Data Structuring

T. Eiting, M. Rosario, C.-Y. Kuo

09.21.2012

1 Introduction

1.1 Welcome

This document covers the basics of how to structure your data. This includes things like performing transformations in your data, substituting other values for NAs, dropping rows or columns from your dataset, merging datasets, etc. As part of this, we discuss how to make simple vectors and other types of data, and how to index your dataset so that you can tell R to apply some function to a specific cell (or cells) within your dataset. If you are following along on your own, we recommend typing the code yourself, so that you become familiar with the nuts and bolts.

1.2 Getting Started

Let's briefly bring our R session up to where we left off last time.

```
# clear our your workspace
rm(list = ls())

# set your working directory
setwd("D:/R")

# check to make sure your working directory is correct
getwd()

## [1] "D:/R"


# read in batbrains
geospiza <- read.table("geospiza.txt", sep = "", header = T, na.strings = "")

# check to make sure import was done correctly
str(geospiza)

## 'data.frame': 13 obs. of  5 variables:
##  $ wingL  : num  4.4 4.35 4.22 4.26 4.24 ...
##  $ tarsusL: num  3.04 2.98 2.9 2.93 2.89 ...
##  $ culmenL: num  2.72 2.65 2.28 2.62 2.41 ...
##  $ beakD  : num  2.82 2.51 2.01 2.14 2.36 ...
##  $ gonysW : num  2.68 2.36 1.93 2.04 2.22 ...
```

2 Data Structuring

After you have brought in your data, chances are very high that there will be at least some actions you need to take before you proceed with your analyses, such as applying a log transformation to one of your variables.

All of these things we refer to as ”data structuring” because you are structuring your dataset so that it is ready for analysis.

2.1 Attaching and indexing your data

Fundamental to the many types of ”structuring” functions is an ability to refer to specific rows, columns, and cells in your dataset. This ability is known as indexing. Type the following code, and notice the output.

```
geospiza[, 4]

##  [1] 2.824 2.514 2.011 2.145 2.363 1.941 2.016 1.191 1.874 2.073
## [11] 1.548 2.347 2.230
```

If you compare that with the actual dataset, you will see that you just reproduced column 4 from the dataset (i.e. beak depth). Another way to perform this **exact same operation** is to use the following code.

```
geospiza$beakD

##  [1] 2.824 2.514 2.011 2.145 2.363 1.941 2.016 1.191 1.874 2.073
## [11] 1.548 2.347 2.230
```

Similar to the column indexing, you can refer just to a row, or a set of rows.

```
geospiza[7:9, ]

##          wingL tarsusL culmenL beakD gonysW
## pallida  4.265   3.089   2.430 2.016  1.949
## fusca    3.975   2.937   2.052 1.191  1.401
## parvulus 4.132   2.973   1.974 1.874  1.813
```

If you are going to be using a particular dataset to do lots of transformations, stats, or whatever, it could be a good idea to ”attach” that dataset so that you don’t always have to use the name of the dataset in your indexing.

```
attach(geospiza)

# now, just type the name of the variable, e.g. 'brain_size,' and
# see what happens
beakD

##  [1] 2.824 2.514 2.011 2.145 2.363 1.941 2.016 1.191 1.874 2.073
## [11] 1.548 2.347 2.230
```

Notice that we did not need to type the full name of the dataset to refer to our variable. The usefulness of this will become more apparent as you continue coding. **CAUTION: make sure to ”detach” your dataset when you are done using it**. This will ensure that you do not wind up doing transformations or other operations to data that you didn’t mean to.

```
detach(geospiza)

# now verify that it was detached. You should get an error when
# typing the following
beakD

## Error:  object 'beakD' not found
```

2.2 Transforming Variables and Adjusting your Dataset

Now let's take it one step further. Say we wanted to prepare a dataset to be included as an appendix to a publication. But in this case we want to exclude "beakD" because we wound up not using it in any of our analyses. One easy way to exclude this variable is as follows.

```
geospiza.apx <- geospiza[, -4]

# to double-check that our new dataset does in fact exclude beakD
str(geospiza.apx)

## 'data.frame': 13 obs. of  4 variables:
##  $ wingL  : num  4.4 4.35 4.22 4.26 4.24 ...
##  $ tarsusL: num  3.04 2.98 2.9 2.93 2.89 ...
##  $ culmenL: num  2.72 2.65 2.28 2.62 2.41 ...
##  $ gonysW : num  2.68 2.36 1.93 2.04 2.22 ...
```

In this hypothetical example, not only did we want to drop beak depth, but we also needed to log-transform tarsus length. Here's the code of how to log-transform tarsus length and append it to the end of the dataset.

```
geospiza.apx$logTarsus <- log(geospiza.apx$tarsusL)
str(geospiza.apx)

## 'data.frame': 13 obs. of  5 variables:
##  $ wingL    : num  4.4 4.35 4.22 4.26 4.24 ...
##  $ tarsusL  : num  3.04 2.98 2.9 2.93 2.89 ...
##  $ culmenL  : num  2.72 2.65 2.28 2.62 2.41 ...
##  $ gonysW   : num  2.68 2.36 1.93 2.04 2.22 ...
##  $ logTarsus: num  1.11 1.09 1.06 1.07 1.06 ...
```

Instead of appending the new log variable, you could have replaced the original tarsus length with log-tarsus length. We'll do this to the original "geospiza" dataset. Here's how.

```
# create a duplicate of the dataset so we don't modify the original
geospiza.test <- geospiza

# perform the transformation and store it in the position of the
# original variable
geospiza.test$tarsusL <- log(geospiza.test$tarsusL)

# rename the variable to reflect your change
names(geospiza.test)[2] <- "logTarsusL"

head(geospiza.test, n = 3)

##              wingL logTarsusL culmenL beakD gonysW
## magnirostris 4.404      1.112   2.725 2.824  2.676
## conirostris  4.350      1.093   2.654 2.514  2.360
## difficilis   4.224      1.064   2.277 2.011  1.930
```

Let's say you later collected some data on the habitat of your various species. You discovered that most of your *Geospiza* species prefer to live and forage near the ground. The only species that don't are *G. magnirostris*, *G. fortis*, *G. fusca*, and *G. parvulus*, who prefer to live and forage around the tops of the shrubs. You will code these two preferences as "ground" and "shrub," and add them back into your dataset. Here's one way to do this.

```r
# we will use the concatenation function, 'c', to perform this task
geospiza.apx$habitat <- c("shrub", "ground", "ground", "ground", "shrub",
    "ground", "ground", "shrub", "shrub", "ground", "ground", "ground",
    "ground")

# check to make sure all of the new values were correctly applied
geospiza.apx

##              wingL tarsusL culmenL gonysW logTarsus habitat
## magnirostris 4.404   3.039   2.725  2.676     1.112   shrub
## conirostris  4.350   2.984   2.654  2.360     1.093  ground
## difficilis   4.224   2.899   2.277  1.930     1.064  ground
## scandens     4.261   2.929   2.622  2.037     1.075  ground
## fortis       4.244   2.895   2.407  2.222     1.063   shrub
## fuliginosa   4.133   2.807   2.095  1.845     1.032  ground
## pallida      4.265   3.089   2.430  1.949     1.128  ground
## fusca        3.975   2.937   2.052  1.401     1.077   shrub
## parvulus     4.132   2.973   1.974  1.813     1.090   shrub
## pauper       4.232   3.036   2.187  1.962     1.111  ground
## Pinaroloxias 4.189   2.980   2.311  1.630     1.092  ground
## Platyspiza   4.420   3.271   2.331  2.282     1.185  ground
## psittacula   4.235   3.049   2.260  2.074     1.115  ground


# also be sure that the structure of your new variable is correct
str(geospiza.apx)

## 'data.frame': 13 obs. of  6 variables:
##  $ wingL    : num  4.4 4.35 4.22 4.26 4.24 ...
##  $ tarsusL  : num  3.04 2.98 2.9 2.93 2.89 ...
##  $ culmenL  : num  2.72 2.65 2.28 2.62 2.41 ...
##  $ gonysW   : num  2.68 2.36 1.93 2.04 2.22 ...
##  $ logTarsus: num  1.11 1.09 1.06 1.07 1.06 ...
##  $ habitat  : chr  "shrub" "ground" "ground" "ground" ...
```

Finally, if we want to write out our new file as a *.csv, use this code.

```r
write.csv(x = geospiza.apx, file = "geospiza_appendix.csv")
```

3 Exercises

3.1 In-class

1. Import ”CYAnoles.csv,” attach the data frame, log-transform ”pdiameter” and append it to the end of your dataset.

3.2 Take-home

1. Using the ”CYAnoles.csv” dataset again, make a new variable (call it whatever you want) that is the product of ”pheight” and ”pdiameter.” Then, drop the original ”pheight” and ”pdiameter” from the dataset.