Stat 462 Lab 7 solutions

March 14, 2014

Question 7.4

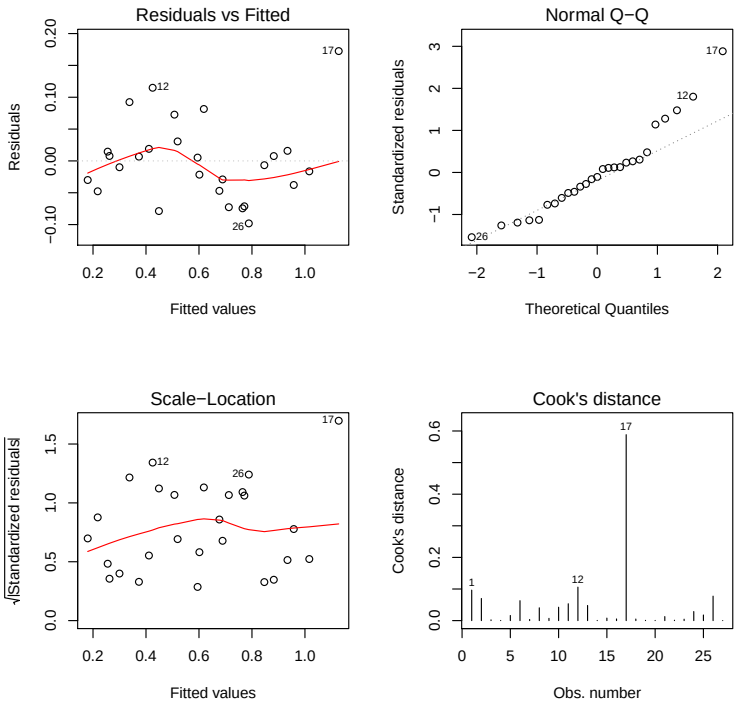
(a)

Fit three lines that have the same slope but different intercepts

```
library(DAAG)
# toycars$car is numeric.  Has to be factor, otherwise
# the car effect for car 2 will be twice car 1 and
# the car effect for car 3 will be thrice car 1.
# (not what the "car" effect means)
toycars$carfac <- as.factor( toycars$car )
lm.7.4.a <- lm( distance ~ carfac + angle, data = toycars )
summary( lm.7.4.a )

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.092524   0.034671   2.669  0.01372 *
carfac2      0.111111   0.031951   3.478  0.00204 **
carfac3     -0.082222   0.031951  -2.573  0.01699 *
angle        0.188541   0.009945  18.958 1.55e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.06778 on 23 degrees of freedom
Multiple R-squared:  0.9451,    Adjusted R-squared:  0.938
F-statistic: 132.1 on 3 and 23 DF,  p-value: 1.219e-14
```

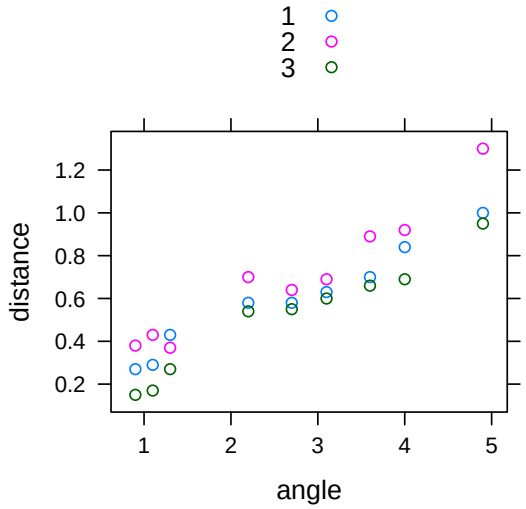
(b)

```
par( mfrow = c(2,2) )
plot( lm.7.4.a, which = 1:4 )
```



There is one point (17) with a moderately large Cook's distance. It is also a moderately large residual. Plotting the data,

```
library(lattice)
xyplot( distance ~ angle, groups = carfac, data = toycars, auto.key = TRUE )
```



It is difficult to argue that the point should be omitted based solely on a plot of the data. It could be due to chance alone rather than a symptom of some pathology.

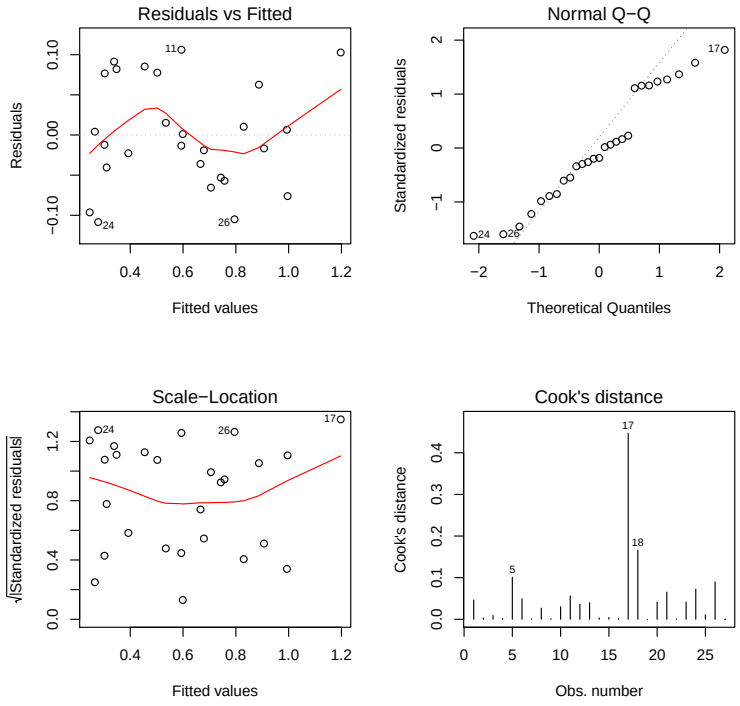
(c)

```
lm.7.4.c <- lm( distance ~ carfac:angle, data = toycars )
summary( lm.7.4.c )
```

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      0.10215    0.03037   3.364  0.00268 **
carfac1:angle    0.18190    0.01215  14.971 2.38e-13 ***
carfac2:angle    0.22349    0.01215  18.393 2.98e-15 ***
carfac3:angle    0.16024    0.01215  13.188 3.29e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.07011 on 23 degrees of freedom
Multiple R-squared:  0.9413,    Adjusted R-squared:  0.9336
F-statistic: 122.9 on 3 and 23 DF,  p-value: 2.653e-14
```

R^2 is indeed smaller for this model with a single intercept and three slopes.

```
par( mfrow = c(2,2) )
plot( lm.7.4.c, which = 1:4 )
```



The point is still somewhat influential, but is no longer an outlier.

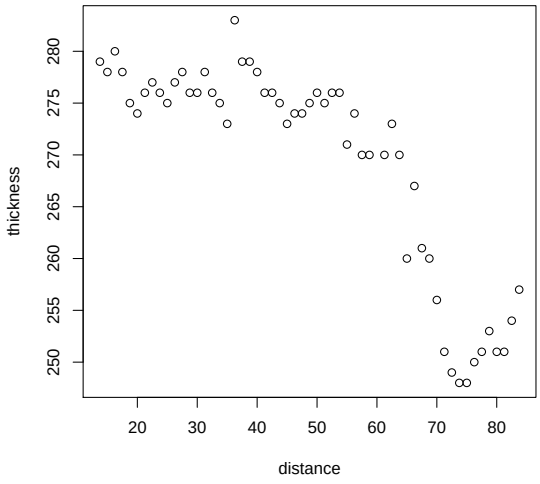
7.7 and 7.8

Polynomial regression (7.7)

Check fits of linear, quadratic, cubic, and quartic polynomials.

```
plot(geophones) # Always plot the data
lm.7.7.full <- lm( thickness ~ poly( distance, 4 ), data = geophones ) # quartic fit
summary( lm.7.7.full )
```

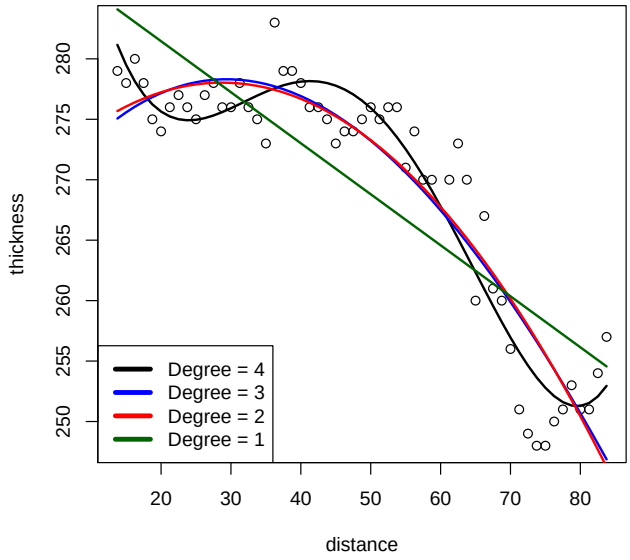
```
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         269.4107     0.4064 662.907  < 2e-16 ***
poly(distance, 4)1  -65.3414     3.0413 -21.485  < 2e-16 ***
poly(distance, 4)2  -29.7868     3.0413  -9.794 2.61e-13 ***
poly(distance, 4)3    1.9622     3.0413   0.645    0.522
poly(distance, 4)4   18.2548     3.0413   6.002 2.03e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 3.041 on 51 degrees of freedom
Multiple R-squared:  0.9209,    Adjusted R-squared:  0.9147
F-statistic: 148.5 on 4 and 51 DF,  p-value: < 2.2e-16
```



Because the poly() function creates an orthogonal polynomial basis, the design matrix is orthogonal, so terms can be removed from the model (starting from highest to lowest) and the coefficients will not change.

```
# predict function that drops higher order polynomial terms
pred.poly <- function( lm.obj, drop = 0 ){
  coefs <- coef( lm.obj ); k <- length( coefs )
  model.matrix( lm.obj )[,1:(k-drop)] %*% coefs[ 1:(k-drop) ]
}
plot( thickness ~ distance, data = geophones )
```

```
cols <- c( "black", "blue", "red", "darkgreen" )
for( i in 0:3 ) lines( geophones$distance, pred.poly( lm.7.7.full, drop = i )
      , col = cols[i+1], lwd = 2 )
legend( "bottomleft", col = cols, lty = 1
      , legend = paste( "Degree =", 4:1 ), lwd = 4 )
```



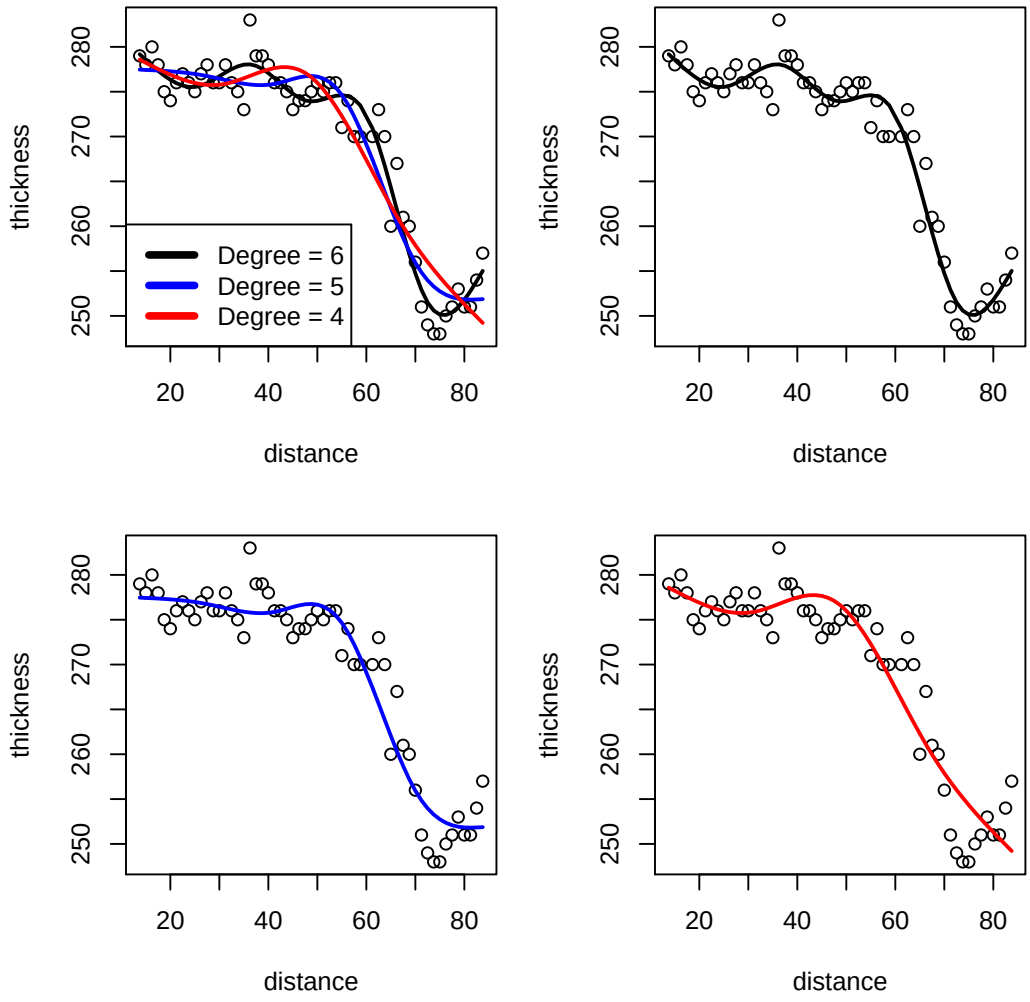
There is almost no difference between the quadratic and cubic curves. The quartic curve provides a significantly better fit, and tracks the data much more closely at small distances. It also would extrapolate much larger values than the quadratic or cubic outside the range of the data. None of the polynomial fits do a good job of describing the thickness at large values of distance.

Spline regression (7.8)

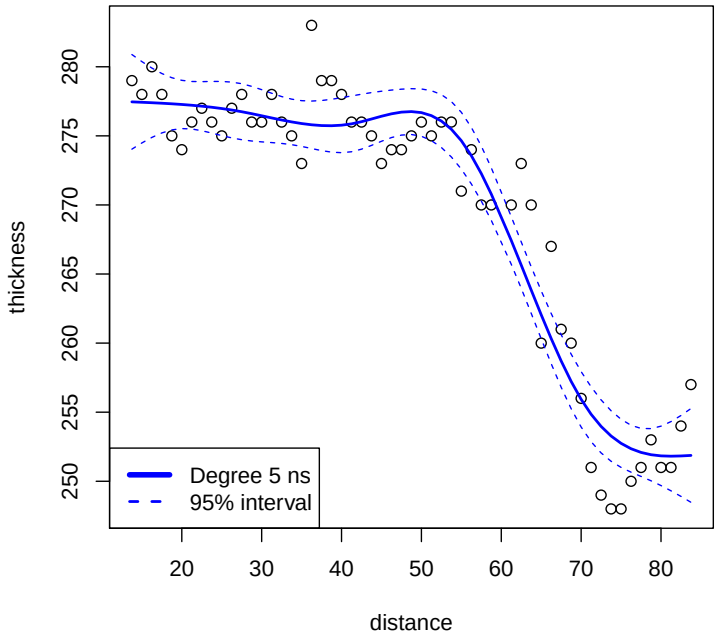
```
library(splines)
lm.7.8.6 <- lm( thickness ~ ns( distance, 6 ), data = geophones )
lm.7.8.5 <- lm( thickness ~ ns( distance, 5 ), data = geophones )
lm.7.8.4 <- lm( thickness ~ ns( distance, 4 ), data = geophones )
anova( lm.7.8.4, lm.7.8.5, lm.7.8.6 )

Analysis of Variance Table
Model 1: thickness ~ ns(distance, 4)
Model 2: thickness ~ ns(distance, 5)
Model 3: thickness ~ ns(distance, 6)
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1     51 611.31
2     50 468.09  1    143.21 26.488 4.669e-06 ***
3     49 264.92  1    203.17 37.578 1.475e-07 ***
---
```

Based on the ANOVA table, the degree 6 natural spline has a much smaller RSS compared to the other splines. Also, it fits better than the quartic polynomial.

```
par( mfrow = c(2,2), mar = c( 5, 4, 1, 1) + 0.1 )
plot( thickness ~ distance, data = geophones )
cols <- c( "black", "blue", "red" )
lines( geophones$distance, predict( lm.7.8.6 ), col = cols[1], lwd = 2 )
lines( geophones$distance, predict( lm.7.8.5 ), col = cols[2], lwd = 2 )
lines( geophones$distance, predict( lm.7.8.4 ), col = cols[3], lwd = 2 )
legend( "bottomleft", col = cols, lty = 1, legend = paste( "Degree =", 6:4 )
    , lwd = 4 )
plot( thickness ~ distance, data = geophones )
lines( geophones$distance, predict( lm.7.8.6 ), col = cols[1], lwd = 2 )
plot( thickness ~ distance, data = geophones )
lines( geophones$distance, predict( lm.7.8.5 ), col = cols[2], lwd = 2 )
plot( thickness ~ distance, data = geophones )
lines( geophones$distance, predict( lm.7.8.4 ), col = cols[3], lwd = 2 )
```



```
# Calculate pointwise confidence bounds for the df=5 case
plot( thickness ~ distance, data = geophones )
prVals <- predict( lm.7.8.5, interval = "confidence" )
lines( geophones$distance, prVals[,1], col = cols[2], lwd = 2 )
lines( geophones$distance, prVals[,2], col = cols[2], lwd = 1, lty = 2 )
lines( geophones$distance, prVals[,3], col = cols[2], lwd = 1, lty = 2 )
legend( "bottomleft", col = cols[c(2,2)], lty = c(1,2), lwd = c(4,2)
     , legend = c( "Degree 5 ns", "95% interval" ) )
```



8.2

Our goal is to come up with a decision rule for the use of CT using a risk threshold of 2.5%. Begin by fitting a logistic model with clinically important brain injury as the response. We can then examine the coefficients of each covariate to develop a decision rule.

```
glm.8.2 <- glm( clinically.important.brain.injury ~ ., data = head.injury
              , family = binomial( link = "logit" ) )
summary( glm.8.2 )
```

```
Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)           -4.4972     0.1629 -27.611  < 2e-16 ***
age.65                 1.3734     0.1827   7.518 5.56e-14 ***
amnesia.before         0.6893     0.1725   3.996 6.45e-05 ***
basal.skull.fracture   1.9620     0.2064   9.504  < 2e-16 ***
```

```
GCS.decrease              -0.2688     0.3680   -0.730 0.465152
GCS.13                     1.0613     0.2820    3.764 0.000168 ***
GCS.15.2hours              1.9408     0.1663   11.669  < 2e-16 ***
high.risk                  1.1115     0.1591    6.984 2.86e-12 ***
loss.of.consciousness      0.9554     0.1959    4.877 1.08e-06 ***
open.skull.fracture        0.6304     0.3151    2.001 0.045424 *
vomiting                   1.2334     0.1961    6.290 3.17e-10 ***
---
```

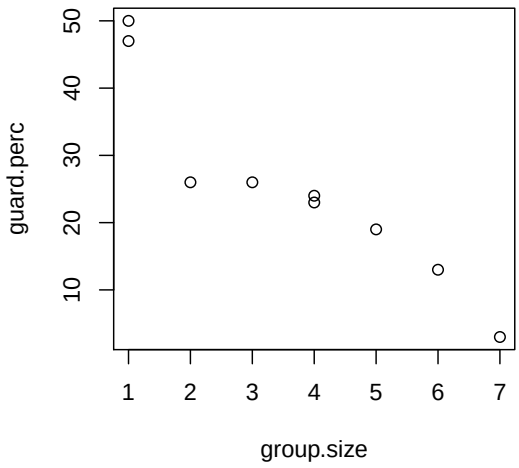
```
qlogis( 0.025 ) # Threshold on the logistic scale
[1] -3.663562
```

On the logistic scale, the 2.5% risk threshold corresponds to -3.66. This threshold corresponds to an increase of 0.834 compared to the intercept. We can achieve this risk threshold in any of the following ways:

1. Age > 65
2. Basal skull fracture present
3. GCS of 15 after two hours
4. Initial GCS of 13 without GCS decrease
5. Assessed by a clinician as high risk
6. Loss of consciousness without GCS decrease
7. Vomiting
8. Amnesia before combined with open skull fracture

7.16

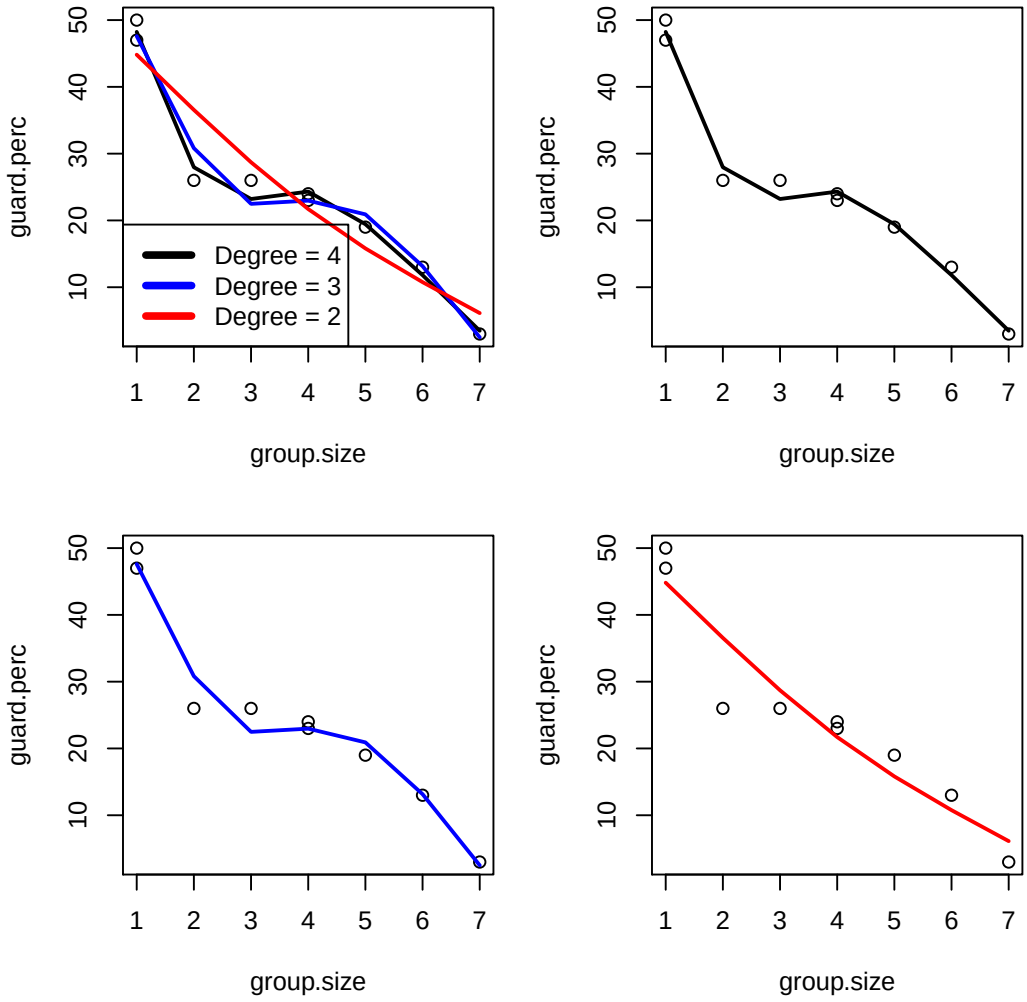
```
meerkats <- data.frame( guard.perc = c( 50, 47, 26, 26, 24, 23, 19, 13, 3 )
                      , group.size = c( 1, 1, 2, 3, 4, 4, 5, 6, 7 ) )
with( meerkats, plot( guard.perc ~ group.size ) )
# Several options here... try some splines
lm.7.16.2 <- lm( guard.perc ~ ns( group.size, 2 ), data = meerkats )
lm.7.16.3 <- lm( guard.perc ~ ns( group.size, 3 ), data = meerkats )
lm.7.16.4 <- lm( guard.perc ~ ns( group.size, 4 ), data = meerkats )
anova( lm.7.16.4, lm.7.16.3, lm.7.16.2 )
```




```
Analysis of Variance Table
Model 1: guard.perc ~ ns(group.size, 4)
Model 2: guard.perc ~ ns(group.size, 3)
Model 3: guard.perc ~ ns(group.size, 2)
  Res.Df     RSS Df Sum of Sq       F    Pr(>F)
1      4  19.946
2      5  46.712 -1   -26.767  5.3679 0.081415 .
3      6 182.266 -1  -135.553 27.1847 0.006454 **
---
```

Based on the ANOVA table, it looks like the order=3 spline does better than the order=2 spline but that the order=4 spline is not significantly better than order=3. Let's plot the fits.

```
par( mfrow = c(2,2), mar = c( 5, 4, 1, 1) + 0.1 )
plot( guard.perc ~ group.size, data = meerkats )
cols <- c( "black", "blue", "red" )
lines( meerkats$group.size, predict( lm.7.16.4 ), col = cols[1], lwd = 2 )
lines( meerkats$group.size, predict( lm.7.16.3 ), col = cols[2], lwd = 2 )
lines( meerkats$group.size, predict( lm.7.16.2 ), col = cols[3], lwd = 2 )
legend( "bottomleft", col = cols, lty = 1
    , legend = paste( "Degree =", 4:2 ), lwd = 4 )

plot( guard.perc ~ group.size, data = meerkats )
lines( meerkats$group.size, predict( lm.7.16.4 ), col = cols[1], lwd = 2 )
plot( guard.perc ~ group.size, data = meerkats )
lines( meerkats$group.size, predict( lm.7.16.3 ), col = cols[2], lwd = 2 )
plot( guard.perc ~ group.size, data = meerkats )
lines( meerkats$group.size, predict( lm.7.16.2 ), col = cols[3], lwd = 2 )
```



As suspected, the degree=4 and degree=3 fits look nearly identical.