Module 12
R Programming

Andrew Jaffe
Instructor
R 'programming'

Now we are going to switch gears a little bit, and talk about some of the more traditional programming that you can do in R.

You can do very flexible things, but at a cost of more difficult notation, and having to actually write programming statements. There are slight notation differences as well, including the use of curly {} brackets

We are going to cover for loops and if statements
'for' Loops

These allow you to iterate over certain observations or subsets of observations.

The syntax is:

```r
for(*var* in seq) {
  do something
}
```

Typically they look something like:

```r
for(i in 1:nrow(dat)) {
  something(dat[i,])
}
```
'for' loops

These are essentially fancier apply statements

For example,

```r
> for(i in 1:10) {
+   print(i)
+ }
```

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 10
```
'for' loops

Here's how they can be more flexible:

```r
> Index = c(3,6,7,20,32,100,234,1000,6543)
> for(i in 1:length(Index)) {
>       print(Index[i])
> }
```

```
[1]  3
[1]  6
[1]  7
[1] 20
[1] 32
[1] 100
[1] 234
[1] 1000
[1] 6543
```

Note that the first time through the body of the loop, \( i \) takes the value 1, then evaluates the body. Then, \( i \) takes the value 2, and evaluates the body, until \( i = \text{length}(\text{Index}) \), then it stops.
'for' loops

They are essentially more useful than apply statements when you are working with two sets of matching datasets or vectors.

```r
> myList = vector("list", length=4)
> mat1 = matrix(rnorm(8), nc = 4)
> mat2 = matrix(rnorm(8), nc = 4)
> mat1

[1,] -0.2854 -0.01147 -0.3855 -0.3403  
[2,] -0.9100  0.52103  1.1067 -1.7502  

> mat2

[1,]  1.429  0.02987 -0.6456  0.5449  
[2,]  1.775  0.30341 -0.3780 -0.6708  
```
> for(i in seq(along=myList)) {
+   myList[[i]] = cbind(mat1[,i],mat2[,i])
+ }
> myList

[[1]]
  [,1] [,2]
[1,] -0.2854  1.429
[2,] -0.9100  1.775

[[2]]
  [,1] [,2]
[1,] -0.01147  0.02987
[2,]  0.52103  0.30341

[[3]]
  [,1] [,2]
[1,] -0.3855  -0.6456
[2,]  1.1067  -0.3780

[[4]]
  [,1] [,2]
[1,] -0.3403   0.5449
[2,] -1.7502  -0.6708
'for' loops

```r
> i=1
> cbind(mat1[,i],mat2[,i])
```

```
[,1]  [,2]
[1,] -0.2854 1.429
[2,] -0.9100 1.775
```

```r
> i=2
> cbind(mat1[,i],mat2[,i])
```

```
[,1]  [,2]
[1,] -0.01147 0.02987
[2,] 0.52103 0.30341
```

```r
> i=3
> cbind(mat1[,i],mat2[,i])
```

```
[,1]  [,2]
[1,] -0.3855 -0.6456
[2,] 1.1067 -0.3780
```
'for' loops

These are useful for making many columns worth of density plots

```
> mat = matrix(rnorm(1000*50), nc = 50)
> plot(density(mat[,1]), ylim = c(0, 0.45))
> for(i in 2:ncol(mat)) {lines(density(mat[,i]))}
```
'for' loops

You can also integrate with lists.

```r
> outList = vector("list",10)
> start=1:10
> end = sample(1:100, 10)
> for(i in seq(along=outList)) {
+   outList[[i]] = start[i]:end[i]
+ }
> outList

[[1]]
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
[24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
[47] 47 48 49 50 51 52 53 54 55 56 57 58 59

[[2]]
 [1]  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
[24] 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
[47] 48 49 50 51 52 53 54

[[3]]
 [1]  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25

[[4]]
 [1]  4  5  6  7  8  9 10 11 12 13 14 15 16 17

[[5]]
 [1]  5  6  7  8  9 10
```
'if' statements

You can put 'if' statements inside of 'for' loops

```r
for(i in 1:nrow(dat)) {
  if(dat$x > num) {
    dat$y[i] = something
  } else {
    dat$y[i] = something else
  }
}
```
Example

```r
> makeIndexes = split(1:nrow(cars), cars$Make)
> lapply(makeIndexes, head, n=4)[1:3]

$ACURA
[1] 10039 13026 13631 14250

$BUICK
[1] 185 233 258 346

$CADILLAC
[1] 3372 4517 8500 9664
```
> pval = rep(NA,length(makeIndexes))
> for(i in 1:length(makeIndexes)) {
>   ind = makeIndexes[[i]]
>   if(length(ind)>1) {
>     f = lm(VehBCost~VehOdo, data=cars,subset=ind)
>     pval[i] = summary(f)$coef[2,4]
>   }
> }
> names(pval)=names(makeIndexes)
>
> i = 1
> ind = makeIndexes[[i]]
> str(ind)

int [1:33] 10039 13026 13631 14250 16392 17289 17889 17979 18166 22044 ...

> f = lm(VehBCost~VehOdo, data=cars,subset=ind)
> summary(f)$coef[2,4]

[1] 0.4932
<table>
<thead>
<tr>
<th>Brand</th>
<th>pval</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACURA</td>
<td>4.932e-01</td>
</tr>
<tr>
<td>BUICK</td>
<td>1.877e-05</td>
</tr>
<tr>
<td>CADILLAC</td>
<td>1.064e-06</td>
</tr>
<tr>
<td>CHEVROLET</td>
<td>2.834e-06</td>
</tr>
<tr>
<td>CHRYSLER</td>
<td>1.128e-78</td>
</tr>
<tr>
<td>DODGE</td>
<td>1.494e-10</td>
</tr>
<tr>
<td>FORD</td>
<td>2.584e-27</td>
</tr>
<tr>
<td>GMC</td>
<td>1.626e-01</td>
</tr>
<tr>
<td>HONDA</td>
<td>2.490e-13</td>
</tr>
<tr>
<td>HUMMER</td>
<td>NA</td>
</tr>
<tr>
<td>HYUNDAI</td>
<td>1.072e-04</td>
</tr>
<tr>
<td>INFINITI</td>
<td>9.737e-04</td>
</tr>
<tr>
<td>ISUZU</td>
<td>2.697e-04</td>
</tr>
<tr>
<td>JEEP</td>
<td>2.723e-16</td>
</tr>
<tr>
<td>KIA</td>
<td>3.765e-18</td>
</tr>
<tr>
<td>LEXUS</td>
<td>5.014e-09</td>
</tr>
<tr>
<td>LINCOLN</td>
<td>7.061e-01</td>
</tr>
<tr>
<td>MAZDA</td>
<td>1.364e-41</td>
</tr>
<tr>
<td>MERCURY</td>
<td>2.953e-04</td>
</tr>
<tr>
<td>MINI</td>
<td>8.709e-02</td>
</tr>
<tr>
<td>MITSUBISHI</td>
<td>3.357e-26</td>
</tr>
<tr>
<td>NISSAN</td>
<td>3.594e-10</td>
</tr>
<tr>
<td>OLDSMOBILE</td>
<td>6.956e-08</td>
</tr>
<tr>
<td>PLYMOUTH</td>
<td>NaN</td>
</tr>
<tr>
<td>PONTIAC</td>
<td>3.305e-154</td>
</tr>
<tr>
<td>SATURN</td>
<td>4.363e-40</td>
</tr>
<tr>
<td>SCION</td>
<td>3.115e-07</td>
</tr>
<tr>
<td>SUBARU</td>
<td>8.062e-01</td>
</tr>
<tr>
<td>SUZUKI</td>
<td>4.285e-32</td>
</tr>
<tr>
<td>TOYOTA</td>
<td>1.395e-16</td>
</tr>
<tr>
<td>SCION</td>
<td>4.738e-06</td>
</tr>
<tr>
<td>VOLKSWAGEN</td>
<td>1.269e-02</td>
</tr>
<tr>
<td>VOLVO</td>
<td></td>
</tr>
</tbody>
</table>
Note you can also do with with `sapply`

```r
> pval2 = sapply(makeIndexes, function(ind) {
+   if(length(ind)>1) {
+     f = lm(VehBCost~VehOdo, data=cars,subset=ind)
+     summary(f)$coef[2,4]
+   } else NA
+ })
> all.equal(pval,pval2)

[1] TRUE
```
Example

Now we can read in many files into a list

```r
> fn = list.files("Reports/", pattern=".txt", full.names=TRUE)
> name = list.files("Reports/", pattern=".txt", full.names=FALSE)
> head(fn)
```

```
[1] "Reports/April_2009_Report.txt"  "Reports/April_2010_Report.txt"
```
```r
> fileList = lapply(fn, read.delim, header=TRUE, as.is=TRUE)
> names(fileList) = name
> sapply(fileList,dim) [,1:5]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>287</td>
<td>324</td>
<td>359</td>
</tr>
<tr>
<td>[2,]</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>August_2009_Report.txt</th>
<th>August_2010_Report.txt</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,]</td>
<td>353</td>
<td>369</td>
</tr>
<tr>
<td>[2,]</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

> lapply(fileList[1:5],head,n=2)

$April_2009_Report.txt

<table>
<thead>
<tr>
<th>id</th>
<th>sex</th>
<th>treat</th>
<th>age</th>
<th>bgDrugs</th>
<th>height</th>
<th>weight</th>
<th>block</th>
<th>recruitDate</th>
<th>bmi</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1072</td>
<td>Female</td>
<td>Control</td>
<td>51.00</td>
<td>asprin</td>
<td>63.84</td>
<td>131.3</td>
<td>d</td>
<td>21</td>
</tr>
<tr>
<td>2</td>
<td>1073</td>
<td>Female</td>
<td>Control</td>
<td>54.81</td>
<td>tylenol</td>
<td>66.10</td>
<td>117.2</td>
<td>b</td>
<td>1</td>
</tr>
</tbody>
</table>

$April_2010_Report.txt

<table>
<thead>
<tr>
<th>id</th>
<th>sex</th>
<th>treat</th>
<th>age</th>
<th>bgDrugs</th>
<th>height</th>
<th>weight</th>
<th>block</th>
<th>recruitDate</th>
<th>bmi</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4337</td>
<td>Female</td>
<td>Case</td>
<td>46.91</td>
<td>none</td>
<td>64.95</td>
<td>140.6</td>
<td>f</td>
<td>25</td>
</tr>
<tr>
<td>2</td>
<td>4338</td>
<td>Female</td>
<td>Case</td>
<td>47.95</td>
<td>none</td>
<td>66.47</td>
<td>143.3</td>
<td>f</td>
<td>14</td>
</tr>
</tbody>
</table>

$April_2011_Report.txt

<table>
<thead>
<tr>
<th>id</th>
<th>sex</th>
<th>treat</th>
<th>age</th>
<th>bgDrugs</th>
<th>height</th>
<th>weight</th>
<th>block</th>
<th>recruitDate</th>
<th>bmi</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7780</td>
<td>Male</td>
<td>Case</td>
<td>53.93</td>
<td>asprin</td>
<td>70.12</td>
<td>175.0</td>
<td>f</td>
<td>29</td>
</tr>
<tr>
<td>2</td>
<td>7781</td>
<td>Male</td>
<td>Control</td>
<td>62.77</td>
<td>tylenol</td>
<td>71.02</td>
<td>153.1</td>
<td>b</td>
<td>29</td>
</tr>
</tbody>
</table>

$August_2009_Report.txt

<table>
<thead>
<tr>
<th>id</th>
<th>sex</th>
<th>treat</th>
<th>age</th>
<th>bgDrugs</th>
<th>height</th>
<th>weight</th>
<th>block</th>
<th>recruitDate</th>
<th>bmi</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2051</td>
<td>Male</td>
<td>Control</td>
<td>56.76</td>
<td>tylenol</td>
<td>70.47</td>
<td>168.0</td>
<td>f</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2052</td>
<td>Male</td>
<td>Case</td>
<td>50.14</td>
<td>asprin</td>
<td>69.56</td>
<td>172.3</td>
<td>c</td>
<td>1</td>
</tr>
</tbody>
</table>