MLE: Lab 1

Andy Ballard

January 13, 2017

Data Structures

R has 5 main data structures that can be defined by a) the number of dimensions (1d, 2d, nd) and whether all items must be of the same type (homogenous) or can be of different types (heterogeneous).

	Homogenous	*Heterogeneous*
1d	Atomic Vector	List
2d	Matrix	Data Frame
nd	Array	

Setting up your workspace

```r
# Start with a clean workspace
rm(list=ls())

#Doing any simulations, randomization, sampling, etc? Set your seed
set.seed(1988)

# Load libraries
library(foreign)
library(xtable)
library(apsrtable)
library(arm)
library(ggplot2)
```

What if we need to install packages and load them? The native R way is pretty simple…

```r
install.packages("package.name.here")
library(package.name.here)
```

…but messy. You need to do this separately for each package. Instead, we can write our own function to 1) check if packages we specify are installed, 2) if not, install them, and 3) load the desired packages. Particularly when we want to load dozens of packages for complicated projects, this is a much easier way to do things.

```r
loadPkg=function(toLoad){
  for(lib in toLoad){
    if(! lib %in% installed.packages()[,1])
    { install.packages(lib, repos='http://cran.rstudio.com/') }
    suppressMessages( library(lib, character.only=TRUE) )
  }
}

toLoad <- c("foreign", "xtable", "apsrtable", "arm", "ggplot2")
loadPkg(toLoad)
```

```
# Other functions that I use frequently
char = function(x){ as.character(x) }
num = function(x){ as.numeric(char(x)) }
detach_package <- function(pkg, character.only = FALSE)
{
  if(!character.only)
  {
    pkg <- deparse(substitute(pkg))
  }
  search_item <- paste("package", pkg, sep = ":")
  while(search_item %in% search())
  {
    detach(search_item, unload = TRUE, character.only = TRUE)
  }
}
```

Fun with R

Monty Hall Monte Carlo

```
doors <- c("1", "2", "3")

correct <- 0
incorrect <- 0

while((correct + incorrect) <= 10000){

prize <- sample(doors, 1)

guess <- sample(doors, 1)


if(guess != prize){
  one.removed <- c(guess, prize)
} else {
  one.removed <- c(guess, sample(doors[doors != guess], 1))
}

final.guess <- one.removed[one.removed != guess]

if(final.guess==prize){
  correct <- correct + 1
} else if (final.guess!=prize){
  incorrect <- incorrect + 1
}

}

correct/10000
```

```
## [1] 0.6679
```

```r
incorrect/10000
```

```
## [1] 0.3322
```