STAT 545

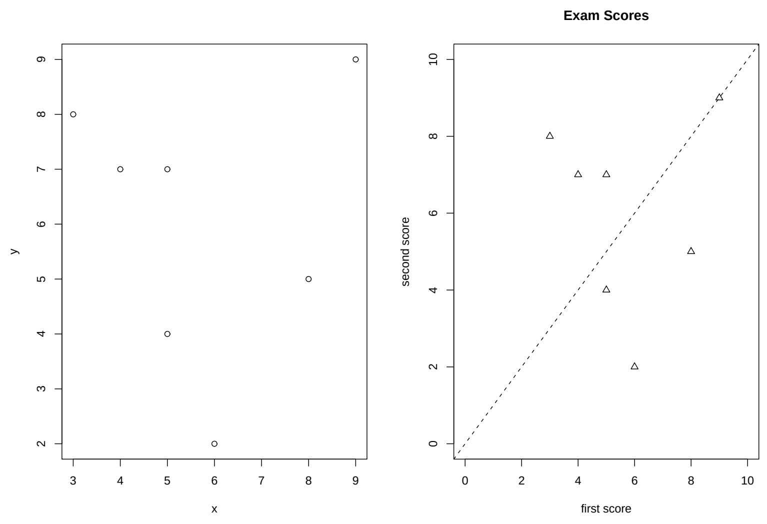
TEN NICE THINGS ABOUT R/S

1. Defaults versus fine control

```
> x <- c(3, 8, 5, 6, 4, 5, 9)
> y <- c(8, 5, 7, 2, 7, 4, 9)

> plot(x,y)

> plot(x,y, pch=2, xlim=c(0,10), ylim=c(0,10),
       xlab="first score", ylab="second score",
       main="Exam Scores")
> abline(0, 1, lty=2)
```



2. Data Frames

```
> !system("cat data.txt")
name exam1 exam2
Fred     56    67
Barney   88    78
Will     70    65
Grace    90    98
Bill     40    30
Hillary 75    85

> testscores <- read.table("data.txt", header=T)
> testscores
     name exam1 exam2
1    Fred    56    67
2  Barney    88    78
3    Will    70    65
4   Grace    90    98
5    Bill    40    30
6 Hillary    75    85

> testscores$exam1
[1] 56 88 70 90 40 75
```

```
> testscores$name
[1] Fred    Barney  Will    Grace   Bill    Hillary
Levels: Barney Bill Fred Grace Hillary Will

> summary(testscores)

      name        exam1             exam2
 Barney :1   Min.   :40.00   Min.   :30.00
 Bill   :1   1st Qu.:59.50   1st Qu.:65.50
 Fred   :1   Median :72.50   Median :72.50
 Grace  :1   Mean   :69.83   Mean   :70.50
 Hillary:1   3rd Qu.:84.75   3rd Qu.:83.25
 Will   :1   Max.   :90.00   Max.   :98.00
```

3. Matrix Facilities

```
> x <- matrix( rnorm(10), nrow=5)
> y <- x %*% c(-1, 1) + rnorm(5, mean=0, sd=0.5)
> beta.hat <- solve ( t(x) %*% x) %*% t(x) %*% y

> beta.hat
[1,] -1.2742084  0.8042694

> var(x)
          [,1]      [,2]
[1,]  3.052888 -1.126367
[2,] -1.126367  0.491465

> x[c(1,3,5),2]
[1]  0.3747817 -0.4067923  0.1223382
```

4. Object Oriented

```
> mysummary <- function(x) {
    tmp1 <- mean(x)
    tmp2 <- median(x)
    tmp3 <- max(x)-min(x)
    list(mean=tmp1, median=tmp2, range=tmp3)
  }

> tmp <- mysummary( c(4, 7, 3, 8, 9) )

> names(tmp)
[1] "mean"   "median" "range"

tmp$median
[1] 7
```

5. Reasonable Looping

```
> p <- 10
> for (i in 0:(p-1)) {
    for (j in 0:(p-1)) {
      for (k in 0:(p-1)) {
        for (m in 0:(p-1)) {
          print(i*p^3 + j*p^2 + k*p^1 + m*p^0)
  } } } }
```

Compare with S-Plus!!!

9

6. "Conversion"

```
> x <- sample( c(T,F), size=10, replace=T)
> y <- as.numeric(x)
> x
 [1]  TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE  ...
> y
 [1] 1 1 0 0 0 0 1 1 1 1
> typeof(x)
[1] "logical"
> typeof(y)
[1] "double"

> mean(y)
[1] 0.6
> mean(x)
[1] 0.6
```

10

7. Functions like apply()

```
> x <- matrix(rnorm(30), nrow=10)

> tmp <- rep(NA, 3)
> for (i in 1:3) {
    tmp[i] <- mean(x[,i]) }

> tmp
[1]  0.5688142  0.2355787 -0.4592990

> apply(x, 2, mean)
[1]  0.5688142  0.2355787 -0.4592990
```

Also see tapply(), sapply(), lapply()

11

8. Function of function

```
> fun <- function(x) {
    -abs(x)*exp(-.5*x^2)
  }

> tmp <- nlm(fun, p=2)

> tmp$minimum
[1] -0.6065307
> tmp$estimate
[1] 0.9999995
```

12

9. Built-in statistics and probability functions

```
> tmp_rbeta(5,15,5)
> tmp
[1] 0.7344866 0.7831696 0.8848570 0.6997509 0.6191256

> qbeta(pbeta(tmp,15,5) , 15,5)
[1] 0.7344866 0.7831696 0.8848570 0.6997509 0.6191256
```

10. Nice Graphics!

```
> demo(graphics)

> demo(image)

> demo(persp)
```