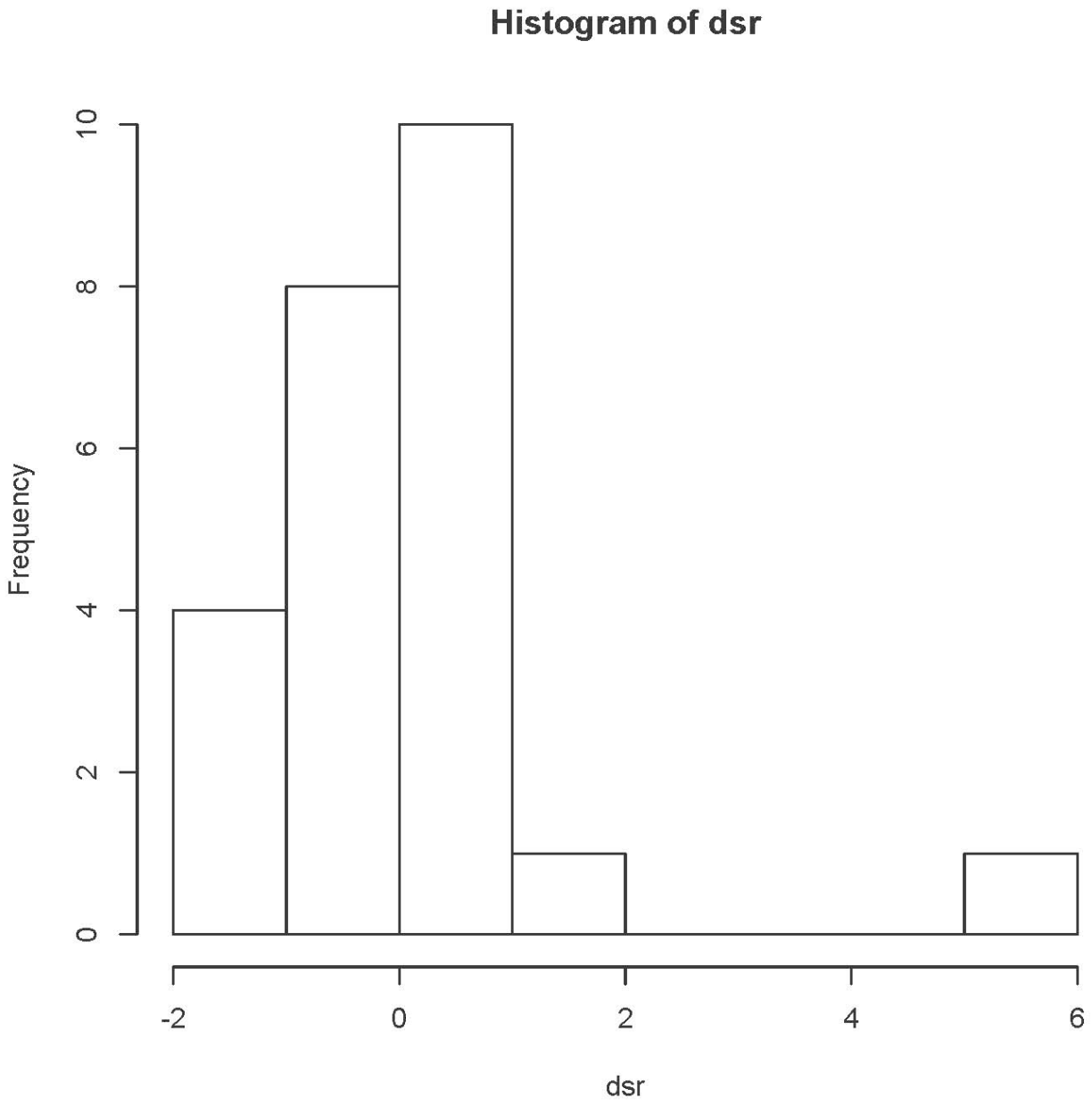
The Little Tubes Data With R

```
 tubes =
read.table("http://www.utstat.toronto.edu/~brunner/appliedf14/code_n_data/lecture/littletubes.data")
> tubes # See the whole data set
   mcg length10 weight
1  198    27.80 0.5996
2  198    28.20 0.6040
3  198    27.60 0.6172
4  198    27.50 0.6053
5  205    24.95 0.6769
6  205    25.70 0.7057
7  205    25.40 0.7271
8  205    25.30 0.6029
9  213    26.85 0.6023
10 213    24.35 0.6976
11 213    24.70 0.7154
12 213    24.35 0.6575
13 221    23.35 0.5958
14 221    23.00 0.6789
15 221    22.30 0.6965
16 221    23.15 0.6433
17 223    24.10 0.5479
18 223    24.55 0.5604
19 223    24.35 0.5446
20 223    24.40 0.5398
21 225    23.55 0.5615
22 225    24.55 0.6363
23 225    24.70 0.5753
24 225    23.85 0.6627
> summary(tubes)
      mcg           length10         weight
 Min.   :198.0   Min.   :22.30   Min.   :0.5398
 1st Qu.:205.0   1st Qu.:24.04   1st Qu.:0.5907
 Median :217.0   Median :24.55   Median :0.6112
 Mean   :214.2   Mean   :24.94   Mean   :0.6273
 3rd Qu.:223.0   3rd Qu.:25.48   3rd Qu.:0.6774
 Max.   :225.0   Max.   :28.20   Max.   :0.7271
> attach(tubes)
> # mcg is categorical. Make it a factor
```

```
> mcg = factor(mcg); table(mcg)
mcg
198 205 213 221 223 225
  4   4   4   4   4   4
>
> # Look what you have to do to get a table of means and standard
deviations.
> aggregate(length10,by=list(mcg),FUN=mean)
  Group.1       x
1     198 27.7750
2     205 25.3375
3     213 25.0625
4     221 22.9500
5     223 24.3500
6     225 24.1625

> # Aggregate returns a data frame. Get a nice table.
> Mean = aggregate(length10,by=list(mcg),FUN=mean)$x
> SD = sqrt(aggregate(length10,by=list(mcg),FUN=var)$x)
> Meanz = data.frame(table(mcg),Mean,SD); Meanz
  mcg Freq    Mean        SD
1 198    4 27.7750 0.3095696
2 205    4 25.3375 0.3092329
3 213    4 25.0625 1.2030344
4 221    4 22.9500 0.4564355
5 223    4 24.3500 0.1870829
6 225    4 24.1625 0.5513242
> contrasts(mcg) # See the dummy variable coding scheme
    205 213 221 223 225
198   0   0   0   0   0
205   1   0   0   0   0
213   0   1   0   0   0
221   0   0   1   0   0
223   0   0   0   1   0
225   0   0   0   0   1
```

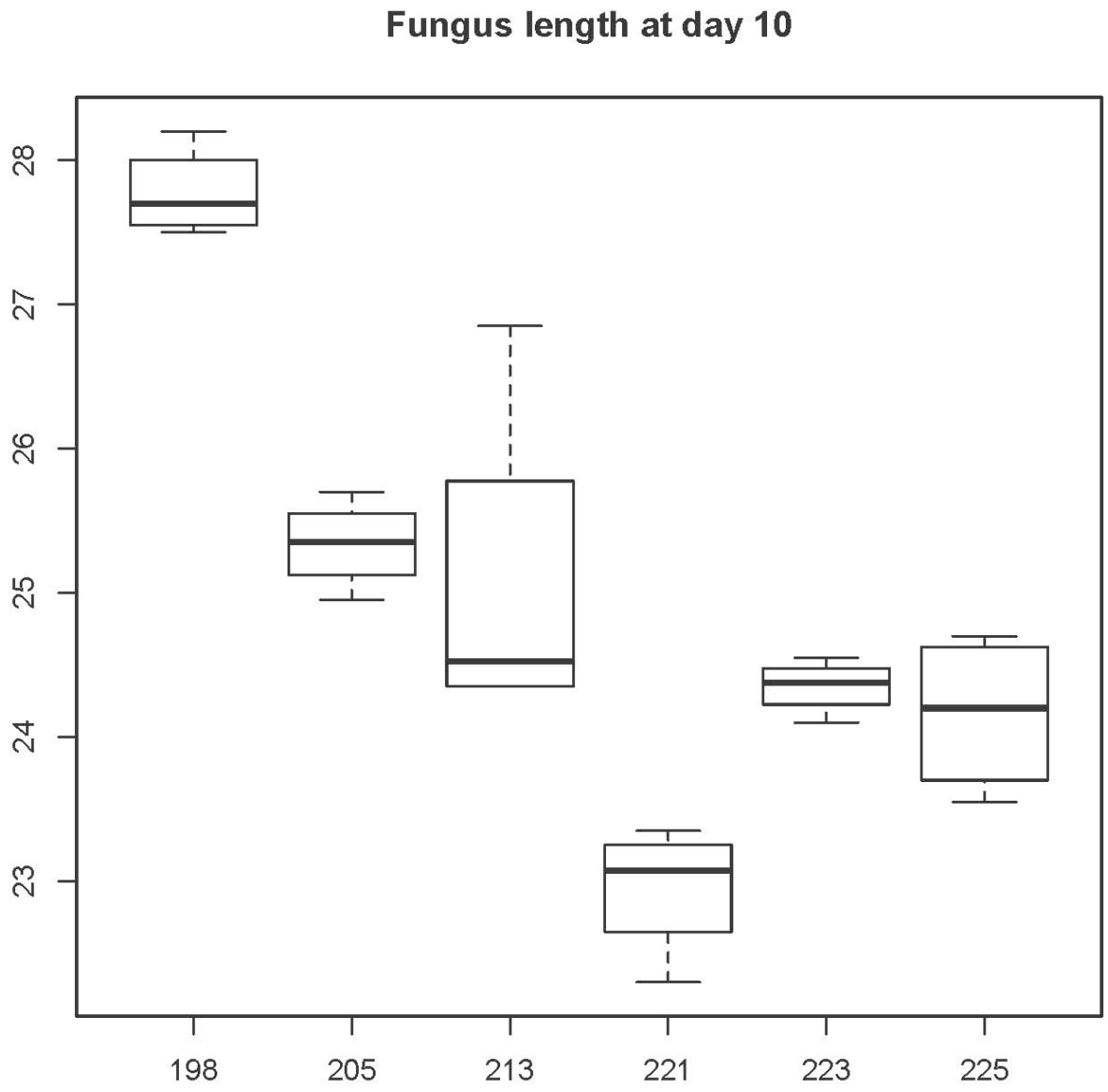
```
> mod1 = lm(length10 ~ mcg); summary(mod1)

Call:
lm(formula = length10 ~ mcg)

Residuals:
    Min      1Q  Median      3Q     Max
-0.7125 -0.3250  0.0125  0.2406  1.7875

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  27.7750     0.3018  92.030  < 2e-16 ***
mcg205       -2.4375     0.4268  -5.711 2.05e-05 ***
mcg213       -2.7125     0.4268  -6.355 5.49e-06 ***
mcg221       -4.8250     0.4268 -11.305 1.31e-09 ***
mcg223       -3.4250     0.4268  -8.025 2.35e-07 ***
mcg225       -3.6125     0.4268  -8.464 1.09e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6036 on 18 degrees of freedom
Multiple R-squared: 0.8889,   Adjusted R-squared: 0.8581
F-statistic: 28.81 on 5 and 18 DF,  p-value: 5.415e-08
```

```
> # Look at residuals
> dsr = rstudent(mod1)
> hist(dsr)
```

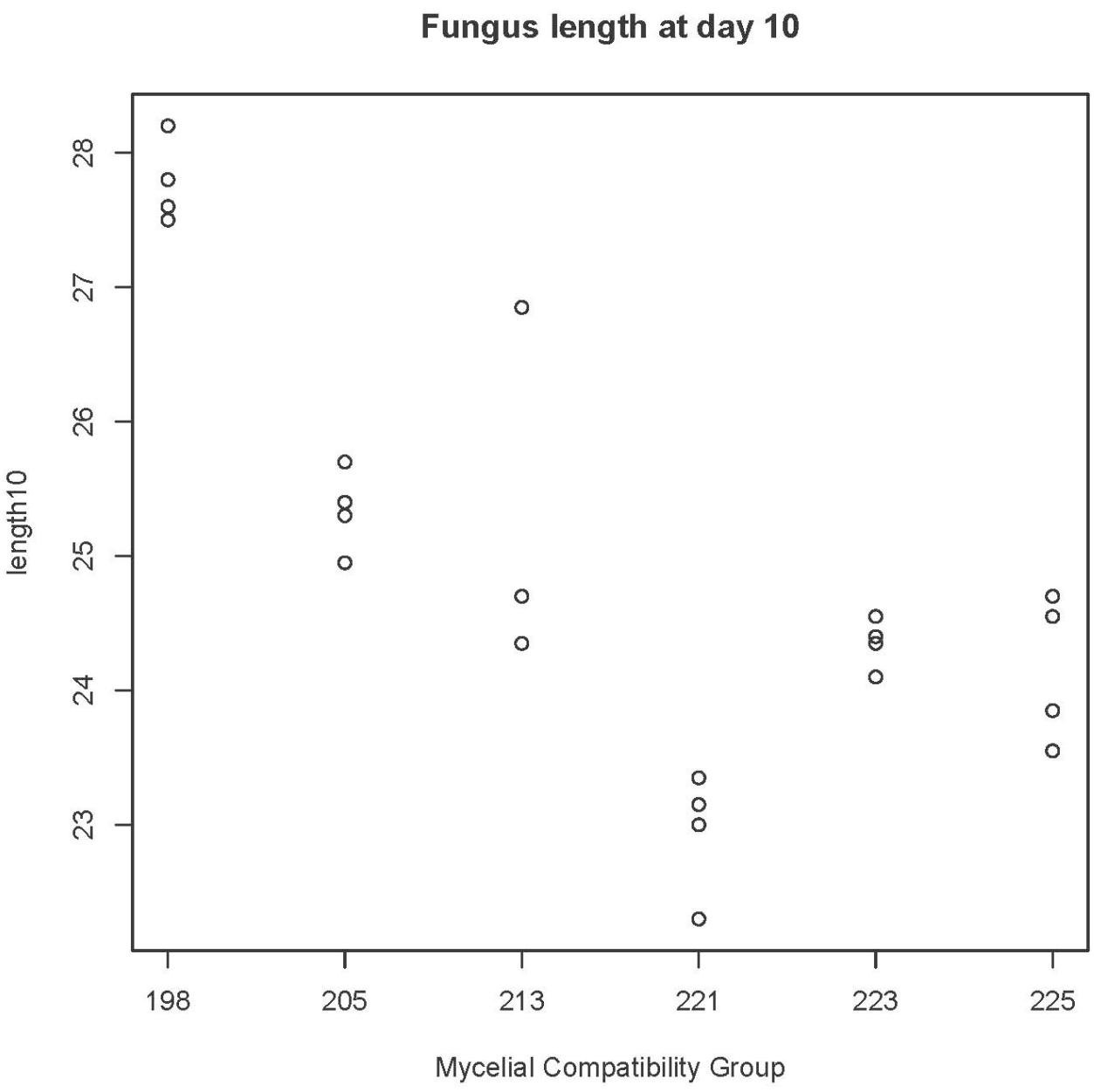


```
> sort(dsr)
         10          12          15          21           5          11
-1.39876217 -1.39876217 -1.26392161 -1.18478153 -0.73165799 -0.68311253
         24           4          17           3           8          19
-0.58682517 -0.51523021 -0.46775769 -0.32636103 -0.06972642  0.00000000
          1          20          14           7          16          18
 0.04648059  0.09297890  0.09297890  0.11624025  0.37334224  0.37334224
          6          22          13           2          23           9
 0.68311253  0.73165799  0.75604063  0.80503969  1.02997597  5.61400431

> # Likely outlier. Check with Bonferroni correction.
> n = length(mcg)
> bcrit = qt(1-0.05/(2*n),17); bcrit
[1] 3.626963
> # Could do it automatically for larger data sets.
> dsr[abs(dsr)>bcrit]
       9
5.614004
>
```

```
> # Should have done this plot first
> plot(mcg,length10); title("Fungus length at day 10") # Oops
```



```
> # Do it right and make it look good
> MCG = as.numeric(mcg) # Has values 1:6
> plot(MCG,length10,xaxt='n',xlab='Mycelial Compatibility Group')
> axis(side=1,at=1:6,labels=unique(mcg))
> title("Fungus length at day 10")
```




```
>
> sort(length10[mcg==213])
[1] 24.35 24.35 24.70 26.85
> length10[9] # That's the outlier.
[1] 26.85
```

```
> #  The botanists persuaded me to throw it out.
> length10[9] = NA # Missing value
>
> # Means and standard deviations again
> Mean = aggregate(length10,by=list(mcg),FUN=mean)$x
> SD = sqrt(aggregate(length10,by=list(mcg),FUN=var)$x)
> Meanz = data.frame(table(mcg),Mean,SD); Meanz
  mcg Freq    Mean        SD
1 198    4 27.7750 0.3095696
2 205    4 25.3375 0.3092329
3 213    4      NA        NA
4 221    4 22.9500 0.4564355
5 223    4 24.3500 0.1870829
6 225    4 24.1625 0.5513242
> # Irritating.

> Mean = aggregate(length10,by=list(mcg),FUN=mean,na.rm=T)$x
> SD = sqrt(aggregate(length10,by=list(mcg),FUN=var,na.rm=T)$x)
> Meanz = data.frame(table(mcg),Mean,SD); Meanz
  mcg Freq     Mean        SD
1 198    4 27.77500 0.3095696
2 205    4 25.33750 0.3092329
3 213    4 24.46667 0.2020726
4 221    4 22.95000 0.4564355
5 223    4 24.35000 0.1870829
6 225    4 24.16250 0.5513242
> # Mean and SD are correct now, but n is still wrong.
> mcg[9] = NA; table(mcg)
mcg
198 205 213 221 223 225
  4   4   3   4   4   4
> Mean = aggregate(length10,by=list(mcg),FUN=mean,na.rm=T)$x
> SD = sqrt(aggregate(length10,by=list(mcg),FUN=var,na.rm=T)$x)
> Meanz = data.frame(table(mcg),Mean,SD); Meanz
  mcg Freq     Mean        SD
1 198    4 27.77500 0.3095696
2 205    4 25.33750 0.3092329
3 213    3 24.46667 0.2020726
4 221    4 22.95000 0.4564355
5 223    4 24.35000 0.1870829
6 225    4 24.16250 0.5513242
>
>
```

```
> # Fit the model again
> mod2 = lm(length10 ~ mcg); summary(mod2)

Call:
lm(formula = length10 ~ mcg)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6500 -0.2125  0.0250  0.2167  0.5375

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  27.7750     0.1838 151.092  < 2e-16 ***
mcg205       -2.4375     0.2600  -9.376 3.95e-08 ***
mcg213       -3.3083     0.2808 -11.782 1.33e-09 ***
mcg221       -4.8250     0.2600 -18.560 1.01e-12 ***
mcg223       -3.4250     0.2600 -13.174 2.38e-10 ***
mcg225       -3.6125     0.2600 -13.896 1.04e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3677 on 17 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared: 0.9584,   Adjusted R-squared: 0.9462
F-statistic: 78.34 on 5 and 17 DF,  p-value: 3.953e-11

> # Compare R^2 = 0.8889 and F = 28.81
>
> # MCG198 clearly grows faster than the others. Compare all the others.
> source("http://www.utstat.utoronto.ca/~brunner/Rfunctions/ftest.txt")
> # Make a table
> L1 = rbind(c( 0, 1,-1, 0, 0, 0),
+            c( 0, 0, 1,-1, 0, 0),
+            c( 0, 0, 0, 1,-1, 0),
+            c( 0, 0, 0, 0, 1,-1)   )
> ftest(mod2,L1)
           F          df1          df2      p-value
2.164083e+01 4.000000e+00 1.700000e+01 1.731025e-06
```

```
>
> # Multiple comparisons. TukeyHSD is easiest.
> aov2 = aov(length10 ~ mcg); summary(aov2)
            Df Sum Sq Mean Sq F value   Pr(>F)
mcg          5  52.94  10.589   78.33 3.95e-11 ***
Residuals   17   2.30   0.135
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
1 observation deleted due to missingness
> TukeyHSD(aov2)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = length10 ~ mcg)

$mcg
               diff       lwr         upr     p adj
205-198 -2.4375000 -3.269076 -1.60592404 0.0000005
213-198 -3.3083333 -4.206538 -2.41012864 0.0000000
221-198 -4.8250000 -5.656576 -3.99342404 0.0000000
223-198 -3.4250000 -4.256576 -2.59342404 0.0000000
225-198 -3.6125000 -4.444076 -2.78092404 0.0000000
213-205 -0.8708333 -1.769038  0.02737136 0.0603187
221-205 -2.3875000 -3.219076 -1.55592404 0.0000007
223-205 -0.9875000 -1.819076 -0.15592404 0.0151272
225-205 -1.1750000 -2.006576 -0.34342404 0.0034487
221-213 -1.5166667 -2.414871 -0.61846197 0.0005762
223-213 -0.1166667 -1.014871  0.78153803 0.9981172
225-213 -0.3041667 -1.202371  0.59403803 0.8814129
223-221  1.4000000  0.568424  2.23157596 0.0005949
225-221  1.2125000  0.380924  2.04407596 0.0025656
225-223 -0.1875000 -1.019076  0.64407596 0.9765713

>
> # Bonferroni and Scheffe will be easier with cell means coding.
> # Make dummy variables ...
> # The missing values create such complications that I will quit R,
> # read the data again, and get rid of tube 9 another way.
>
```

```
> rm(list=ls())
> tubes =
read.table("http://www.utstat.toronto.edu/~brunner/appliedf14/code_n_data/l
ecture/littletubes.data")
> tubes = tubes[-9,] # All rows but 9, all columns
> tubes
   mcg length10 weight
1  198    27.80 0.5996
2  198    28.20 0.6040
3  198    27.60 0.6172
4  198    27.50 0.6053
5  205    24.95 0.6769
6  205    25.70 0.7057
7  205    25.40 0.7271
8  205    25.30 0.6029
10 213    24.35 0.6976
11 213    24.70 0.7154
12 213    24.35 0.6575
13 221    23.35 0.5958
14 221    23.00 0.6789
15 221    22.30 0.6965
16 221    23.15 0.6433
17 223    24.10 0.5479
18 223    24.55 0.5604
19 223    24.35 0.5446
20 223    24.40 0.5398
21 225    23.55 0.5615
22 225    24.55 0.6363
23 225    24.70 0.5753
24 225    23.85 0.6627
> attach(tubes)
> # Make indicator dummy variables
> mcg198 = mcg205 = mcg213 = mcg221 = mcg223 = mcg225 = numeric(23)
> mcg198[mcg==198] = 1
> mcg205[mcg==205] = 1
> mcg213[mcg==213] = 1
> mcg221[mcg==221] = 1
> mcg223[mcg==223] = 1
> mcg225[mcg==225] = 1
```

```
> cbind(mcg,mcg198,mcg205,mcg213,mcg221,mcg223,mcg225)
      mcg mcg198 mcg205 mcg213 mcg221 mcg223 mcg225
 [1,] 198      1      0      0      0      0      0
 [2,] 198      1      0      0      0      0      0
 [3,] 198      1      0      0      0      0      0
 [4,] 198      1      0      0      0      0      0
 [5,] 205      0      1      0      0      0      0
 [6,] 205      0      1      0      0      0      0
 [7,] 205      0      1      0      0      0      0
 [8,] 205      0      1      0      0      0      0
 [9,] 213      0      0      1      0      0      0
[10,] 213      0      0      1      0      0      0
[11,] 213      0      0      1      0      0      0
[12,] 221      0      0      0      1      0      0
[13,] 221      0      0      0      1      0      0
[14,] 221      0      0      0      1      0      0
[15,] 221      0      0      0      1      0      0
[16,] 223      0      0      0      0      1      0
[17,] 223      0      0      0      0      1      0
[18,] 223      0      0      0      0      1      0
[19,] 223      0      0      0      0      1      0
[20,] 225      0      0      0      0      0      1
[21,] 225      0      0      0      0      0      1
[22,] 225      0      0      0      0      0      1
[23,] 225      0      0      0      0      0      1
```

```
> # Fit a model with no intercept
> mod3 = lm(length10 ~ 0 + mcg198+mcg205+mcg213+mcg221+mcg223+mcg225)
> summary(mod3) # Look out

Call:
lm(formula = length10 ~ 0 + mcg198 + mcg205 + mcg213 + mcg221 +
    mcg223 + mcg225)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6500 -0.2125  0.0250  0.2167  0.5375

Coefficients:
       Estimate Std. Error t value Pr(>|t|)
mcg198  27.7750     0.1838   151.1   <2e-16 ***
mcg205  25.3375     0.1838   137.8   <2e-16 ***
mcg213  24.4667     0.2123   115.3   <2e-16 ***
mcg221  22.9500     0.1838   124.8   <2e-16 ***
mcg223  24.3500     0.1838   132.5   <2e-16 ***
mcg225  24.1625     0.1838   131.4   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3677 on 17 degrees of freedom
Multiple R-squared: 0.9998,   Adjusted R-squared: 0.9998
F-statistic: 1.759e+04 on 6 and 17 DF,  p-value: < 2.2e-16

> # Compare means
> Mean = aggregate(length10,by=list(mcg),FUN=mean)$x
> SD = sqrt(aggregate(length10,by=list(mcg),FUN=var)$x)
> Meanz = data.frame(table(mcg),Mean,SD); Meanz
  mcg Freq     Mean        SD
1 198    4 27.77500 0.3095696
2 205    4 25.33750 0.3092329
3 213    3 24.46667 0.2020726
4 221    4 22.95000 0.4564355
5 223    4 24.35000 0.1870829
6 225    4 24.16250 0.5513242
>
```

```
> # Overall F was 78.34: Check
> source("http://www.utstat.utoronto.ca/~brunner/Rfunctions/ftest.txt")
> L2 = rbind(c( 1,-1, 0, 0, 0, 0),
+            c( 0, 1,-1, 0, 0, 0),
+            c( 0, 0, 1,-1, 0, 0),
+            c( 0, 0, 0, 1,-1, 0),
+            c( 0, 0, 0, 0, 1,-1)   )
> ftest(mod3,L2)
           F          df1          df2      p-value
7.833542e+01 5.000000e+00 1.700000e+01 3.952683e-11
>
> # Bonferroni pairwise comparisons
> choose(6,2)
[1] 15
> # Make a matrix of p-values and test statistics
> testmatrix = diag(1,6,6) # Start with an identity matrix.
> labelz = as.character(unique(mcg))
> rownames(testmatrix) = labelz; colnames(testmatrix) = labelz
> testmatrix
    198 205 213 221 223 225
198   1   0   0   0   0   0
205   0   1   0   0   0   0
213   0   0   1   0   0   0
221   0   0   0   1   0   0
223   0   0   0   0   1   0
225   0   0   0   0   0   1
>
> for(i in 1:5)
+     {
+     for(j in (i+1):6)
+         {
+         LL = rbind(c(0,0,0,0,0,0))
+         LL[i]=1; LL[j]=-1
+         # print(LL) # Just to check
+         Fstat = ftest(mod3,L=LL)
+         testmatrix[i,j] = Fstat[1]; testmatrix[j,i]=min(Fstat[4]*15,1)
+         } # Next j
+     } # Next i
>
```

```
> # Test statistics (F with df=1,17) are in the upper triangle,
> # p-values in lower
> round(testmatrix,4)
    198     205      213      221      223      225
198   1 87.9091 138.8086 344.4604 173.5665 193.0903
205   0  1.0000   9.6176  84.3396  14.4284  20.4277
213   0  0.0973   1.0000  29.1728   0.1726   1.1733
221   0  0.0000   0.0007   1.0000  29.0002  21.7524
223   0  0.0215   1.0000   0.0007   1.0000   0.5202
225   0  0.0045   1.0000   0.0033   1.0000   1.0000
>
> # Scheffe Tests: Adjusted p-value is the tail area beyond F/(p-1)
> # Using the null distribution of the INITIAL test
> # Just re-compute the adjusted p-values
> for(i in 1:5)
+     {
+     for(j in (i+1):6)
+         {
+         testmatrix[j,i] = 1-pf(testmatrix[i,j]/5,5,17)
+         } # Next j
+     } # Next i
> round(testmatrix,4)
    198     205      213      221      223      225
198   1 87.9091 138.8086 344.4604 173.5665 193.0903
205   0  1.0000   9.6176  84.3396  14.4284  20.4277
213   0  0.1431   1.0000  29.1728   0.1726   1.1733
221   0  0.0000   0.0026   1.0000  29.0002  21.7524
223   0  0.0459   0.9993   0.0027   1.0000   0.5202
225   0  0.0128   0.9419   0.0099   0.9899   1.0000
>
```
