

# Additional Course Materials

*Dillon Niederhut*

*24 May, 2016*

## Introduction

The following are materials that do not fit into the course as currently taught, but that may be useful for students later on.

## Data does not need to be in the local filesystem

R has an interface to curl called RCurl

```
#install.packages('RCurl')
library(RCurl)
```

```
## Loading required package: bitops
##
## Attaching package: 'RCurl'
##
## The following object is masked from 'package:tidyr':
##
##      complete
```

```
#install.packages("XML")
library(XML)
```

you can use this to access remote data

you may just want to read text lines from a webpage

```
RJ <- readLines("http://shakespeare.mit.edu/romeo_juliet/full.html")
RJ[1:25]
```

```
## [1] "<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN\""
## [2] " \"http://www.w3.org/TR/REC-html40/loose.dtd\">"
## [3] " <html>"
## [4] " <head>"
## [5] " <title>Romeo and Juliet: Entire Play"
## [6] " </title>"
## [7] " <meta http-equiv=\"Content-Type\" content=\"text/html; charset=iso-8859-1\">"
## [8] " <LINK rel=\"stylesheet\" type=\"text/css\" media=\"screen\""
## [9] "      href=\"/shake.css\">"
## [10] " </HEAD>"
## [11] " <body bgcolor=\"#ffffff\" text=\"#000000\">"
## [12] ""
```

```
## [13] "<table width=\"100%\" bgcolor=\"#CCF6F6\">"
## [14] "<tr><td class=\"play\" align=\"center\">Romeo and Juliet"
## [15] "<tr><td class=\"nav\" align=\"center\">"
## [16] "      <a href=\"/Shakespeare\">Shakespeare homepage</A> "
## [17] "      | <A href=\"/romeo_juliet/\">Romeo and Juliet</A> "
## [18] "      | Entire play"
## [19] "</table>"
## [20] ""
## [21] "<H3>ACT I</h3>"
## [22] "<h3>PROLOGUE</h3>"
## [23] "<blockquote>"
## [24] "<A NAME=1.0.1>Two households, both alike in dignity,</A><br>"
## [25] "<A NAME=1.0.2>In fair Verona, where we lay our scene,</A><br>"
```

and use the kinds of string manipulation we learned yesterday to retrieve the first lines of an act or a scene

```
RJ[grep("<h3>", RJ, perl=T)]
```

```
## [1] "<h3>PROLOGUE</h3>"
## [2] "<h3>SCENE I. Verona. A public place.</h3>"
## [3] "<h3>SCENE II. A street.</h3>"
## [4] "<h3>SCENE III. A room in Capulet's house.</h3>"
## [5] "<h3>SCENE IV. A street.</h3>"
## [6] "<h3>SCENE V. A hall in Capulet's house.</h3>"
## [7] "<h3>PROLOGUE</h3>"
## [8] "<h3>SCENE I. A lane by the wall of Capulet's orchard.</h3>"
## [9] "<h3>SCENE II. Capulet's orchard.</h3>"
## [10] "<h3>SCENE III. Friar Laurence's cell.</h3>"
## [11] "<h3>SCENE IV. A street.</h3>"
## [12] "<h3>SCENE V. Capulet's orchard.</h3>"
## [13] "<h3>SCENE VI. Friar Laurence's cell.</h3>"
## [14] "<h3>SCENE I. A public place.</h3>"
## [15] "<h3>SCENE II. Capulet's orchard.</h3>"
## [16] "<h3>SCENE III. Friar Laurence's cell.</h3>"
## [17] "<h3>SCENE IV. A room in Capulet's house.</h3>"
## [18] "<h3>SCENE V. Capulet's orchard.</h3>"
## [19] "<h3>SCENE I. Friar Laurence's cell.</h3>"
## [20] "<h3>SCENE II. Hall in Capulet's house.</h3>"
## [21] "<h3>SCENE III. Juliet's chamber.</h3>"
## [22] "<h3>SCENE IV. Hall in Capulet's house.</h3>"
## [23] "<h3>SCENE V. Juliet's chamber.</h3>"
## [24] "<h3>SCENE I. Mantua. A street.</h3>"
## [25] "<h3>SCENE II. Friar Laurence's cell.</h3>"
## [26] "<h3>SCENE III. A churchyard; in it a tomb belonging to the Capulets.</h3>"
```

```
RJ[grep("<h3>", RJ, perl=TRUE)]
```

```
## [1] "<h3>PROLOGUE</h3>"
## [2] "<h3>SCENE I. Verona. A public place.</h3>"
## [3] "<h3>SCENE II. A street.</h3>"
## [4] "<h3>SCENE III. A room in Capulet's house.</h3>"
## [5] "<h3>SCENE IV. A street.</h3>"
```

```
## [6] "<h3>SCENE V. A hall in Capulet's house.</h3>"
## [7] "<h3>PROLOGUE</h3>"
## [8] "<h3>SCENE I. A lane by the wall of Capulet's orchard.</h3>"
## [9] "<h3>SCENE II. Capulet's orchard.</h3>"
## [10] "<h3>SCENE III. Friar Laurence's cell.</h3>"
## [11] "<h3>SCENE IV. A street.</h3>"
## [12] "<h3>SCENE V. Capulet's orchard.</h3>"
## [13] "<h3>SCENE VI. Friar Laurence's cell.</h3>"
## [14] "<h3>SCENE I. A public place.</h3>"
## [15] "<h3>SCENE II. Capulet's orchard.</h3>"
## [16] "<h3>SCENE III. Friar Laurence's cell.</h3>"
## [17] "<h3>SCENE IV. A room in Capulet's house.</h3>"
## [18] "<h3>SCENE V. Capulet's orchard.</h3>"
## [19] "<h3>SCENE I. Friar Laurence's cell.</h3>"
## [20] "<h3>SCENE II. Hall in Capulet's house.</h3>"
## [21] "<h3>SCENE III. Juliet's chamber.</h3>"
## [22] "<h3>SCENE IV. Hall in Capulet's house.</h3>"
## [23] "<h3>SCENE V. Juliet's chamber.</h3>"
## [24] "<h3>SCENE I. Mantua. A street.</h3>"
## [25] "<h3>SCENE II. Friar Laurence's cell.</h3>"
## [26] "<h3>SCENE III. A churchyard; in it a tomb belonging to the Capulets.</h3>"
```

or maybe pull information out of an RSS feed

```
link <- "http://rss.nytimes.com/services/xml/rss/nyt/HomePage.xml"
page <- getURL(url = link)
xmlParse(file = page)
```

R also has libraries for pulling and parsing web pages

```
link<-"http://clerk.house.gov/evs/2014/ROLL_000.asp"
readHTMLTable(doc=link, header=T, which=1, stringsAsFactors=F)[1:10, ]
```

```
##      Roll   Date      Issue
## 1    99   6-Mar  H RES 501
## 2    98   5-Mar  H R 2126
## 3    97   5-Mar  H R 4118
## 4    96   5-Mar  H R 4118
## 5    95   5-Mar   H R 938
## 6    94   5-Mar  H RES 497
## 7    93   5-Mar  H RES 497
## 8    92   4-Mar  H RES 488
## 9    91   4-Mar  H R 3370
## 10   90  28-Feb   H R 899
##
##                                     Question Result
## 1                                     On Ordering the Previous Question      P
## 2      On Motion to Suspend the Rules and Pass, as Amended                  P
## 3                                     On Passage                             P
## 4                                     On Motion to Recommit with Instructions    F
## 5      On Motion to Suspend the Rules and Pass, as Amended                  P
## 6                                     On Agreeing to the Resolution              P
```

```
## 7                On Ordering the Previous Question      P
## 8 On Motion to Suspend the Rules and Agree, as Amended P
## 9 On Motion to Suspend the Rules and Pass, as Amended  P
## 10                On Passage                          P
##
## 1 Providing for consideration of the bill (H.R. 2824) Preventing Government Waste and Protecting Co
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
```

## Connecting to a database

why read from a database? they use less memory, are faster, create their own backups, and offer optimized querying/joining

databases generally come in two flavors, relational and non-relational, which has to do with how important schemas are (and is a bit beyond the scope of an R intro)

two popular relational databases are SQL (or one of its many flavors)

```
#are there websites that allow you to connect to test servers?
install.packages("RMySQL")
library(RMySQL)
con <- dbConnect(MySQL(),
  user="", password="",
  dbname="", host="localhost")
data <- fetch(dbSendQuery(con, "select * from table"), n=10)
con.exit(dbDisconnect(con))
```

and postgres

```
install.packages("RPostgreSQL")
library(RPostgreSQL)
con <- dbConnect(dbDriver("PostgreSQL"),
  dbname="",
  host="localhost",
  port=1234,
  user="",
  password="")
data <- dbReadTable(con, c("column1", "column2"))
dbDisconnect(con)
```

a popular non-relational database is MongoDB

```
install.packages("rmongodb")
library(rmongodb)
con <- mongo.create(host = localhost,
```

```

        name = "",
        username = "",
        password = "",
        db = "admin")
if(mongo.is.connected(con) == TRUE) {
  data <- mongo.find.all(con, "collection", list("city" = list( "$exists" = "true")))
}
mongo.destroy(con)

```

one quirk about mongo is that your connection always authenticates to the authentication database, not the database you are querying - this db is usually called 'admin'

## Data tidying with plyr

### enter plyr

- *plyr* is the go-to package for all your splitting-applying-combining needs
- Among its many benefits (above base R capabilities):
  - a) Don't have to worry about different name, argument, or output consistencies
  - b) Easily parallelized
  - c) Input from, and output to, data frames, matrices, and lists
  - d) Progress bars for lengthy computation
  - e) Informative error messages

### group-wise operations/plyr/selecting functions

- Two essential questions:
  1. What is the class of your input object?
  2. What is the class of your desired output object?
- If you want to split a **data** frame, and return results as a **data** frame, you use **ddply**
- If you want to split a **data** frame, and return results as a **list**, you use **dlply**
- If you want to split a **list**, and return results as a **data** frame, you use **ldply**

```

# plyr package
mydata <- read.csv("http://www.ats.ucla.edu/stat/data/binary.csv")
# Consider the case where we want to calculate descriptive statistics across admits and not-admits
# from the dataset and return them as a data.frame
ddata <- ddply(mydata, c("admit"), summarize,
               gpa.over3 = length(gpa[gpa>=3]),
               gpa.over3.5 = length(gpa[gpa>=3.5]),
               gpa.over3per = length(gpa[gpa>=3])/length(gpa),
               gpa.over3.5per = length(gpa[gpa>=3.5])/length(gpa))
)

```

## Group-wise Operations/plyr/functions

- plyr can accomodate any user-defined function, but it also comes with some pre-defined functions that assist with the most common split-apply-combine tasks

- We've already seen **summarize**, which creates user-specified vectors and combines them into a data.frame. Here are some other helpful functions:

**transform**: applies a function to a data.frame and adds new vectors (columns) to it

## add a column containing the average gre score of students

```
mydata <- dplyr::transform(mydata, c("admit"), transform,
                             gre.ave=mean(x=gre, na.rm=T),
                             gre.sd = sd(x=gre, na.rm=T))
head(mydata)
unique(mydata$gre.ave)
)
```

side note: note that **transform** can't do transformations that involve the results of *other* transformations from the same call

Another very useful function is **arrange**, which orders a data frame on the basis of column contents

```
# Another very useful function is arrange, which orders a data frame on the basis of column contents
# arrange by "rank"
mydata.rank <- plyr::arrange(mydata, rank)
# arrange by "rank", descending
mydata.rank <- plyr::arrange(mydata, desc(rank))
# arrange by "rank", then "gre", then "gpa"
mydata.comb <- plyr::arrange(mydata, rank, desc(gre), desc(gpa))
head(mydata.comb)
```