Business Analytics

using the R language

Learning Objectives

- how to input data in R using various ways
- how to check for correct data input
- how to use special packages for fast data input
- how to input data from statistical file formats
- how to input data from databases
- how to input data from web (web scraping)

What will you learn from this lesson

- data input from various kinds of format
- efficient data input via various packages
- sql to R
- web scraping
- piping in R
- using json in R

Environment

ls() -lists objects

rm()-removes an object

gc() -does garbage collection and frees up memory

File Edit Code View Plots Session Build Debug Tools Help

Go to file/functior

Project: (None)

Console ~/

```
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: i686-pc-linux-gnu (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> ls()
[1] "a"     "i"     "iris3"
> rm(a)
> ls()
[1] "i"     "iris3"
> gc()
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 334887  9.0     597831 16.0   407500 10.9
Vcells 624499  4.8    1215808  9.3  1215802  9.3
>
```

Environment | **History**

Import Dataset | Clear | List

Global Environment

Data

iris3	50 obs. of 12 variables

Values

i	90L

Files | **Plots** | **Packages** | **Help** | **Viewer**

R: Search Results | Find in Topic

Search Results

The search string was **"kmeans"**

Vignettes:

broom::kmeans	kmeans with dplyr+broom	HTML	source	R code

Help pages:

amap::Kmeans	K-Means Clustering
broom::augment.kmeans	Tidying methods for kmeans objects
e1071::cmeans	Fuzzy C-Means Clustering

File Edit Code View Plots Session Build Debug Tools Help

Go to file/function

Project: (None)

Console ~/

```
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: i686-pc-linux-gnu (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> ls()
[1] "a"     "i"     "iris3"
> rm(a)
> ls()
[1] "i"     "iris3"
> gc()
         used (Mb) gc trigger (Mb) max used (Mb)
Ncells 334887  9.0     597831 16.0   407500 10.9
Vcells 624499  4.8    1215808  9.3  1215802  9.3
>
```

Environment | **History**

Import Dataset | Clear | List

Global Environment

Data	
iris3	50 obs. of 12 variables
Values	
i	90L

Files | **Plots** | **Packages** | **Help** | **Viewer**

R: Search Results | Find in Topic

Search Results

The search string was **"kmeans"**

Vignettes:

broom::kmeans	kmeans with dplyr+broom	HTML	source	R code

Help pages:

amap::Kmeans	K-Means Clustering
broom::augment.kmeans	Tidying methods for kmeans objects
e1071::cmeans	Fuzzy C-Means Clustering

File System

getwd()- get working directory

setwd()- set or change working directory

dir() - lists files in working directory

File Edit Code View Plots Session Build Debug Tools Help

Go to file/functior

Project: (None)

Console ~/Desktop/new/

```
> getwd()
[1] "/home/ajay/Desktop"
> setwd("/home/ajay/Desktop/new")
> dir()
[1] "obama"
>
```

Environment | **History**

Import Dataset | Clear | List

Global Environment

Data

iris3	50 obs. of 12 variables

Values

i	90L

Files | **Plots** | **Packages** | **Help** | **Viewer**

New Folder | Delete | Rename | More

Home

Name	Size	Modified
.RData	3.8 KB	May 2, 2015, 11:47 AM
.Rhistory	10.7 KB	May 10, 2015, 2:20 PM
17811636-Brain-function-as-gears-and-cogs-in-the-shape-of-a-human-head-as-a-medical-symbol-of-mental-health-c-Stock-Photo.jpg	139.8 KB	Apr 16, 2015, 9:41 AM
21.png	352.7 KB	May 4, 2015, 5:18 PM
2167434.jpg	32.8 KB	May 4, 2015, 5:36 PM
a.out	7.7 KB	May 2, 2015, 2:26 PM
anaconda		
animation		
animation2		
backports-3.18.1-1		
backports-3.18.1-1.tar.xz	8.6 MB	Dec 22, 2014, 3:14 AM
Call R and Python from base SAS.html	48.4 KB	May 5, 2015, 12:53 PM

File Edit Code View Plots Session Build Debug Tools Help

Go to file/functior

Project: (None)

Console ~/Desktop/new/

```
> getwd()
[1] "/home/ajay/Desktop"
> setwd("/home/ajay/Desktop/new")
> dir()
[1] "obama"
>
```

Environment | **History**

Import Dataset | Clear | List

Global Environment

Data

iris3	50 obs. of 12 variables

Values

i	90L

Files | **Plots** | **Packages** | **Help** | **Viewer**

New Folder | Delete | Rename | More

Home

Name	Size	Modified
.RData	3.8 KB	May 2, 2015, 11:47 AM
.Rhistory	10.7 KB	May 10, 2015, 2:20 PM
17811636-Brain-function-as-gears-and-cogs-in-the-shape-of-a-human-head-as-a-medical-symbol-of-mental-health-c-Stock-Photo.jpg	139.8 KB	Apr 16, 2015, 9:41 AM
21.png	352.7 KB	May 4, 2015, 5:18 PM
2167434.jpg	32.8 KB	May 4, 2015, 5:36 PM
a.out	7.7 KB	May 2, 2015, 2:26 PM
anaconda		
animation		
animation2		
backports-3.18.1-1		
backports-3.18.1-1.tar.xz	8.6 MB	Dec 22, 2014, 3:14 AM
Call R and Python from base SAS.html	48.4 KB	May 5, 2015, 12:53 PM

File Edit Code View Plots Session Build Debug Tools Help

Go to file/functior

Project: (None)

Console ~/Desktop/new/

```
> getwd()
[1] "/home/ajay/Desktop"
> setwd("/home/ajay/Desktop/new")
> dir()
[1] "obama"
>
```

Environment | **History**

Import Dataset | Clear | List

Global Environment

Data

iris3	50 obs. of 12 variables

Values

i	90L

Files | **Plots** | **Packages** | **Help** | **Viewer**

New Folder | Delete | Rename | More

Home

Name	Size	Modified
.RData	3.8 KB	May 2, 2015, 11:47 AM
.Rhistory	10.7 KB	May 10, 2015, 2:20 PM
17811636-Brain-function-as-gears-and-cogs-in-the-shape-of-a-human-head-as-a-medical-symbol-of-mental-health-c-Stock-Photo.jpg	139.8 KB	Apr 16, 2015, 9:41 AM
21.png	352.7 KB	May 4, 2015, 5:18 PM
2167434.jpg	32.8 KB	May 4, 2015, 5:36 PM
a.out	7.7 KB	May 2, 2015, 2:26 PM
anaconda		
animation		
animation2		
backports-3.18.1-1		
backports-3.18.1-1.tar.xz	8.6 MB	Dec 22, 2014, 3:14 AM
Call R and Python from base SAS.html	48.4 KB	May 5, 2015, 12:53 PM

Assigning

objectname=read.csv(filepath,parameters)

OR

objectname<-read.csv(filepath,parameters)

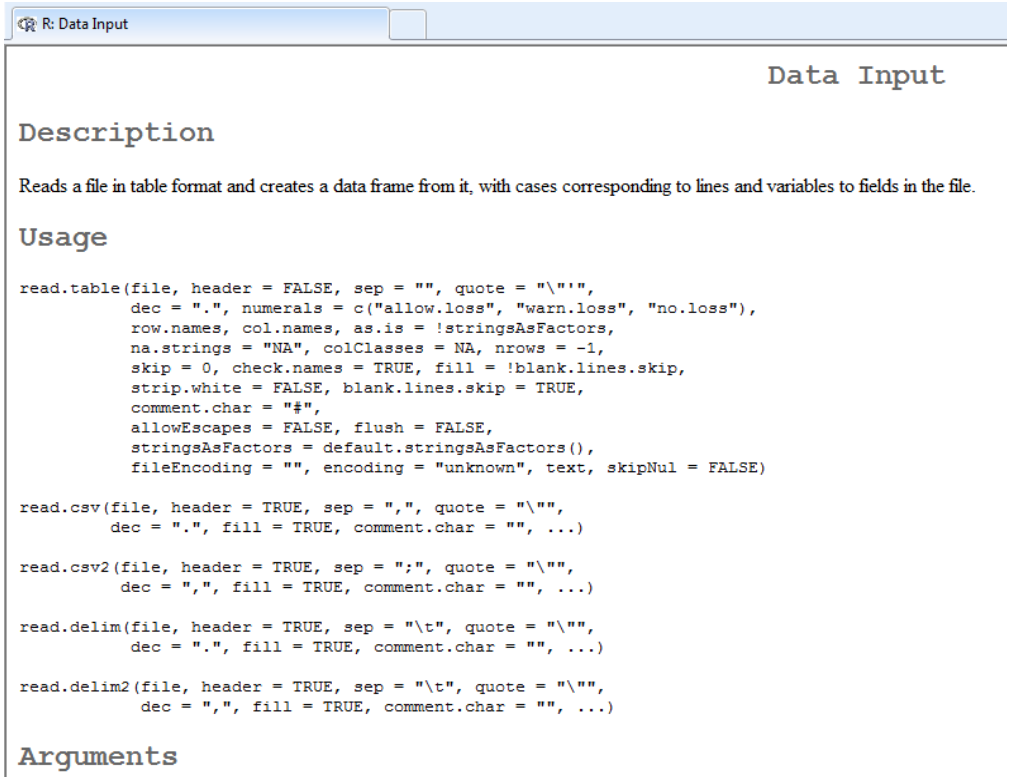
Data Input

read.table() or read.csv()

read.spss()

read.sas7bdat()

read.table()



R: Data Input

Data Input

Description

Reads a file in table format and creates a data frame from it, with cases corresponding to lines and variables to fields in the file.

Usage

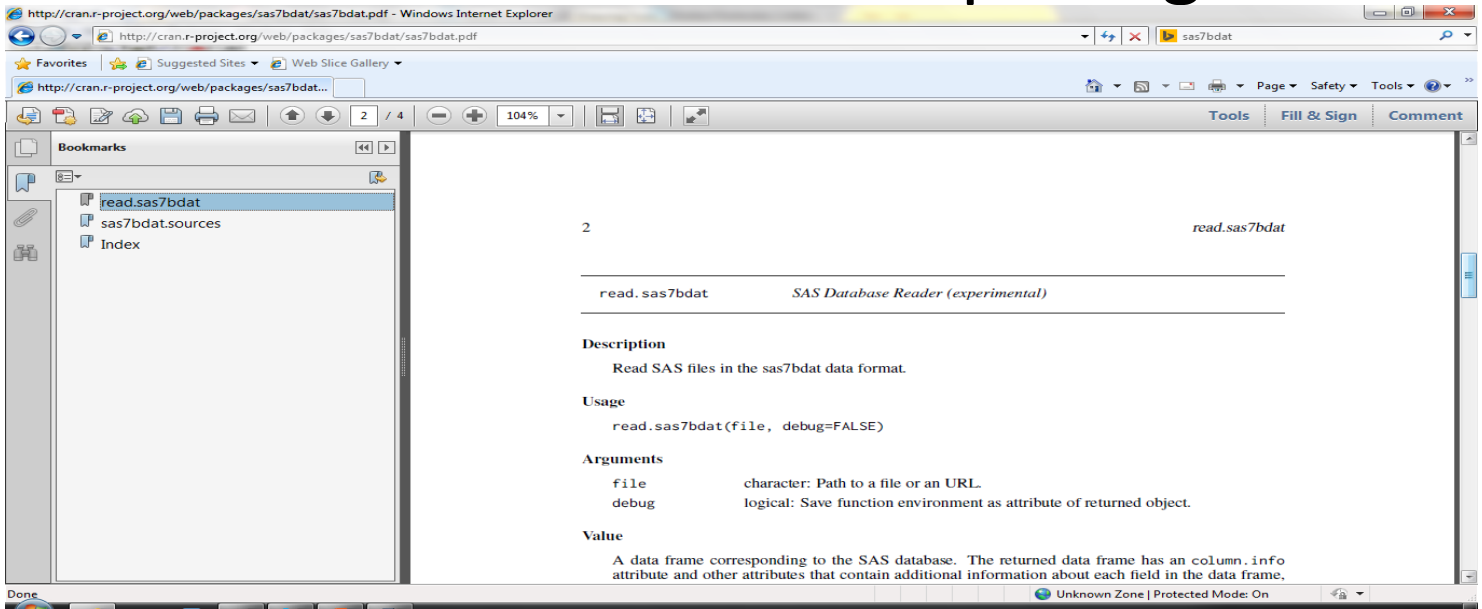
```
read.table(file, header = FALSE, sep = "", quote = "\"'",
           dec = ".", numerals = c("allow.loss", "warn.loss", "no.loss"),
           row.names, col.names, as.is = !stringsAsFactors,
           na.strings = "NA", colClasses = NA, nrows = -1,
           skip = 0, check.names = TRUE, fill = !blank.lines.skip,
           strip.white = FALSE, blank.lines.skip = TRUE,
           comment.char = "#",
           allowEscapes = FALSE, flush = FALSE,
           stringsAsFactors = default.stringsAsFactors(),
           fileEncoding = "", encoding = "unknown", text, skipNul = FALSE)

read.csv(file, header = TRUE, sep = ",", quote = "\"",
         dec = ".", fill = TRUE, comment.char = "", ...)

read.csv2(file, header = TRUE, sep = ";", quote = "\"",
          dec = ",", fill = TRUE, comment.char = "", ...)

read.delim(file, header = TRUE, sep = "\t", quote = "\"",
           dec = ".", fill = TRUE, comment.char = "", ...)

read.delim2(file, header = TRUE, sep = "\t", quote = "\"",
            dec = ",", fill = TRUE, comment.char = "", ...)
```

Arguments

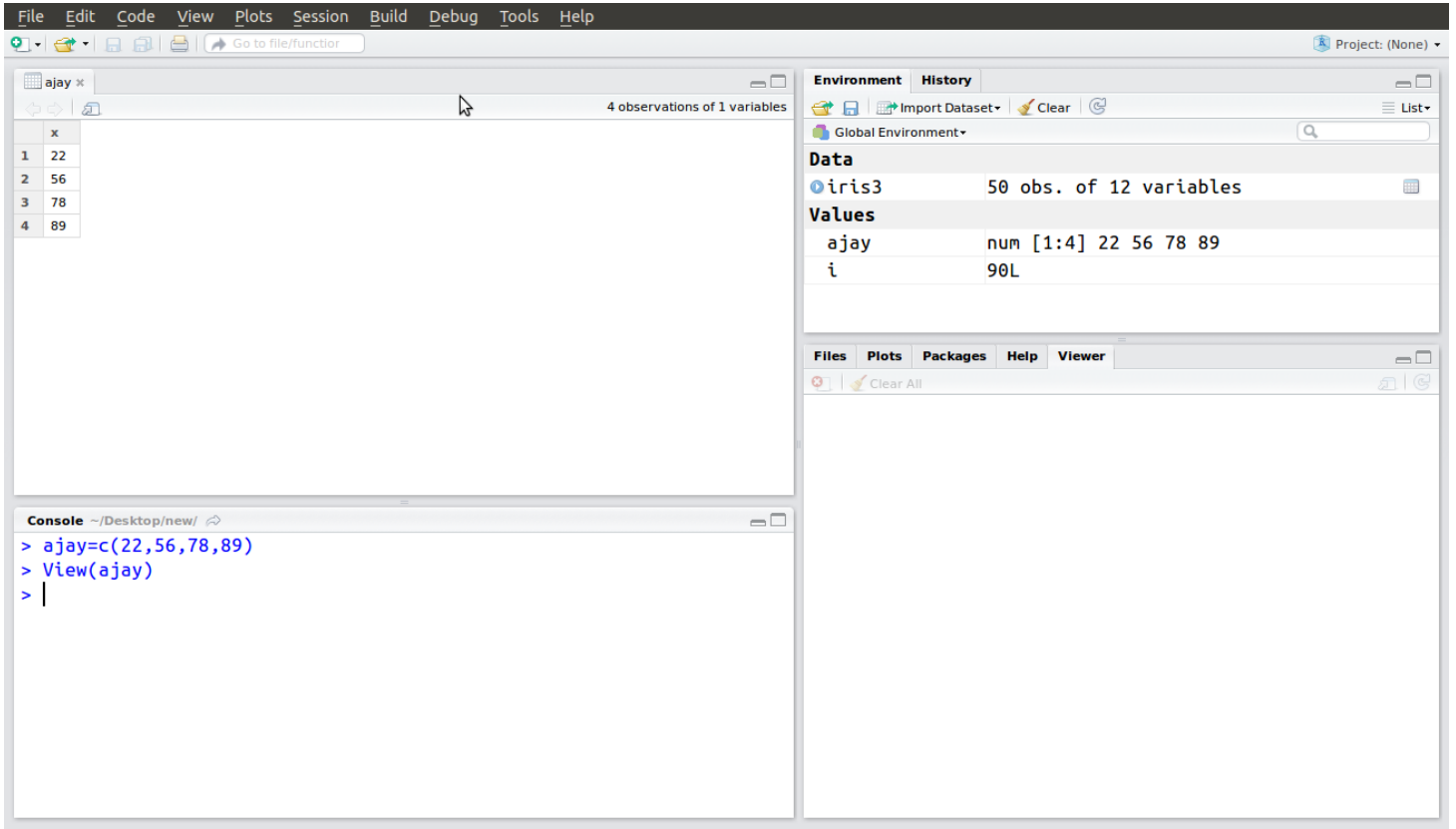
https://stat.ethz.ch/R-manual/R-devel/library/utils/html/read.table.html

Statistical formats

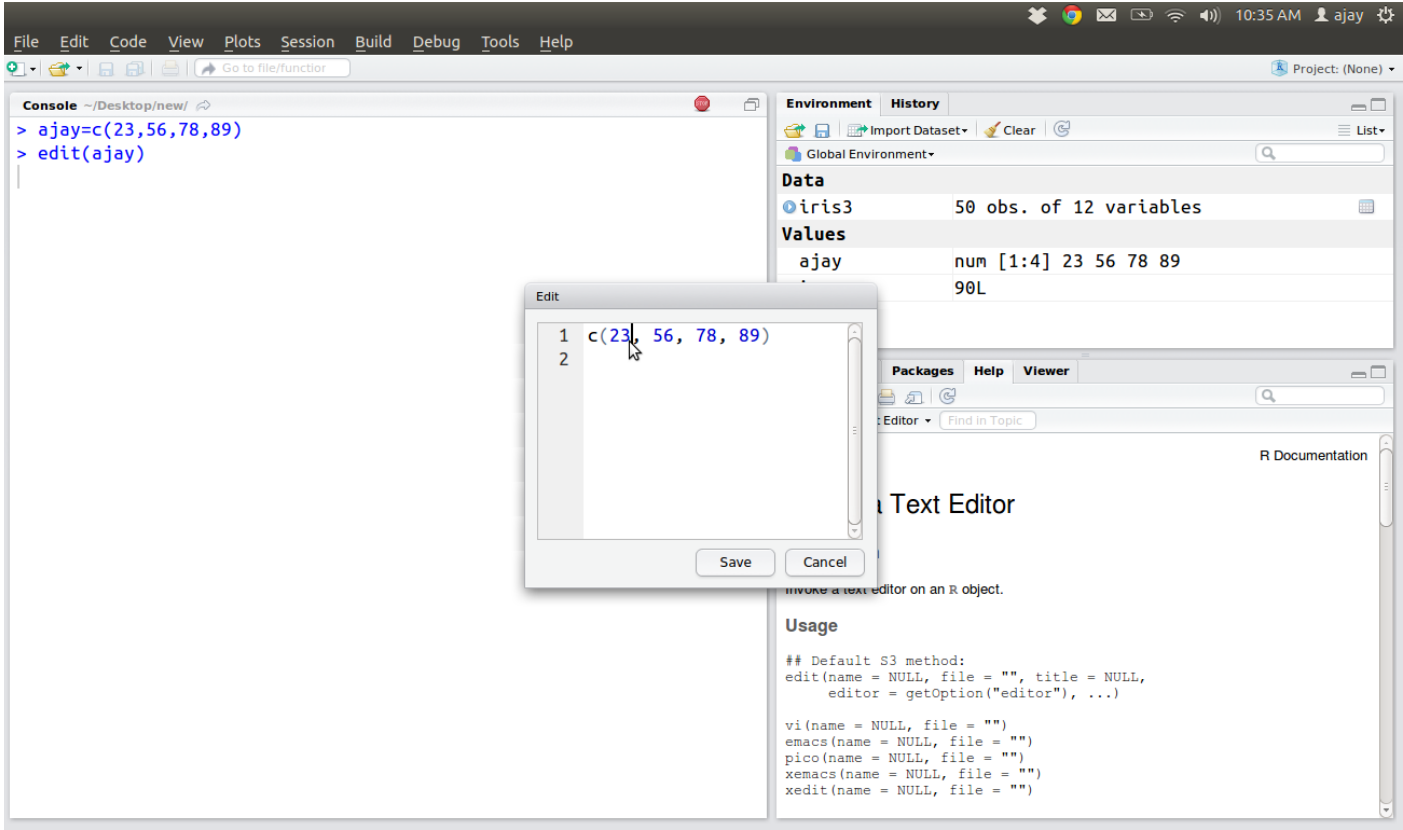
- read.spss from foreign package
- read.sas7bdat from sas7bdat package



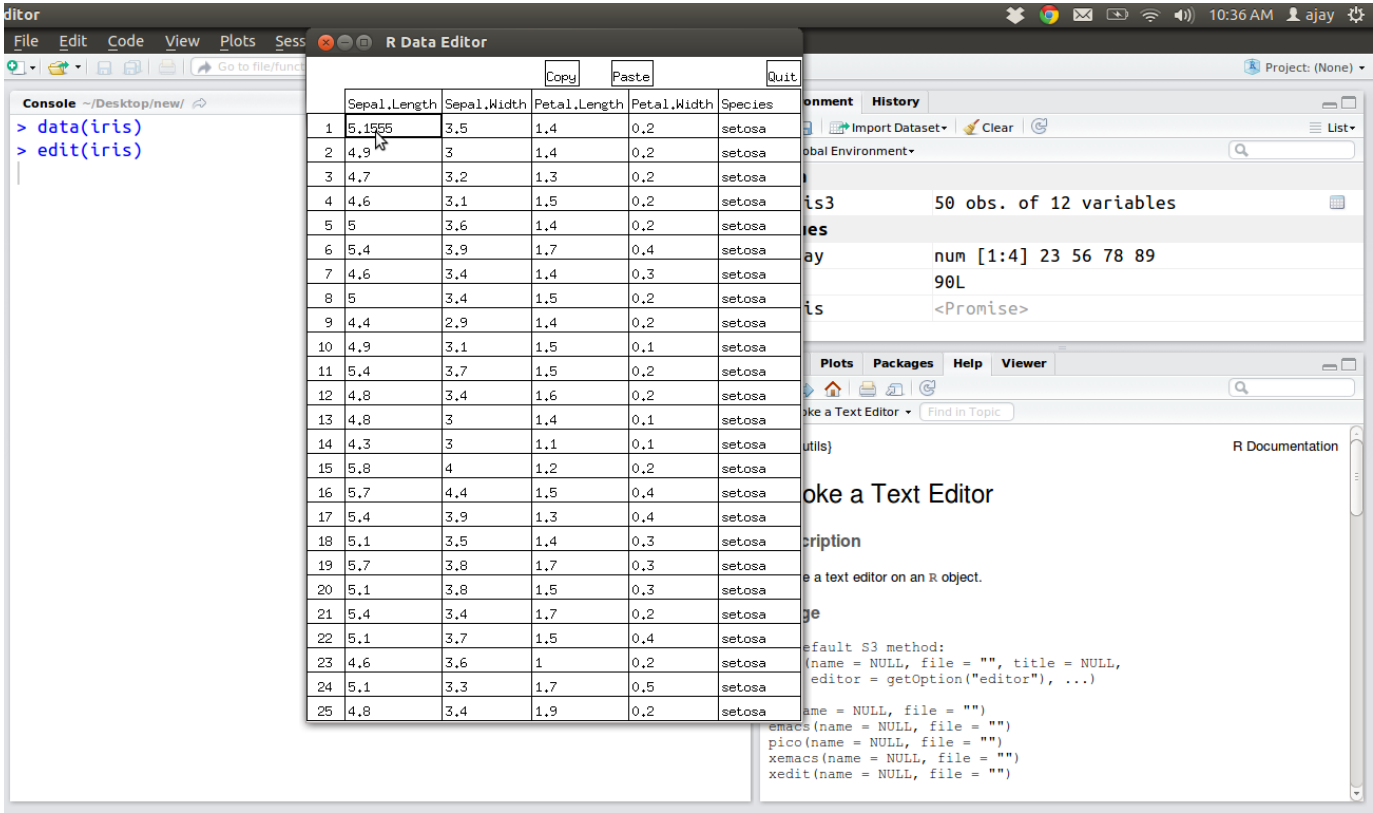
Manual Entry



Manual Editing



Manual Editing



readr from Hadley

The goal of readr is to provide a fast and friendly way to read tabular data into R. The most important functions are:

- Read delimited files: `read_delim()`, `read_csv()`, `read_tsv()`, `read_csv2()`.
- Read fixed width files: `read_fwf()`, `read_table()`.
- Read lines: `read_lines()`.
- Read whole file: `read_file()`.
- Re-parse existing data frame: `type_convert()`.

https://github.com/hadley/readr

readr from Hadley

Source Data - https://bit.ly/dsdata

```
> library(readr)
> system.time(read_csv("BigDiamonds.csv"))
|======================================================================================| 100%   49 MB
   user  system elapsed
  2.396   0.068   2.448
Warning message:
597311 problems parsing 'BigDiamonds.csv'. See problems(...) for more details.
```

readxl from Hadley

Readxl supports both the legacy `.xls` format and the modern xml-based `.xlsx` format. `.xls`support is made possible the with libxls C library, which abstracts away many of the complexities of the underlying binary format. To parse `.xlsx`, we use the RapidXML C++ library.

```
read_excel("my-old-spreadsheet.xls")
read_excel("my-new-spreadsheet.xlsx")

read_excel("my-spreadsheet.xls", sheet = "data")
read_excel("my-spreadsheet.xls", sheet = 2)

read_excel("my-spreadsheet.xls", na = "NA")
```

https://github.com/hadley/readxl

data.table

fread is the fastest way to read data

```
> b=fread("BigDiamonds.csv")
Read 598024 rows and 13 (of 13) columns from 0.049 GB file in 00:00:04
```

data.table

fread is the fastest way to read data

```
> b=fread("BigDiamonds.csv")
Read 598024 rows and 13 (of 13) columns from 0.049 GB file in 00:00:04
```

data.table

fread is the fastest way to read data

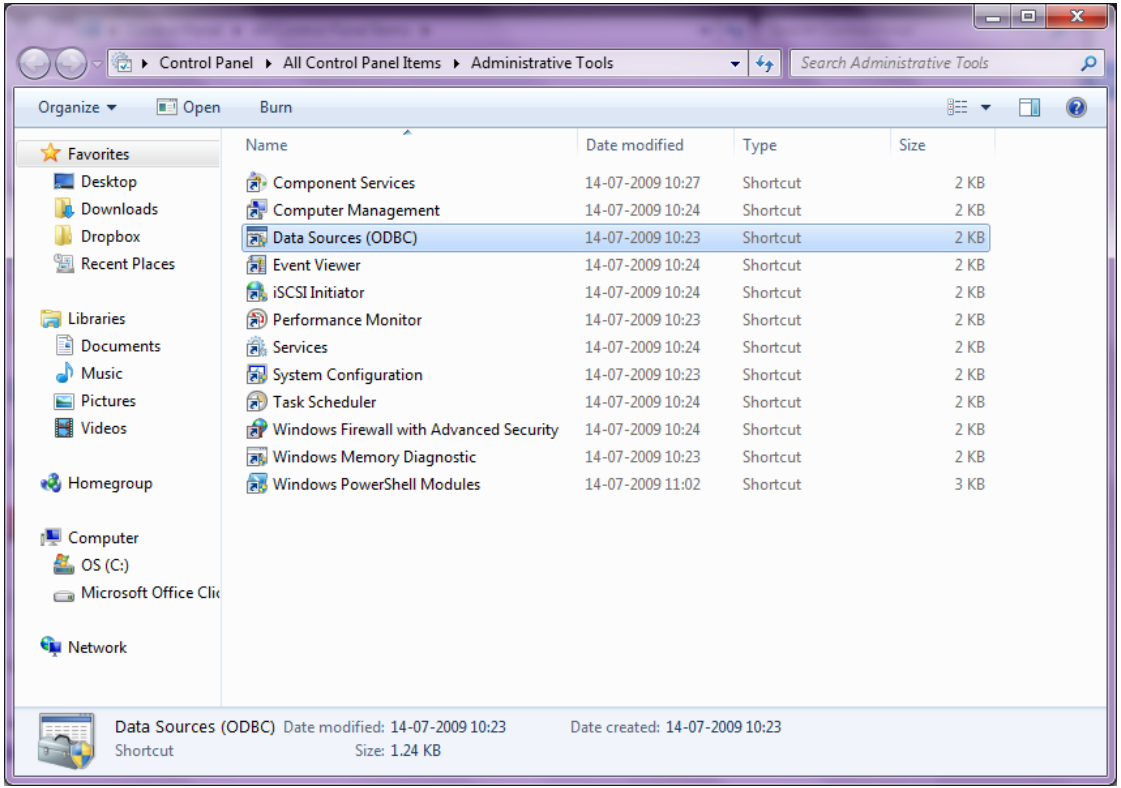
```
> system.time(read_csv("BigDiamonds.csv"))
   user  system elapsed
  2.552   0.028   2.581
Warning message:
597311 problems parsing 'BigDiamonds.csv'. See problems(...) for more details.
> system.time(fread("BigDiamonds.csv"))
   user  system elapsed
  1.532   0.012   1.540
> system.time(read.csv("BigDiamonds.csv"))
   user  system elapsed
 10.892   0.032  10.922
```

Some learnings

1. Multiple packages can do the same thing faster or slower in R
2. Knowing the right package is the essential difference as a data scientist
3. Putting code within system.time() helps measure speed

also see http://adv-r.had.co.nz/Profiling.html for advanced ways to spped up code

Creating DSN



Creating DSN (in Windows)

A Data Source Name (DSN) is the logical name that is used by Open Database Connectivity (ODBC) to refer to the drive and other information that is required to access data. The name is use by Internet Information Services for a connection to an ODBC data source, such as a Microsoft SQL Server database.

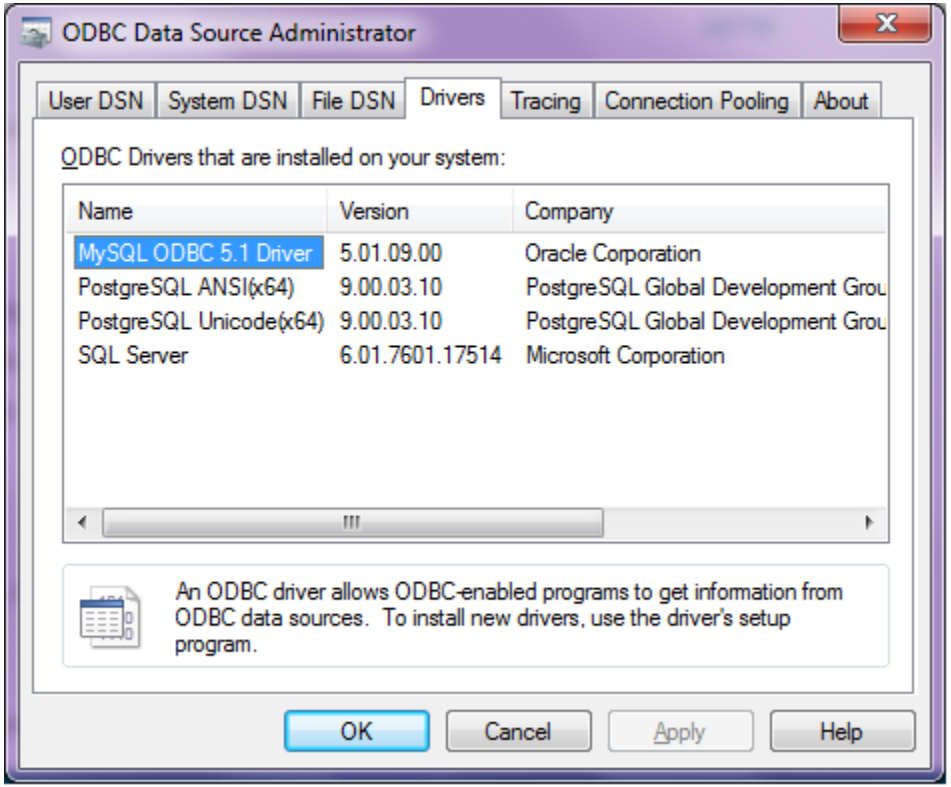
https://support.microsoft.com/en-us/kb/kbview/300596

Creating DSN (in Windows)

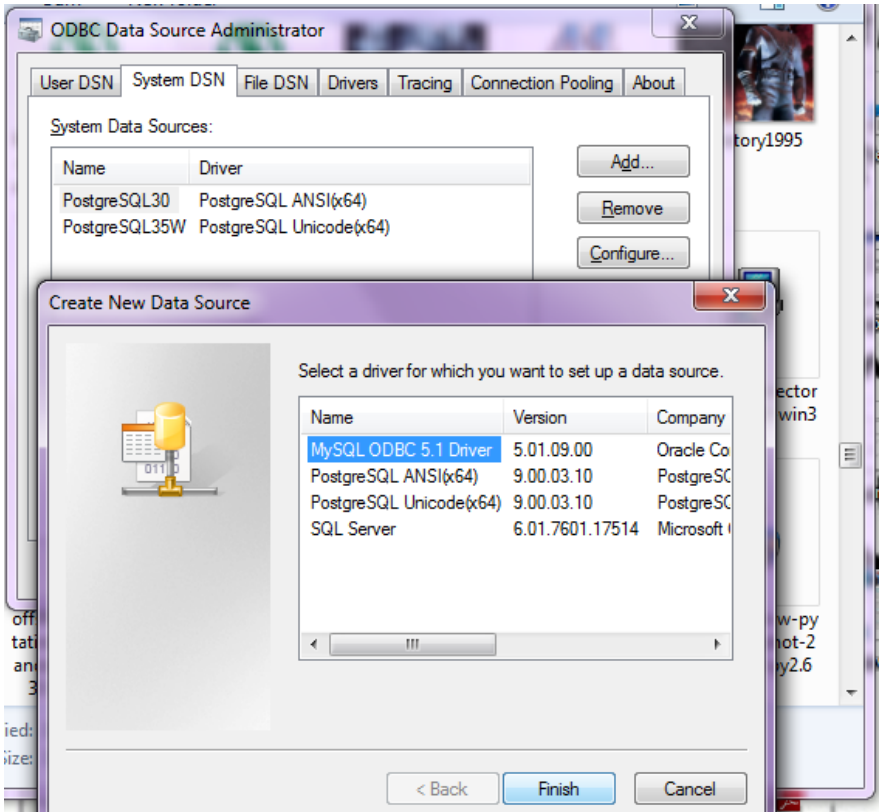
1. Click **Start**, point to **Control Panel**, double-click **Administrative Tools**, and then double-click **Data Sources(ODBC)**.
2. Click the **System DSN** tab, and then click **Add**.
3. Click the database driver that corresponds with the database type to which you are connecting, and then click **Finish**.
4. Type the data source name. Make sure that you choose a name that you can remember. You will need to use this name later.
5. Click **Select**.
6. Click the correct database, and then click **OK**.
7. Click **OK**, and then click **OK**.

https://support.microsoft.com/en-us/kb/kbview/300596

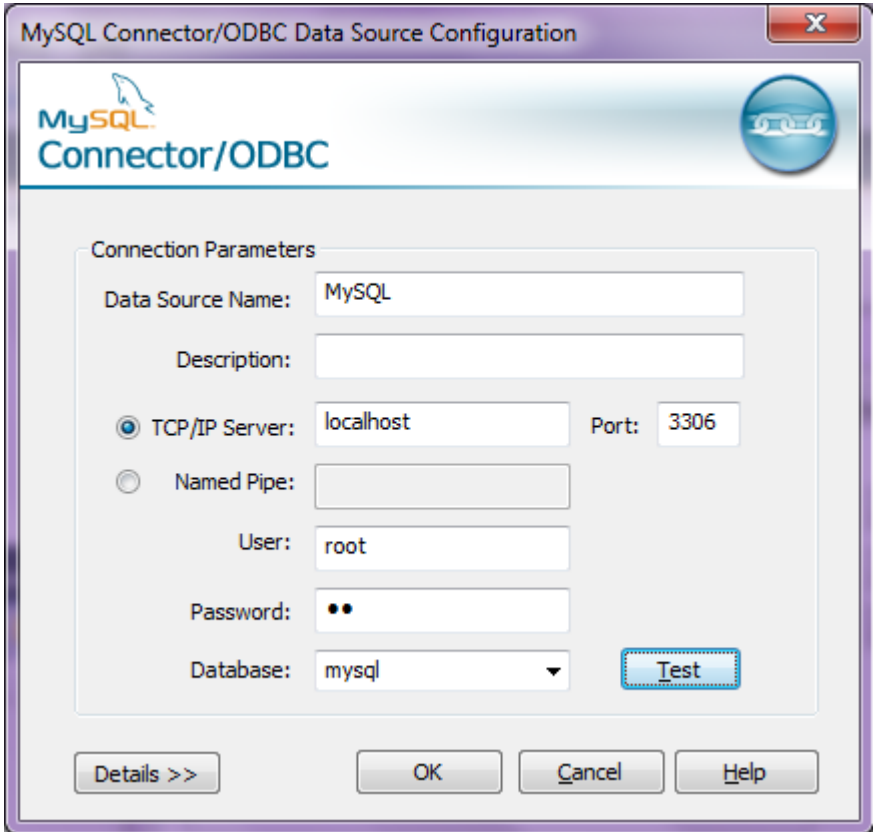
Creating DSN



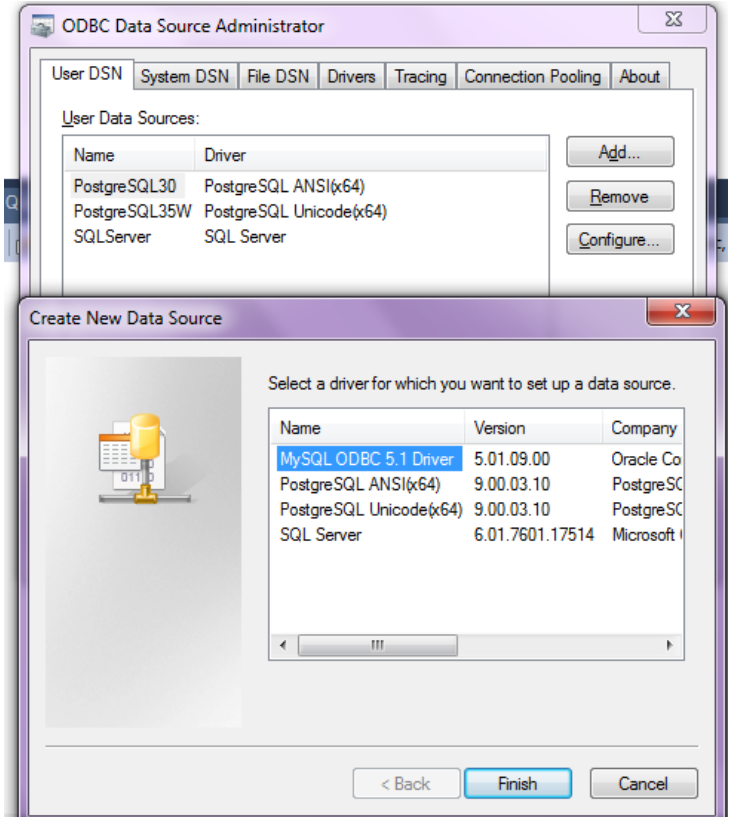
Creating DSN



Creating DSN



Creating DSN



RODBC

```
> library(RODBC)
> odbcDataSources()
> ajay=odbcConnect(“MySQL”,uid=”root”,pwd=”XX”)
> ajay
> sqlTables(ajay)
>tested=sqlFetch(ajay,”host”)
```

From Databases

The **RODBC** package provides access to databases through an **ODBC** interface.

The primary functions are

- **odbcConnect(*dsn*, uid="", pwd="")** Open a connection to an ODBC database
- **sqlFetch(*channel*, *sqltable*)** Read a table from an ODBC database into a data frame

Hint- a good site to revise R http://www.statmethods.net

sqlite

http://cran.r-project.org/web/packages/RSQLite/RSQLite.pdf embeds the SQLite database engine in R and provides an interface compliant with the DBI package.

SQLite is a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine.

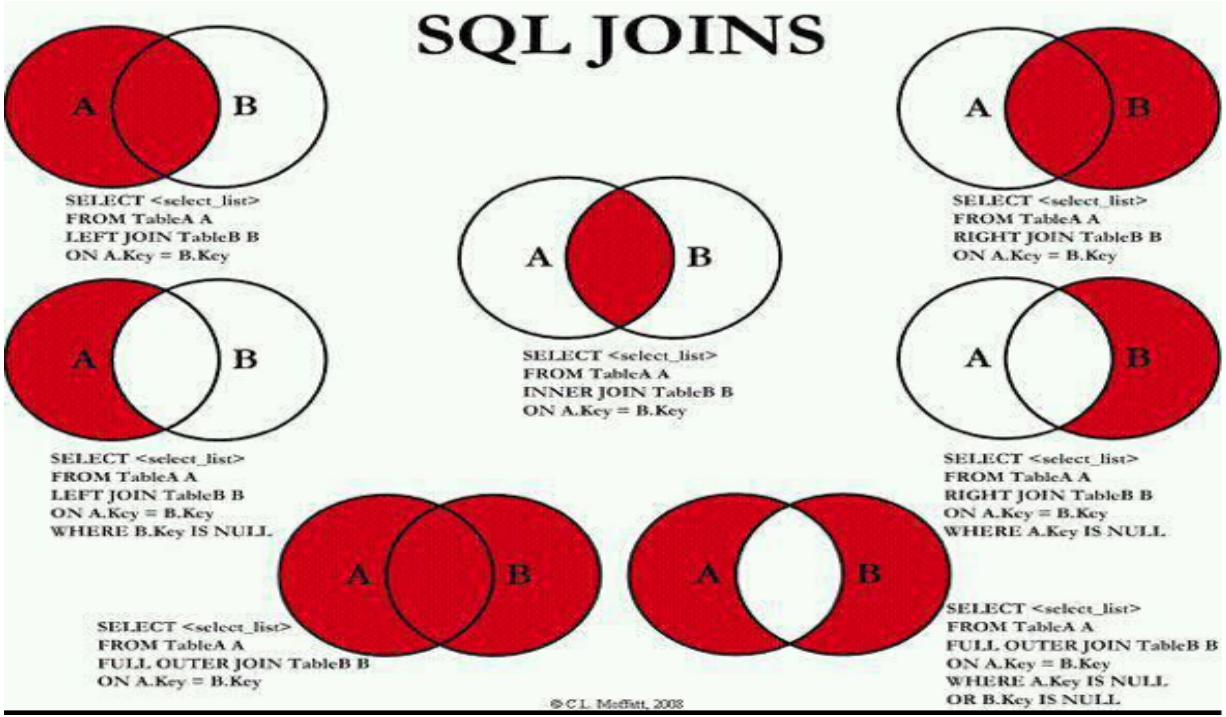
SQLite is the most widely deployed database engine in the world

```
library(RSQLite)
con <- dbConnect("SQLite", dbname = "sample_db")
 # read csv file into sql database
dbWriteTable(con, name="sample_data", value="sample_data.csv", row.names=FALSE, header=TRUE, sep = ",")
```

http://cran.r-project.org/web/packages/sqldf/index.html Manipulate R data frames using SQL

`read.csv.sql` in the `sqldf` package imports data into a temporary SQLite database and then reads it into R.

A Detour to SQL



RMySQL

```
install.packages("RMySQL")
library(RMySQL)

mydb = dbConnect(MySQL(), user='user', password='password', dbname='database_name', host='host')


dbListTables(mydb)

dbListFields(mydb, 'some_table')


dbSendQuery(mydb, 'drop table if exists some_table, some_other_table')

dbWriteTable(mydb, name='table_name', value=data.frame.name)
```

Other databases

Teradata https://github.com/Teradata/teradataR

PostgreSQL http://cran.r-project.org/web/packages/RPostgreSQL/

MongoDB http://cran.r-project.org/web/packages/mongolite/index.html

couchDB http://cran.r-project.org/web/packages/couchDB/index.html

MonetDB http://cran.r-project.org/web/packages/MonetDB.R/index.html

Other data sources

Cassandra with R http://cran.r-project.org/web/packages/RCassandra/RCassandra.pdf

Neo4j with R

http://things-about-r.tumblr.com/post/47392314578/venue-recommendation-a-simple-use-case-connecting-r

R with Hadoop Stack https://github.com/RevolutionAnalytics/RHadoop/wiki

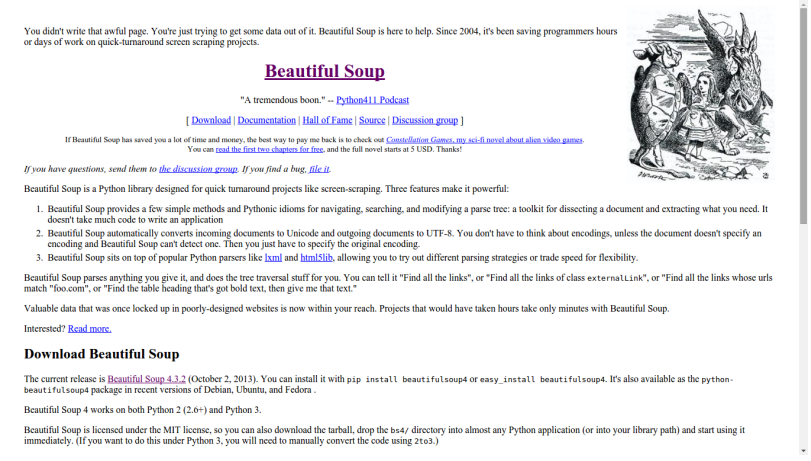
- NEW! ravro - read and write files in avro format
- plyrmr - higher level plyr-like data processing for structured data, powered by `rmr`
- rmr - functions providing Hadoop MapReduce functionality in R
- rhdfs - functions providing file management of the HDFS from within R
- rhbase - functions providing database management for the HBase distributed database from within R

https://amplab-extras.github.io/SparkR-pkg/ SparkR is an R package to use Spark from R.

Web Scraping

Web scraping (**web** harvesting or **web** data extraction) is a computer software technique of extracting information from websites.

example - python (scrapy and beautiful soup)


You didn't write that awful page. You're just trying to get some data out of it. Beautiful Soup is here to help. Since 2004, it's been saving programmers hours or days of work on quick-turnaround screen scraping projects.

Beautiful Soup

"A tremendous boon." -- Python411 Podcast

[Download | Documentation | Hall of Fame | Source | Discussion group]

If Beautiful Soup has saved you a lot of time and money, the best way to pay me back is to check out *Constellation Games*, my sci-fi novel about alien video games. You can read the first two chapters for free, and the full novel starts at 5 USD. Thanks!

If you have questions, send them to the discussion group. If you find a bug, file it.

Beautiful Soup is a Python library designed for quick turnaround projects like screen-scraping. Three features make it powerful:

1. Beautiful Soup provides a few simple methods and Pythonic idioms for navigating, searching, and modifying a parse tree: a toolkit for dissecting a document and extracting what you need. It doesn't take much code to write an application
2. Beautiful Soup automatically converts incoming documents to Unicode and outgoing documents to UTF-8. You don't have to think about encodings, unless the document doesn't specify an encoding and Beautiful Soup can't detect one. Then you just have to specify the original encoding.
3. Beautiful Soup sits on top of popular Python parsers like lxml and html5lib, allowing you to try out different parsing strategies or trade speed for flexibility.

Beautiful Soup parses anything you give it, and does the tree traversal stuff for you. You can tell it "Find all the links", or "Find all the links of class `externalLink`", or "Find all the links whose urls match "foo.com", or "Find the table heading that's got bold text, then give me that text."

Valuable data that was once locked up in poorly-designed websites is now within your reach. Projects that would have taken hours take only minutes with Beautiful Soup.

Interested? Read more.

Download Beautiful Soup

The current release is Beautiful Soup 4.3.2 (October 2, 2013). You can install it with `pip install beautifulsoup4` or `easy_install beautifulsoup4`. It's also available as the `python-beautifulsoup4` package in recent versions of Debian, Ubuntu, and Fedora .

Beautiful Soup 4 works on both Python 2 (2.6+) and Python 3.

Beautiful Soup is licensed under the MIT license, so you can also download the tarball, drop the `bs4/` directory into almost any Python application (or into your library path) and start using it immediately. (If you want to do this under Python 3, you will need to manually convert the code using `2to3`.)

Web Scraping

- readlines

Hint : R is case sensitive
readlines is not the same as readLines

```
> url="http://nytimes.com"
> ajay=readlines(url)
Error: could not find function "readlines"
> ajay=readLines(url)
> head(ajay)
[1] "<!DOCTYPE html>"                                                                                                          $
[2] "<!--[if (gt IE 9)|!(IE)]> <!--> <html lang=\"en\" class=\"no-js edition-domestic app-homepage\"  itemscope xmlns:og$
[3] "<!--[if IE 9]> <html lang=\"en\" class=\"no-js ie9 lt-ie10 edition-domestic app-homepage\" xmlns:og=\"http://opengr$
[4] "<!--[if IE 8]> <html lang=\"en\" class=\"no-js ie8 lt-ie10 lt-ie9 edition-domestic app-homepage\" xmlns:og=\"http:/$
[5] "<!--[if (lt IE 8)]> <html lang=\"en\" class=\"no-js lt-ie10 lt-ie9 lt-ie8 edition-domestic app-homepage\" xmlns:og=$
[6] "<head>"                                                                                                                   $
> tail(ajay)
[1] "<div id=\"ab3\" class=\"ad ab3-ad hidden\"></div>"                                                                        $
[2] "<div id=\"prop1\" class=\"ad prop1-ad hidden\"></div>"                                                                    $
[3] "<div id=\"prop2\" class=\"ad prop2-ad hidden\"></div>"                                                                    $
[4] "<div id=\"Anchor\" class=\"ad anchor-ad hidden\"></div>"                                                                  $
[5] "<script type=\"text/javascript\">window.NREUM||(NREUM={});NREUM.info={\"beacon\":\"beacon-6.newrelic.com\",\"licens$
[6] "</html>"                                                                                                                  $
> 
```

Hint : Use head() and tail() to inspect objects

Other packages are XML and Curl
Case Study- http://decisionstats.com/2013/04/14/using-r-for-cricket-analysis-rstats/

curl

cURL is a computer software project providing a library and command-line tool for transferring data using various protocols. The **cURL** project produces two products, libcurl and **cURL**.

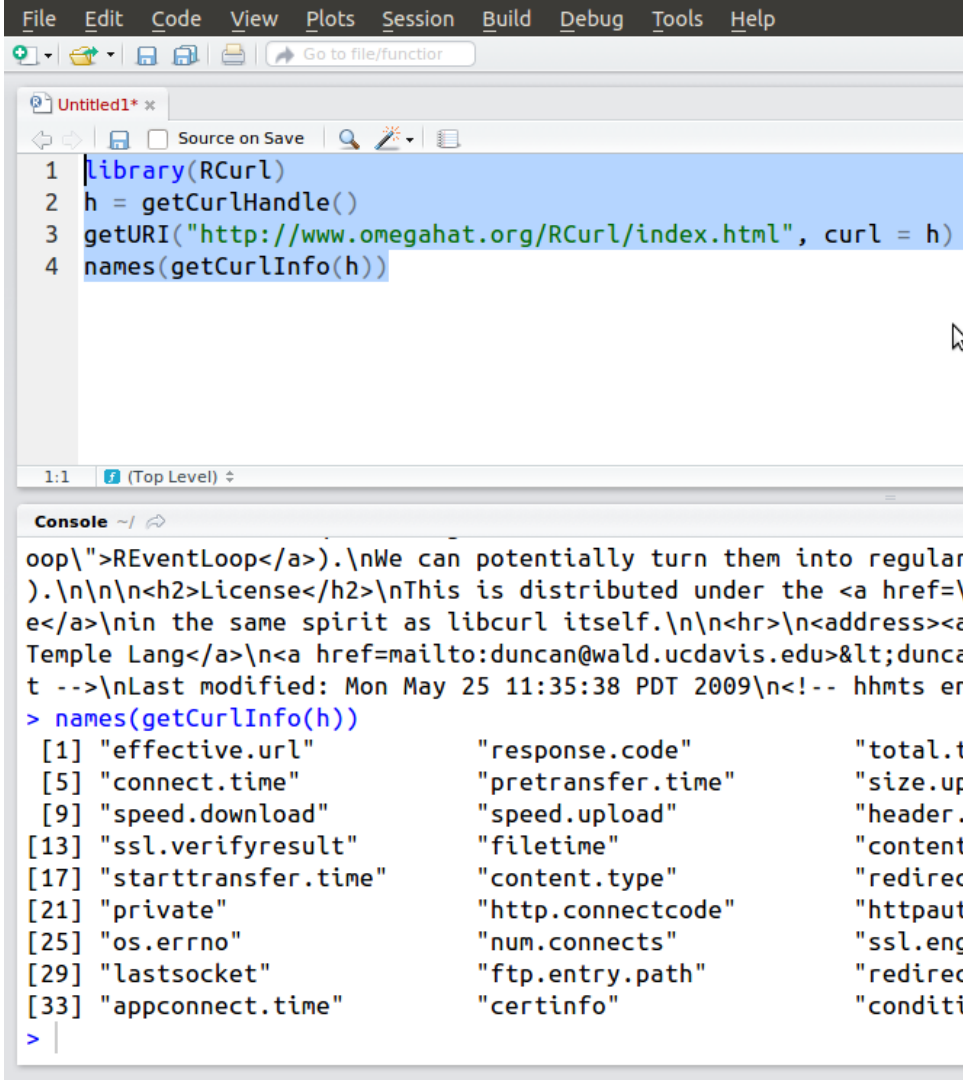
The RCurl package is an R-interface to the libcurl library that provides HTTP facilities. This allows us to download files from Web servers, post forms, use HTTPS (the secure HTTP), use persistent connections, upload files, use binary content, handle redirects, password authentication, etc.

The primary top-level entry points are

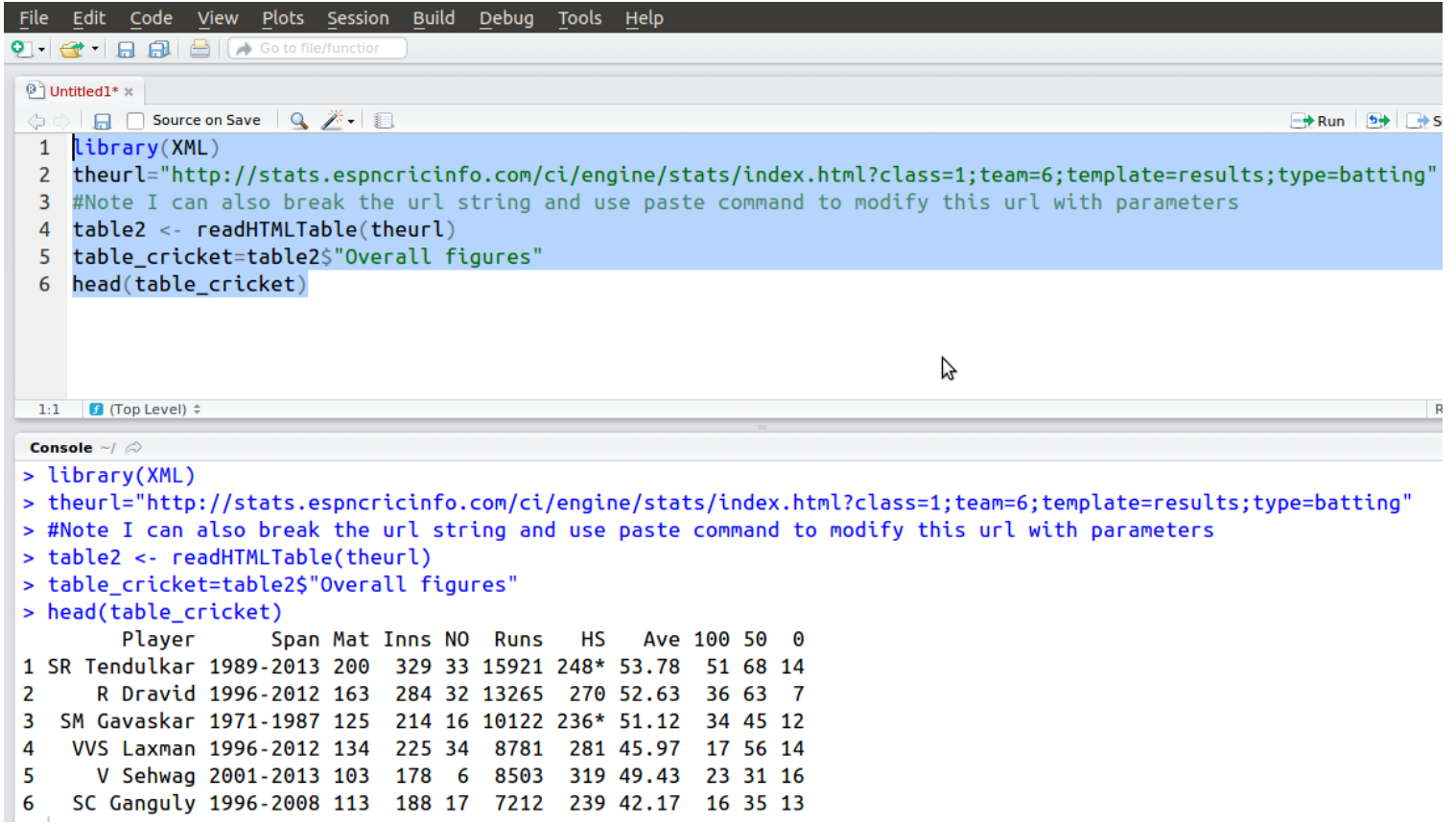
- getURL()
- getURLContent()
- getForm()
- postForm()

http://www.omegahat.org/RCurl/RCurlJSS.pdf

Rcurl


File Edit Code View Plots Session Build Debug Tools Help
Untitled1*
Source on Save
1 library(RCurl)
2 h = getCurlHandle()
3 getURI("http://www.omegahat.org/RCurl/index.html", curl = h)
4 names(getCurlInfo(h))
1:1 (Top Level)
Console ~/
oop\">REventLoop).\nWe can potentially turn them into regular
).\n\n\n<h2>License</h2>\nThis is distributed under the <a href=\
e\nin the same spirit as libcurl itself.\n\n<hr>\n<address><a
Temple Lang\n<a href=mailto:duncan@wald.ucdavis.edu><dunca
t -->\nLast modified: Mon May 25 11:35:38 PDT 2009\n<!-- hhmts en
> names(getCurlInfo(h))
[1] "effective.url" "response.code" "total.t
[5] "connect.time" "pretransfer.time" "size.up
[9] "speed.download" "speed.upload" "header.
[13] "ssl.verifyresult" "filetime" "content
[17] "starttransfer.time" "content.type" "redirec
[21] "private" "http.connectcode" "httpaut
[25] "os.errno" "num.connects" "ssl.eng
[29] "lastsocket" "ftp.entry.path" "redirec
[33] "appconnect.time" "certinfo" "conditi
>

XML


File Edit Code View Plots Session Build Debug Tools Help
Untitled1*
Source on Save
Run
1 library(XML)
2 theurl="http://stats.espncricinfo.com/ci/engine/stats/index.html?class=1;team=6;template=results;type=batting"
3 #Note I can also break the url string and use paste command to modify this url with parameters
4 table2 <- readHTMLTable(theurl)
5 table_cricket=table2$"Overall figures"
6 head(table_cricket)
1:1
(Top Level)
Console ~/
> library(XML)
> theurl="http://stats.espncricinfo.com/ci/engine/stats/index.html?class=1;team=6;template=results;type=batting"
> #Note I can also break the url string and use paste command to modify this url with parameters
> table2 <- readHTMLTable(theurl)
> table_cricket=table2$"Overall figures"
> head(table_cricket)
Player Span Mat Inns NO Runs HS Ave 100 50 0
1 SR Tendulkar 1989-2013 200 329 33 15921 248* 53.78 51 68 14
2 R Dravid 1996-2012 163 284 32 13265 270 52.63 36 63 7
3 SM Gavaskar 1971-1987 125 214 16 10122 236* 51.12 34 45 12
4 VVS Laxman 1996-2012 134 225 34 8781 281 45.97 17 56 14
5 V Sehwag 2001-2013 103 178 6 8503 319 49.43 23 31 16
6 SC Ganguly 1996-2008 113 188 17 7212 239 42.17 16 35 13

a note on piping using magrittr

This stylistic option has several advantages:

1. Reduced requirements of nested parenthesizes
2. Order of functional operations now read from left to right
3. Organizational style of the code may be improved

The library uses a operator %>% which basically tells R to take the value of that which is to the left and pass it to the right as an argument

http://www.econometricsbysimulation.com/2014/07/more-readable-code-with-pipes-in-r.html

```
1 + 8 %>% sqrt
# Returns 3.828427

# Rather than
(1 + 8) %>% sqrt
# [1] 3
```

a note on piping using magrittr

```
let (|>) x f = f x.

some_value |> some_function other_value
```

```
babynames %>%
    filter(name %>% substr(1, 3) %>% equals("Ste")) %>%
    group_by(year, sex) %>%
    summarize(total = sum(n)) %>%
    qplot(year, total, color = sex, data = ., geom = "line") %>%
    add(ggtitle('Names starting with "Ste"')) %>%
    print
```

INTERPRET

1. take the baby data, then
2. filter it such that the name sub-string from character 1 to 3 equals "Ste", then
3. group it by year and sex, then
4. summarize it by computing total sum for each group, then
5. plot the resuls, coloring by sex, then
6. add a title, then
7. print it to the canvas.

rvest by Hadley Wickham

```r
library(rvest)
page=html("http://stats.espncricinfo.com/ci/engine/stats/index.html?class=1;team=6;template=results;type=batting")
data <-
  page %>%
  html_nodes("table") %>%
  .[[3]] %>%
  html_table()
head(data)
```

8:11 (Top Level) R Script

Console R Markdown

```
~/
```

```
        Player      Span Mat Inns NO  Runs   HS   Ave 100 50  0
1 SR Tendulkar 1989-2013 200  329 33 15921 248* 53.78  51 68 14
2     R Dravid 1996-2012 163  284 32 13265  270 52.63  36 63  7
3  SM Gavaskar 1971-1987 125  214 16 10122 236* 51.12  34 45 12
4   VVS Laxman 1996-2012 134  225 34  8781  281 45.97  17 56 14
5     V Sehwag 2001-2013 103  178  6  8503  319 49.43  23 31 16
6   SC Ganguly 1996-2008 113  188 17  7212  239 42.17  16 35 13
```

json format

jsonlite for json data

http://arxiv.org/abs/1403.2805

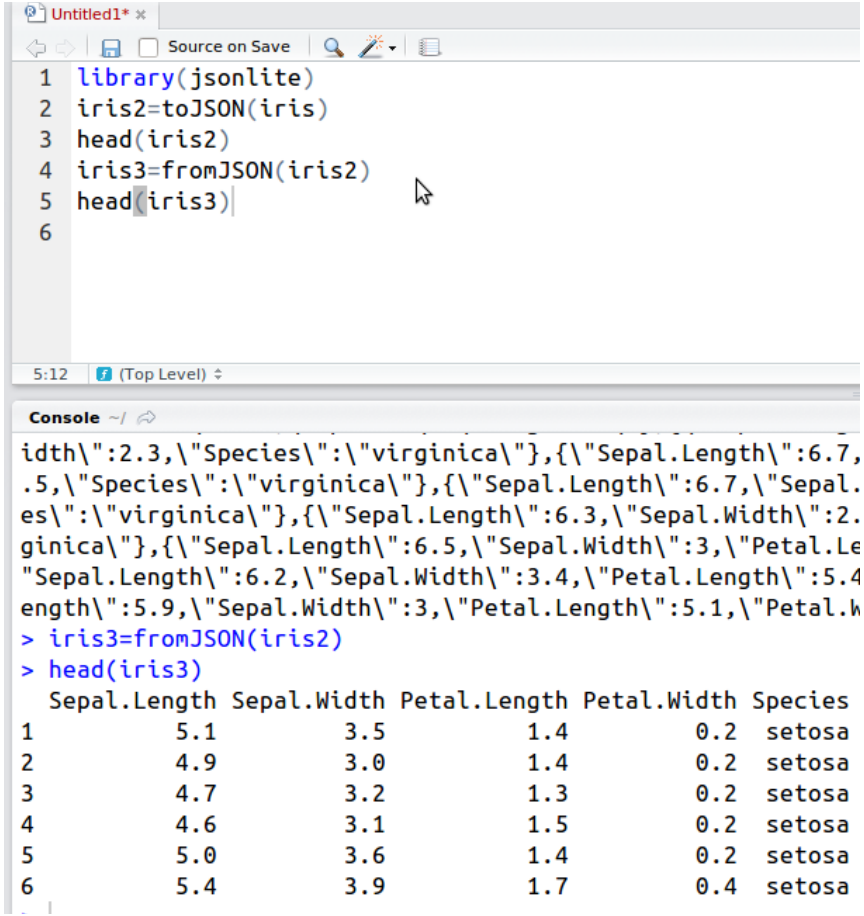
```
> library(jsonlite)

Attaching package: ‘jsonlite’

The following object is masked from ‘package:utils’:

    View

> library(httr)
> library(curl)
> zips <- stream_in(curl("https://media.mongodb.org/zips.json"))
opening curl input connection.
Found 29353 lines...
binding pages together (no custom handler).
closing curl input connection.
>
> head(zips)
    _id        city                  loc   pop state
1 01001      AGAWAM -72.62274, 42.07021 15338    MA
2 01002     CUSHMAN -72.51565, 42.37702 36963    MA
3 01005       BARRE -72.10835, 42.40970  4546    MA
4 01007 BELCHERTOWN -72.41095, 42.27510 10579    MA
5 01008   BLANDFORD -72.93611, 42.18295  1240    MA
6 01010   BRIMFIELD -72.18846, 42.11654  3706    MA
```


```
{ "_id" : "01001", "city" : "AGAWAM", "loc" : [ -72.622739, 42.070206 ], "pop" : 15338, "state" : "MA" }
{ "_id" : "01002", "city" : "CUSHMAN", "loc" : [ -72.51564999999999, 42.377017 ], "pop" : 36963, "state" : "MA" }
{ "_id" : "01005", "city" : "BARRE", "loc" : [ -72.10835400000001, 42.409698 ], "pop" : 4546, "state" : "MA" }
{ "_id" : "01007", "city" : "BELCHERTOWN", "loc" : [ -72.41095300000001, 42.275103 ], "pop" : 10579, "state" : "MA" }
{ "_id" : "01008", "city" : "BLANDFORD", "loc" : [ -72.936114, 42.182949 ], "pop" : 1240, "state" : "MA" }
{ "_id" : "01010", "city" : "BRIMFIELD", "loc" : [ -72.188455, 42.116543 ], "pop" : 3706, "state" : "MA" }
{ "_id" : "01011", "city" : "CHESTER", "loc" : [ -72.988761, 42.279421 ], "pop" : 1688, "state" : "MA" }
{ "_id" : "01012", "city" : "CHESTERFIELD", "loc" : [ -72.833309, 42.38167 ], "pop" : 177, "state" : "MA" }
{ "_id" : "01013", "city" : "CHICOPEE", "loc" : [ -72.607962, 42.162046 ], "pop" : 23396, "state" : "MA" }
{ "_id" : "01020", "city" : "CHICOPEE", "loc" : [ -72.576142, 42.176443 ], "pop" : 31495, "state" : "MA" }
{ "_id" : "01022", "city" : "WESTOVER AFB", "loc" : [ -72.558657, 42.196672 ], "pop" : 1764, "state" : "MA" }
{ "_id" : "01026", "city" : "CUMMINGTON", "loc" : [ -72.905767, 42.435296 ], "pop" : 1484, "state" : "MA" }
{ "_id" : "01027", "city" : "MOUNT TOM", "loc" : [ -72.67992099999999, 42.264319 ], "pop" : 16864, "state" : "MA" }
{ "_id" : "01028", "city" : "EAST LONGMEADOW", "loc" : [ -72.505565, 42.067203 ], "pop" : 13367, "state" : "MA" }
{ "_id" : "01030", "city" : "FEEDING HILLS", "loc" : [ -72.675077, 42.07182 ], "pop" : 11985, "state" : "MA" }
{ "_id" : "01031", "city" : "GILBERTVILLE", "loc" : [ -72.19858499999999, 42.332194 ], "pop" : 2385, "state" : "MA" }
{ "_id" : "01032", "city" : "GOSHEN", "loc" : [ -72.844092, 42.466234 ], "pop" : 122, "state" : "MA" }
```

json format

jsonlite for json data

http://arxiv.org/abs/1403.2805



Using APIs for data

https://ropensci.org/

CRAN Task View: Web Technologies and Services

Maintainer: Scott Chamberlain, Thomas Leeper, Patrick Mair, Karthik Ram, Christopher Gandrud
Contact: scott at ropensci.org
Version: 2015-03-20

This task view contains information about using R to obtain and parse data from the web. The base version of R does not ship with many tools for interacting with the web. Thankfully, there are an increasingly large number of tools for interacting with the web. A list of available packages and functions is presented below, grouped by the type of activity. If you have any comments or suggestions for additions or improvements for this taskview, go to GitHub and submit an issue , or make some changes and submit a pull request . If you can't contribute on GitHub, send Scott an email . If you have an issue with one of the packages discussed below, please contact the maintainer of that package. If you know of a web service, API, data source, or other online resource that is not yet supported by an R package, consider adding it to the package development to do list on GitHub .

Tools for Working with the Web from R

Parsing Data from the Web

- *downloading files* : `download.file()` is in base R and commonly used way to download a file. However, downloading files over HTTPS is not supported in R's internal method for `download.file()`. The `download()` function in the package downloader wraps `download.file()`, and takes all the same arguments, but works for https across platforms.
- *tabular data as txt, csv, etc.* : You can use `read.table()`, `read.csv()`, and friends to read a table directly from a URL, or after acquiring the csv file from the web via e.g., `getURL()` from RCurl. `read.csv()` works with http but not https, i.e.: `read.csv("http://...")`, but not `read.csv("https://...")`. You can download a file first before reading the file in R, and you can use downloader to download over https. `read.table()` and friends also have a `text` parameter so you can read a table if a table is encoded as a string with line breaks, etc.
- *JSON I/O* : JSON is *javascript object notation* . There are three packages for reading and writing JSON: rjson, RJSONIO, and jsonlite. jsonlite includes a different parser from RJSONIO called yajl . We recommend using jsonlite. Check out the paper describing jsonlite by Jeroen Ooms http://arxiv.org/abs/1403.2805 .
- *XML/HTML I/O* : The package XML contains functions for parsing XML and HTML, and supports xpath for searching XML (think regex for strings). A helpful function to read data from one or more HTML tables is `readHTMLTable()`. XML also includes XPATH parsing ability, see `xpathApply()` and `xpathSApply()`. The XML2R package is a collection of convenient functions for coercing XML into data frames (development version on GitHub). An alternative to XML is selectr , which parses CSS3 Selectors and translates them to XPath 1.0 expressions. XML package is often used for parsing xml and html, but selectr translates CSS selectors to XPath, so can use the CSS selectors instead of XPath. The selectorgadget browser extension can be used to identify page elements. RHTMLForms reads HTML documents and obtains a description of each of the forms it contains, along with the different elements and hidden fields. scrapeR provides additional tools for scraping data from HTML and XML documents.
- rvest : rvest scrapes html from web pages, and is designed to work with magrittr to make it easy to express common web scraping tasks.
- The tldextract package extract top level domains and subdomains from a host name. It's a port of a Python library of the same name .
- webutils: Utility functions for developing web applications. Parsers for `application/x-www-form-urlencoded` as well as `multipart/form-data`. Source on Github
- urltools: URL encoding, decoding, parsing, and parameter extraction. Source on Github
- The repmis package contains a `source_data()` command to load and cache plain-text data from a URL (either http or https). It also includes `source_Dropbox()` for downloading/caching plain-text data from non-public Dropbox folders and `source_XlsxData()` for downloading/caching Excel xlsx sheets.
- rsdmx provides tools to read data and metadata documents exchanged through the Statistical Data and Metadata Exchange (SDMX) framework. The package currently focuses on the SDMX XML standard format (SDMX-ML). project website (Github) .

Curl, HTTP, FTP, HTML, XML, SOAP

- RCurl: A low level curl wrapper that allows one to compose general HTTP requests and provides convenient functions to fetch URIs, get/post forms, etc. and process the results returned by the Web server. This provides a great deal of control over the HTTP/FTP connection and the form of the request while providing a higher-level interface than is available just using R socket connections. It also provide tools for Web authentication.

ff package

http://cran.r-project.org/web/packages/ff/index.html

The ff package provides data structures that are stored on disk but behave (almost) as if they were in RAM by transparently mapping only a section (pagesize) in main memory - the effective virtual memory consumption per ff object.

http://cran.r-project.org/web/packages/ffbase/index.html

Basic (statistical) functionality for package ff

Example- http://www.bnosac.be/index.php/blog/22-if-you-are-into-large-data-and-work-a-lot-package-ff

```
> require(ffbase)
> hhp <- read.table.ffdf(file="/home/jan/Work/RForgeBNOSAC/github/RBelgium_HeritageHealthPrize/Data/Claims.
csv", FUN = "read.csv", na.strings = "")
```

Also see http://cran.r-project.org/web/packages/bigmemory/index.html

Create, store, access, and manipulate massive matrices. Matrices are allocated to shared memory and may use memory-mapped files. Packages biganalytics, bigtabulate, synchronicity, and bigalgebra provide advanced functionality

RevoScaleR package

RevoScaleR has its own file format, XDF, which is able to rapidly access data by row or by column and to read some data sequentially. XDF file data is stored in the same binary format used in memory, which eliminates the need for conversion when it is brought into memory.

http://www.revolutionanalytics.com/revolution-r-enterprise-scaler

rhdf5

This R/Bioconductor package provides an interface between HDF5 and R. HDF5's main features are the ability to store and access very large and/or complex datasets and a wide variety of metadata on mass storage (disk) through a completely portable file format.

http://www.bioconductor.org/packages/release/bioc/html/rhdf5.html

HDF5 is a data model, library, and file format for storing and managing data. It supports an unlimited variety of datatypes, and is designed for flexible and efficient I/O and for high volume and complex data. HDF5 is portable and is extensible, allowing applications to evolve in their use of HDF5.

https://www.hdfgroup.org/HDF5/

HDF5 simplifies the file structure to include only two major types of object:

- Datasets, which are multidimensional arrays of a homogeneous type
- Groups, which are container structures which can hold datasets and other groups

rhdf5

platforms all | downloads top 5% | posts 10 / 2 / 0.9 / 1 | in Bioc > 10 years
build ok | commits 0.83 | test coverage unknown

HDF5 interface to R

Bioconductor version: Release (3.1)

This R/Bioconductor package provides an interface between HDF5 and R. HDF5's main features are the ability to store and access very large and/or complex datasets and a wide variety of metadata on mass storage (disk) through a completely portable file format. The rhdf5 package is thus suited for the exchange of large and/or complex datasets between R and other software package, and for letting R applications work on datasets that are larger than the available RAM.

Author: Bernd Fischer, Gregoire Pau

Maintainer: Bernd Fischer <b.fischer at dkfz.de>

Citation (from within R, enter `citation("rhdf5")`):

Fischer B and Pau G. *rhdf5: HDF5 interface to R*. R package version 2.12.0.

Installation

To install this package, start R and enter:

```
source("http://bioconductor.org/biocLite.R")
biocLite("rhdf5")
```

Documentation

To view documentation for the version of this package installed in your system, start R and enter:

```
browseVignettes("rhdf5")
```

PDF	R Script	rhdf5
PDF		Reference Manual
Text		NEWS

Documentation »

Bioconductor

- Package vignettes and manuals.
- Workflows for learning and use.
- Course and conference material.
- Videos.
- Community resources and tutorials.

R / CRAN packages and documentation

Support »

Please read the posting guide. Post questions about Bioconductor to one of the following locations:

- Support site - for questions about Bioconductor packages
- Bioc-devel mailing list - for package developers

Functions Used in this lesson

- toJSON and fromJSON

Packages

- data.table
- jsonlite
- rvest

Citations and References

M Dowle, T Short, S Lianoglou, A Srinivasan with contributions from R Saporta and E Antonyan (2014) data.table: Extension of data.frame. R package version 1.9.4. http://CRAN.R-project.org/package=data.table

Jeroen Ooms (2014). The jsonlite Package: A Practical and Consistent Mapping Between JSON Data and R Objects. arXiv:1403.2805 [stat.CO] URL http://arxiv.org/abs/1403.2805

Hadley Wickham (2015). rvest: Easily Harvest (Scrape) Web Pages. R package version 0.2.0. http://CRAN.R-project.org/package=rvest