Lecture 8 – Data Analytics with R

Dr. Jing Li
Department of Computing
The Hong Kong Polytechnic University

15&17 Mar 2022

Looking Back (Lecture 7)

- Install and attach *ggplot2* package
- Barplot, histogram, and scatterplots
- Analyzing statistics of *Big Mart Sales Datasets*

Roadmap

- Simulations
  - Generate Random Numbers
  - Random Number Seeds
  - Simulating a Linear Model
  - Random Sampling
- A case study of changes in PM 2.5 in the U.S.
  - Data
  - Results

Roadmap

- **<u>Simulations</u>**
  - Generate Random Numbers
  - Random Number Seeds
  - Simulating a Linear Model
  - Random Sampling
- A case study of changes in PM 2.5 in the U.S.
  - Data
  - Results

Why Simulation?

- A ***simulation*** is an *approximate imitation* of the operation of a process or system
- Why we do simulations?
  - To estimate the parameters for statistical models (i.e., probability distributions).
  - Performance tuning or optimizing.
  - Test out a hypothesis or statistical method.

Roadmap

- Simulations
  - **Generate Random Numbers**
  - Random Number Seeds
  - Simulating a Linear Model
  - Random Sampling
- A case study of changes in PM 2.5 in the U.S.
  - Data
  - Results

Generate Random Numbers

Why?

- R comes with a set of *pseudo-random* number generators that allow you to simulate from well-known probability distributions.
  - *rnorm*: *generate random* Normal variates with a given mean and standard deviation
  - *dnorm*: evaluate the Normal probability density (with a given mean/SD) at a point (or vector of points)
  - *pnorm*: evaluate the cumulative distribution function for a Normal distribution

r: random number generation

p: cumulation distribution

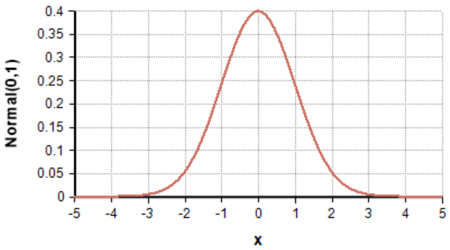
d: density

```
> pnorm(2)
[1] 0.9772499
```

The probability of a random standard Normal variable of being less than 2

Example: Generate Normal Random Numbers

- Generate *standard* Normal random numbers




```
> ## Simulate standard Normal random numbers
> x <- rnorm(10)
> x
 [1]  0.01874617 -0.18425254 -1.37133055 -0.59916772  0.29454513
 [6]  0.38979430 -1.20807618 -0.36367602 -1.62667268 -0.25647839
```

- Generate random numbers from $N(20, 2^2)$

```
> x <- rnorm(10, 20, 2)
> x
 [1] 22.20356 21.51156 19.52353 21.97489 21.48278 20.17869 18.09011
 [8] 19.60970 21.85104 20.96596
> summary(x)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  18.09   19.75   21.22   20.74   21.77   22.20
```

Roadmap

- Simulations
  - Generate Random Numbers
  - **<u>Random Number Seeds</u>**
  - Simulating a Linear Model
  - Random Sampling
- A case study of changes in PM 2.5 in the U.S.
  - Data
  - Results

Random Number Seed

- A *random seed* is a number used to initialize a pseudorandom number generator (a starting point).
- Ensure reproducibility of the sequence of random numbers.
- Setting the random number seed with *set.seed()*

```
> set.seed(1)
> rnorm(5)
[1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
```

- *What if you call rnorm(5) again?*

Random Number Seed

- Setting the random number seed with *set.seed()*

```
> set.seed(1)
> rnorm(5)
[1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
```

- *What if you call rnorm(5) again?*

```
> rnorm(5)
[1] -0.8204684  0.4874291  0.7383247  0.5757814 -0.3053884
```

- *Reset the seed with set.seed(1).*

```
> set.seed(1)
> rnorm(5)     ## Same as before
[1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
```

Roadmap

- Simulations
  - Generate Random Numbers
  - Random Number Seeds
  - **Simulating a Linear Model**
  - Random Sampling
- A case study of changes in PM 2.5 in the U.S.
  - Data
  - Results

Simulating a Linear Model

- Suppose we want to simulate from the following linear model
  - $y = \beta_0 + \beta_1 x + \epsilon$
  - $\epsilon \sim N(0,2^2)$
- Assume $x \sim N(0,1^2)$, $\beta_0 = 0.5$ and $\beta_1 = 2$
- The variable x might represent an important predictor of the outcome y. But how?

Simulating a Linear Model

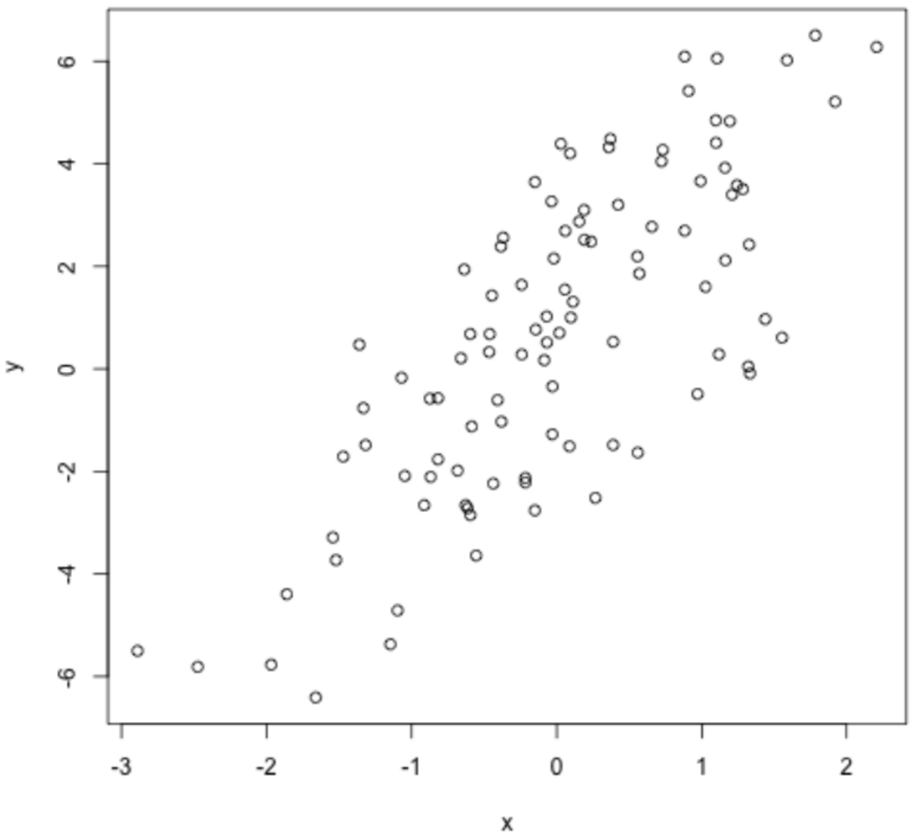
- $y = \beta_0 + \beta_1 x + \epsilon$ ($\beta_0 = 0.5$ and $\beta_1 = 2$)
  - $\epsilon \sim N(0, 2^2)$
  - $x \sim N(0, 1^2)$
-

```
> ## Always set your seed!
> set.seed(20)
>
> ## Simulate predictor variable
> x <- rnorm(100)
>
> ## Simulate the error term
> e <- rnorm(100, 0, 2)
>
> ## Compute the outcome via the model
> y <- 0.5 + 2 * x + e
> summary(y)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-6.4080 -1.5400  0.6789  0.6893  2.9300  6.5050
```

Draw out a scatterplot with *plot(x,y)*

Simulating a Linear Model

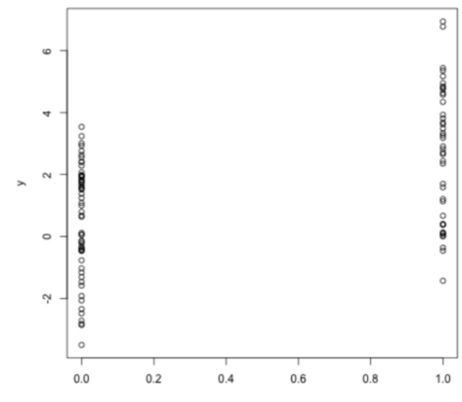
- $y = \beta_0 + \beta_1 x + \epsilon$ ($\beta_0 = 0.5$ and $\beta_1 = 2$)
  - $\epsilon \sim N(0,2^2)$
  - $x \sim N(0,1^2)$
-



Simulating a Linear Model

- What if we wanted to simulate a predictor variable x that is binary?
- We can use the rbinom() function to simulate binary random variables.

```
> set.seed(10)
> x <- rbinom(100, 1, 0.5)
> str(x)      ## 'x' is now 0s and 1s
 int [1:100] 1 0 0 1 0 0 0 0 1 0 ...
```



- Proceed the rest

```
> e <- rnorm(100, 0, 2)
> y <- 0.5 + 2 * x + e
> plot(x, y)
```

Roadmap

- Simulations
    - Generate Random Numbers
    - Random Number Seeds
    - Simulating a Linear Model
    - **Random Sampling**
- A case study of changes in PM 2.5 in the U.S.
    - Data
    - Results

Random Sampling

- The *sample()* function draws randomly from a specified set of (scalar) objects allowing you to sample from arbitrary distributions of numbers.

```
> set.seed(1)
> sample(1:10, 4)
[1] 3 4 5 7
> sample(1:10, 4)
[1] 3 9 8 5
>
```

Sample Numbers

```
> ## Doesn't have to be numbers
> sample(letters, 5)
[1] "q" "b" "e" "x" "p"
>
```

Sample Letters

Random Permutation

```
> ## Do a random permutation
> sample(1:10)
 [1]  4  7 10  6  9  2  8  3  1  5
> sample(1:10)
 [1]  2  3  4  1  9  5 10  8  6  7
>
```

Sample with Replacement

```
> ## Sample w/replacement
> sample(1:10, replace = TRUE)
 [1] 2 9 7 8 2 8 5 9 7 8
```

Random Sampling

- To sample more complicated things, such as rows from a data frame or a list, you can sample the indices into an object rather than the elements of the object itself.

```
> library(datasets)
> data(airquality)
> head(airquality)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
```

Sample rows from a data frame?

Random Sampling

- Create the index vector indexing the rows of the data frame and sample directly from that index vector.

```
> set.seed(20)
>
```

Always specify the seeds

```
> ## Create index vector
> idx <- seq_len(nrow(airquality))
>
```

A vector from 1 to 153 (the record number)

```
> ## Sample from the index vector
> samp <- sample(idx, 6)
```

Generate 6 random numbers

```
> airquality[samp, ]
    Ozone Solar.R Wind Temp Month Day
135    21     259 15.5   76     9  12
117   168     238  3.4   81     8  25
43     NA     250  9.2   92     6  12
80     79     187  5.1   87     7  19
144    13     238 12.6   64     9  21
146    36     139 10.3   81     9  23
```

Sample 6 rows according to the random numbers generated

Roadmap

- **Simulations**
  - **Generate Random Numbers**
  - **Random Number Seeds**
  - **Simulating a Linear Model**
  - **Random Sampling**
- A case study of changes in PM 2.5 in the U.S.
  - Data
  - Results

Highlights for Simulation

- Drawing samples from specific probability distributions can be done with *“r” functions*
- Standard distributions are built in: *Normal*, *Binomial*, etc.
- The *sample()* function can be used to draw random samples from arbitrary vectors.
- Setting the random number generator seed via *set.seed()* is critical for *reproducibility*.

Roadmap

- Simulations
  - Generate Random Numbers
  - Random Number Seeds
  - Simulating a Linear Model
  - Random Sampling
- **A case study of changes in PM 2.5 in the U.S.**
  - Data
  - Results

Case Study: Background

- Analyze changes in PM2.5 outdoor air pollution in the United States between the years 1999 and 2012.
- Obtained PM 2.5 data from the U.S. Environmental Protection Agency (EPA), which is collected from monitors sited across the U.S. (Uploaded to BB).

```
# RD|Action Code|State Code|County Code|Site ID|Parameter|POC|Sample Duration|Unit|Method|Date|Start Time|Sample Value|Null Data Code|Sampling Frequency|Monitor Protocol (MP) ID|Qualifier - 1|Qualifier - 2|Qualifier - 3|Qualifier - 4|Qualifier - 5|Qualifier - 6|Qualifier - 7|Qualifier - 8|Qualifier - 9|Qualifier - 10|Alternate Method Detectable Limit|Uncertainty
# RC|Action Code|State Code|County Code|Site ID|Parameter|POC|Unit|Method|Year|Period|Number of Samples|Composite Type|Sample Value|Monitor Protocol (MP) ID|Qualifier - 1|Qualifier - 2|Qualifier - 3|Qualifier - 4|Qualifier - 5|Qualifier - 6|Qualifier - 7|Qualifier - 8|Qualifier - 9|Qualifier - 10|Alternate Method Detectable Limit|Uncertainty
RD|I|01|003|0010|88101|1|7|105|118|20120101|00:00|6.7||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120104|00:00|9||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120107|00:00|6.5||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120110|00:00|7||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120113|00:00|5.8||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120116|00:00|8||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120119|00:00|7.9||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120122|00:00|8||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120125|00:00|6||3|||||||||||||
RD|I|01|003|0010|88101|1|7|105|118|20120128|00:00|9.6||3|||||||||||||
```

Roadmap

- Simulations
  - Generate Random Numbers
  - Random Number Seeds
  - Simulating a Linear Model
  - Random Sampling
- A case study of changes in PM 2.5 in the U.S.
  - **Data**
  - Results

Case Study: Data

- ***Reading in the 1999 data***
  - Fields are delimited with the | character
  - Missing values are coded as blank fields.
  - Skip some commented lines in the beginning of the file
  - Do not read the header data.

```
> pm0 <- read.table("pm25_data/RD_501_88101_1999-0.txt", comment.char = "#", hea\
der = FALSE, sep = "|", na.strings = "")
```

Case Study: Data

- Check the number of records and the attributes.

```
> dim(pm0)
[1] 117421     28
```

- Examine the first few rows.

```
> head(pm0[, 1:13])
  V1 V2 V3 V4 V5    V6 V7 V8  V9 V10      V11   V12    V13
1 RD  I  1 27  1 88101  1  7 105 120 19990103 00:00     NA
2 RD  I  1 27  1 88101  1  7 105 120 19990106 00:00     NA
3 RD  I  1 27  1 88101  1  7 105 120 19990109 00:00     NA
4 RD  I  1 27  1 88101  1  7 105 120 19990112 00:00  8.841
5 RD  I  1 27  1 88101  1  7 105 120 19990115 00:00 14.920
6 RD  I  1 27  1 88101  1  7 105 120 19990118 00:00  3.878
```

Case Study: Data

- Attach the column headers to the dataset and make sure that they are properly formatted for R data frames.

```
> cnames <- readLines("pm25_data/RD_501_88101_1999-0.txt", 1)
> cnames <- strsplit(cnames, "|", fixed = TRUE)
> ## Ensure names are properly formatted
> names(pm0) <- make.names(cnames[[1]])
> head(pm0[, 1:13])
  X..RD Action.Code State.Code County.Code Site.ID Parameter POC
1    RD           I          1          27       1     88101   1
2    RD           I          1          27       1     88101   1
3    RD           I          1          27       1     88101   1
4    RD           I          1          27       1     88101   1
5    RD           I          1          27       1     88101   1
6    RD           I          1          27       1     88101   1
  Sample.Duration Unit Method     Date Start.Time Sample.Value
1               7  105    120 19990103      00:00           NA
2               7  105    120 19990106      00:00           NA
3               7  105    120 19990109      00:00           NA
4               7  105    120 19990112      00:00        8.841
5               7  105    120 19990115      00:00       14.920
6               7  105    120 19990118      00:00        3.878
```

Case Study: Data

- The column we are interested in is the *Sample.Value* column, which contains the PM 2.5 measurements.

Missing values are a common problem with environmental data. What matters here is the proportion of the observations are missing.

```
> x0 <- pm0$Sample.Value
> summary(x0)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   0.00    7.20   11.50   13.74   17.90  157.10   13217
```

Are the missing values important here?

```
> mean(is.na(x0))
[1] 0.1125608
```

Case Study: Data

- ***Reading in the 2012 data.***

```
> pm1 <- read.table("pm25_data/RD_501_88101_2012-0.txt", comment.char = "#",
+                   header = FALSE, sep = "|", na.strings = "", nrow = 1304290)
```

Why?

Much more data records than 1999

- We also set the column names (the same as the 1999 dataset) and extract the *Sample.Value* column from this dataset.

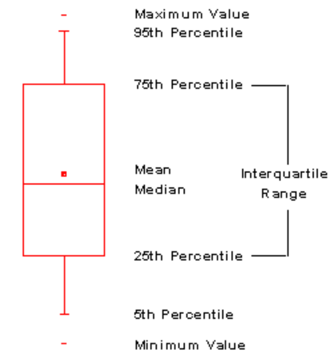
```
> names(pm1) <- make.names(cnames[[1]])
> x1 <- pm1$Sample.Value
```

Roadmap

- Simulations
  - Generate Random Numbers
  - Random Number Seeds
  - Simulating a Linear Model
  - Random Sampling
- A case study of changes in PM 2.5 in the U.S.
  - Data
  - **Results**

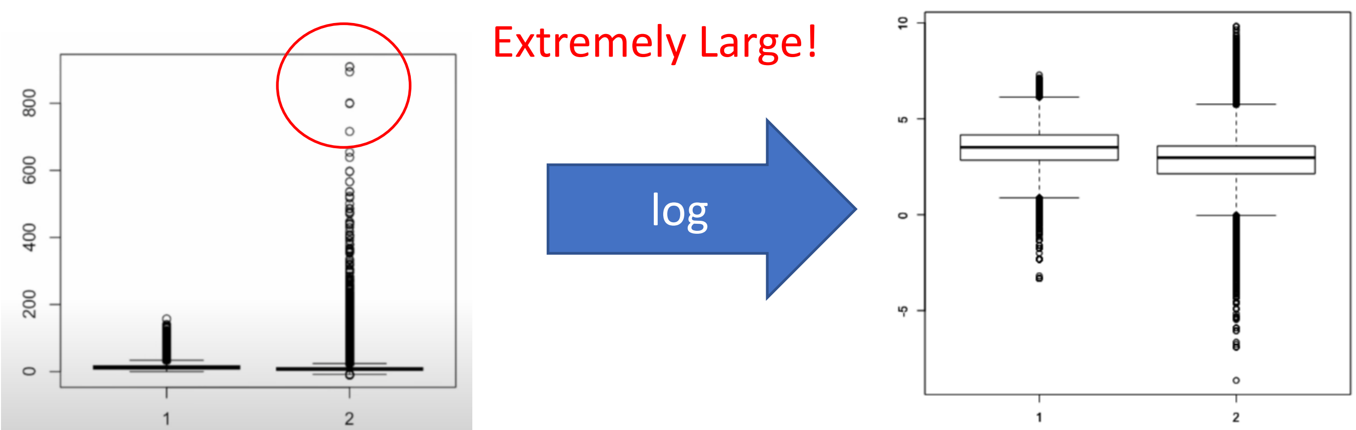
Results: Entire U.S. analysis

- Show aggregate changes in PM across the entire monitoring network.
- Make *boxplots* of all monitor values in 1999 and 2012




```
> boxplot(log2(x0), log2(x1))
```

We take the log of the PM values to adjust for the skew in the data



Results: Entire U.S. analysis

- Show aggregate changes in PM across the entire monitoring network.
- Make boxplots of all monitor values in 1999 and 2012

```
> summary(x0)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   0.00    7.20   11.50   13.74   17.90  157.10   13217
> summary(x1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
 -10.00    4.00    7.63    9.14   12.00  909.00   73133
```

Strange: Negative Values!

What proportion?

```
> negative <- x1 < 0
> mean(negative, na.rm = T)
[1] 0.0215034
```

Results: Entire U.S. analysis

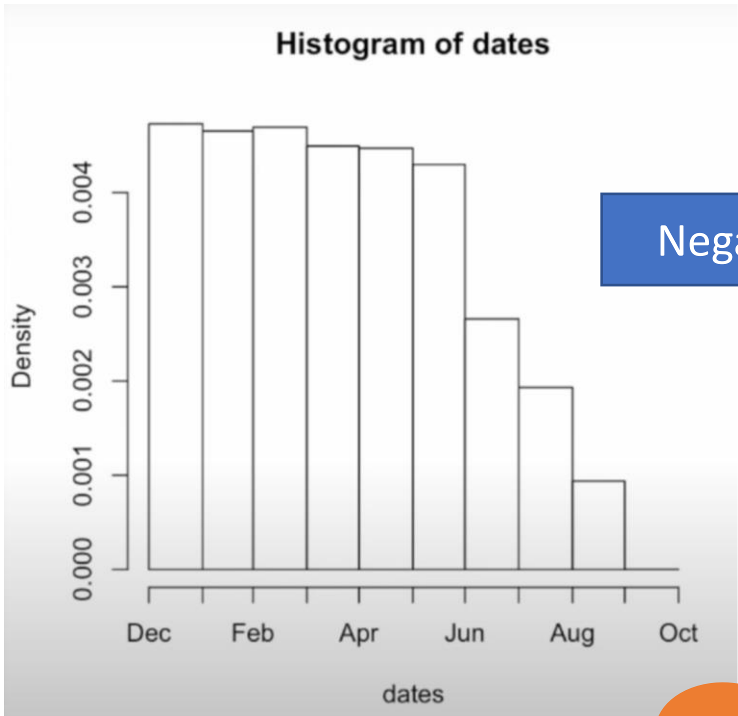
- Extract the date of each measurement from the original data frame.
- The idea here is that negative values may occur more often in some parts of the year.
- The original data are formatted as *character strings* so we convert them to *R's Date format*.

```
> dates <- pm1$Date
> dates <- as.Date(as.character(dates), "%Y%m%d")
```

```
RD|I|01|027|0001|88101|1|7|105|120|19990103|00:00||AS|3||||||||||||
RD|I|01|027|0001|88101|1|7|105|120|19990106|00:00||AS|3||||||||||||
RD|I|01|027|0001|88101|1|7|105|120|19990109|00:00||AS|3||||||||||||
```

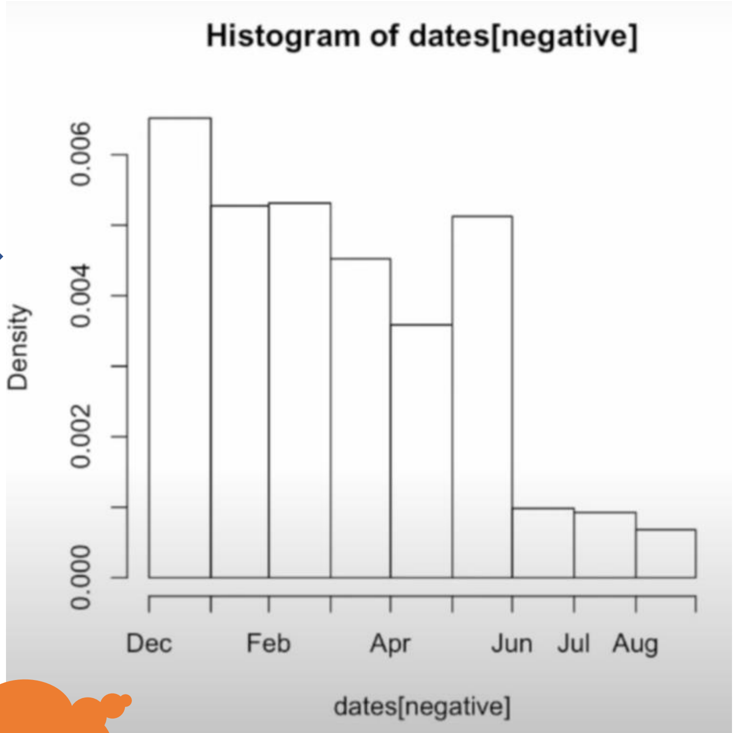
Case Study: Results

> hist(dates, “month”)



Negative

> hist(dates[negative], “month”)



Less often
in summer?

Results: An Individual Monitor

- One issue with the previous analysis is that the *monitoring network* could have changed in the time period between 1999 and 2012.
  - For example, if more monitors concentrated in cleaner parts in 2012, then it might appear the PM levels decreased
  - Focus on a single monitor in New York State to see if PM levels at that monitor decreased from 1999 to 2012.

```
> site0 <- unique(subset(pm0, State.Code == 36, c(County.Code, Site.ID)))
> site1 <- unique(subset(pm1, State.Code == 36, c(County.Code, Site.ID)))
```

Data from New York State

Only county code and site ID considered

Results: An Individual Monitor

- Focus on a single monitor in New York State to see if PM levels at that monitor decreased from 1999 to 2012.

```
> site0 <- unique(subset(pm0, State.Code == 36, c(County.Code, Site.ID)))
> site1 <- unique(subset(pm1, State.Code == 36, c(County.Code, Site.ID)))
```

Create a new variable that combines the county code and the site ID into a single string

```
> site0 <- paste(site0[,1], site0[,2], sep = ".")
> site1 <- paste(site1[,1], site1[,2], sep = ".")
> str(site0)
 chr [1:33] "1.5" "1.12" "5.73" "5.80" "5.83" "5.110" ...
> str(site1)
 chr [1:18] "1.5" "1.12" "5.80" "5.133" "13.11" "29.5" ...
```

Results: An Individual Monitor

- Focus on a single monitor in New York State to see if PM levels at that monitor decreased from 1999 to 2012.

```
> site0 <- paste(site0[,1], site0[,2], sep = ".")
> site1 <- paste(site1[,1], site1[,2], sep = ".")
> str(site0)
 chr [1:33] "1.5" "1.12" "5.73" "5.80" "5.83" "5.110" ...
> str(site1)
 chr [1:18] "1.5" "1.12" "5.80" "5.133" "13.11" "29.5" ...
```

- Find out the intersection between the sites present in 1999 and 2012
- So, we might choose a monitor that has data in both periods.

10 found! How to choose?

```
> both <- intersect(site0, site1)
> print(both)
 [1] "1.5"      "1.12"     "5.80"     "13.11"    "29.5"     "31.3"     "63.2008"
 [8] "67.1015"  "85.55"    "101.3"
```

Results: An Individual Monitor

- Choose one that had a reasonable amount of data in each year.

```
> ## Find how many observations available at each monitor
> pm0$county.site <- with(pm0, paste(County.Code, Site.ID, sep = "."))
> pm1$county.site <- with(pm1, paste(County.Code, Site.ID, sep = "."))
> cnt0 <- subset(pm0, State.Code == 36 & county.site %in% both)
> cnt1 <- subset(pm1, State.Code == 36 & county.site %in% both)
```

Create a new attribute with *county.site*

Extract a subset with the records in New York and from the monitors that overlap between 1999 and 2012

Results: An Individual Monitor

- Choose one that had a reasonable amount of data in each year.

Data frame containing values to be divided into groups.

```
> ## 1999
> sapply(split(cnt0, cnt0$county.site), nrow)
   1.12      1.5    101.3    13.11     29.5     31.3     5.80  63.2008 67.1015 
     61      122      152       61       61      183       61      122     122 
  85.55 
      7 
> ## 2012
> sapply(split(cnt1, cnt1$county.site), nrow)
   1.12      1.5    101.3    13.11     29.5     31.3     5.80  63.2008 67.1015 
     31       64       31       31       33       15       31       30      31 
  85.55 
     31 
```

Pick Up!

Results: An Individual Monitor

- Choose one that had a reasonable amount of data in each year.
- Pick up the records from New York (*State.Code==36*) and *county.ID=63, site.ID=2008.*

```
> both.county <- 63
> both.id <- 2008
>
> ## Choose county 63 and side ID 2008
> pm1sub <- subset(pm1, State.Code == 36 & County.Code == both.county & Site.ID \
== both.id)
> pm0sub <- subset(pm0, State.Code == 36 & County.Code == both.county & Site.ID \
== both.id)
```

Results: An Individual Monitor

- Plot the time series data of PM for the monitor in both years
  - X-axis: dates; Y-axis: Sample Values

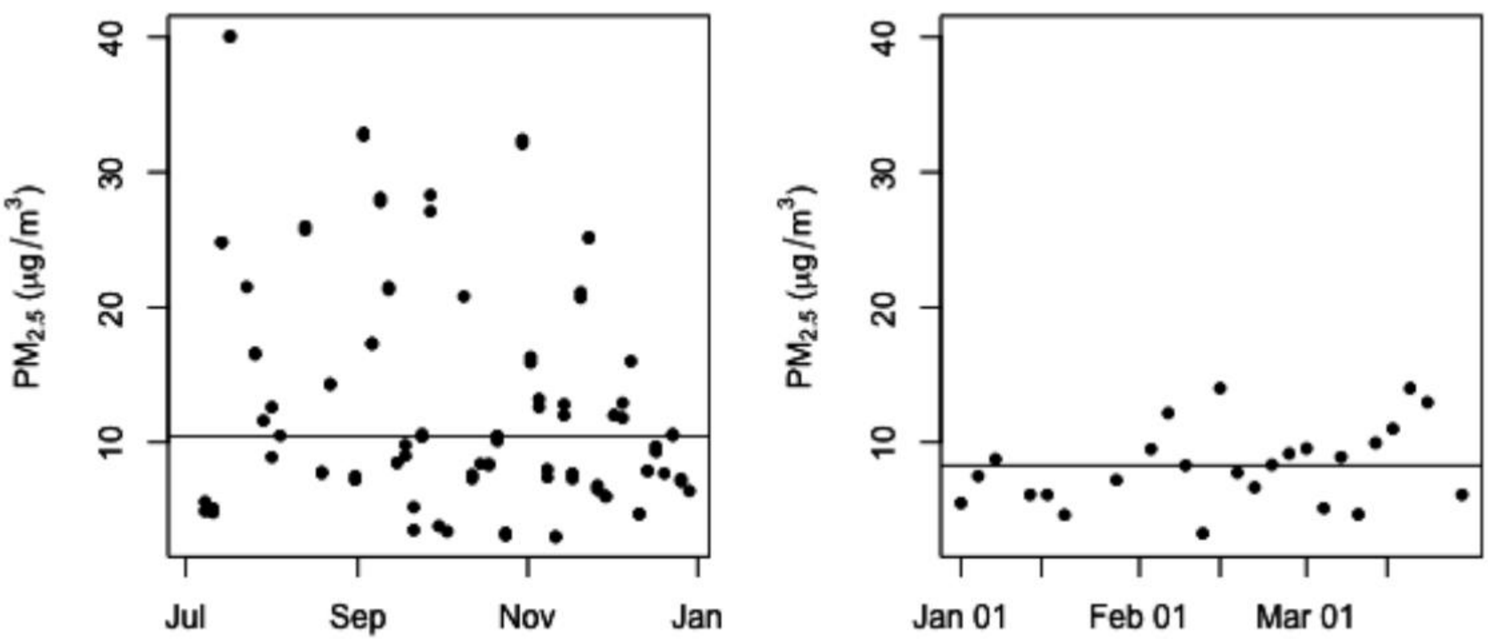
```
> dates1 <- as.Date(as.character(pm1sub$Date), "%Y%m%d")
> x1sub <- pm1sub$Sample.Value
> dates0 <- as.Date(as.character(pm0sub$Date), "%Y%m%d")
> x0sub <- pm0sub$Sample.Value
>
> ## Find global range
> rng <- range(x0sub, x1sub, na.rm = T)
> par(mfrow = c(1, 2), mar = c(4, 5, 2, 1))
> plot(dates0, x0sub, pch = 20, ylim = rng, xlab = "", ylab = expression(PM[2.5]\
 * " (" * mu * g/m^3 * ")"))
> abline(h = median(x0sub, na.rm = T))
> plot(dates1, x1sub, pch = 20, ylim = rng, xlab = "", ylab = expression(PM[2.5]\
 * " (" * mu * g/m^3 * ")"))
> abline(h = median(x1sub, na.rm = T))
```

Set both the scatter plots with the same y range for comparison

Draw the median line

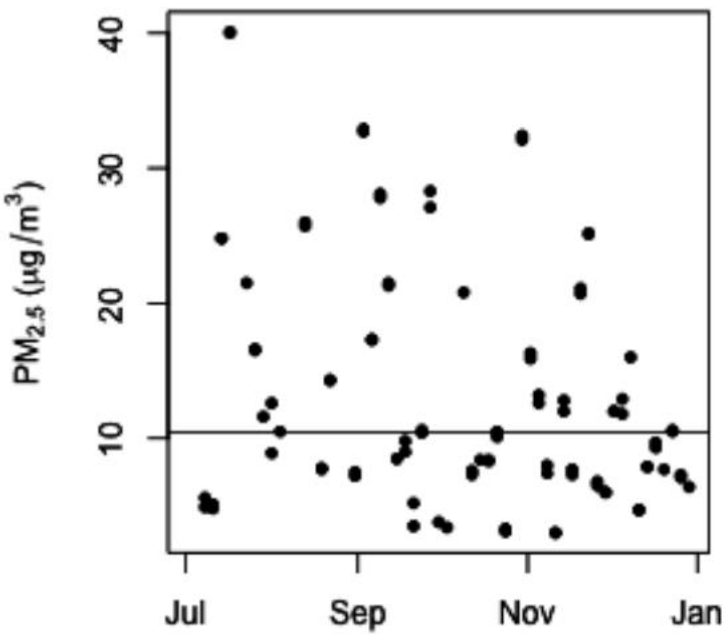
Results: An Individual Monitor

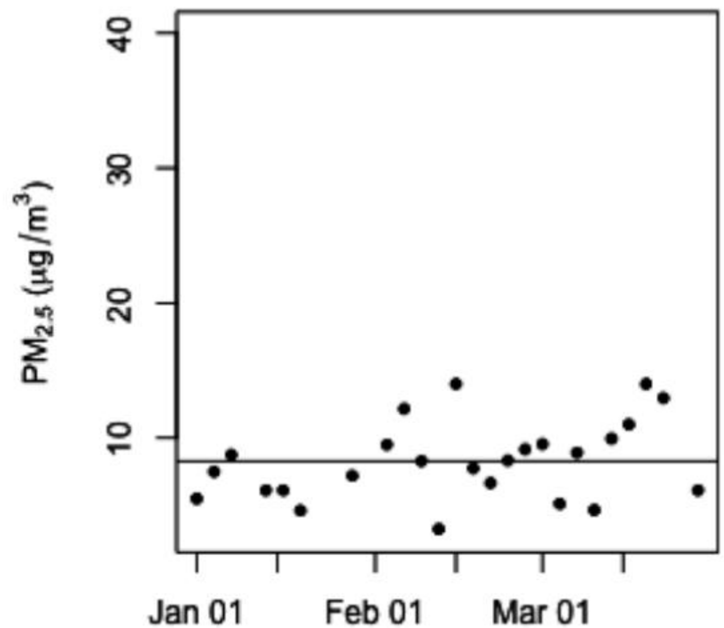
- **Observation 1**: Median levels of PM (horizontal solid line) have decreased a little.



Results: An Individual Monitor

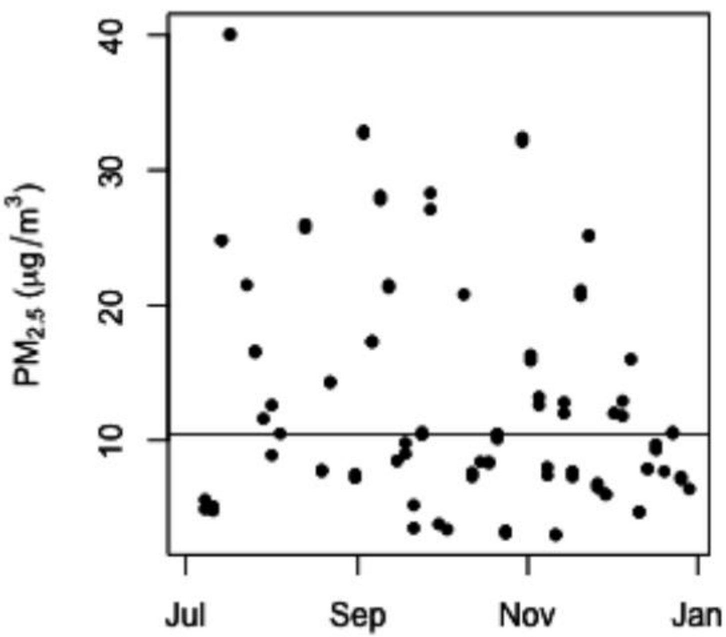
- **Observation 2**: The variation (spread) in the PM values in 2012 is much smaller than it was in 1999.
  - Fewer large spikes from day to day!

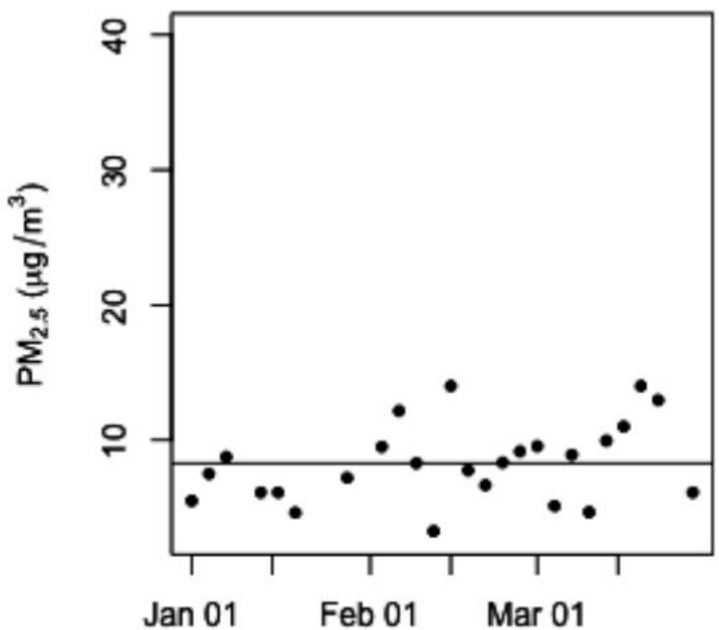




Results: An Individual Monitor

- **Possible Issue**. The 1999 data are from July through December while the 2012 data are recorded in January through April.
    - *Better if we have full-year data for both 1999 and 2012.*





Results: State-wide PM Levels

- The actual reduction and management of PM is left to the individual states!
- Calculate the mean of PM for each state in 1999 and 2012

Tapply(): Apply a function to each cell of a ragged array, that is to each (non-empty) *group of values* given by a unique combination of the levels of certain factors.

```
> ## 1999
> mn0 <- with(pm0, tapply(Sample.Value, State.Code, mean, na.rm = TRUE))
> ## 2012
> mn1 <- with(pm1, tapply(Sample.Value, State.Code, mean, na.rm = TRUE))
>
> ## Make separate data frames for states / years
> d0 <- data.frame(state = names(mn0), mean = mn0)
> d1 <- data.frame(state = names(mn1), mean = mn1)
> mrg <- merge(d0, d1, by = "state")
> head(mrg)
```

	state	mean.x	mean.y
1	1	19.956391	10.126190
2	10	14.492895	11.236059
3	11	15.786507	11.991697
4	12	11.137139	8.239690
5	13	19.943240	11.321364
6	15	4.861821	8.749336

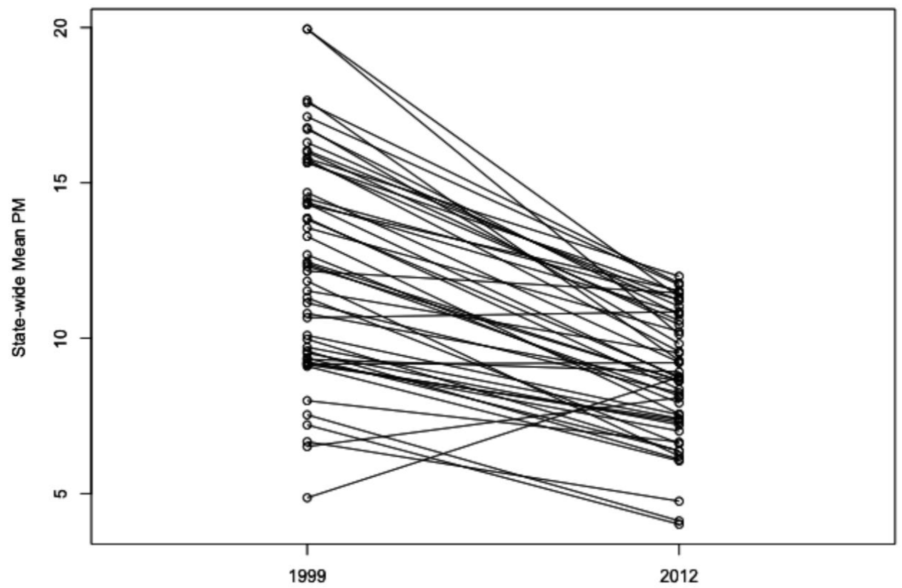
Results: State-wide PM Levels

- Now make a plot that shows the 1999 state-wide means in one “column” and the 2012 state-wide means in another columns.
- We then draw a line connecting the means for each year in the same state to highlight the trend.

```
> par(mfrow = c(1, 1))
> rng <- range(mrg[,2], mrg[,3])
> with(mrg, plot(rep(1, 52), mrg[, 2], xlim = c(.5, 2.5), ylim = rng, xaxt = "n"\
, xlab = "", ylab = "State-wide Mean PM"))
> with(mrg, points(rep(2, 52), mrg[, 3]))
> segments(rep(1, 52), mrg[, 2], rep(2, 52), mrg[, 3])
> axis(1, c(1, 2), c("1999", "2012"))
```

Results: State-wide PM Levels

- Now make a plot that shows the 1999 state-wide means in one “column” and the 2012 state-wide means in another columns.
- We then draw a line connecting the means for each year in the same state to highlight the trend.



Observation: Many states have decreased the average PM levels from 1999 to 2012 (although a few states actually increased their levels).

A slide to take away

- **Simulations**
  - Why we need to do simulations?
  - How to generate random numbers from some certain distributions?
  - Why random seeds are important?
  - How to do random samplings?
- **Case Study**
  - How to read data?
  - How to analyze the results?