Help Pages

Help pages can be accessed from within the open R environment in by:

(1) selecting "HTML help" from the HELP menu
(2) ?topic
(3) help(topic)
(4) help.search("concept")

Selecting R HELP from the Help Menu will open an html file with links to installed R packages. Let's follow the links through a test run of the CTFS R package.

Open R Help, click on "Packages"

An index of all available R packages that came with the R installation and any locally installed packages appears.

Click on "CTFS"

An index of all available functions in the CTFS package appear with a short (title) description of each function. You can follow these links to help pages on individual files. Note that at the top of the page "overview" and "directory" are provided.

Click on "overview"

An index to CTFS "vignettes" appears. A "vignette" is a document that can take on just about any form and any content. These documents are (by default) viewed in pdf format. In the case of CTFS, the vignettes are these manual chapters.

Explore the links and get an understanding of how the help pages are interconnected.

From the R Console, *?topic* or *help(topic)* will bring up the html pages for the specific function that is named *topic*. If the name is not correctly typed you get:

```
> help(mort)
Error in help(mort) : No documentation for 'mort' in specified packages and libraries: you could try 'help.search("mort")'
```

If you don't know the name of the function, try the *help.search("topic")* form of help. R will provide a list of help pages that contain information on *topic*. Chose the ones you want and each will appear in a separate browser window.

For a topic specified by special characters, the special characters must be enclosed in double or single quotes. The use of double quotes makes the special characters a character string. Examples of such special characters are: "!", "%%", "&" and "[[". See the help pages of these special characters to learn about what they do. The use of double quotes is also necessary to find the help pages for words which have special syntactic meaning in R. For example syntactic words include: "if", "for", "while", "repeat" and "function". See the help pages of these words to learn what they do. For example:

```
> ?"[["
> help("[[")
> ?"if"
> help("if")
```

3.3 Interpreting Help Pages

Help pages, including the CTFS help pages, follow a general format that closely resembles that of UNIX man pages. The names of the function and of its package are written at the top of each help page. Each help page contains a brief description of the function's application, and is divided into sections. The most important sections are: Description, Usage, Arguments, Details, Value and Examples. The names and number of sections vary from help page to help page depending on the author of the help page. The help page of the CTFS function *growth()* will be used to illustrate how to read a help page.

The **Title** is a single sentence description of the function and is what appears in the index lists for function description.
Annual Growth Rates by Categories (User defined groups)

The **Description** section consists of a short paragraph which details an explanation of the function and its purpose.

Description

Computes annual growth rate for all trees or any user defined categorization of trees. Two growth rates can be computed: simple change in dbh over time and relative growth rate. Unrealistically high and low values are removed from the summary values. Growth rate in mm dbh per year, relative growth rate in % change in dbh per year, standard deviation or 95% confidence limits, and sample size are provided. Two datasets must be used, one for each census.

The **Usage** section contains basic format for invoking a function which is called the usage line. It consists of the function name followed by all of the function's arguments inside parenthesis. For example, a usage line would appear in the following format:

function name (argument 1, argument 2, argument 3, ...)

Usage
growth(census1, census2, rounddown = FALSE, method = "I", stdev = FALSE,
mindbh = 10, err.limit = 4, maxgrow = 75, split1 = NULL, split2 = NULL)

Argument(s) are the variable(s) provided to a function when it is used either when typed into the R console by the user or when used by another function ("called" from inside another function). Arguments provide functions with the values that are needed to perform their purposes. Oftentimes, some of the arguments specified in the usage line are set equal to specific values or variables. These are referred to as "default" values. If no default values are provided, then the function must be used with a name for the variable without a default value

Because the arguments *cens1*, *cens2*, *rounddown*, *method*, *stdev*, *split1* and *split2* are all set equal to values in the usage line, the user can infer that these arguments have default values. The function *growth ()* must, as a minimum, be passed the name of a *datafile.*

Here is the detailed description of some of the arguments in *growth()*:

census1: name of census datafile for first census, must be a ***data frame****, must be of same length as census2.*

census2: name of census datafile for second census, must be a ***data frame****, must be of same length as census1.*

rounddown: logical value in caps. When TRUE, if either of census is < 55, then the floor of the dbh value / 5 is provided. When FALSE, no change in the dbh is made.

census1 and *census2* must be provided because no defaults exist. The rest of the parameters have defaults and those values will be used unless specified otherwise in the use of the function. These other parameters must be addressed by their name.

Look down at the bottom of the page for examples on how to use the function and its arguments. Note that these examples (should be) can be run in the R session if the appropriate datafiles are available.

1. Default use of growth()

```
        > growth(tst.bci90.full,tst.bi95.full) -> growthI.out
>growth(tst.bci90.full,tst.bci95.full,method="E") -> growthE.out
```

The **Details** section generally describes the process and computations that a function undergoes as it runs. For example, if the function is computing a complex mathematical equation this section would explain the mathematics performed by the function. Help Page authors often include additional miscellaneous information in this section. In the CTFS package there is an additional help page for analysis purpose of a collection of functions. For instance for *growth()* there is the CTFS.growth help page. The direct link to this help page is provided in the Details section of lower on the page in the See Also section.

The **Value** section explains the value that the function returns, i.e. the output of the function. Many functions return complex data and this section is crucial to interpreting the different aspects of that data. Although the user might be able to guess what these variable names represent, this section of the help page clearly explains what each name actually represents

In this example, *growth()* returns a list and the component of the list and their names are provided so the user can address them properly to view the values that *growth()* computes. The names on the list are specified by the $.

In the case of *growth()* some of the complexities of the *split1* and *split2* variables are explored. The variables appear to be very complicated in the usage statement. However, this is only because the function is computing default, effectively "empty" values for these variables unless the user provides otherwise. Here is the example of providing user defined values for an argument of some complexity.

The **See Also** section contains links to functions that are related to *growth ()*. Reading these help pages will further the user's understanding of this function.

The **Examples** section provides examples of how the user would call the function in the R console. For example, if the user has the CTFS package, appropriate datasets and objects loaded in the R environment, which can be copied from CTFS help pages and paste them unmodified into the R console. Here is the example for creating a vector to use for calculating mean growth rates for each species instead of the default of an overall mean growth rate for all trees.

```
> tst.bci9095.full$sp->spp.vct
> dim(tst.bci90.full)
[1] 12421    14
> length(spp.vct)
[1] 12421
```

A vector of species names is made from the dataset such that the vector has a row for the name of each tree and in the same order as the trees in the dataset.

```
    > growth(tst.bci90.full,tst.bci95.full,split1=spp.vct) -> growth.spp.out
```

The function is run using the user created value of *split1*. Note that all the user does is set *split1* = *spp.vct*, the new variable created. All the information provided by the function the default value for *split1* is producing is ignored and only the contents of *spp.vct* are used in the function.

Here is what the function returns. Note that it is, indeed, a list.

```
> growth.spp.out
$N
      all
alsebl 7111
psycde   20
socrex  483

$rate
          all
alsebl 0.5776014
psycde 0.1742765
socrex 1.6349520

$clim
           all
alsebl 0.03320377
psycde 0.16768391
socrex 0.25021546

$dbhmean
          all
alsebl 49.93489
psycde 13.55000
socrex 91.02070

$meanyrs
          all
alsebl 4.768006
psycde 4.836961
socrex 4.795853

$date1
          all
```

```
alsebl 3507.439
psycde 3405.200
socrex 3437.186

$date2
         all
alsebl 5248.954
psycde 5171.900
socrex 5188.872
```

3.4 Viewing Functions in R
The source code of functions that are currently loaded into any R environment can be viewed in R by typing the function's name without the (). The source code for functions not currently loaded into R can be viewed by opening the file in which the function is saved using a text editor. The individual functions of a locally installed package can be viewed in *$R_HOME/library/PackageName/R-ex.*