Lecture 11
String manipulation and graphics

GEOG 489 SPRING 2020

String manipulation

1) grep(pattern, x): look for a substring pattern in a vector of strings

```
mystrings <- c("Equator","North Pole","South Pole")
grep("Pole",mystrings)
```

2) nchar(): the number of characters in a string

```
nchar("South Pole") # spaces count
```

String manipulation

3) paste(): concatenates strings into one string

```
paste("North","Pole")

x <- "and"

paste("North",x,"South","Poles")
```

4) sprintf(): string print. It returns a character vector containing a formatted combination of text and variable values

```
i <- 8

s <- sprintf("the square of %d is %d",i,i^2)
```

String manipulation

5) substr(): return a substring given a range of characters

```
# return the 3rd through 5th character
substr("Equator",start=3,stop=5)
```

6) regexpr(): Find the character position of the first instance of the pattern within the string.

```
regexpr(pattern="uat",text="Equator")
```

gregexpr(): find the character position of all instances of the pattern

```
gregexpr("iss","Mississippi")[[1]]
```

String manipulation

7) Regular expressions

```
mystrings <- c("Equator","North Pole","South Pole")

# Bracket expressions search for a single character that is found within the brackets
grep("[au]",mystrings)

# A period represents a single-character wildcard
grep("o.e",mystrings)
# Each period represents a wildcard for one character, so we can put multiple periods together:
grep("N..t",mystrings)
```

Graphics

There are basically two approaches to graphing.

The first is the default graphing functions that R comes with.

The second is a package called "lattice".

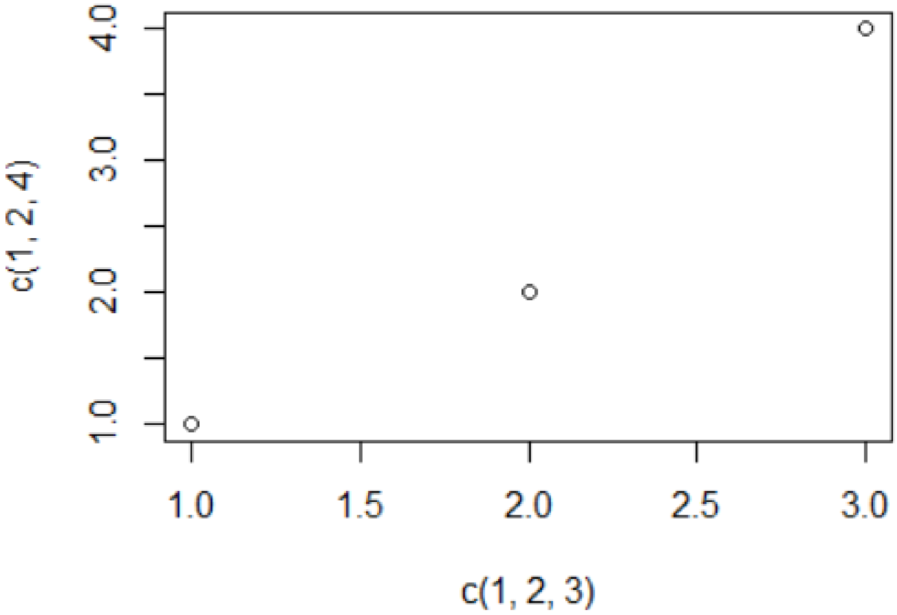
We will focus on the former, but mention that lattice is a VERY widely used package, and allows for vastly expanded graphing capabilities over the default package.

Graphics

1) plot() function

Plot() is a generic function, and is used by a large number of objects. For instance, if we look at plotting two vectors:

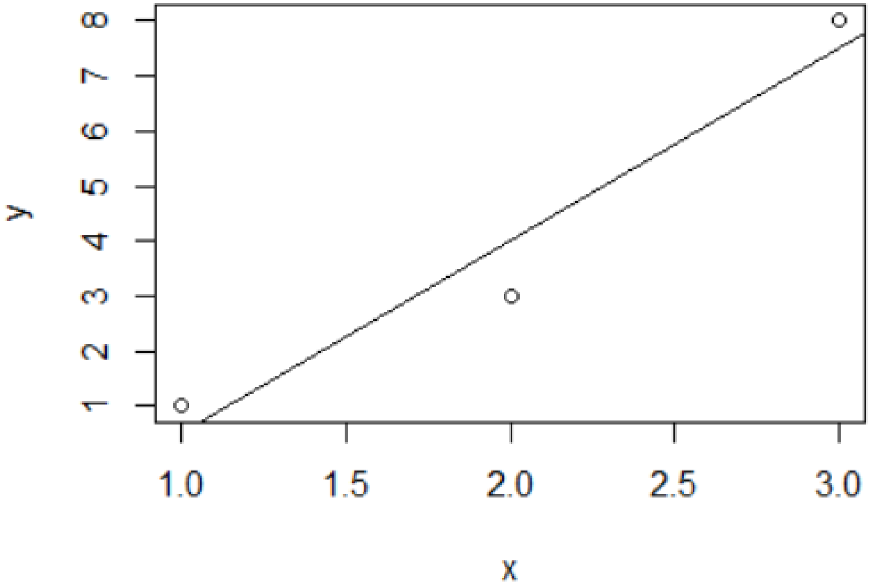
plot(x=c(1,2,3),y=c(1,2,4))



Graphics

2) abline() function

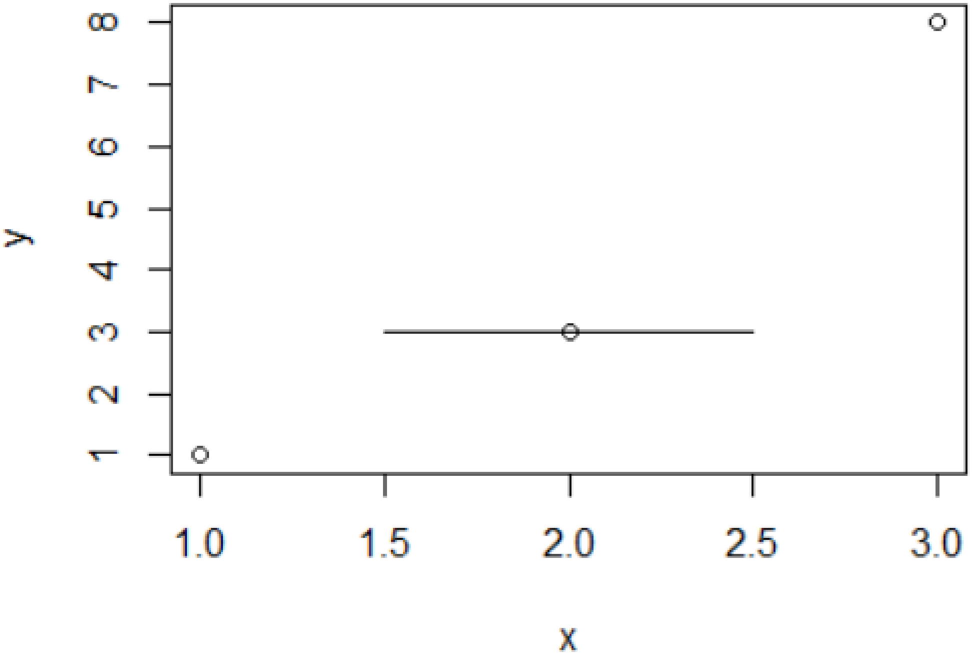
```
x <- c(1,2,3)
y <- c(1,3,8)
plot(x,y,xlab="x",ylab="y")

lmout <- lm(y ~ x)
# add a line using linear regression
abline(lmout)
```



Graphics

3) lines() function

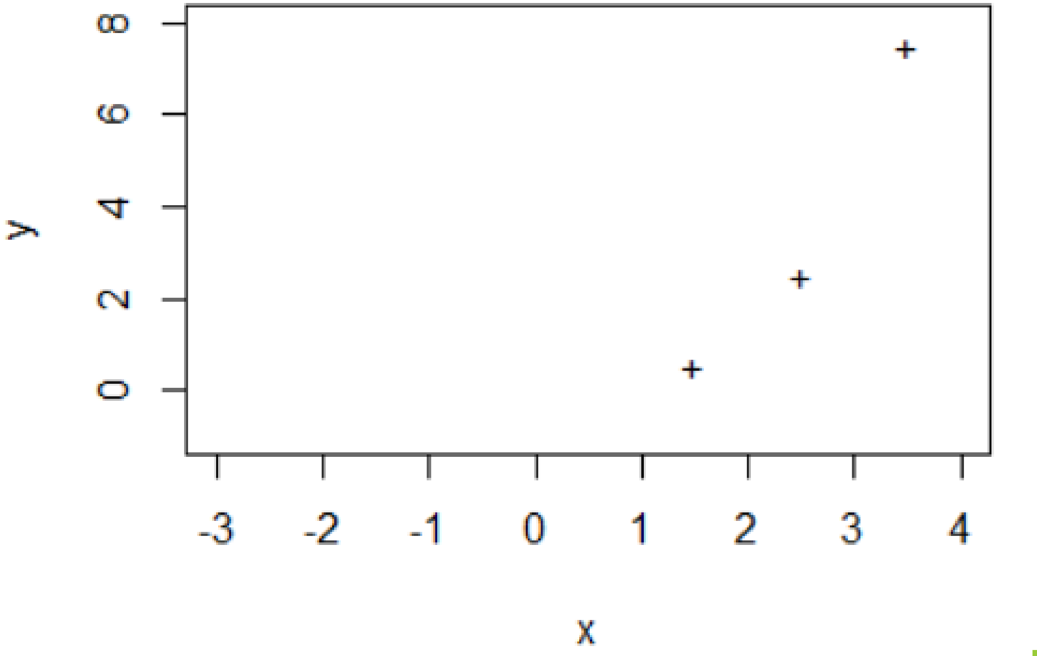
```
x <- c(1,2,3)
y <- c(1,3,8)
plot(x,y,xlab="x",ylab="y")

# add a line with starting and stopping coordinates
lines(x=c(1.5,2.5),y=c(3,3))

# plot a line
plot(x=x,y=y,type="l")
```



Graphics

4) points() function

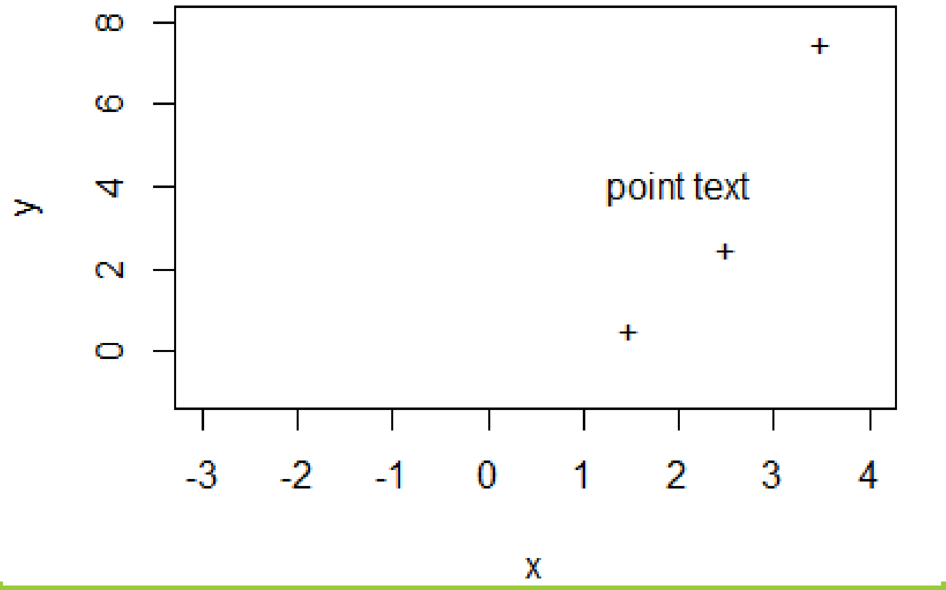
```
plot(x=c(-3,4), y=c(-1,8), type="n", xlab="x", ylab="y")
# add points
points(x=c(1.5,2.5,3.5), y=c(0.5,2.5,7.5), pch="+")
```



Graphics

5) text() function

```
# add text on a graph
plot(x=c(-3,4),y=c(-1,8),type="n",xlab="x",ylab="y")
points(x=c(1.5,2.5,3.5),y=c(0.5,2.5,7.5),pch="+")
text(x=2,y=4,"point text")
```



Graphics

5) Customize graphs

- change character sizes: the cex options
- change the range of axes: the xlim and ylim options

```
plot(x,y,xlim=c(-10,10),ylim=c(-20,20), xlab="myx", ylab="myy")
```

- add a polygon: the polygon() function

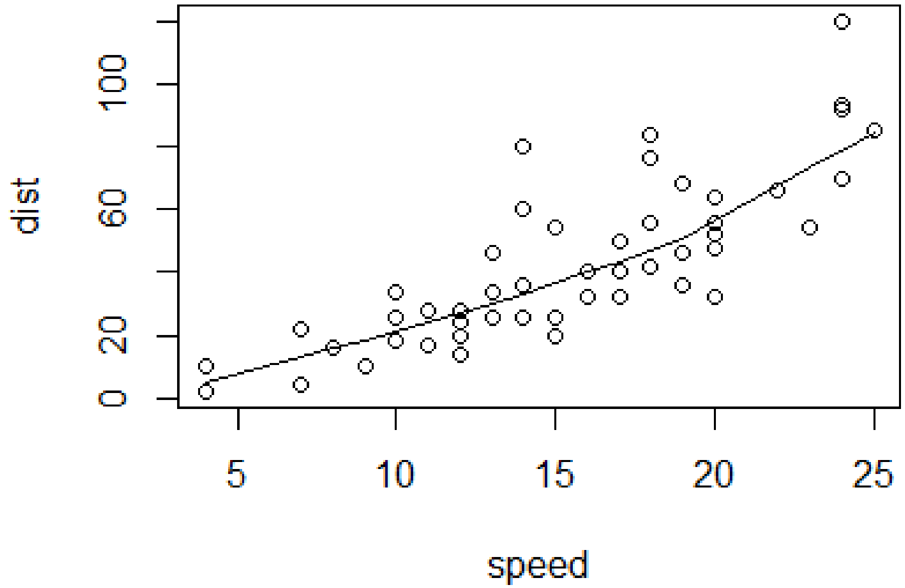
```
f <- function(x) return(1-exp(-x))

curve(expr=f,from=0,to=2)

polygon(x=c(1.2,1.4,1.4,1.2),y=c(0,0,f(1.3),f(1.3)),col="gray")
```

Graphics

5) Customize graphs

- smoothing points: the lowess() and loess() functions

```
plot(cars)

lines(lowess(cars))
```



Graphics

5) Save graphs to files

- Each graphic window or file we want to "print" to is considered a graphics device.

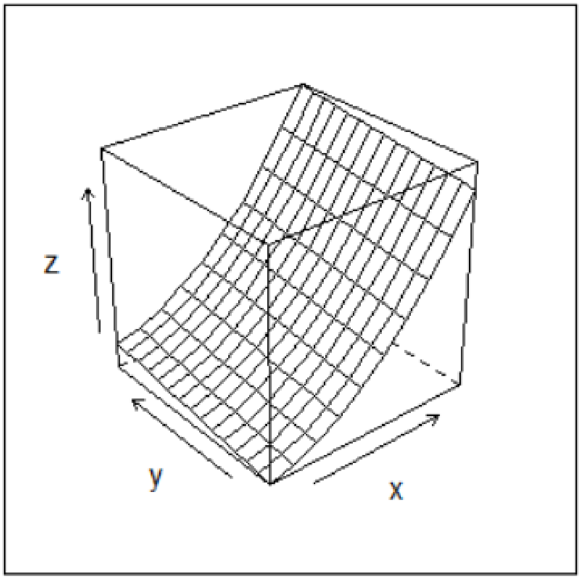
```
# We can see all the devices currently opened via:
dev.list()
# Let's open a pdf file to save a plot to:
pdf("d14_multiple.pdf")
curve(expr=f,from=0,to=3)
dev.off() # Turn it off (close it).
# Set the current graphics device to be ID #2
dev.set(2)
```

Graphics

5) Create 3-d plots

- load up more powerful library “lattice”

```
library(lattice)

# Create all possible combinations of a and b:

a <- 1:10

b <- 1:15

eg <- expand.grid(x=a,y=b)

eg$z <- eg$x^2 + eg$x + eg$y

wireframe(z ~ x+y,eg)
```



Quiz 5

Write a function to remove “NA” from a string

For example:

Input: x <- "Programming NA GIS NA"

Output: y <- "Programming GIS “

The R file needs to be named: LastName_FirstName_Quiz5.R

Please submit the quiz R file on Compass by the end of this class.