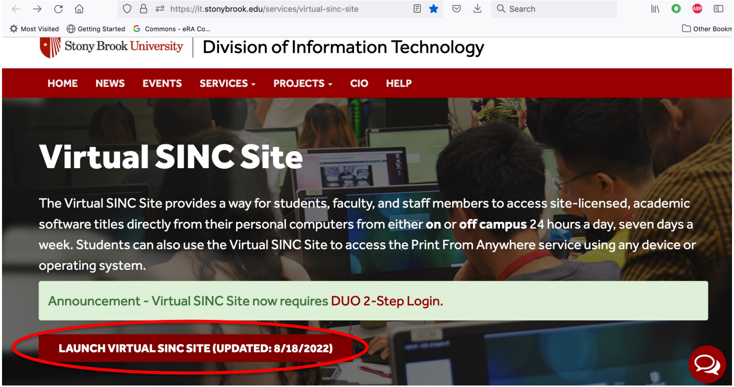
AMS 572 Data Analysis I
Introduction to R

Pei-Fen Kuan

Applied Math and Stats, Stony Brook University

Getting started with SAS

- SAS can be downloaded from `https://softweb.cc.stonybrook.edu/`
- You can use SAS program from Virtual SINC Site: `https://it.stonybrook.edu/services/virtual-sinc-site`

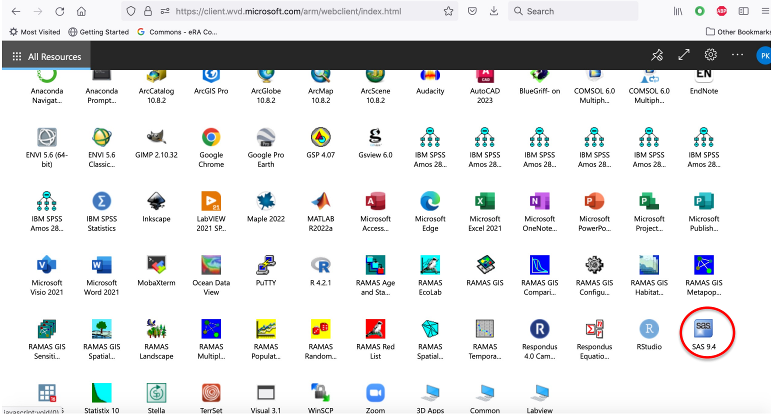


Virtual SINC Site

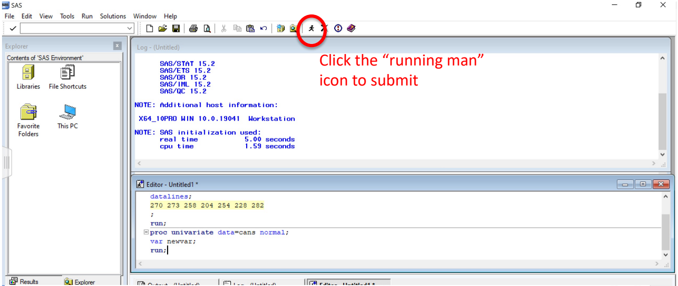
The Virtual SINC Site provides a way for students, faculty, and staff members to access site-licensed, academic software titles directly from their personal computers from either **on** or **off campus** 24 hours a day, seven days a week. Students can also use the Virtual SINC Site to access the Print From Anywhere service using any device or operating system.

Announcement - Virtual SINC Site now requires DUO 2-Step Login.

Getting started with SAS



Getting started with SAS



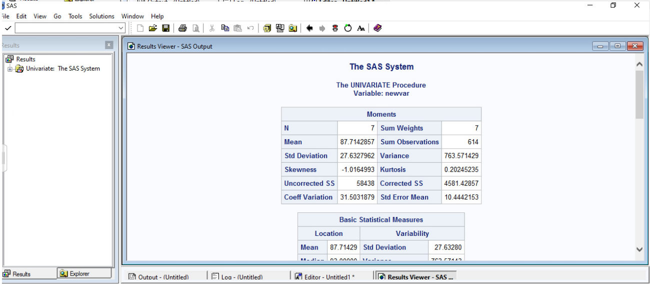
Click the "running man" icon to submit

```
     SAS/STAT 15.2
     SAS/ETS 15.2
     SAS/OR 15.2
     SAS/IML 15.2
     SAS/QC 15.2
NOTE: Additional host information:

 X64_10PRO WIN 10.0.19041  Workstation

NOTE: SAS initialization used:
     real time           5.00 seconds
     cpu time            1.59 seconds
```

```
   datalines;
   270 273 258 204 254 228 282
   ;
   run;
proc univariate data=cans normal;
   var newvar;
   run;
```



The SAS System

The UNIVARIATE Procedure
Variable: newvar

Moments			
N	7	Sum Weights	7
Mean	87.7142857	Sum Observations	614
Std Deviation	27.6327962	Variance	763.571429
Skewness	-1.0164993	Kurtosis	0.20245235
Uncorrected SS	58438	Corrected SS	4581.42857
Coeff Variation	31.5031879	Std Error Mean	10.4442153

Basic Statistical Measures			
Location		Variability	
Mean	87.71429	Std Deviation	27.63280
Median	93.00000	Variance	763.57143

Using SAS as a calculator

```
data compute;
x=(8/9)*2+2**4+exp(2);
put "the answer is:" x;
run;
/* this is comment in SAS */
/* ** means power */
```

SAS Procedures

- SAS procedures perform various computations on SAS datasets.
- A general syntax of a SAS procedure

```
proc procname options;
statements / statement options;
.
.
.
statements / statement options;
run;

/* Example */
proc freq data = mydata;
tables gender / nocum;
run;
```

What is R?

A language and software environment for statistical computing and graphics.

- R is free! Can be downloaded from `https://cran.r-project.org/`
- Some students also use R studio `https://rstudio.com/` which runs R and includes a nice console, syntax-highlighting editor that supports direct code execution.
- It is open-source and involves many developers.
- The R system is developing rapidly.
- Straightforward simple calculations and analysis.
- Allows low level control for some tasks.
- Extensive graphical abilities.
- Sometimes R is slow.
- Introduction to R for data science `https://r4ds.had.co.nz/introduction.html`

Using R as a calculator

```
> # How many seconds in a year?
> 60*60*24*365
[1] 31536000
>
> # remainder
> 56%%10
[1] 6
>
> # natural log, log base 10 and log base 2
> log(100)
[1] 4.60517
>
> log10(100)
[1] 2
>
> log2(4)
[1] 2
```

Using R as a calculator

```
> # exponential
> exp(1)
[1] 2.718282
>
> # power
> 2^3
[1] 8
>
> # square root
> sqrt(16)
[1] 4
```

Scalar Variables

```
> # Define a variable
> x = 12.3
> x
[1] 12.3
>
> # R language is case sensitive
> X
Error: object ’X’ not found
>
> # another way to define a variable
> z <-  456.7
> z + x
[1] 469
>
> # be careful
> w < - 12
Error: object ’w’ not found
```

Vectors

```
> # Define a vector
> v = c(1,2,3,4,5)
> v
[1] 1 2 3 4 5
> v*3
[1]  3  6  9 12 15
>
> # summation
> sum(v)
[1] 15
>
> # mean and standard deviation
> mean(v)
[1] 3
> sd(v)
[1] 1.581139
```

```
> v = c(1,3,-5,10,-7)
> summary(v)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   -7.0    -5.0     1.0     0.4     3.0    10.0
>
> # vector length
> length(v)
[1] 5
>
> # choose a subvector
> 1:3
[1] 1 2 3
> v[1:3]
[1]  1  3 -5
>
> v1 = v[which(v>0)]
> v1
[1]  1  3 10
```

Matrices

```
> m1 = matrix(1:9, nrow=3, ncol=3)
> m1
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
>
> m2 = matrix(1:9, nrow=3, ncol=3, byrow=TRUE)
> m2
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
>
> m1 + m2
     [,1] [,2] [,3]
[1,]    2    6   10
[2,]    6   10   14
[3,]   10   14   18
```

```
> # matrix dimension
> dim(m1)
[1] 3 3
>
> # element-wise multiplication
> m1*m2
     [,1] [,2] [,3]
[1,]    1    8   21
[2,]    8   25   48
[3,]   21   48   81
>
> m3 = matrix(1:6, nrow=3)
> m3
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
> m1*m3
Error in m1 * m3 : non-conformable arrays
```

```
> # matrix multiplication
> m1 %*% m2
     [,1] [,2] [,3]
[1,]   66   78   90
[2,]   78   93  108
[3,]   90  108  126
>
> m1 %*% m3
     [,1] [,2]
[1,]   30   66
[2,]   36   81
[3,]   42   96
```

```
> m1
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
>
> # submatrix
> m1[2,2]
[1] 5
>
> m1[1:2,]
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
>
> m1[c(1,3),2:3]
     [,1] [,2]
[1,]    4    7
[2,]    6    9
```

```
> diag(1,3)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
>
> m4 = diag(1,3) + matrix(c(0,1,2,0,0,1,0,0,0),nrow=3)
> m4
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    1    1    0
[3,]    2    1    1
>
> # matrix transpose
> t(m4)
     [,1] [,2] [,3]
[1,]    1    1    2
[2,]    0    1    1
[3,]    0    0    1
```

```
> # matrix transpose
> t(m4)
     [,1] [,2] [,3]
[1,]    1    1    2
[2,]    0    1    1
[3,]    0    0    1
>
> # matrix inverse
> m5 = solve(m4)
> m5
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]   -1    1    0
[3,]   -1   -1    1
>
> m4 %*% m5
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

seq/rep

```
> # Generate a sequence
> # seq(from, to, by/length)
> x = seq(1, 10, by=3)
> x
[1]  1  4  7 10
> y = seq(1, 10, length.out=5)
> y
[1]  1.00  3.25  5.50  7.75 10.00
>
> # replicate a vector x
> # rep(x, times)
> z = rep(1, 5)
> z
[1] 1 1 1 1 1
```

rbind/cbind

```
> x = c(1, 2, 3)
> y = c(4, 5, 6)
> # row-wise bind
> rbind(x,y)
  [,1] [,2] [,3]
x    1    2    3
y    4    5    6
>
> # column-wise bind
> cbind(x,y)
     x y
[1,] 1 4
[2,] 2 5
[3,] 3 6
```

Types of Variables

```
> v = 1:5
> v
[1] 1 2 3 4 5
> mode(v)
[1] "numeric"
>
> a = "Hello AMS 572 students :)"
> a
[1] "Hello AMS 572 students :)"
> mode(a)
[1] "character"
>
> b = v == 2
> b
[1] FALSE  TRUE FALSE FALSE FALSE
> mode(b)
[1] "logical"
```

```
> # factor
> cars = c("bmw","toyota","hyundai","ford")
>
> # sample() draws a sample with/without replacement
> mysample = sample(cars,10, replace=TRUE)
>
> # as.factor() forces its argument to be an object of class factor
> as.factor(mysample)
 [1] toyota  ford    ford    bmw     ford    toyota  bmw     bmw    
 [9] hyundai hyundai
Levels: bmw ford hyundai toyota
>
> # frequency table
> table(mysample)
mysample
    bmw    ford hyundai  toyota 
      3       3       2       2 
```

R functions, datasets, and packages

- R functions and datasets are stored in packages. They are available when a package is loaded.
- By default, some standard packages (e.g., base, stats) are included in the binary distribution of R and they are loaded into the R environment automatically when one opens the R interface.
- Some recommended packages are included in the binary R distribution, but are not loaded automatically.
- Contributed packages need to be installed before one can load and use them.

Help

- How does one know which function to use?
- Suppose one is looking for something related to the uniform distribution
  - help(package="stats")
  - help.search("uniform")
  - google it
- How to use a function?
  - ?runif
  - help(runif)

Loops and Conditional Execution

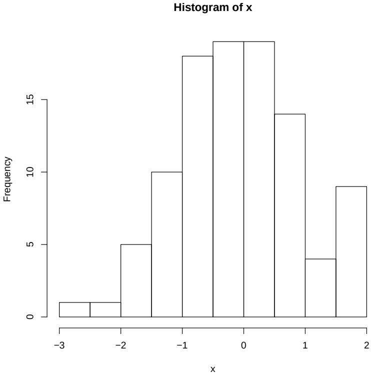
```
> x = runif(100)
> x[1:5]
[1] 0.7829481 0.6258223 0.1174721 0.4679859 0.7647583
> summary(x)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
0.005991 0.301300 0.477400 0.482900 0.644000 0.958800 
> 
> sum(x[x>0.5 & x<0.8])
[1] 23.28054
> 
> y = 0
> for(i in 1:length(x)){
+   if(x[i]>0.5 & x[i]<0.8){
+     y = y + x[i]
+   }
+ }
> y
[1] 23.28054
```

Reading/writing data from/to files

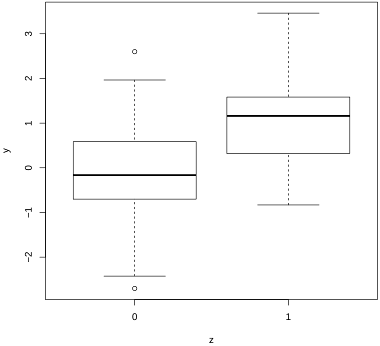
```
> mydat = read.table("AMS572dat1.txt", header = TRUE, sep = "\t")
> write.table(mydat, file = "AMS572dat2.txt", append = FALSE,
    quote = FALSE,sep = "\t", row.names = FALSE, col.names = TRUE)
```

Basic graphics in R

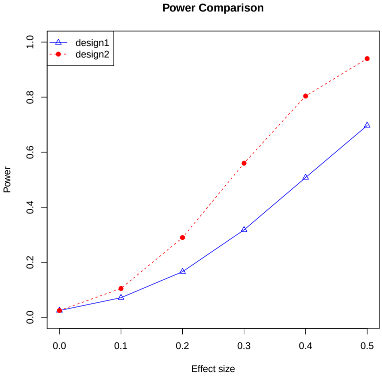
- Scatter plot: `plot(x,y)`
- Histogram: `hist(x)`
- Boxplot: `boxplot(x)`
- User's control on plotting
  - add points: `points(x,y)`
  - add lines: `lines(x,y)`, `abline(a,b)`
  - add text: `text(x,y,labels)`
  - add legend: `legend(x,y,legend)`



```
> x <- rnorm(100)
> hist(x)
```



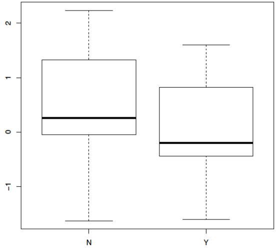
```
> y = c(rnorm(100, 0, 1),rnorm(100, 1, 1))
> z = rep(c(0,1),each=100)
> boxplot(y~z, xlab="z", ylab="y")
> x = c(0,0.1,0.2,0.3,0.4,0.5)
> y1 = c(0.0250, 0.0715, 0.1660, 0.3180, 0.5080, 0.6970)
> y2 = c(0.025, 0.105, 0.290, 0.560, 0.804, 0.940)
>
> plot(c(0,0.5), c(0,1), type = "n" ,xlab = "Effect size",
   ylab = "Power",main = "Power Comparison")
> points(x, y1, pch=2, col="blue")
> lines(x, y1, lty=1, col="blue")
> points(x, y2, pch=19, col="red")
> lines(x, y2, lty=2, col="red")
> legend("topleft",legend = c("design1","design2"),
   pch = c(2,19),lty = c(1,2),col = c("blue","red"))
```

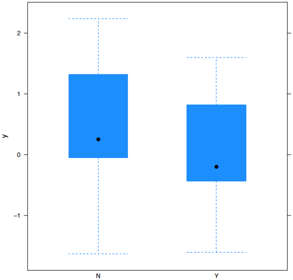

Power Comparison
design1
design2
Power
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.1
0.2
0.3
0.4
0.5
Effect size

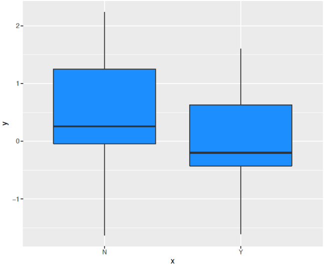
Advanced graphics in R

- R packages `lattice`, `ggplot2`.

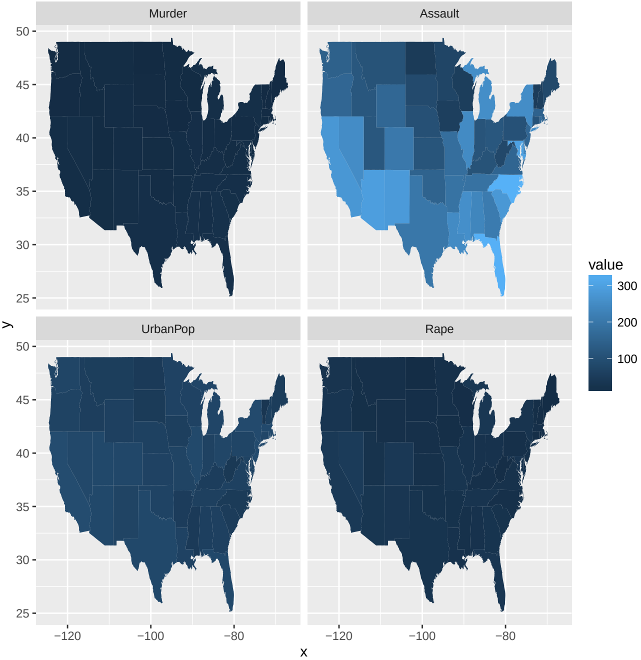
```
> library(ggplot2)
> library(lattice)
>
> x <- rep(c('Y','N'),each=20)
> y <- c(rnorm(20),rnorm(20,0.5))
>
> dat <- data.frame(x=x,y=y)
> ### basic
> boxplot(y~x,cols='dodgerblue')
> ## lattice
> bwplot(y~x,par.settings =
+ list(box.rectangle = list(fill= c('dodgerblue','dodgerblue'))))
> ## ggplot2
> ggplot(dat, aes(x = x, y = y)) + geom_boxplot(fill='dodgerblue')
```


2
1
0
−1
N
Y


y
2
1
0
−1
N
Y


y
2
1
0
−1
N
Y
x

```
> library(ggplot2)
> library(maps)
> crimes <- data.frame(state = tolower(rownames(USArrests)), USArrests)
> crimesm <- reshape2::melt(crimes, id = 1)
> if (require(maps)) {
+   states_map <- map_data("state")
+   ggplot(crimes, aes(map_id = state)) +
+     geom_map(aes(fill = Murder), map = states_map) +
+     expand_limits(x = states_map$long, y = states_map$lat)
+
+   last_plot() + coord_map()
+   ggplot(crimesm, aes(map_id = state)) +
+     geom_map(aes(fill = value), map = states_map) +
+     expand_limits(x = states_map$long, y = states_map$lat) +
+     facet_wrap( ~ variable)
+ }
```


Murder
Assault
UrbanPop
Rape
value
300
200
100
50
45
40
35
30
25
y
x
−120
−100
−80

R Markdown

- A convenient tool to create reproducible web-based reports.
- The simplest way to convert R Markdown file to HTML is via R studio.
- Example of an R Markdown file: `http://www.ams.sunysb.edu/~pfkuan/RNAAgeCalc/RNAAge-vignette.html`

Read Chapter 4