Data Storage

STAT 133

Gaston Sanchez

Department of Statistics, UC–Berkeley

gastonsanchez.com
github.com/gastonstat
Course web: gastonsanchez.com/teaching/stat133

Data Storage

Awesome Resource

Introduction to Data Technologies

- **ItDT** by Paul Murrell
- Basic introduction to a number of computer technologies
- e.g. HTML, XML, Databases, SQL, regular expressions
- Book's website:
  https://www.stat.auckland.ac.nz/~paul/ItDT/
- PDF version:
  https://www.stat.auckland.ac.nz/~paul/ItDT/itdt-2013-03-26.pdf
- Paul's website
  https://www.stat.auckland.ac.nz/~paul/

From ItDT

In a book on data analysis, we would assume that the data are already in a form that can be conveniently loaded into statistical software, and the emphasis would be on how to analyze these data. However, that is not the focus of this book. Here, we are interested in all of the steps that must be taken before the data can be conveniently loaded into statistical software.

Paul Murrell

From ItDT

As anyone who has worked with data knows, it often takes more time and effort to get the data ready than it takes to perform the data analysis. And yet there are many more books on how to analyze data than there are on how to prepare data for analysis. This book aims to redress that balance.

Paul Murrell

Motivation

Various formats

- We don't usually have control over the format in which data is given to us
- Consider the **IBTrACS Data**

  https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-data
- IBTrACS available formats: ATCF, CSV, cXML, HURDAT, netCDF, WMO

Case Study

IBTrACS Data

IBTrACS

- International Best Track Archive for Climate Stewardship
- https://www.ncdc.noaa.gov/ibtracs/index.php
- Tropical cyclone best track data
- Aims at understanding the distribution, frequency, and intensity of tropical cyclones worldwide

Why use IBTrACS?

- Contains the most complete global set of historical tropical cyclones available
- Combines information from numerous tropical cyclone datasets
- Simplifies inter-agency comparisons by providing storm data from multiple sources in one place
- Provides data in popular formats to facilitate analysis
- Checks the quality of storm inventories, positions, pressures, and wind speeds, passing the information on to the user

IBTrACS-WMO Data

https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-data

Data files are available in the following ways:

- **By storm**: each storm is in a separate file.
- **All storms**: all IBTrACS storms in one file.
- **Storms by basin**: all IBTrACS storms for a particular basin (all years).
- **Storms by year**: all IBTrACS storms for a particular year (all basins).
- **Storms by hemisphere**: all IBTrACS storms for a hemisphere.

IBTrACS-WMO Formats

https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-data

Format	Data File Access		
ATCF	All storms (2 MB)	Storms by basin	Storms by year
CSV	All storms (2 MB)	Storms by basin	Storms by year
cXML	All storms (2 MB)	Storms by basin	Storms by year
HURDAT format	All storms (1 MB)	Storms by basin	Storms by hemisphere NH - SH
netCDF	All storms (2 MB)	Storms by basin	Storms by year
WMO	All storms (2 MB)	Storms by basin	Storms by year

IBTrACS-WMO Data Format Samples

Data Format Samples:

- https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-samples#atcf
- https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-samples#csv
- https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-samples#cxml
- https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-samples#netcdf
- https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-samples#hurdat
- https://www.ncdc.noaa.gov/ibtracs/index.php?name=wmo-samples#wmo

IBTrACS Formats

HURDAT / NOAA Tape Format / TD9697 format

Data from IBTrACS file:*Basin.WP.ibtracs.v03r06.tape* which includes the corection to report 10-min winds.

```
00001 07/18/2006 M= 5  1 SNBR=   1 BERYL       XING=0 SSS=0
00002 07/18*0000000   0    0*0000000   0    0*3232867  30 1010*3302867  35 1008*
00003 07/19*3382865  35 1006*3452863  35 1005*3522864  40 1004*3592865  50 1003*
00004 07/20*3662868  50 1002*3742868  50 1001*3832870  50 1002*3912875  45 1002*
00005 07/21*3982882  45 1003*4102895  45 1000*4242916  40 1002E4382937  35 1000*
00006 07/22E4552967  35 1002E4723000  35 1003E4853035  30 1004*0000000   0    0*
00007 TS SRC=atcf  S/N=2006200N32287
```

ATCF

Data from IBTrACS file: *Year.1969.ibtracs.v03r06.atcf*

```
NA,0,2006071812,00,0,32.30N, 73.30W, 30.0,1010.0,TS
NA,0,2006071818,00,0,33.00N, 73.30W, 35.0,1008.0,TS
NA,0,2006071900,00,0,33.80N, 73.50W, 35.0,1006.0,TS
NA,0,2006071906,00,0,34.50N, 73.70W, 35.0,1005.0,TS
NA,0,2006071912,00,0,35.20N, 73.60W, 40.0,1004.0,TS
NA,0,2006071918,00,0,35.90N, 73.50W, 50.0,1003.0,TS
NA,0,2006072000,00,0,36.60N, 73.20W, 50.0,1002.0,TS
NA,0,2006072006,00,0,37.40N, 73.20W, 50.0,1001.0,TS
NA,0,2006072012,00,0,38.30N, 73.00W, 50.0,1002.0,TS
NA,0,2006072018,00,0,39.10N, 72.50W, 45.0,1002.0,TS
NA,0,2006072100,00,0,39.80N, 71.80W, 45.0,1003.0,TS
NA,0,2006072106,00,0,41.00N, 70.50W, 45.0,1000.0,TS
NA,0,2006072106,45,0,41.30N, 70.10W, 45.0,1000.0,TS
NA,0,2006072112,00,0,42.40N, 68.40W, 40.0,1002.0,TS
NA,0,2006072118,00,0,43.80N, 66.30W, 35.0,1000.0,ET
NA,0,2006072200,00,0,45.50N, 63.30W, 35.0,1002.0,ET
NA,0,2006072206,00,0,47.20N, 60.00W, 35.0,1003.0,ET
NA,0,2006072212,00,0,48.50N, 56.50W, 30.0,1004.0,ET
```

IBTrACS Formats

cXML

The following is from *Storm.1969287N11150.ibtracs_wmo.v03r06.cxml*.

```
<?xml version="1.0" encoding="UTF-8"?>
<cxml  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:noNamespaceSchemaLocation="http://www.bom.gov.au/bmrc/projects/THORPEX/CXML/cxml.1.1.xsd">
  <header>
    <product>IBTrACS</product>
    <generatingApplication>
      <applicationType>IBTrACS Merged Analysis</applicationType>
    </generatingApplication>
    <productionCenter>NCDC</productionCenter>
    <moreInfo>http:/www.ncdc.noaa.gov/wdc/ibtracs/</moreInfo>
    <creationTime>2014-08-21T13:48:31</creationTime>
    <missing>-999</missing>
  </header>
  <data type="analysis">
    <disturbance ID="2006071812_323N_733W">
      <cycloneName>BERYL</cycloneName>
      <fix>
        <validTime>2006-07-18T12:00:00</validTime>
        <latitude  units="deg N"> 32.30</latitude>
        <longitude units="deg E"> -73.30</longitude>
        <cycloneData>
          <minimumPressure>
            <pressure units="hPa">1010.0</pressure>
          </minimumPressure>
          <maximumWind>
            <speed units="kt"> 30.0</speed>
            <averagingPeriod units="min">1</averagingPeriod>
          </maximumWind>
        </cycloneData>
      </fix>
```

IBTrACS Formats

WMO - WMO Standard format

Data from IBTrACS file: *Year.1969.ibtracs.v03r06.wmo*

```
03 NA2006BERYL     2006 718121323  1 733  99999 30101999991010999999999999999
03 NA2006BERYL     2006 718181330  1 733  99999 35101999991008999999999999999
03 NA2006BERYL     2006 719 01338  1 735  99999 35101999991006999999999999999
03 NA2006BERYL     2006 719 61345  1 737  99999 35101999991005999999999999999
03 NA2006BERYL     2006 719121352  1 736  99999 40101999991004999999999999999
03 NA2006BERYL     2006 719181359  1 735  99999 50101999991003999999999999999
03 NA2006BERYL     2006 720 01366  1 732  99999 50101999991002999999999999999
03 NA2006BERYL     2006 720 61374  1 732  99999 50101999991001999999999999999
03 NA2006BERYL     2006 720121383  1 730  99999 50101999991002999999999999999
03 NA2006BERYL     2006 720181391  1 725  99999 45101999991002999999999999999
03 NA2006BERYL     2006 721 01398  1 718  99999 45101999991003999999999999999
03 NA2006BERYL     2006 721 61410  1 705  99999 45101999991000999999999999999
03 NA2006BERYL     2006 721 61413  1 701  99999 45101999991000999999999999999
03 NA2006BERYL     2006 721121424  1 684  99999 40101999991002999999999999999
03 NA2006BERYL     2006 721181438  1 663  99999 35101999991000999999999999999
03 NA2006BERYL     2006 722 01455  1 633  99999 35101999991002999999999999999
03 NA2006BERYL     2006 722 61472  1 600  99999 35101999991003999999999999999
03 NA2006BERYL     2006 722121485  1 565  99999 30101999991004999999999999999
```

Description

ftp://eclipse.ncdc.noaa.gov/pub/ibtracs/documents/Best-Track_WMO-format_Revised2002.pdf

WMO Format Description

Position	Content
1- 9	Cyclone identification code composed by 2 digit numbers in order within the cyclone season, area code and year code. 01SWI2000 shows the 1st system observed in South-West Indian Ocean basin during the 2000/2001 season. Area codes are as follows: ARB = Arabian Sea ATL = Atlantic Ocean AUB = Australian Region (Brisbane) AUD = Australian Region (Darwin) AUP = Australian Region (Perth) BOB = Bay of Bengal CNP = Central North Pacific Ocean ENP = Eastern North Pacific Ocean ZEA = New Zealand Region SWI = South-West Indian Ocean SWP = South-West Pacific Ocean WNP = Western North Pacific Ocean and South China Sea
10-19	Storm Name
20-23	Year
24-25	Month (01-12)
26-27	Day (01-31)
28-29	Hour- universal time (at least every 6 hourly position -00Z,06Z,12Z and 18Z)
	Latitude indicator: 1=North latitude; 2=South latitude
31-33	Latitude (degrees and tenths)
34-35	Check sum (sum of all digits in the latitude)
36	Longitude indicator: 1=West longitude; 2=East longitude
37-40	Longitude (degrees and tenths)

check pdf to see complete metadata

Storm Tracker

CSV data

```
2012296N14283,2012,18, NA, MM,SANDY,2012-10-21 18:00:00, DS, 14.30, -77.40, 25.0, 1006.0,atcf,   4.767,  26.983,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-22 00:00:00, DS, 13.90, -77.80, 25.0, 1005.0,atcf,   4.767,  32.120,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-22 06:00:00, DS, 13.50, -78.20, 25.0, 1003.0,atcf,   4.767,  41.712,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-22 12:00:00, TS, 13.10, -78.60, 30.0, 1002.0,atcf,  17.256,  45.083,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-22 18:00:00, TS, 12.70, -78.70, 35.0, 1000.0,atcf,  32.416,  50.576,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-23 00:00:00, TS, 12.60, -78.40, 40.0,  998.0,atcf,  42.517,  57.193,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-23 06:00:00, TS, 12.90, -78.10, 40.0,  998.0,atcf,  42.517,  57.193,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-23 12:00:00, TS, 13.40, -77.90, 40.0,  995.0,atcf,  42.517,  62.912,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-23 18:00:00, TS, 14.00, -77.60, 45.0,  993.0,atcf,  50.140,  67.740,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-24 00:00:00, TS, 14.70, -77.30, 55.0,  990.0,atcf,  64.208,  71.051,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-24 06:00:00, TS, 15.60, -77.10, 60.0,  987.0,atcf,  69.630,  74.924,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-24 12:00:00, TS, 16.60, -76.90, 65.0,  981.0,atcf,  73.788,  81.333,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-24 18:00:00, TS, 17.70, -76.70, 75.0,  972.0,atcf,  81.938,  87.720,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-24 19:00:00, TS, 17.90, -76.60, 75.0,  971.0,atcf,  81.938,  88.215,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-25 00:00:00, TS, 18.90, -76.40, 85.0,  964.0,atcf,  87.354,  91.835,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-25 05:25:00, TS, 20.00, -76.00,100.0,  954.0,atcf,  92.955,  95.040,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-25 06:00:00, TS, 20.10, -76.00,100.0,  954.0,atcf,  92.955,  95.040,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-25 09:00:00, TS, 20.90, -75.70, 95.0,  960.0,atcf,  91.721,  92.945,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-25 12:00:00, TS, 21.70, -75.50, 95.0,  966.0,atcf,  91.721,  90.872,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-25 18:00:00, TS, 23.30, -75.30, 90.0,  963.0,atcf,  89.071,  92.111,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-26 00:00:00, TS, 24.80, -75.90, 75.0,  965.0,atcf,  81.938,  91.220,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-26 06:00:00, TS, 25.70, -76.40, 70.0,  968.0,atcf,  78.805,  90.148,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-26 12:00:00, TS, 26.40, -76.90, 65.0,  970.0,atcf,  73.788,  88.562,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-26 18:00:00, TS, 27.00, -77.20, 65.0,  971.0,atcf,  73.788,  88.215,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-27 00:00:00, TS, 27.50, -77.10, 60.0,  969.0,atcf,  69.630,  89.887,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-27 06:00:00, TS, 28.10, -76.90, 60.0,  968.0,atcf,  69.630,  90.148,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-27 12:00:00, TS, 28.80, -76.50, 70.0,  956.0,atcf,  78.805,  94.455,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-27 18:00:00, TS, 29.70, -75.60, 70.0,  960.0,atcf,  78.805,  92.945,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-28 00:00:00, TS, 30.50, -74.70, 65.0,  960.0,atcf,  73.788,  92.945,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-28 06:00:00, TS, 31.30, -73.90, 65.0,  959.0,atcf,  73.788,  93.787,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-28 12:00:00, TS, 32.00, -73.00, 65.0,  954.0,atcf,  73.788,  95.040,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-28 18:00:00, TS, 32.80, -72.00, 65.0,  952.0,atcf,  73.788,  95.432,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-29 00:00:00, TS, 33.90, -71.00, 70.0,  950.0,atcf,  78.805,  95.874,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-29 06:00:00, TS, 35.30, -70.50, 80.0,  947.0,atcf,  85.083,  96.939,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-29 12:00:00, TS, 36.90, -71.00, 85.0,  945.0,atcf,  87.354,  97.282,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-29 18:00:00, TS, 38.30, -73.20, 80.0,  940.0,atcf,  85.083,  98.086,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-29 21:00:00, ET, 38.80, -74.00, 75.0,  943.0,atcf,  81.938,  97.660,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-29 23:30:00, ET, 39.40, -74.40, 70.0,  945.0,atcf,  78.805,  97.282,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-30 00:00:00, ET, 39.50, -74.50, 70.0,  946.0,atcf,  78.805,  97.078,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-30 06:00:00, ET, 39.90, -76.20, 55.0,  960.0,atcf,  64.208,  92.945,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-30 12:00:00, ET, 40.10, -77.80, 50.0,  978.0,atcf,  58.394,  84.164,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-30 18:00:00, ET, 40.40, -78.90, 40.0,  986.0,atcf,  42.517,  77.007,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-31 00:00:00, ET, 40.70, -79.80, 35.0,  992.0,atcf,  32.416,  68.884,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-31 06:00:00, ET, 41.10, -80.30, 35.0,  993.0,atcf,  32.416,  67.740,main
2012296N14283,2012,18, NA, MM,SANDY,2012-10-31 12:00:00, ET, 41.50, -80.70, 30.0,  995.0,atcf,  17.256,  62.912,main
```

Some variables in CSV - IBTrACS

Variables	Variables
Serial_Num	Wind(WMO)
Season	Pres(WMO)
Num	Center
Basin	Wind(WMO) Percentile
Sub_basin	Pres(WMO) Percentile
Name	Track_type N/A
ISO_time	Year
Nature	#
Latitude	BB
Longitude	YYYY-MM-DD HH:MM:SS

IBTrACS Formats

- We're able to read and see most IBTrACS data files
- IBTrACS formats are plain text formats
- We can look at the sample formats
- We can read the metadata of each format

Motivation

Various formats

- There is no overall best storage format
- The correct choice will depend on:
  - the size and complexity of the data set
  - what the data set will be used for
- Every storage format has its strengths and weaknesses

Reading Files from the Web

Motivation

- We've talked before about importing data tables
- e.g. field-delimited files
- But not all data come in the form of a table
- We'll see other ways in which we can import data in R

Motivation

We'll cover a variety of situations you most likely will find yourself dealing with:

- reading raw (plain) text
- reading tabular (spreadsheet-like) data
- reading structured data (xml, html) as text

Keep in mind

All the material described in this presentation relies on 3 key assumptions:

- we know where the data is located
- we know how the data is stored (i.e. type of file)
- all we want is to import the data in R

Basics First

R is equipped with a set of handy functions that allow us to read a wide range of data files

The trick to use those functions depends on the format of the data we want to read, and the way R handles the imported values:

- what type of objets (e.g. `vector`, `list`, `data.frame`)
- what kind of modes (e.g. `character`, `numeric`, `factor`)

Basics First

Fundamentals

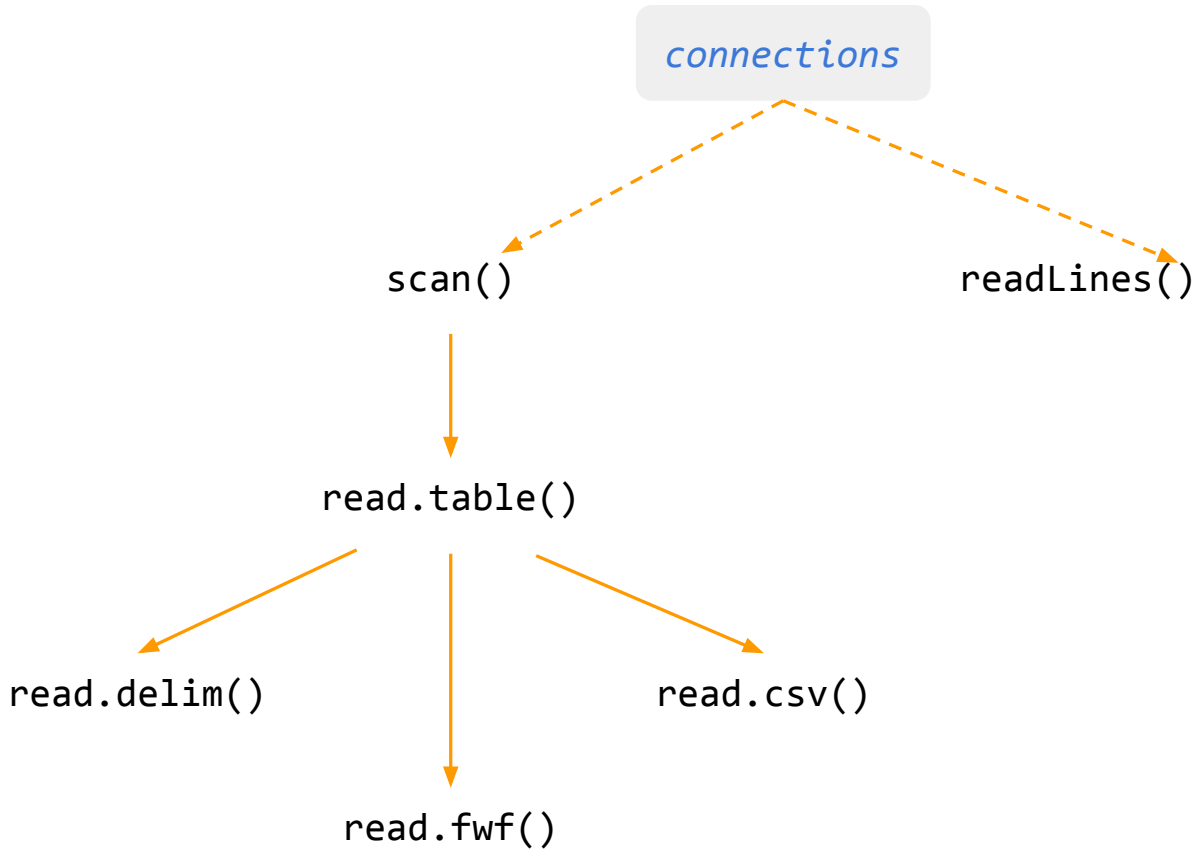
Let's start with the basic reading functions and some R technicalities

- `scan()`
- `readLines()`
- `connections`

Built-in reading functions

- The primary functions to read files in R are `scan()` and `readLines()`
- `readLines()` is the workhorse function to read raw text in R as character strings
- `scan()` is a low-level function for reading data values, and it is extended by `read.table()` and its related functions
- When reading files, there's the special concept under the hood called R `connections`
- Both `scan()` and `readLines()` take a `connection` as input

R Reading Files



Connections

R connections?

Connection is the term used in R to define a mechanism for handling input (reading) and output (writing) operations.

What do they do?

A **connection** is just an object that tells R to be prepared for opening a data source (eg file in local directory, or a file url)
See the full help documentation with: `?connections`

Connections (con't)

Usefulness

Connections provide a **means to have more control** over the way R will "comunicate" with the resources to be read (or written).

Keep in mind

Most of the times you don't need to use or worry about connections. However, you should know that they can play an important role behind the built-in fuctions to read data in R.

About Connections

Should we care?

- Again, most of the times we don't need to explicitly use `url()`
- Connections can be used anywhere a file name could be passed to functions like `scan()` or `read.table()`.
- Usually, the reading functions —eg `readLines()`, `read.table()`, `read.csv()`— will take care of the URL connection for us.
- However, there may be occassions in which we will need to specify a `url()` connection.

Reading Text

Reading Text Files As Text

In this section we'll talk about reading text files with the purpose of importing their contents as *raw text* (ie character strings) in R.

About Text Files

Some considerations so we can all be on the same page:

- By **text files** we mean *plain text files*
- *Plain text* as an umbrella term for any file that is in a human-readable form (e.g. `.txt`, `.csv`, `.xml`, `.html`)
- Text files stored as a sequence of characters
- Each character stored as a single byte of data
- Data is arranged in rows, with several values stored on each row
- Text files that can be read and manipulated with a text editor

Reading Text Functions

Functions for reading text

- `readLines()` is the main function to read text files as *raw text* in R
- `scan()` is another function that can be used to read text files. It is more generic and low-level but we can specify some of its parameters to import content as text

About `readLines()`

Function `readLines()`

- `readLines()` is the workhorse function to read text files as *raw text* in R
- The main input is the file to be read, either specified with a `connection` or with the file name
- `readLines()` treats each line as a string, and it returns a character vector
- The output vector will contain as many elements as number of lines in the read file

readLines()

Using `readLines()`

```
readLines(con = stdin(), n = -1L, ok = TRUE,
          warn = TRUE, encoding = "unknown")
```

- `con` a connection, which in our case will be a complete URL
- `n` the number of lines to read
- `ok` whether to reach the end of the connection
- `warn` warning if there is no End-Of-Line
- `encoding` types of encoding for input strings

Example

Project Gutenberg

A great collection of texts are available from the **Project Gutenberg** which has a catalog of more than 25,000 free online books:

`http://www.gutenberg.org`

Moby Dick

Let's consider the famous novel **Moby Dick** by Herman Melville. A plain text file of Moby Dick can be found at:

`http://www.gutenberg.org/cache/epub/2701/pg2701.txt`

Reading Raw text

Here's how you could read the first 500 lines of cotent with `readLines()`

```
# url of Moby Dick (project Gutenberg)
moby_url <- url("http://www.gutenberg.org/cache/epub/2701/pg2701.txt")

# reading the content (first 500 lines)
moby_dick <- readLines(moby_url, n = 500)
```

Each line read is stored as an element in the character vector `moby_dick`

Goot to Know

Terms of Service

Some times, reading data directly from a website may be against the **terms of use** of the site.

Web Politeness

When you're reading (and "playing" with) content from a web page, make a local copy as a courtesy to the owner of the web site so you don't overload their server by constantly rereading the page. To make a copy from inside of R, look at the `download.file()` function.

Download Moby Dick

Downloading

It is good advice to download a copy of the file to your computer, and then play with it.

Let's use `download.file()` to save a copy in our working directory. In this case we create the file `"mobydick.txt"`

```
# download a copy in the working directory
download.file("http://www.gutenberg.org/cache/epub/2701/pg2701.txt",
              "mobydick.txt")
```

Abut scan()

Function `scan()`

Another very useful function that we can use to read text is `scan()`. By default, `scan()` expects to read numeric values, but we can change this behavior with the argument `what`

```
scan(file = "", what = double(), nmax = -1, n = -1, sep = "",
     quote = if(identical(sep, "\n")) "" else "'\"", dec = ".",
     skip = 0, nlines = 0, na.strings = "NA",
     flush = FALSE, fill = FALSE, strip.white = FALSE,
     quiet = FALSE, blank.lines.skip = TRUE, multi.line = TRUE,
     comment.char = "", allowEscapes = FALSE,
     fileEncoding = "", encoding = "unknown", text)
```

Function scan() (con't)

Some `scan()` arguments

- `file` the file name or a connection
- `what` type of data to be read
- `n` maximum number of data values to read
- `sep` type of separator
- `skip` number of lines to skip before reading values
- `nlines` maximum number of lines to be read

Reading Some Lines

If we want to read just a pre-specified number of lines, we have to loop over the file lines and read the content with `scan()`. For instance, let's skip the first 535 lines, and then read the following 10 lines of Chapter 1

```
# empty vector to store results
moby_dick_chap1 <- rep("", 10)

# number of lines to skip until Chapter 1
skip = 535

# reading 10 lines (line-by-line using scan)
for (i in 1L:10) {
  one_line = scan("mobydick.txt", what = "", skip = skip, nlines = 1)
  # pasting the contents in one_line
  moby_dick_chap1[i] = paste(one_line, collapse = " ")
  skip = skip + 1
}
```

Reading an HTML file

HTML File

Our third example involves reading the contents of an html file. We're just illustrating how to import html content as raw text in R. (We are not *parsing* html; we'll see that topic in the next lecture)

Egyptian Skulls

Let's consider the file containing information about the Egyptian Skulls data set by Thomson *et al*:

`http://lib.stat.cmu.edu/DASL/Datafiles/EgyptianSkulls.html`

Skulls Data

To read the html content we use `readLines()`

```
# read html file content as a string vector
skulls <- readLines("http://lib.stat.cmu.edu/DASL/Datafiles/EgyptianSkulls.html")
```

Iris Example

Iris Data

Data set `"iris"` from UCI Machine Learning Repo

`https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data`

Iris Data

How do we read the data?

If you try to simply use `read.csv()`, you'll be disappointed:

```
# URL of data file
uci <- "https://archive.ics.uci.edu/ml/machine-learning-databases/"
iris_file <- paste0(uci, "iris/iris.data")
```

```
# this won't work
iris_data <- read.csv(iris_file, header = FALSE)
```

Note that the URL starts with `https://`, that means a secured connection.

Iris Data

One solution is to use the package `"readr"`. Another solution requires some special functions:

- We need to use the R package `"RCurl"` to make an `HTTPS` request with `getURL()`
- We also need to use `textConnection()` inside `read.csv()`

Reading Iris Data

Here's how to read the iris data in R (without `"readr"`):

```
# URL of data file
uci <- "https://archive.ics.uci.edu/ml/machine-learning-databases/"
iris_file <- paste0(uci, "iris/iris.data")
```

`getURL` downloads the specified URL (and handles `https`).
Then we pass it as a text-connection to a read-table function:

```
library(RCurl)
iris_url <- getURL(iris_file)
iris_data <- read.csv(textConnection(iris_url), header = FALSE)
```

Wikipedia Table

Let's read an HTML table from Wikipedia. This is not technically a file, but a piece of content from an html document

#	Time	Name	Nationality	Date	Meet	Location
1	22:48.4	Henry Taylor	Great Britain	Jul 25, 1908	Olympic Games	London, United Kingdom
2	22:00.0	George Hodgson	Canada	Jul 10, 1912	Olympic Games	Stockholm, Sweden
3	21:35.3	Arne Borg	Sweden	Jul 8, 1923	-	Gothenburg, Sweden
4	21:15.0	Arne Borg	Sweden	Jan 30, 1924	-	Sydney, Australia
5	21:11.4	Arne Borg	Sweden	Jul 13, 1924	-	Paris, France

https://en.wikipedia.org/wiki/World_record_progression_1500_metres_freestyle

Since Jun-12-2015, wikipedia uses *HTTPS*, so the first step is to get the URL content via `getURL` (from `"RCurl"`). Then we parse the HTML document, and finally import tables with `readHTMLTable()`

```
library(RCurl)
library(XML)

# url
swim_url <- "https://en.wikipedia.org/wiki/World_record_progression_1500_metres_freestyle"
```

```
# parse HTML content
swim_wiki <- htmlParse(getURL(swim_url))

# reading HTML table
swim1500 <- readHTMLTable(swim_wiki)
```

The output is a list with as many data.frames as tables in the read document

Reading data in an HTML Table

Since we want just the first table, we can use the `which = 1` argument:

```
# reading HTML table
swim1500men <- readHTMLTable(swim_wiki, which = 1,
                             stringsAsFactors = FALSE)
```

Note that we can pass `data.frame()` parameters, in this case `stringsAsFactors = FALSE`

```
head(swim1500)
```

```
##   #    Time                                Name   Nationality
## 1 1 22:48.4      Taylor , Henry Henry Taylor Great Britain
## 2 2 22:00.0  Hodgson , George George Hodgson        Canada
## 3 3 21:35.3            Borg , Arne Arne Borg        Sweden
## 4 4 21:15.0            Borg , Arne Arne Borg        Sweden
## 5 5 21:11.4            Borg , Arne Arne Borg        Sweden
## 6 6 20:06.6      Charlton , Boy Boy Charlton     Australia
##                          Date          Meet
## 1 01908-07-25-0000Jul 25, 1908 Olympic Games
## 2 01912-07-10-0000Jul 10, 1912 Olympic Games
## 3  01923-07-08-0000Jul 8, 1923             -
## 4 01924-01-30-0000Jan 30, 1924             -
## 5 01924-07-13-0000Jul 13, 1924             -
## 6 01924-07-15-0000Jul 15, 1924 Olympic Games
##                                        Location Ref
## 1 United Kingdom, London !  London, United Kingdom
## 2           Sweden, Stockholm !  Stockholm, Sweden
## 3         Sweden, Gothenburg !  Gothenburg, Sweden
## 4           Australia, Sydney !  Sydney, Australia
## 5                  France, Paris !  Paris, France
## 6                  France, Paris !  Paris, France
```