Tutorial 2: Properties of Random Variables

Anh Le

August 25, 2015

Agenda (and learning goals)

1. Implement formulas for Expected Values, Variance, etc. in R

- learn vectorized operation

2. Download data automatically from the web

- learn `help()` in R
- learn reproducible analysis even at the downloading data step

3. Draw the plots you saw from lectures in R (histograms, density plots)

- learn how to generate random sample
- learn how to inspect the distribution of real data

4. Tips and tricks

1. Implement expected value and variance formula

Calculate Expected Value:

Use `sum()` (to get the sum) and `length()` (to get the number of elements in a vector). Calculate:

$$E(X) = \frac{1}{n}\sum_{i=1}^{n} X_i$$

```
X <- rnorm(1000)
sum(X) / length(X)
```

```
## [1] 0.02244788
```

```
mean(X)
```

```
## [1] 0.02244788
```

Calculate Variance:

$$Var(X) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - E(X))^2$$

Let's break down this formula. Mathematically, the formula mean that for each element `X_i` in the vector `X`: - subtract `E(X)` from `X_i`, square the result - then we add up all the results and divide by `n - 1`

So we can naively translate that into code as follows:

```
myVec <- rnorm(1000, mean = 2, sd = 5)

myVar1 <- function(X) {
  n <- length(X)

  sum = 0
  # For each element X_i
  for (i in 1:n) {
    # Subtract E(X), square the result, then add the results together
    sum = sum + (X[i] - mean(X)) ** 2
  }

  return(sum / (n - 1))
}

myVar1(myVec)
```

```
## [1] 25.01291
```

```
var(myVec)
```

```
## [1] 25.01291
```

But loops in R are notoriously slow! We should use vectorized operation instead. For example,

```
X <- 1:5

# To subtract E(X) from each element
X - mean(X)
```

```
## [1] -2 -1  0  1  2
```

```
# To square all elements
X ** 2
```

```
## [1]  1  4  9 16 25
```

```
# To calculate the sum of squares
sum(X ** 2)
```

```
## [1] 55
```

Let's use this to rewrite `myVar1` so that it's faster:

```r
myVar2 <- function(X) {
  return(sum((X - mean(X)) ** 2) / (length(X) - 1))
}

myVar2(myVec)
```

```
## [1] 25.01291
```

```r
myVar1(myVec)
```

```
## [1] 25.01291
```

```r
var(myVec)
```

```
## [1] 25.01291
```

Let's compare the speed:

```r
library(rbenchmark) # install.packages if you don't have the package
benchmark(myVar1(myVec), myVar2(myVec))
```

```
##            test replications elapsed relative user.self sys.self
## 1 myVar1(myVec)          100   0.691      691     0.687    0.001
## 2 myVar2(myVec)          100   0.001        1     0.002    0.000
##   user.child sys.child
## 1          0         0
## 2          0         0
```

In-class exercise: Implement covariance formula

You'll learn about the properties of covariance next week. For now, you can implement the following formula of covariance in R.

$$cov(X,Y) = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \bar{X})(Y_i - \bar{Y})$$

```r
X <- rnorm(100)
Y <- X + rnorm(10)
myCov(X, Y)
```

```
## [1] 0.8882114
```

```r
cov(X, Y)
```

```
## [1] 0.8882114
```

2. Download data automatically from the web

```r
# install.packages("WDI")
library(WDI)
```

```
## Loading required package: RJSONIO
```

```r
help(WDI)
```

Let's download GDP data:

```r
d_gdp <- WDI(country = "all", indicator = "NY.GDP.MKTP.KD",
             extra = TRUE, start = 2010, end = 2011)
head(d_gdp)
```

```
##   iso2c                              country NY.GDP.MKTP.KD year iso3c
## 1    1A                           Arab World   1.563499e+12 2011   ARB
## 2    1A                           Arab World   1.509096e+12 2010   ARB
## 3    1W                                World   5.264624e+13 2010   WLD
## 4    1W                                World   5.414223e+13 2011   WLD
## 5    4E East Asia & Pacific (developing only)   5.330219e+12 2011   EAP
## 6    4E East Asia & Pacific (developing only)   4.914852e+12 2010   EAP
##       region capital longitude latitude     income    lending
## 1 Aggregates                             Aggregates Aggregates
## 2 Aggregates                             Aggregates Aggregates
## 3 Aggregates                             Aggregates Aggregates
## 4 Aggregates                             Aggregates Aggregates
## 5 Aggregates                             Aggregates Aggregates
## 6 Aggregates                             Aggregates Aggregates
```

Note how the dataset includes regions' aggregate data as well. We can exclude those rows as follows:

```r
# Note that the region variable is available because we specified WDI(extra=TRUE)
d_gdp <- d_gdp[d_gdp$region != "Aggregates", ]
head(d_gdp)
```

```
##    iso2c              country NY.GDP.MKTP.KD year iso3c
## 11    AD              Andorra     2693180721 2011   AND
## 12    AD              Andorra     2829050839 2010   AND
## 13    AE United Arab Emirates   213372925637 2011   ARE
## 14    AE United Arab Emirates   203434595050 2010   ARE
## 15    AF          Afghanistan    10243250247 2010   AFG
## 16    AF          Afghanistan    10869490318 2011   AFG
##                                        region          capital
## 11      Europe & Central Asia (all income levels) Andorra la Vella
## 12      Europe & Central Asia (all income levels) Andorra la Vella
## 13 Middle East & North Africa (all income levels)        Abu Dhabi
## 14 Middle East & North Africa (all income levels)        Abu Dhabi
## 15                                     South Asia            Kabul
## 16                                     South Asia            Kabul
##    longitude latitude               income        lending
## 11    1.5218  42.5075 High income: nonOECD Not classified
```

```
## 12    1.5218  42.5075 High income: nonOECD Not classified
## 13   54.3705  24.4764 High income: nonOECD Not classified
## 14   54.3705  24.4764 High income: nonOECD Not classified
## 15   69.1761  34.5228           Low income            IDA
## 16   69.1761  34.5228           Low income            IDA
```

3. Draw the plots you saw from lectures in R (histograms, density plots)

We can generate random samples from various distributions in R, using `rbinom`, `rnorm`, `rpois`, etc.

Binomial distribution:

```r
binomdraws <- rbinom(n=1000, size=100, prob=0.33)
head(binomdraws)
```

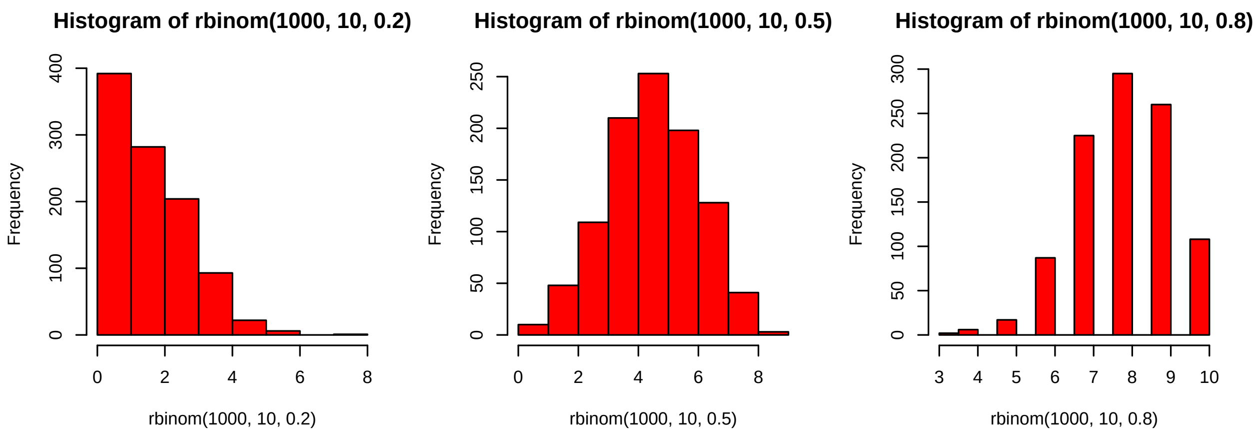
```
## [1] 30 31 28 31 29 29
```

```r
mean(binomdraws)
```

```
## [1] 33.169
```

In-class exercise: Replicate binomial histogram in your lecture slides

```r
par(mfrow=c(1, 3))
hist(rbinom(1000, 10, 0.2), col = 'red')
hist(rbinom(1000, 10, 0.5), col = 'red')
hist(rbinom(1000, 10, 0.8), col = 'red')
```




```r
par(mfrow=c(1, 1))
```

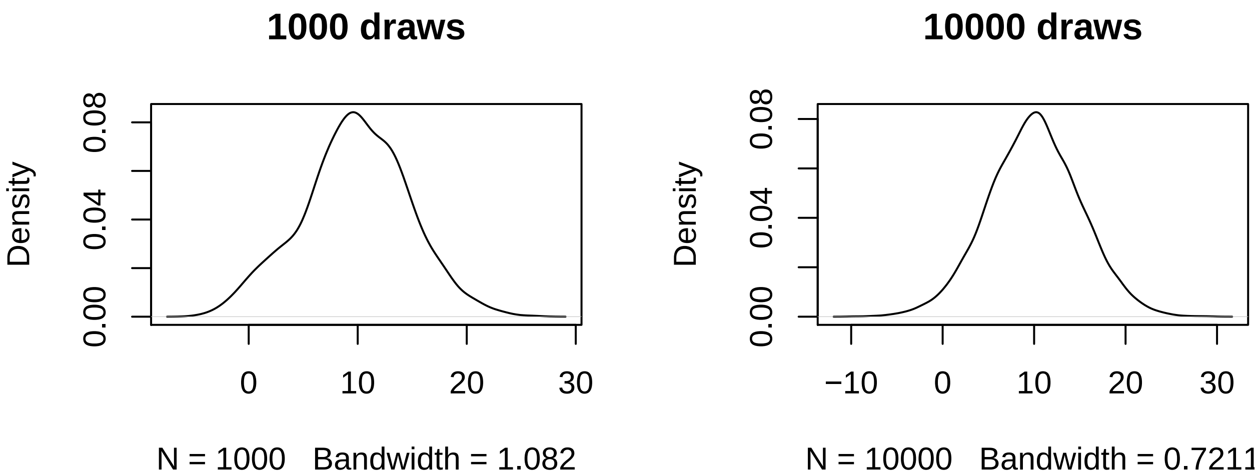
Normal (Gaussian) distribution:

Draw normal samples

```r
normdraws <- rnorm(n = 1000, mean = 10, sd = 5)
```

Plot the density. With 1000 draws, the density does not look exactly normal. With 10000 draws, it looks much closer to "textbook" normal.

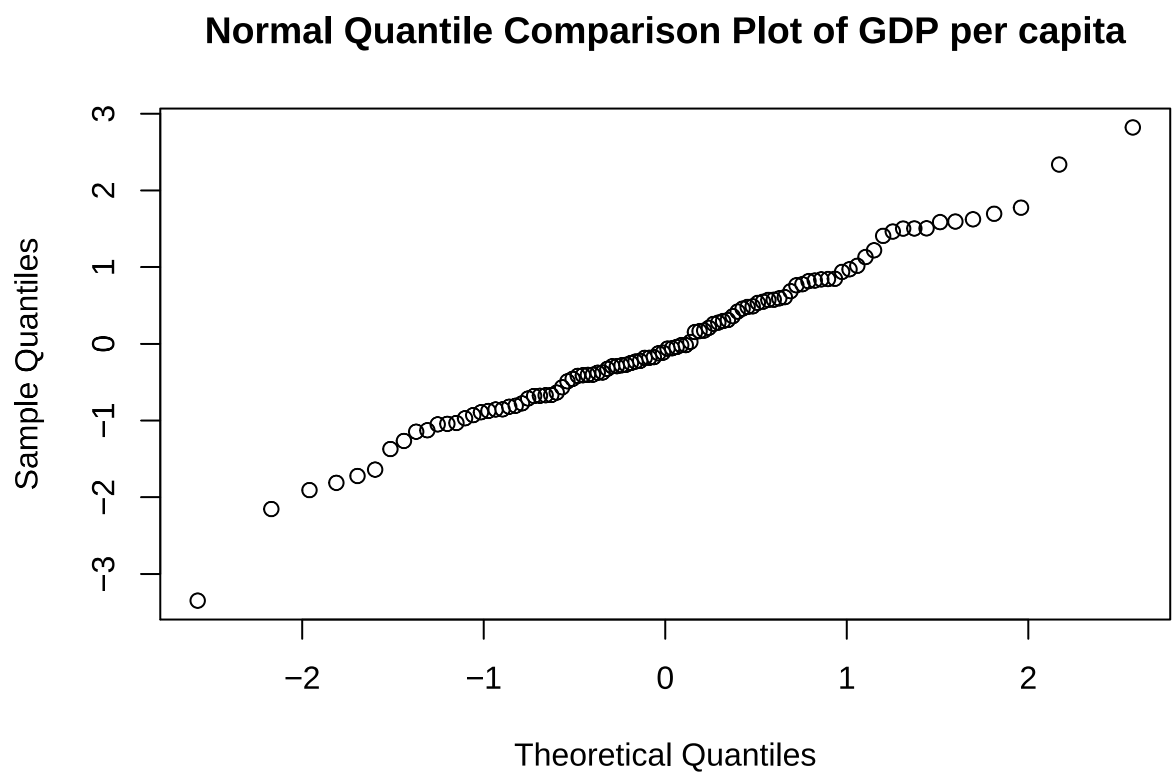
```r
par(mfrow = c(1, 2))
normdensity <- density(normdraws)
plot(normdensity, main="1000 draws")

plot(density(rnorm(n = 10000, mean = 10, sd = 5)), main = "10000 draws")
```




```r
par(mfrow = c(1, 1))
```

Another way to check whether a variable is normally distributed is the "normal quantile comparison plot".

```r
qqnorm(rnorm(100), main="Normal Quantile Comparison Plot of GDP per capita")
```




```r
# qqline(happy$gdp2002)
```

Box plot

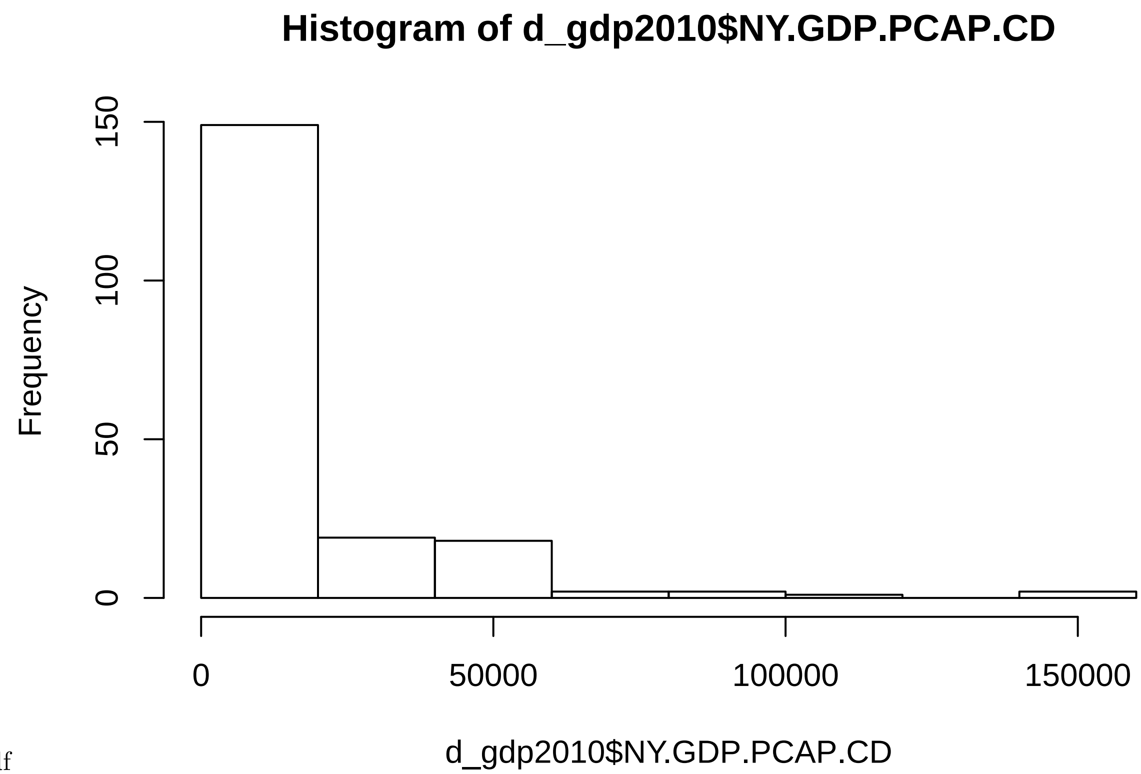
library(car) row.names(happy)<-happy$country_name Boxplot(~ gdp2002, data=happy, main="Box Plot of 2002 GDP per capita", ylab="GDP per capita")

scatter plot

plot(happygdp2002, $happy$happins, main="Scatter Plot of Happiness and GDP", ylab="Happiness Index", xlab="GDP per capita")

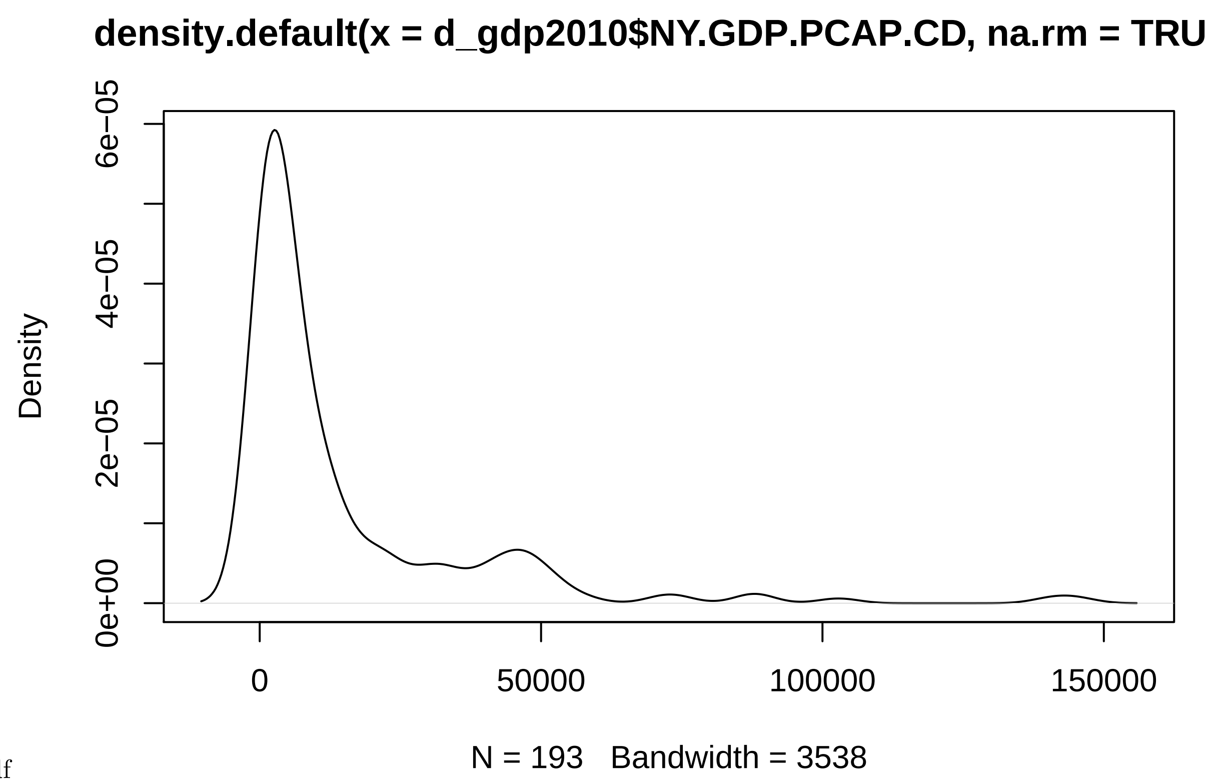
In-class exercise: Plot the histogram and density of all countries' GDP per capita in 2010

```r
d_gdp2010 <- WDI(country = "all", indicator = "NY.GDP.PCAP.CD",
                 start = 2010, end = 2010, extra = TRUE)
d_gdp2010 <- d_gdp2010[d_gdp2010$region != "Aggregates", ]
hist(d_gdp2010$NY.GDP.PCAP.CD)
```



GDP histogram-1.pdf

```r
plot(density(d_gdp2010$NY.GDP.PCAP.CD, na.rm = TRUE))
```

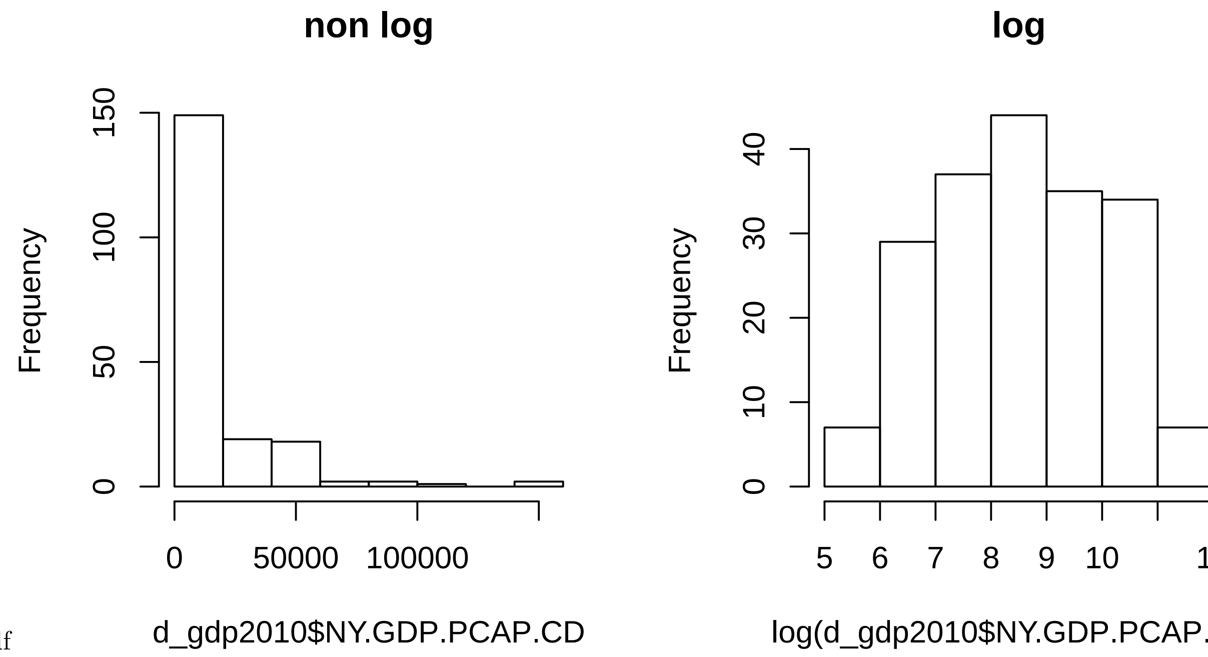


GDP histogram-2.pdf

The distribution of GDP is has a long right tail. This is because a country's GDP can go very high but cannot go lower than 0 (this phenomenon is called "left-censored").

Because of this, GDP is NOT normally distributed, and can misbehave in models that assume normality. A common way to deal with this is to take the `log(GDP)` instead.

```
par(mfrow=c(1, 2))
hist(d_gdp2010$NY.GDP.PCAP.CD, main="non log")
hist(log(d_gdp2010$NY.GDP.PCAP.CD), main="log")
```



GDP histogram-1.pdf

```
par(mfrow=c(1, 1))
```

4. Tips and tricks

1. You can name your knitr chunk
2. You can divide your R code into sections