LAMAR UNIVERSITY

MEMBER THE TEXAS STATE UNIVERSITY SYSTEM™

Civil and Environmental Engineering

Professional Development Friday Workshop Series

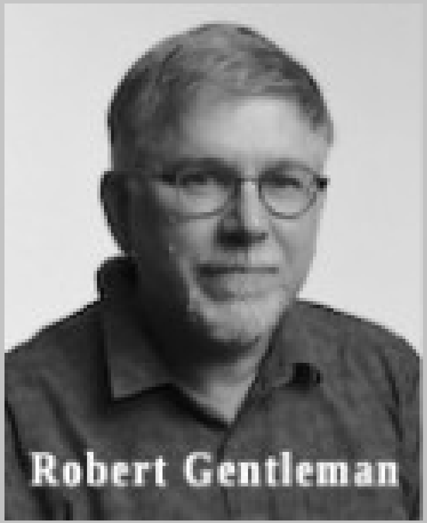
Introduction to R (Part I)

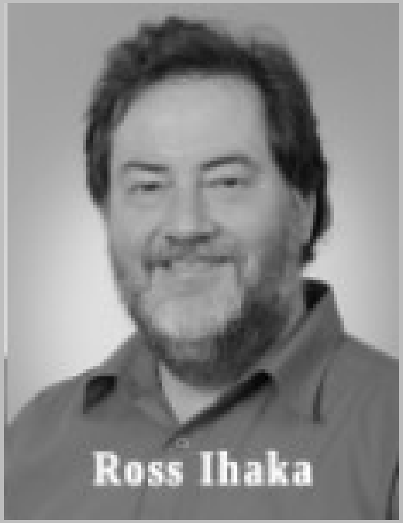
Venki Uddameri, Ph.D. P.E., F. AWRA
William B and Mary G. Mitchell Endowed Chair
Professor and Chair, Civil & Environmental Eng.

R Background

What is R?

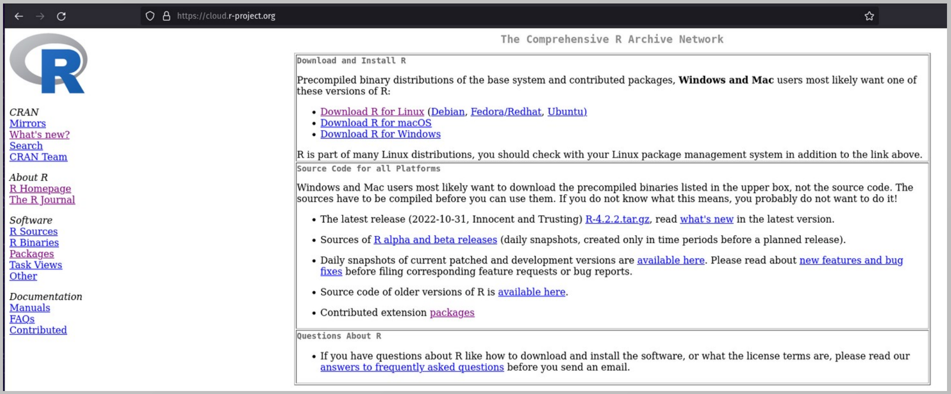
- R is a statistical and computing Ecosystem
  - Developed by Ross Ihaka and Robert Gentleman
- Based on a program called “S”
  - John Chambers at AT&T Bell Labs
- R is an open-source and free software
  - Free as in free speech and free beer!!
- R is now owned and managed by R foundation
- First public release was in 2000
  - Currently version 4.2.2





How to Obtain R

- R can be downloaded from CRAN repository
- Comprehensive R Archive Network
  - https://cran.r-project.org
- R is available on all major platforms
  - Windows, MacOS and Linux
- CRAN has many mirrors across the globe



The Comprehensive R Archive Network

CRAN
Mirrors
What's new?
Search
CRAN Team

About R
R Homepage
The R Journal

Software
R Sources
R Binaries
Packages
Task Views
Other

Documentation
Manuals
FAQs
Contributed

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux (Debian, Fedora/Redhat, Ubuntu)
- Download R for macOS
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

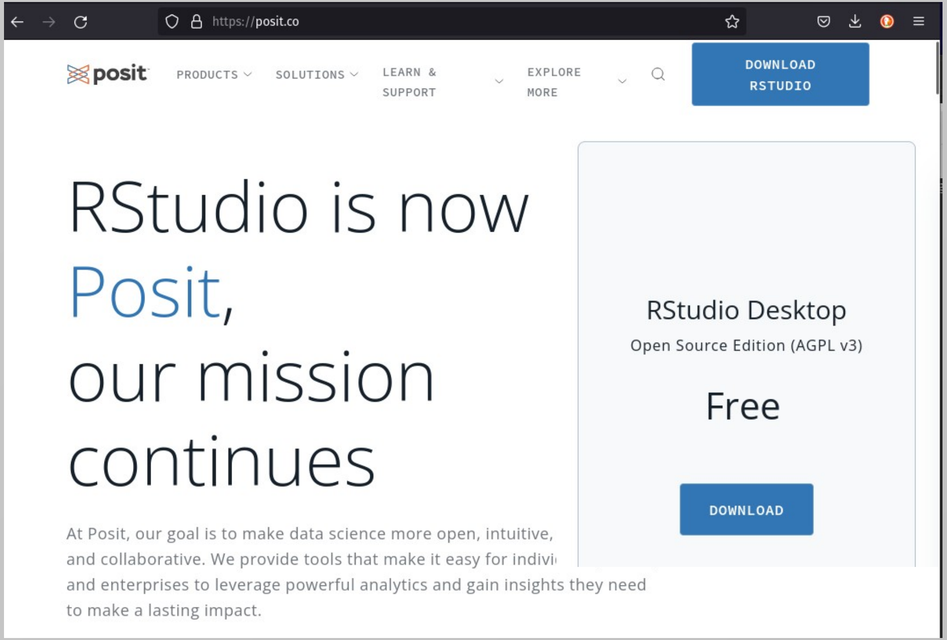
- The latest release (2022-10-31, Innocent and Trusting) R-4.2.2.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

Windows and MacOS versions come with basic GUI. You can also run from command line

RStudio



https://posit.co

RStudio is now Posit, our mission continues

At Posit, our goal is to make data science more open, intuitive, and collaborative. We provide tools that make it easy for indivi and enterprises to leverage powerful analytics and gain insights they need to make a lasting impact.

RStudio Desktop

Open Source Edition (AGPL v3)

Free

- Rstudio is a popular IDE for R
  - IDE: Interactive Development Environment
- Rstudio is developed and maintained by Posit.co
  - Formerly known as rstudio.com
- Rstudio has both free and commercial versions
  - You can also get desktop and server versions
- Windows users can download latest stable .exe file
- Linux users may find installing daily builds easier

You MUST install R before you install RStudio

Other R IDEs

- In addition to Rstudio there are other IDEs available for R as well
- R can also be run using the Jupyter Environment
  - Notebooks and Jupyter Lab
- Running R with Jupyter and Jupyter Notebooks helps integrate R with HTML to create reproducible notebooks
- Most Data Science Applications of R are built using either Rstudio or Jupyter Notebooks

Easiest Way to Install Jupyter is through Anaconda Distribution

Take this route if you are interested in using both R and Python

You can download Anaconda Developer from https://anaconda.com

Running R in Jupyter Environment required you to have IRKernel library and Python

R Philosophy and Resources

R Program Philosophy

- Built on lean philosophy
- Software provides some basic functionality
- Much of the advanced functionality is supported to external packages
  - Libraries
- R comes pre-installed with some libraries
  - Some are loaded along with R
  - Others are installed on your hard-drive but not loaded during runtime.
- External libraries have to be downloaded onto your computer and then loaded into your active memory (RAM)
  - Helps manage your memory resources better

CRAN has a repository of over 19,000 packages

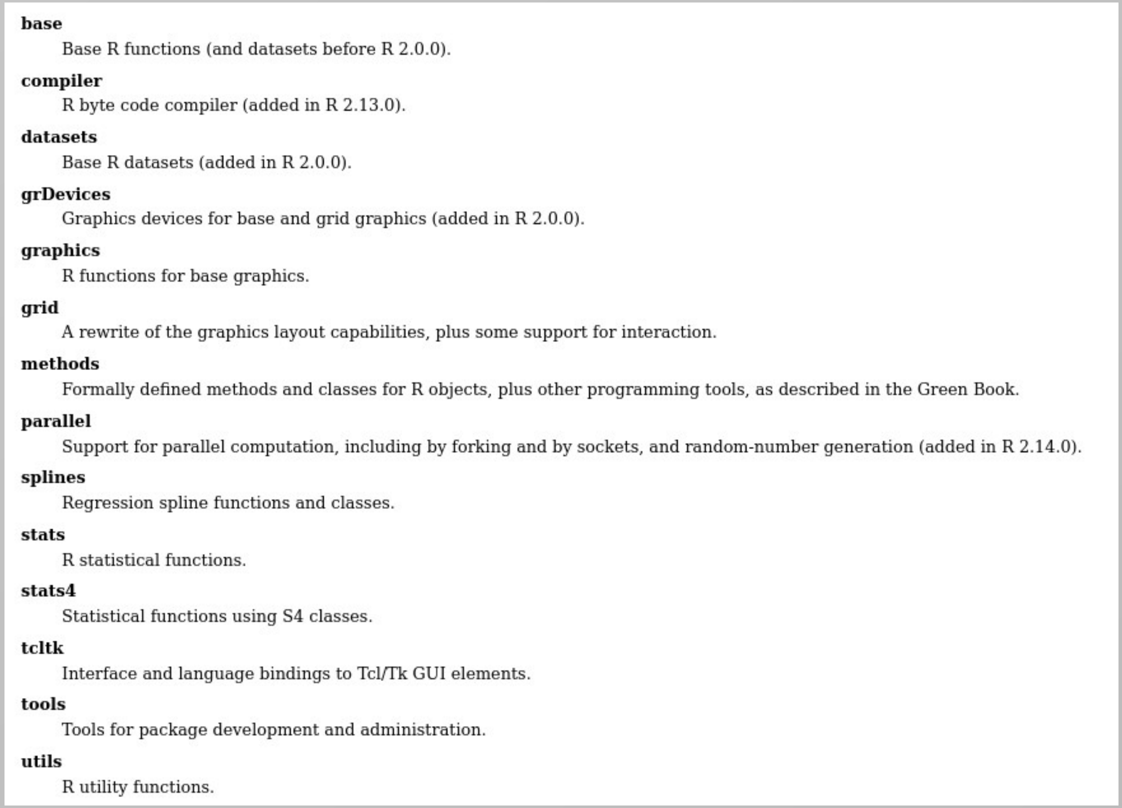
There are other repositories – Bioconductor; Github

More than 25000 packages available

Significantly leaner than EXCEL not as lean as Python

R Base Packages

- R Base packages are installed with R
  - Loaded at runtime
- You can use the functions in these libraries without calling them explictly
- Much of the native functionality of R is actually embedded in these libraries



base
Base R functions (and datasets before R 2.0.0).

compiler
R byte code compiler (added in R 2.13.0).

datasets
Base R datasets (added in R 2.0.0).

grDevices
Graphics devices for base and grid graphics (added in R 2.0.0).

graphics
R functions for base graphics.

grid
A rewrite of the graphics layout capabilities, plus some support for interaction.

methods
Formally defined methods and classes for R objects, plus other programming tools, as described in the Green Book.

parallel
Support for parallel computation, including by forking and by sockets, and random-number generation (added in R 2.14.0).

splines
Regression spline functions and classes.

stats
R statistical functions.

stats4
Statistical functions using S4 classes.

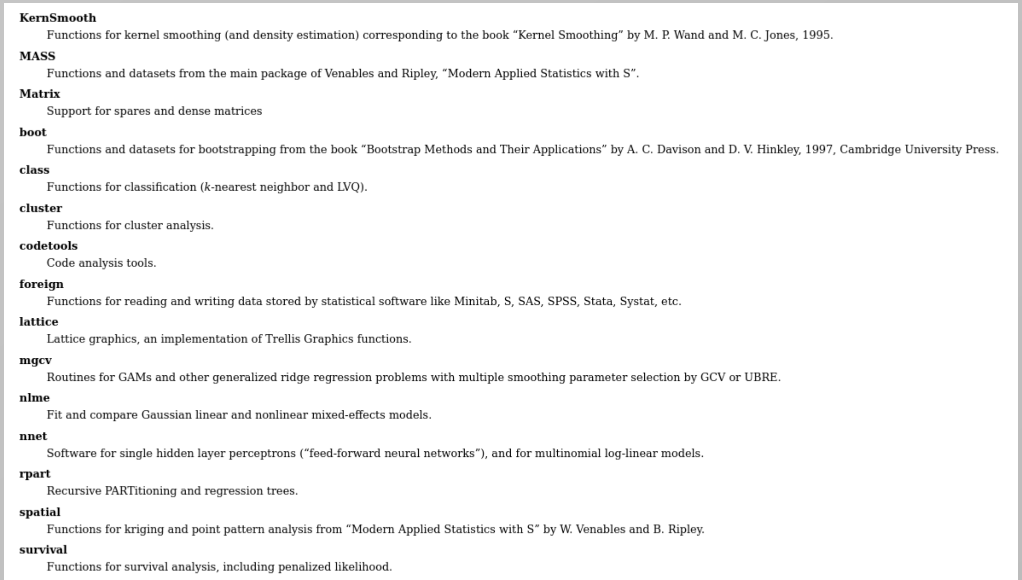
tcltk
Interface and language bindings to Tcl/Tk GUI elements.

tools
Tools for package development and administration.

utils
R utility functions.

R Recommended Packages

- Recommended packages are installed as part of your standard installation
- They are therefore on your hard-drive
- They are not automatically loaded into active memory when R is started
  - You have to call them in your program
  - Use library(package-name) command



KernSmooth
Functions for kernel smoothing (and density estimation) corresponding to the book “Kernel Smoothing” by M. P. Wand and M. C. Jones, 1995.

MASS
Functions and datasets from the main package of Venables and Ripley, “Modern Applied Statistics with S”.

Matrix
Support for spares and dense matrices

boot
Functions and datasets for bootstrapping from the book “Bootstrap Methods and Their Applications” by A. C. Davison and D. V. Hinkley, 1997, Cambridge University Press.

class
Functions for classification (*k*-nearest neighbor and LVQ).

cluster
Functions for cluster analysis.

codetools
Code analysis tools.

foreign
Functions for reading and writing data stored by statistical software like Minitab, S, SAS, SPSS, Stata, Systat, etc.

lattice
Lattice graphics, an implementation of Trellis Graphics functions.

mgcv
Routines for GAMs and other generalized ridge regression problems with multiple smoothing parameter selection by GCV or UBRE.

nlme
Fit and compare Gaussian linear and nonlinear mixed-effects models.

nnet
Software for single hidden layer perceptrons (“feed-forward neural networks”), and for multinomial log-linear models.

rpart
Recursive PARTitioning and regression trees.

spatial
Functions for kriging and point pattern analysis from “Modern Applied Statistics with S” by W. Venables and B. Ripley.

survival
Functions for survival analysis, including penalized likelihood.

You can now load a specific function from a library without loading it entirely
Use: library-name::function(args)

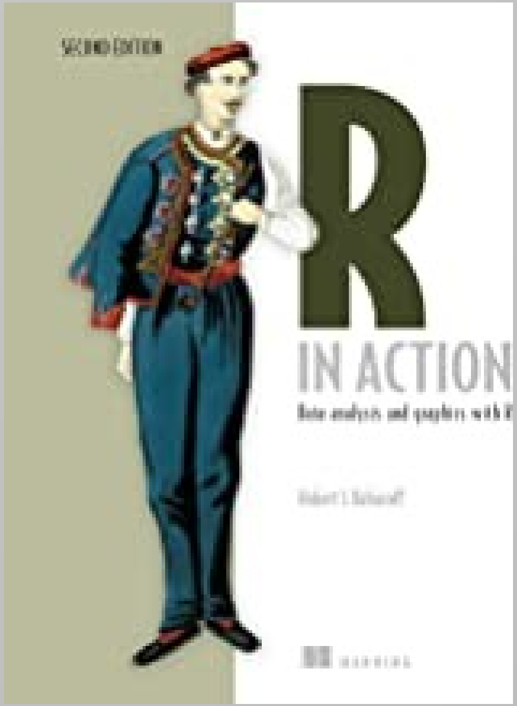
R Other Libraries

- It is estimated that there are over 25000 libraries available in R
- CRAN is the official repository of packages
  - A package submitted must meet reproducibility requirements
    - Proper documentation
    - Example Data
    - Dependencies on other libraries
- More recently packages are being made available on GitHub
  - Less cumbersome process
  - But not as standardized.
- Microsoft and Bioconductor are other repositories for packages

Goto:
https://cran.r-project.org/web/packages/

For a comprehensive listing of all packages available thru CRAN
Over 19000 package

R Support Ecosystem

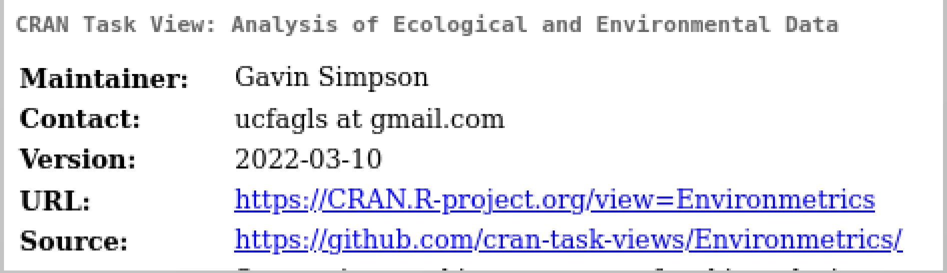


- R has an extensive user community
- Very helpful in answering questions
  - Stackoverflow is a great resource
- Youtube and the web have several excellent resources
- Several free and paid books are available
- Several R programming courses on Coursera
  - Not geared towards engineers

Somewhat Steep Learning Curve – But well worth the Effort!!!

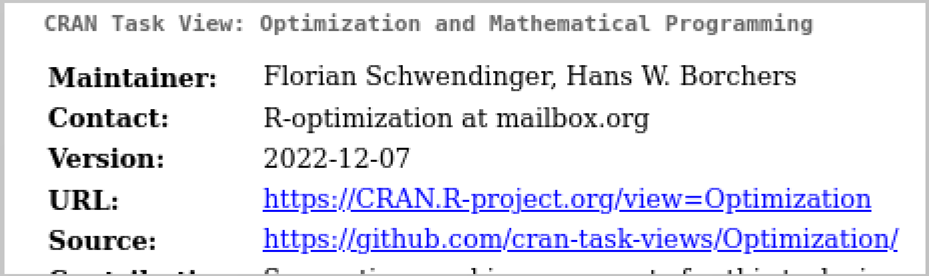
R Support

- R Taskviews are very helpful to get discipline specific packages
  - https://cran.r-project.org/
    - click on Task Views on the right



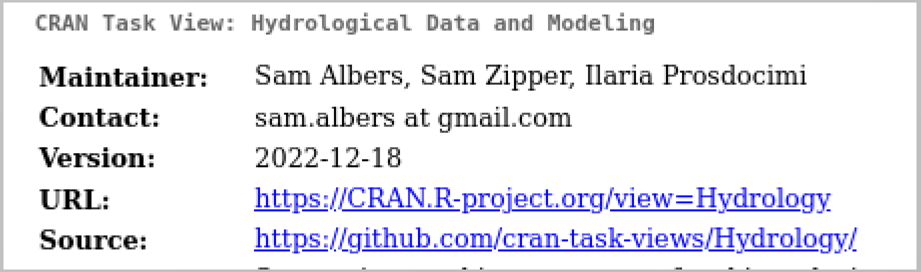
CRAN Task View: Analysis of Ecological and Environmental Data

Maintainer:	Gavin Simpson
Contact:	ucfagls at gmail.com
Version:	2022-03-10
URL:	https://CRAN.R-project.org/view=Environmetrics
Source:	https://github.com/cran-task-views/Environmetrics/



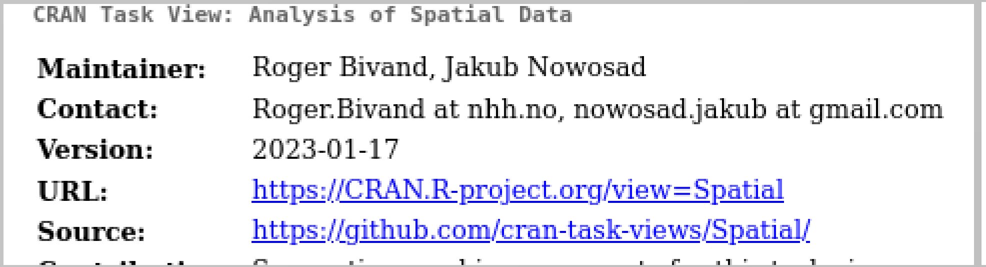
CRAN Task View: Optimization and Mathematical Programming

Maintainer:	Florian Schwendinger, Hans W. Borchers
Contact:	R-optimization at mailbox.org
Version:	2022-12-07
URL:	https://CRAN.R-project.org/view=Optimization
Source:	https://github.com/cran-task-views/Optimization/



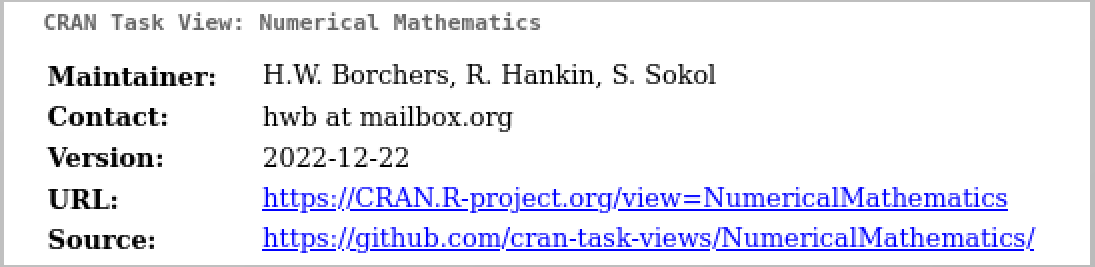
CRAN Task View: Hydrological Data and Modeling

Maintainer:	Sam Albers, Sam Zipper, Ilaria Prosdocimi
Contact:	sam.albers at gmail.com
Version:	2022-12-18
URL:	https://CRAN.R-project.org/view=Hydrology
Source:	https://github.com/cran-task-views/Hydrology/



CRAN Task View: Analysis of Spatial Data

Maintainer:	Roger Bivand, Jakub Nowosad
Contact:	Roger.Bivand at nhh.no, nowosad.jakub at gmail.com
Version:	2023-01-17
URL:	https://CRAN.R-project.org/view=Spatial
Source:	https://github.com/cran-task-views/Spatial/

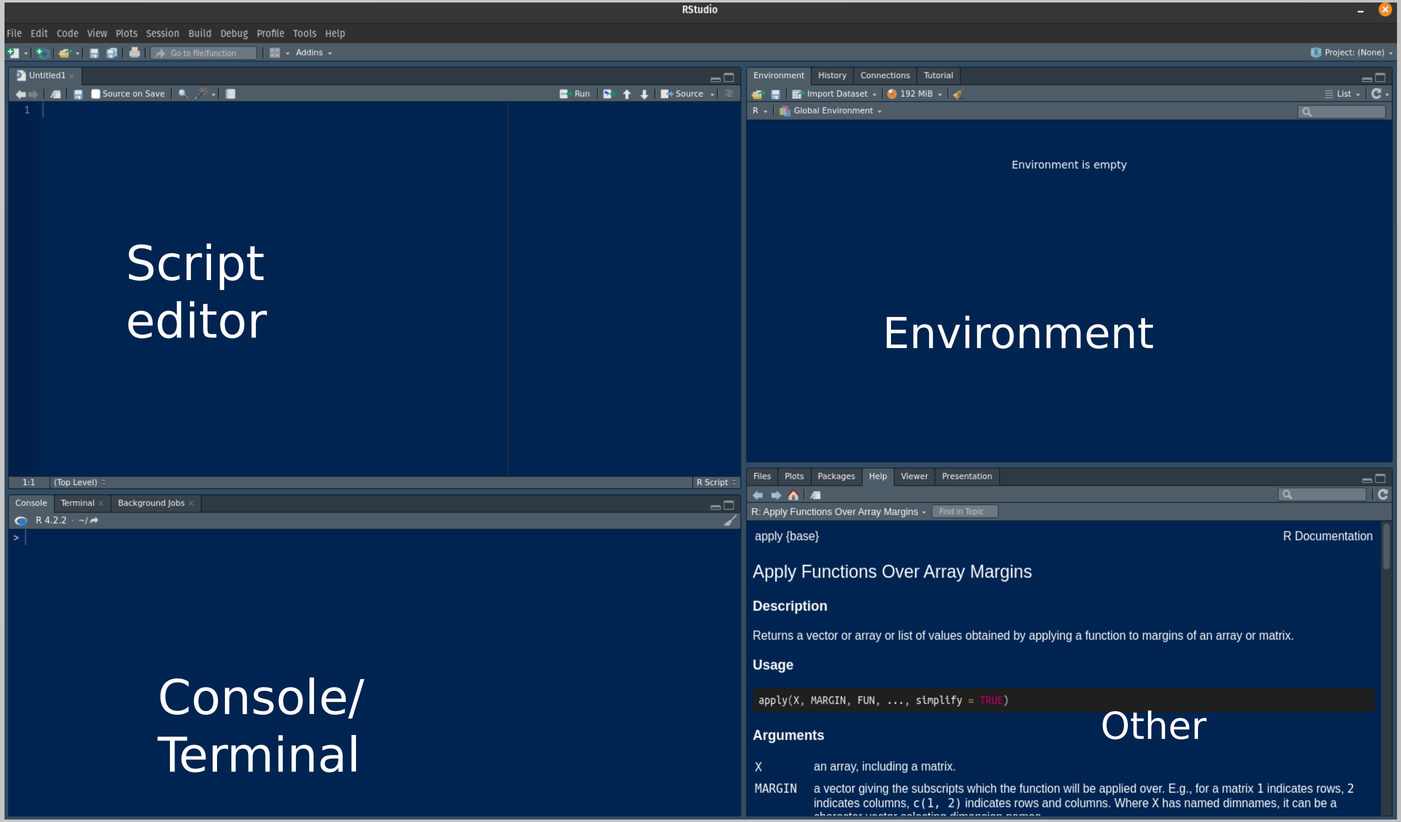


CRAN Task View: Numerical Mathematics

Maintainer:	H.W. Borchers, R. Hankin, S. Sokol
Contact:	hwb at mailbox.org
Version:	2022-12-22
URL:	https://CRAN.R-project.org/view=NumericalMathematics
Source:	https://github.com/cran-task-views/NumericalMathematics/

Using R

RStudio



apply {base} R Documentation

Apply Functions Over Array Margins

Description

Returns a vector or array or list of values obtained by applying a function to margins of an array or matrix.

Usage

```
apply(X, MARGIN, FUN, ..., simplify = TRUE)
```

Arguments

X an array, including a matrix.

MARGIN a vector giving the subscripts which the function will be applied over. E.g., for a matrix 1 indicates rows, 2 indicates columns, c(1, 2) indicates rows and columns. Where X has named dimnames, it can be a character vector selecting dimension names.

RStudio Components

- Script Editor
- Console/Terminal
- Environment
- Other

R Studio has a cheatsheet in the help menu is helpful

Using R General Steps

- Understand the problem to be solved
  - What questions do you want anwered from your data
- Create a workflow of analysis
  - Clean up and prepare data
  - Read data into R
  - Do additional cleanup or subsetting
  - Perform analysis
    - May need additional libraries
    - Have to write custom functions
    - Know the syntax
  - Make plots
  - Export the results for report writing

Recognize that the workflow is not linear

Several steps need to be carried out iteratively

R is used interactively:

- Some steps are automated
- Some steps involve manual intervention

R Primitive Data Structures

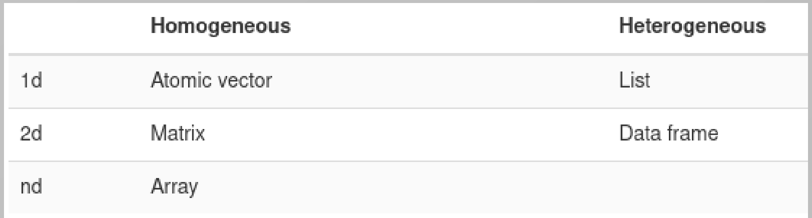
R Language

- Some basic syntax of R language and an understanding of Data Types and Structures is important before you delve in
- I will get you started on some basic concepts here
  - Reduce your first-use anxeity
- We shall follow up with data analysis exercises to further explore R
  - Next week workshop

Data Structures and Types

- Scalars are variables with a single value
- Integer
  - Numbers without decimals
- Floating point
  - Numbers with decimals
- Character
  - Strings and texts
- Complex
  - Complex numbers 3 + 2i

- Vectors are variables with multiple values (sequences)
- Vectors contain a set of scalars
  - Homogeneous
  - Heterogeneous
- 1D data – Vectors and Lisis
- 2D data – Matrix, Data.Frame
- Multidimensional - Arrays

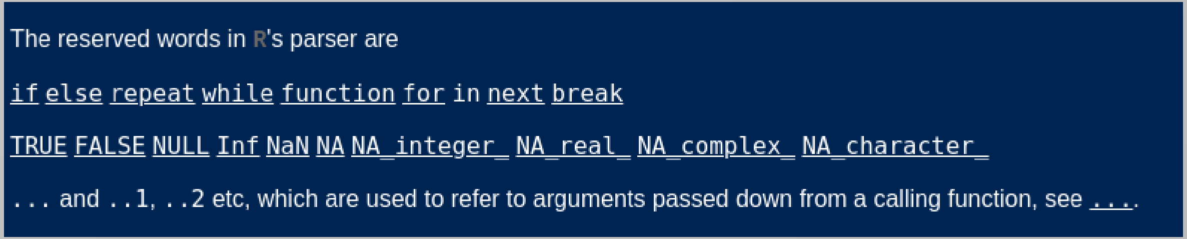


	Homogeneous	Heterogeneous
1d	Atomic vector	List
2d	Matrix	Data frame
nd	Array	

Variables and Objects

- Variables are used to store data
- Variables point to a memory location of where data are stored
  - Helps you work at a higher level
- You can define variables and assign values
- Variable naming has a few rules:
  - Variables in R are case-sensitive
  - Variables use alphanumeric characters
    - Cannot start with a number
  - You can use dot (.) and under_score(_)
    - Use of dot is not recommended
  - You cannot use reserve words
    - Words with special meaning to R
- You do not have to declare variables in the beginning

Everything in R is an Object
So variables are also objects!!



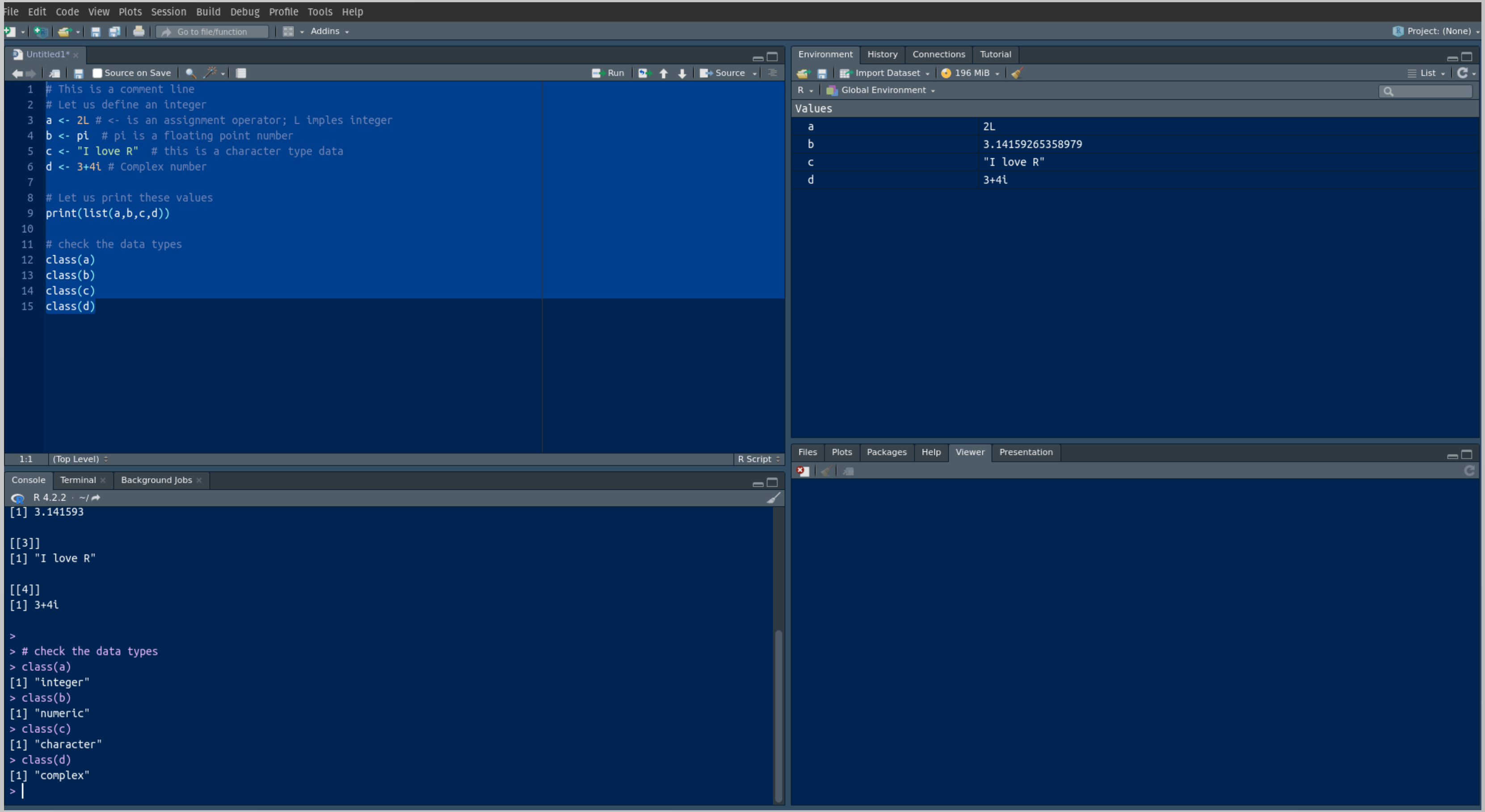
The reserved words in R's parser are

`if` `else` `repeat` `while` `function` `for` in `next` `break`

`TRUE` `FALSE` `NULL` `Inf` `NaN` `NA` `NA_integer_` `NA_real_` `NA_complex_` `NA_character_`

`...` and `..1`, `..2` etc, which are used to refer to arguments passed down from a calling function, see `...`.

Example 1: Constants



```r
# This is a comment line
# Let us define an integer
a <- 2L # <- is an assignment operator; L imples integer
b <- pi  # pi is a floating point number
c <- "I love R"  # this is a character type data
d <- 3+4i # Complex number

# Let us print these values
print(list(a,b,c,d))

# check the data types
class(a)
class(b)
class(c)
class(d)
```

Values

```
a        2L
b        3.14159265358979
c        "I love R"
d        3+4i
```

```
[1] 3.141593

[[3]]
[1] "I love R"

[[4]]
[1] 3+4i

>
> # check the data types
> class(a)
[1] "integer"
> class(b)
[1] "numeric"
> class(c)
[1] "character"
> class(d)
[1] "complex"
>
```

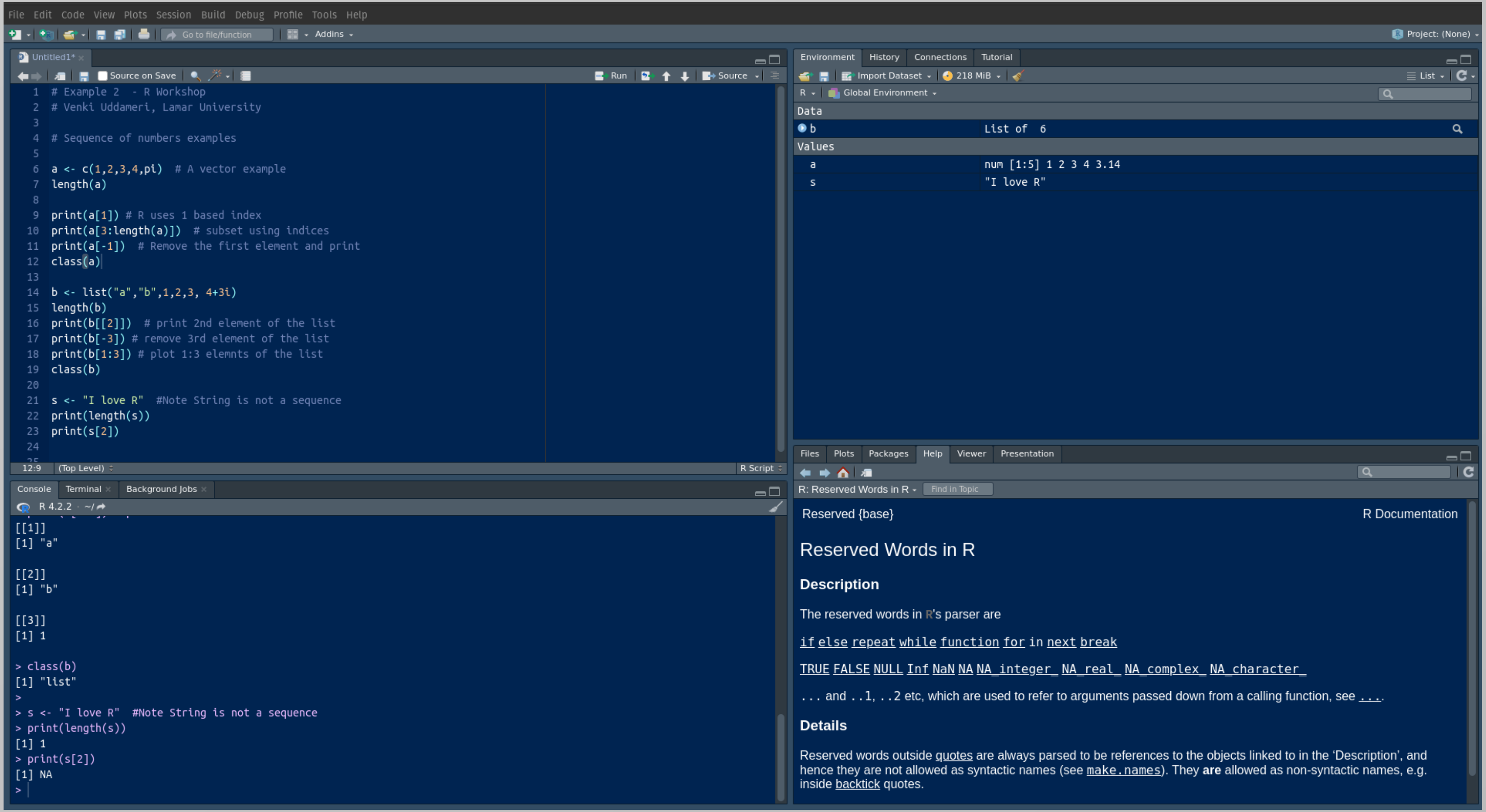
Syntax Used

- # is used to write a comment line
  - Most important part of any R script
- “<-” is the (left assignment) operator
  - Assigns a value on the right to a value on the left
- Putting L after a whole number tells R it is an integer
- We used list(a,b,c,d) to combine different data types
- We used class(x) function to get the data type
  - Class is the blueprint from which an object (variable) is instantiated
- We used the print() function to print out the list

Notice we embedded the list function into the print function. This is called Nesting

Nest is cool but confusing!!

Example 2 – Sequence Data Types



```r
# Example 2  - R Workshop
# Venki Uddameri, Lamar University

# Sequence of numbers examples

a <- c(1,2,3,4,pi)  # A vector example
length(a)

print(a[1]) # R uses 1 based index
print(a[3:length(a)])  # subset using indices
print(a[-1])  # Remove the first element and print
class(a)

b <- list("a","b",1,2,3, 4+3i)
length(b)
print(b[[2]])  # print 2nd element of the list
print(b[-3]) # remove 3rd element of the list
print(b[1:3]) # plot 1:3 elemnts of the list
class(b)

s <- "I love R"  #Note String is not a sequence
print(length(s))
print(s[2])

```

```
[[1]]
[1] "a"

[[2]]
[1] "b"

[[3]]
[1] 1

> class(b)
[1] "list"
>
> s <- "I love R"  #Note String is not a sequence
> print(length(s))
[1] 1
> print(s[2])
[1] NA
>
```

```
Data
b                         List of  6
Values
  a                       num [1:5] 1 2 3 4 3.14
  s                       "I love R"
```

Reserved {base}   R Documentation

Reserved Words in R

Description

The reserved words in R's parser are

`if` `else` `repeat` `while` `function` `for` `in` `next` `break`

`TRUE` `FALSE` `NULL` `Inf` `NaN` `NA` `NA_integer_` `NA_real_` `NA_complex_` `NA_character_`

`...` and `..1`, `..2` etc, which are used to refer to arguments passed down from a calling function, see `...`.

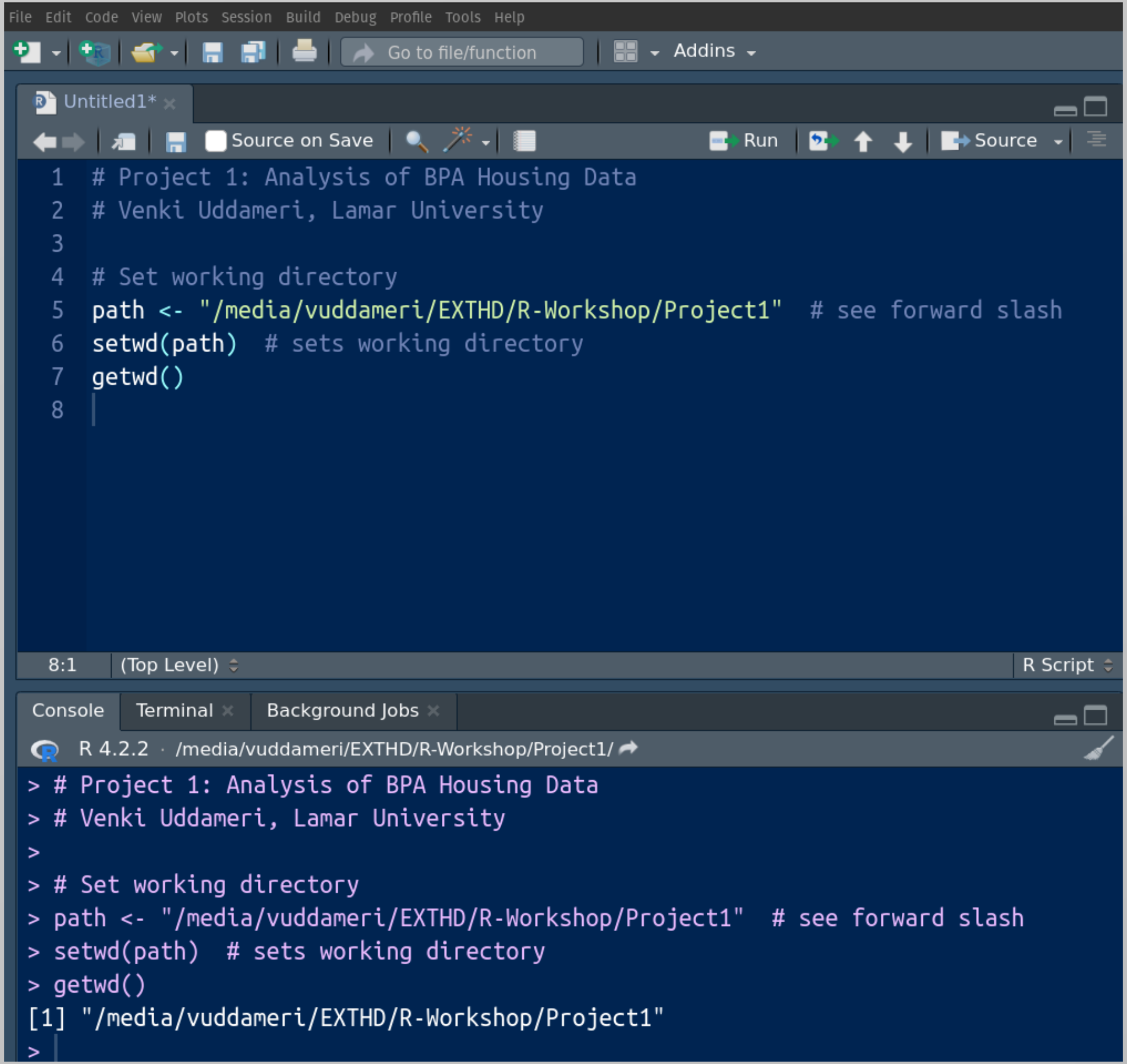
Details

Reserved words outside quotes are always parsed to be references to the objects linked to in the 'Description', and hence they are not allowed as syntactic names (see `make.names`). They **are** allowed as non-syntactic names, e.g. inside backtick quotes.

Setting Working Directory

Working Directory

- A working directory is a file folder where you store all your data and R Code relevant to your project
- Set a separate working directory for each project
- Always set the working directory on your Hard-Drive
  - You can archive it on an external or cloud drive
  - Always keep a duplicate copy after substantial work
    - Avoids loss of work if the system crashes
- You can use setwd(path) for setting your working directory
  - Path: Use “/” forward slash or “\\” Double back-slash when writing path
- You need to have the directory created before calling it as a working directory


File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Untitled1*
Source on Save
Run
Source
8:1
(Top Level)
R Script
Console
Terminal
Background Jobs
R 4.2.2 · /media/vuddameri/EXTHD/R-Workshop/Project1/

```r
# Project 1: Analysis of BPA Housing Data
# Venki Uddameri, Lamar University

# Set working directory
path <- "/media/vuddameri/EXTHD/R-Workshop/Project1"  # see forward slash
setwd(path)  # sets working directory
getwd()

```

```
> # Project 1: Analysis of BPA Housing Data
> # Venki Uddameri, Lamar University
>
> # Set working directory
> path <- "/media/vuddameri/EXTHD/R-Workshop/Project1"  # see forward slash
> setwd(path)  # sets working directory
> getwd()
[1] "/media/vuddameri/EXTHD/R-Workshop/Project1"
>
```

Setting Working Directory

1) Set a path variable: Easier to change later

2) Use setwd(path)

3) Use getwd() to see if the set working directory is correct.

Reading Data in R

R and Data Files

- R is capable of reading a variety of data types
  - Spreadsheet type data is the most common
    - Table format data
- R can read Time-series data
- R can read images and GIS datasets
  - Both vector and raster datasets
- R can read Audio files
- R can read Videofiles

Volume
Value
Big Data
Velocity
Veracity
Variety

Special Data formats such as NetCDF and HDF5

Reading Tabular Data

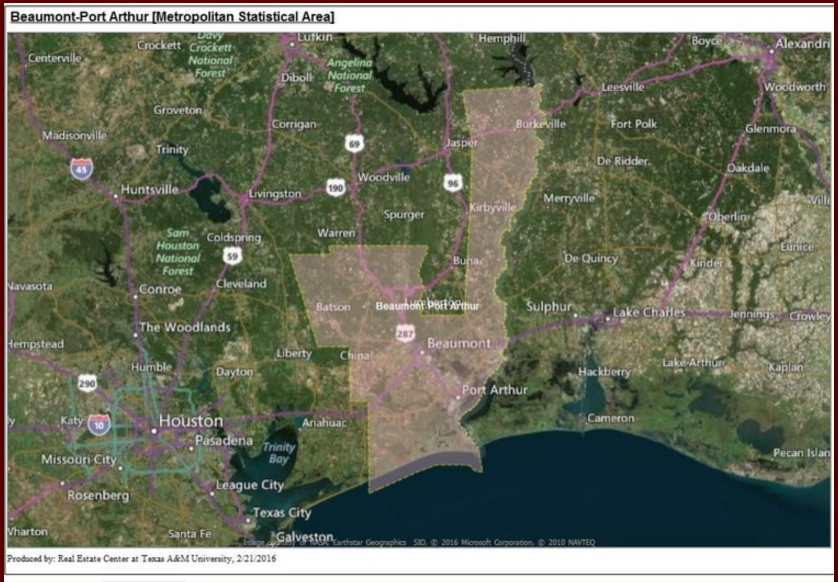
- Avoid using proprietary formats
  - MS EXCEL (xls and xlsx)
- Convert to platform independent formats
  - Comma Separated Value (CSV) files
- Let us use a simple dataset to explore some properties
  - BPAHousing.csv
  -

Dataset

- Housing data from Beaumont-Port Arthur Metropolitan Statistical Area (MSA)
- Monthly Housing sales data from Jan 1990 – Dec 2022
- Sales: # of houses sold
- Volume: Total value in $
- AvPrice: Average Price in $, Affected by outliers
- MedPrice: Medan Prince in $, Robust to outliers
- TotalListings: Total # of houses available
- Inventory: # Av.Months on inventory before being sold in that month
- Month: Month
- Year:Year

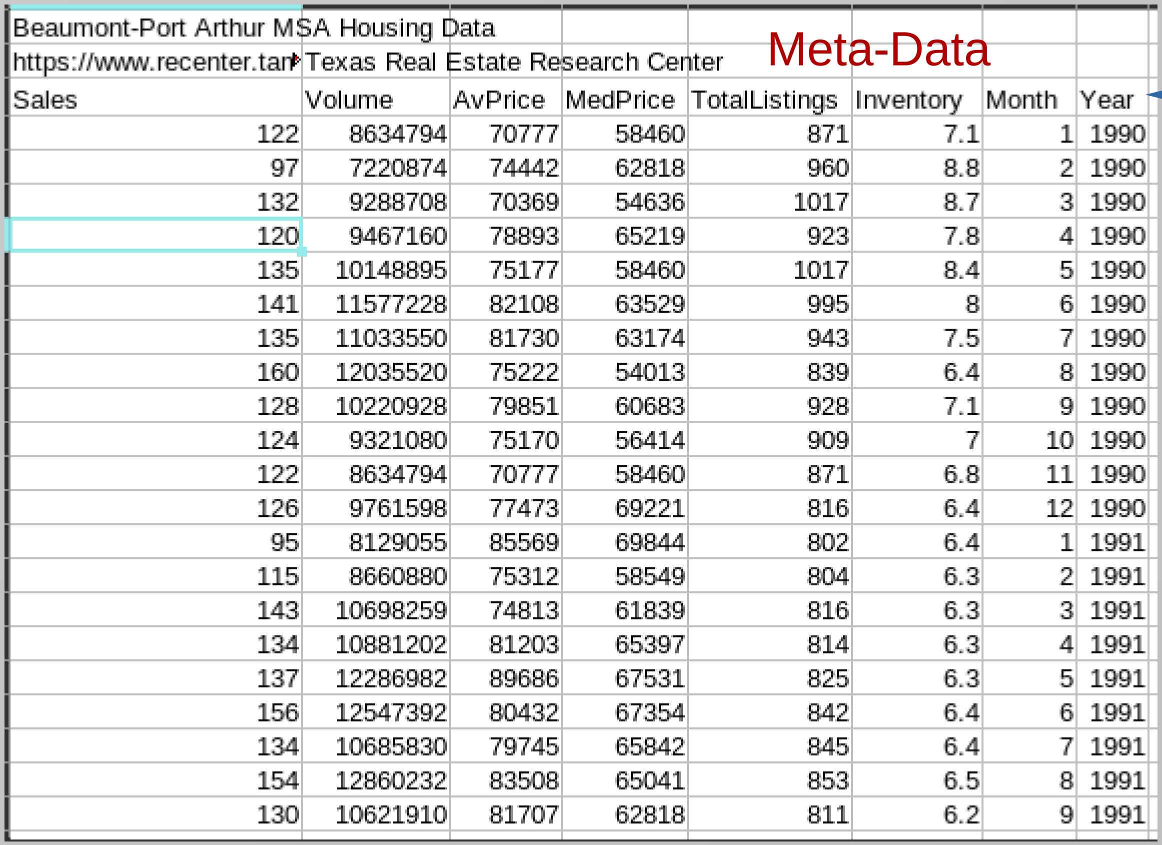
Data from Texas Real Estate Center

https://www.recenter.tamu.edu/data/housing-activity/#!/activity/MSA/Beaumont-Port_Arthur


Beaumont-Port Arthur [Metropolitan Statistical Area]

Produced by: Real Estate Center at Texas A&M University, 2/21/2016

Data Structure



Beaumont-Port Arthur MSA Housing Data							
https://www.recenter.tan	Texas Real Estate Research Center						
Sales	Volume	AvPrice	MedPrice	TotalListings	Inventory	Month	Year
122	8634794	70777	58460	871	7.1	1	1990
97	7220874	74442	62818	960	8.8	2	1990
132	9288708	70369	54636	1017	8.7	3	1990
120	9467160	78893	65219	923	7.8	4	1990
135	10148895	75177	58460	1017	8.4	5	1990
141	11577228	82108	63529	995	8	6	1990
135	11033550	81730	63174	943	7.5	7	1990
160	12035520	75222	54013	839	6.4	8	1990
128	10220928	79851	60683	928	7.1	9	1990
124	9321080	75170	56414	909	7	10	1990
122	8634794	70777	58460	871	6.8	11	1990
126	9761598	77473	69221	816	6.4	12	1990
95	8129055	85569	69844	802	6.4	1	1991
115	8660880	75312	58549	804	6.3	2	1991
143	10698259	74813	61839	816	6.3	3	1991
134	10881202	81203	65397	814	6.3	4	1991
137	12286982	89686	67531	825	6.3	5	1991
156	12547392	80432	67354	842	6.4	6	1991
134	10685830	79745	65842	845	6.4	7	1991
154	12860232	83508	65041	853	6.5	8	1991
130	10621910	81707	62818	811	6.2	9	1991

Meta-Data

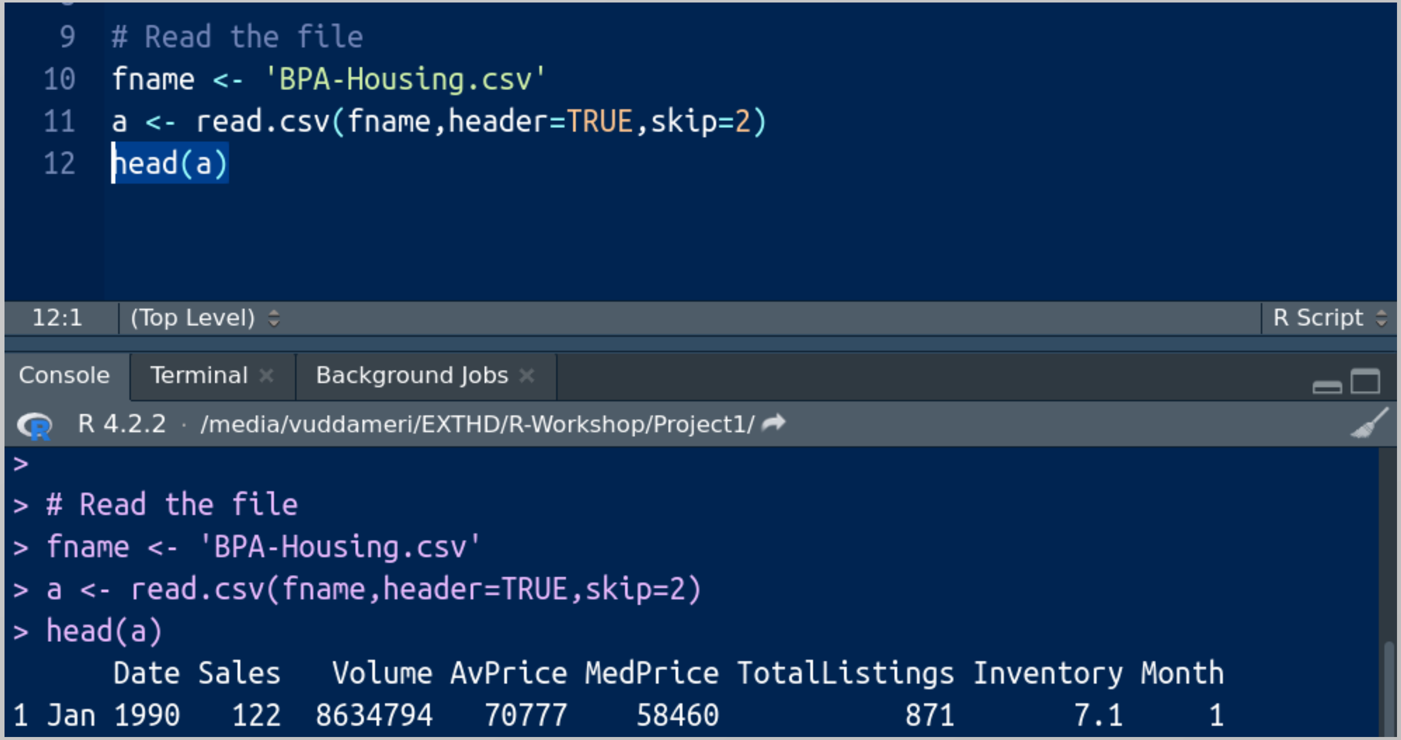
Header (Attribute Names)

Rows (Data)

Use read.csv(fname, …) for reading the file

Make sure the file is in the Working Directory

Reading a Data File



```
# Read the file
fname <- 'BPA-Housing.csv'
a <- read.csv(fname,header=TRUE,skip=2)
head(a)
```

```
>
> # Read the file
> fname <- 'BPA-Housing.csv'
> a <- read.csv(fname,header=TRUE,skip=2)
> head(a)
      Date Sales   Volume AvPrice MedPrice TotalListings Inventory Month
1 Jan 1990   122  8634794   70777    58460           871       7.1     1
```

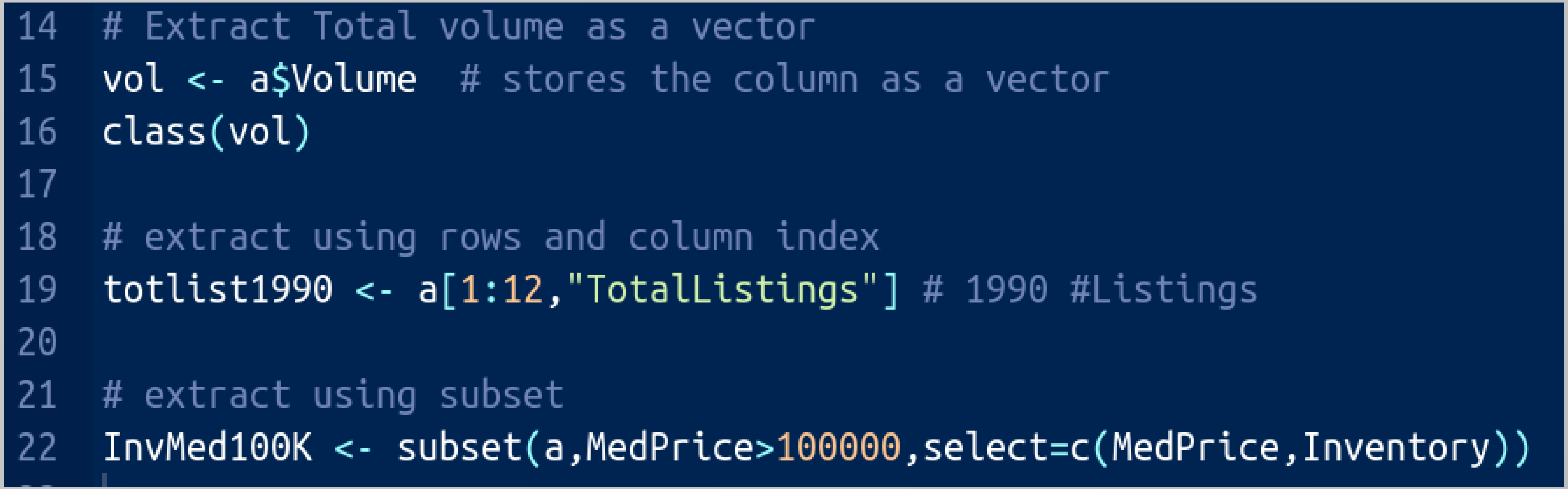
Functions Used:
read.csv
head(a)

Entire content of the file is stored in object **a**

Subsetting Data

- Subsetting can happen in 3 different ways:
  - Extract one or more columns
  - Extract some rows of all columns
  - Extract some rows of some columns
- All 3 operations can be done easily using R
  - Use of $ for extracting a column as a vector
  - Use row indices to extract rows and columns
  - Use subset() function to extract rows based on some criteria

Slicing and Dicing Data



```r
# Extract Total volume as a vector
vol <- a$Volume  # stores the column as a vector
class(vol)

# extract using rows and column index
totlist1990 <- a[1:12,"TotalListings"] # 1990 #Listings

# extract using subset
InvMed100K <- subset(a,MedPrice>100000,select=c(MedPrice,Inventory))
```

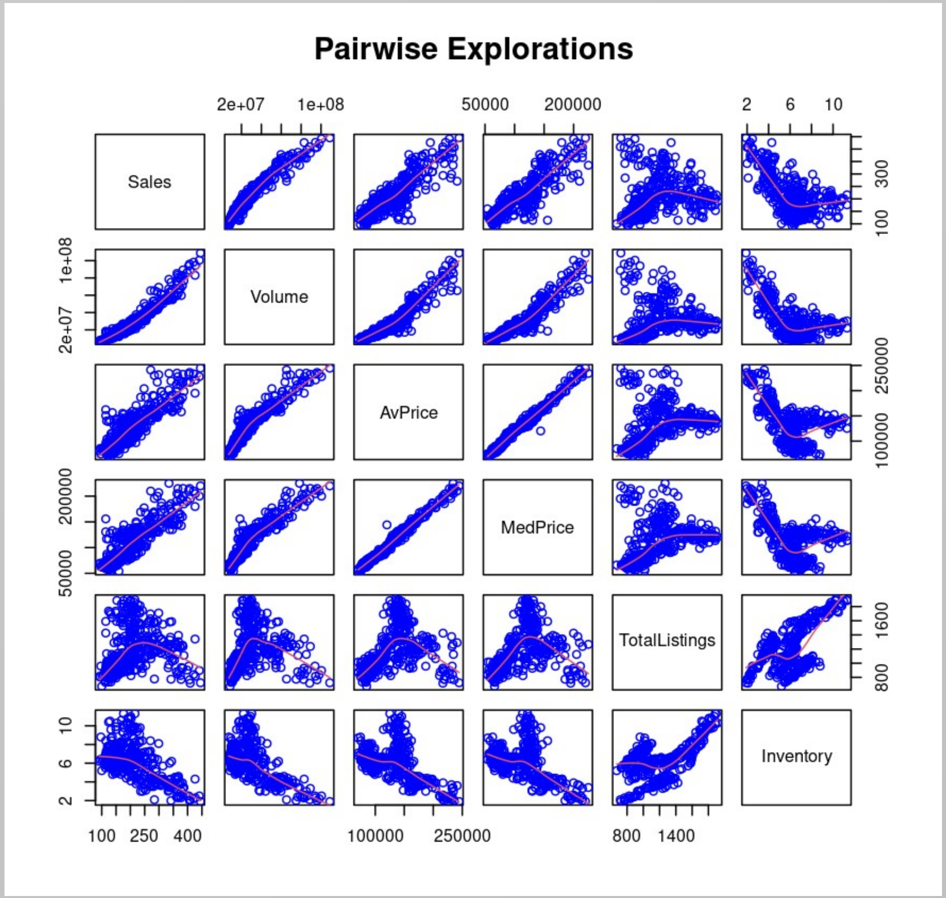
R has a very clean syntax. F() Parenthesis is used Functions. [] Square Brackets is used for matrix and data.frame and {} are used for code-block.

Visualization in R

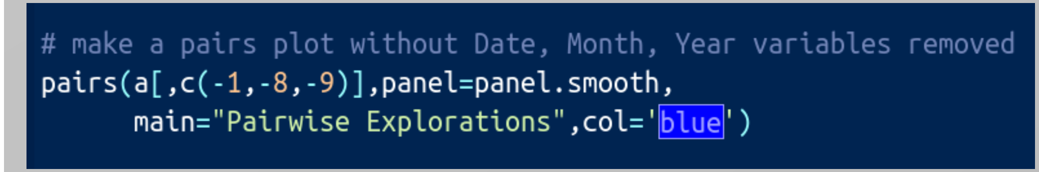
Visualization in R

- R really shines in the visualizations department
- You can create high-quality graphics for both print and other media
- A variety of different types of charts and plots using R
  - Can be used to plot both 2D and 3D graphs
- R is very good to do Exploratory Data Analysis (EDA)
  - Work with multi-dimensional data
  - Find relationships
  - Data reduction methods

Exploratory Data Analysis (EDA)

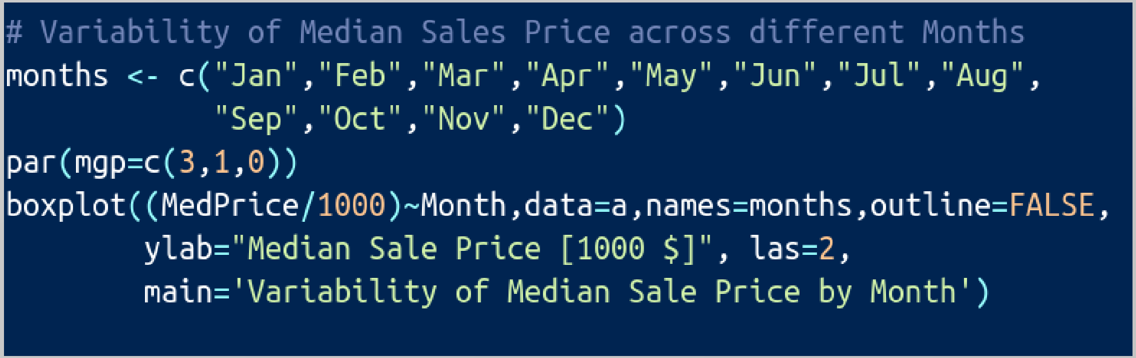


Pairs plot gives a useful way to assess relationships between several variables

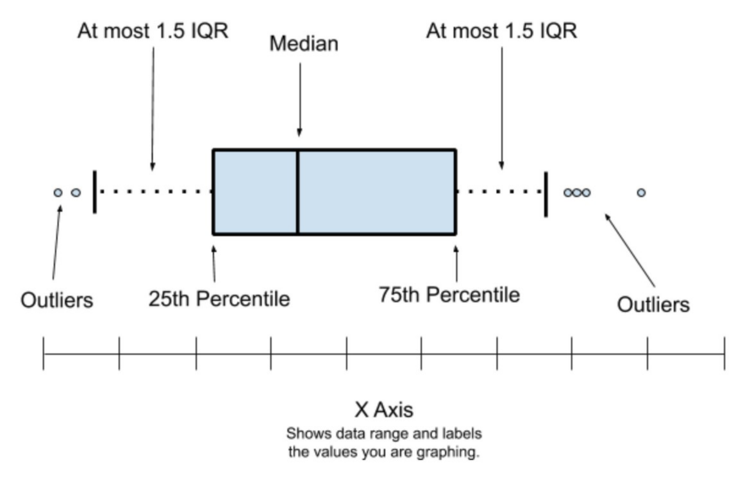


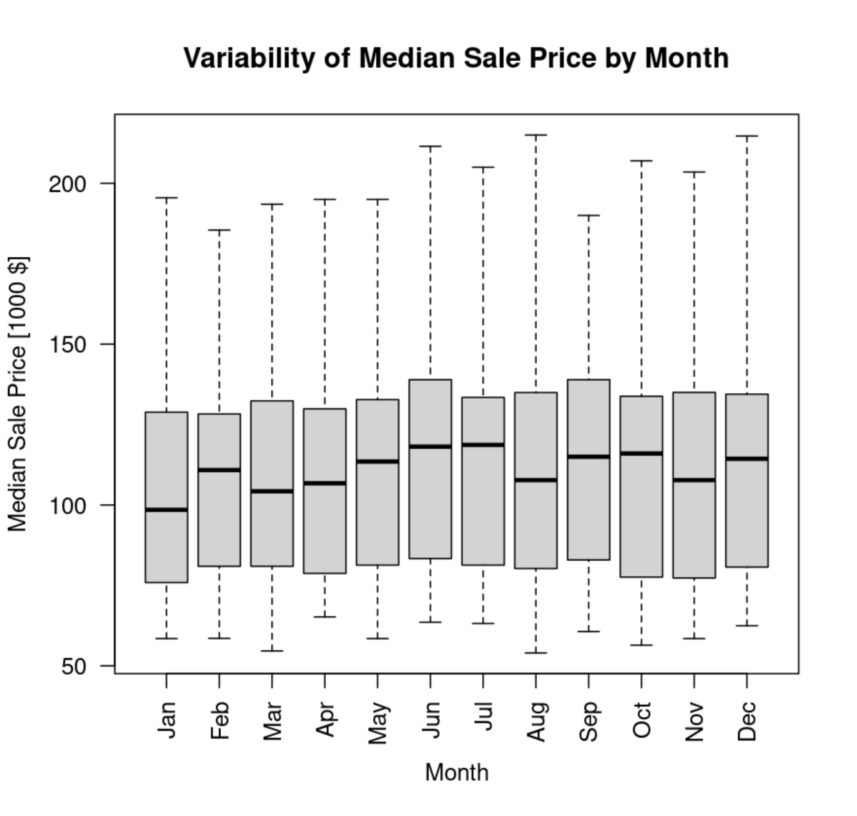
```
# make a pairs plot without Date, Month, Year variables removed
pairs(a[,c(-1,-8,-9)],panel=panel.smooth,
      main="Pairwise Explorations",col='blue')
```

Variability in Median Sale Price Across Months

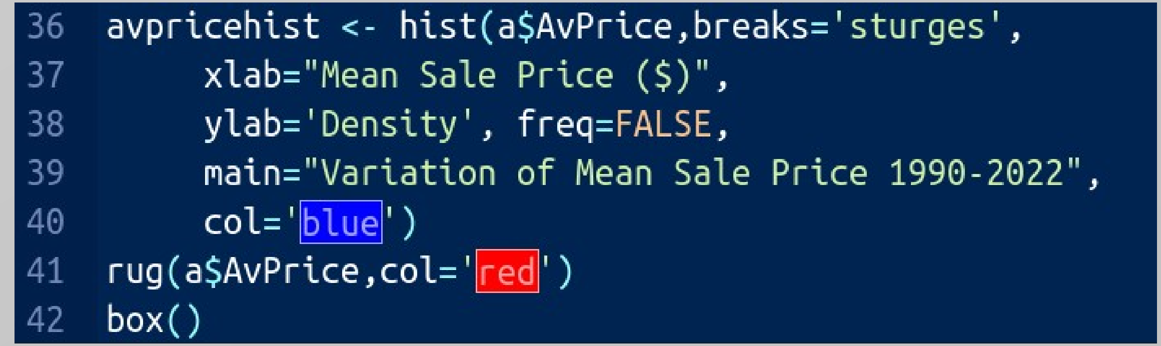


```
# Variability of Median Sales Price across different Months
months <- c("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug",
            "Sep","Oct","Nov","Dec")
par(mgp=c(3,1,0))
boxplot((MedPrice/1000)~Month,data=a,names=months,outline=FALSE,
        ylab="Median Sale Price [1000 $]", las=2,
        main='Variability of Median Sale Price by Month')
```

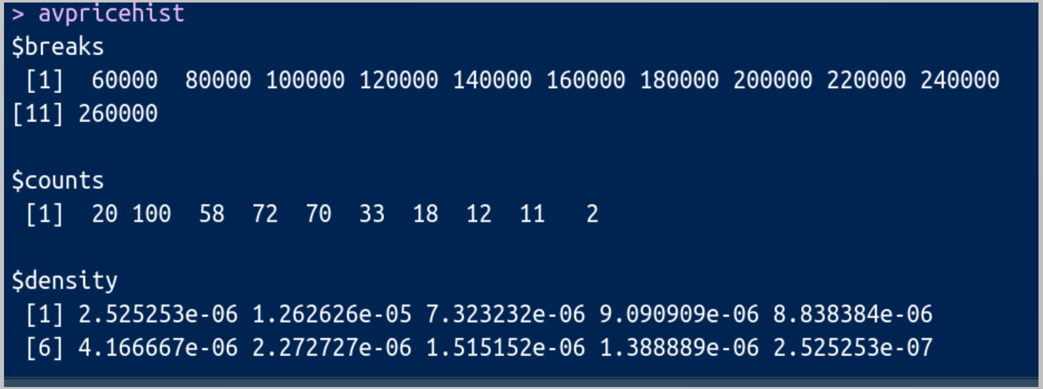




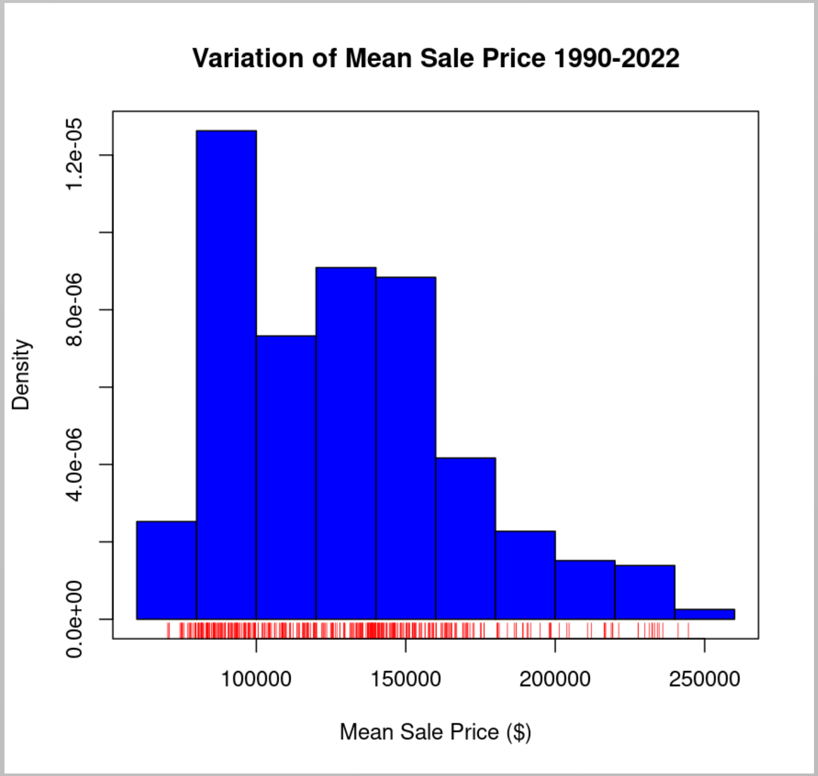
Histogram Plots



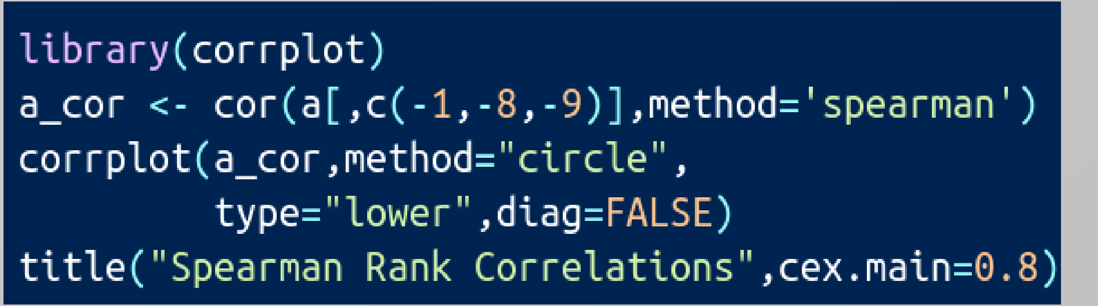
```
avpricehist <- hist(a$AvPrice,breaks='sturges',
          xlab="Mean Sale Price ($)",
          ylab='Density', freq=FALSE,
          main="Variation of Mean Sale Price 1990-2022",
          col='blue')
rug(a$AvPrice,col='red')
box()
```



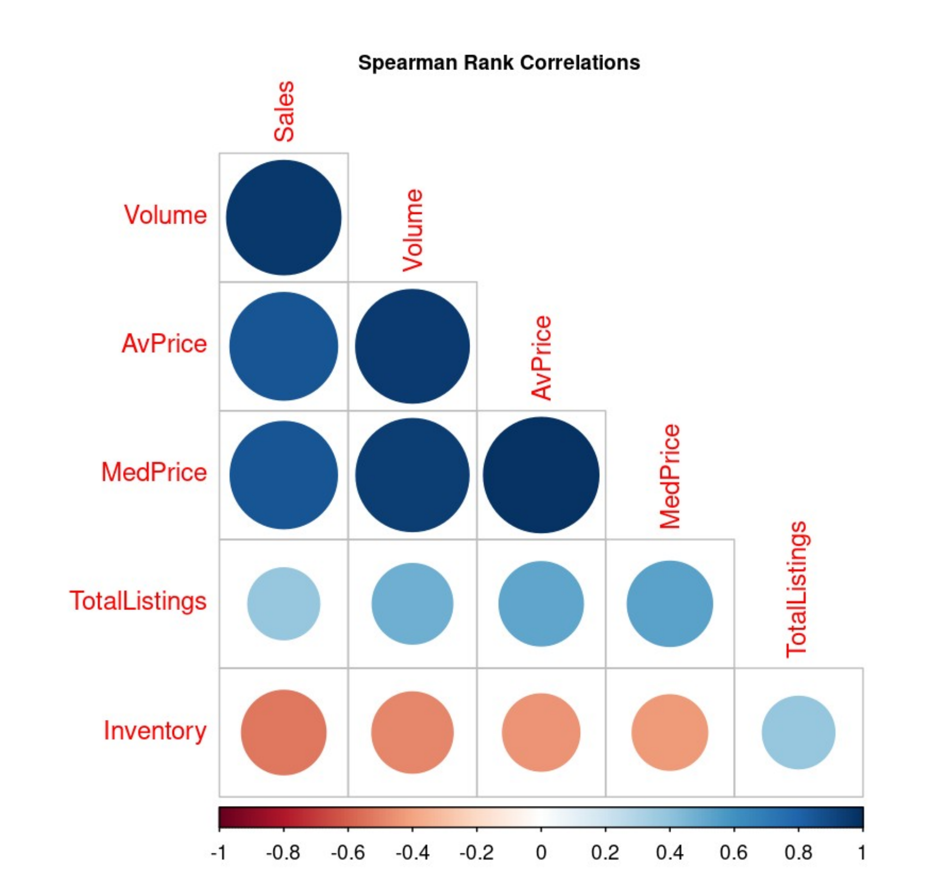
```
> avpricehist
$breaks
 [1]  60000  80000 100000 120000 140000 160000 180000 200000 220000 240000
[11] 260000

$counts
 [1]  20 100  58  72  70  33  18  12  11   2

$density
 [1] 2.525253e-06 1.262626e-05 7.323232e-06 9.090909e-06 8.838384e-06
 [6] 4.166667e-06 2.272727e-06 1.515152e-06 1.388889e-06 2.525253e-07
```



Correlation Among Variables

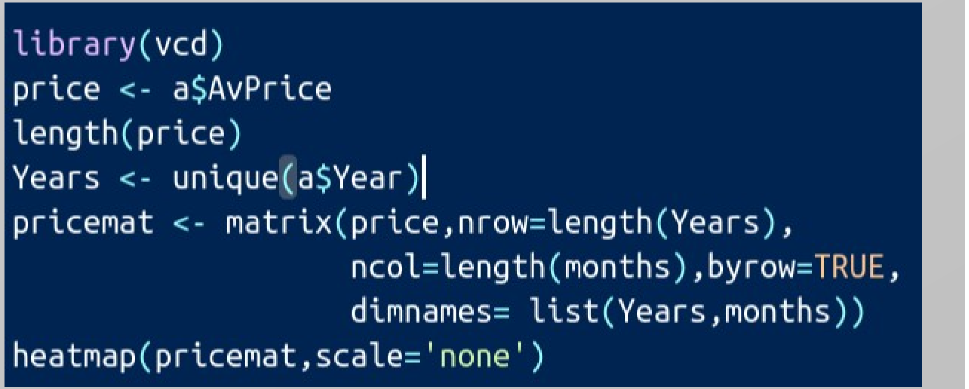


```
library(corrplot)
a_cor <- cor(a[,c(-1,-8,-9)],method='spearman')
corrplot(a_cor,method="circle",
         type="lower",diag=FALSE)
title("Spearman Rank Correlations",cex.main=0.8)
```

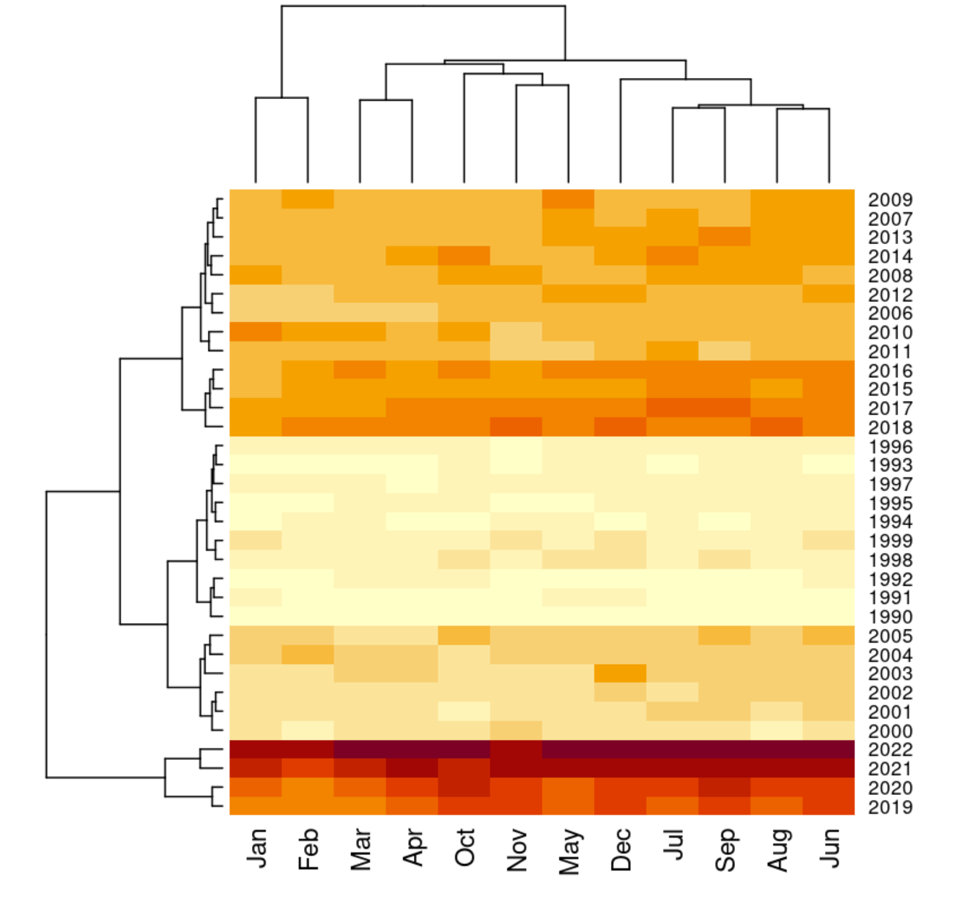


Heatmaps

- Data across two categories
  - Clusters the data



```
library(vcd)
price <- a$AvPrice
length(price)
Years <- unique(a$Year)
pricemat <- matrix(price,nrow=length(Years),
                   ncol=length(months),byrow=TRUE,
                   dimnames= list(Years,months))
heatmap(pricemat,scale='none')
```



What did we Learn

- What is R programming language
  - Its genesis and philosophy
- What is R studio
  - Why use this IDE
- Basic Data Types in R
- How to read a CSV data frame
- How to perform Exploratory Data Analysis

What is coming next

- Programming Concepts in R
- Probabilistic Modeling – Risk Assessment
- How to perform Regression Modeling
- Spatial Analysis using R

Thank You

Workshop Materials – Files, Codes and Datasets will be posted at: https://github.com/vuddameri/RWorkshop