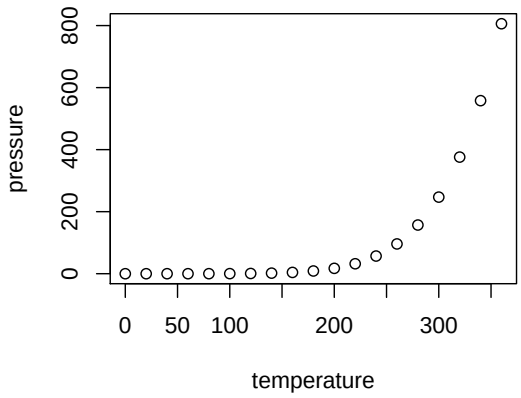
Stat 462 Lab 6 solutions

March 2, 2014

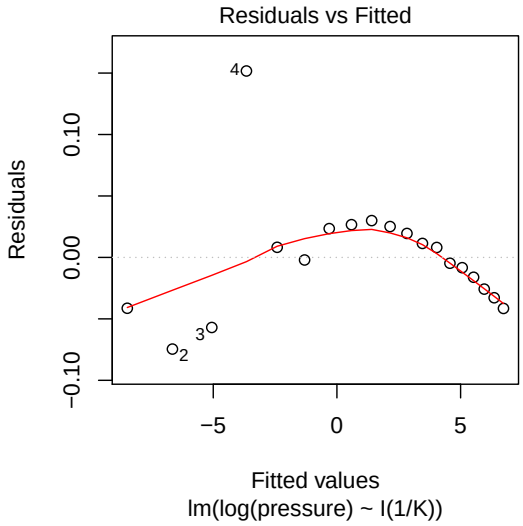
5.5 and 5.6

A plot of pressure as a function of temperature shows that pressure appears to increase as a power function.



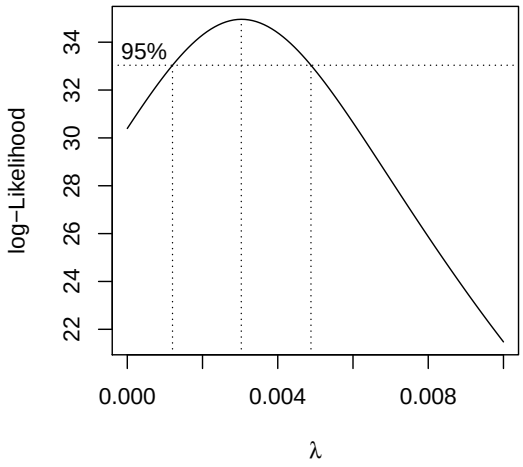
According to the text, the relationship should be log(pressure) ~ 1/K, where K is degrees Kelvin ("absolute temperature").

```
> pressure$K <- pressure$temperature + 273.15
> press.fit <- lm( log( pressure ) ~ I(1/K), data = pressure )
> plot( press.fit , which = 1 )
```



It looks like there is a systematic difference between the hypothesized relationship and the data. The logarithmic relationship proposed by the Claudius-Clapeyron equation is not quite capturing the shape. A generalized approach using Box-Cox should work better here.

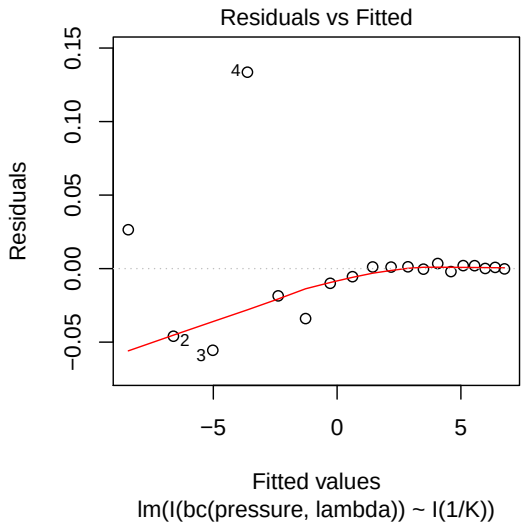
```
> boxcox( pressure ~ I(1/K), data = pressure, lambda = seq(0,0.01,0.001) )
```



With 95% confidence, the Box-Cox λ should lie between about 0.0012 and 0.0048. The maximum likelihood estimate is $\lambda = 0.003$.

```
> lambda <- press.bx$x[which.max(press.bx$y)]
# The Box–Cox transformation
> bc <- function( x, lam ) ( x^lam - 1 )/lam
```

```
# Fit using the maximum likelihood lambda value
> press.fit2 <- lm( I(bc( pressure, lambda )) ~ I(1/K), data = pressure )
> plot( press.fit2, which = 1 )
```



Although the relationship breaks down at low temperature (possibly due to additional noise present at low pressure?), the systematic departures seen previously are now all but gone. We may wish to downweight the observations at low temperature using a robust or weighted regression procedure as a next step.

5.10

```
> set.seed(5) # For reproducibility
> linsim <- function( x ){
            data.frame( x = x, y = 2 + 3*x + rnorm( length(x) ) ) }
> dat1 <- linsim( runif( 10, -1, 1 ) )
> dat2 <- linsim( c( rep(-1,5), rep(1,5) ) )
> summary( lm( y ~ x, data = dat1 ) )
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.6203     0.2287   7.085 0.000103 ***
x             3.0927     0.3621   8.540 2.72e-05 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
Residual standard error: 0.72 on 8 degrees of freedom
Multiple R-squared:  0.9011,    Adjusted R-squared:  0.8888

> summary( lm( y ~ x, data = dat2 ) )
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.1897     0.2205   9.931 8.94e-06 ***
```

```
x             3.7775      0.2205   17.132 1.37e-07 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
Residual standard error: 0.6973 on 8 degrees of freedom
Multiple R-squared:  0.9735,     Adjusted R-squared:  0.9701
```

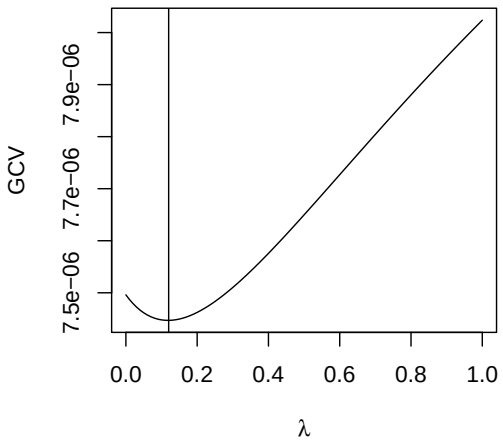
The standard error in the slope estimate for the second design is smaller than that first due to increased contrast in x. Note however that this does not guarantee a more accurate estimate for any given sample (the true slope is 3). What we lose however is the ability to verify that the relationship is indeed linear – this makes interpolation between (-1,1) risky with the second design.

6.7 and 6.8

```
> litters.lm <- lm( brainwt ~ bodywt + lsize , data = litters )
> vif( litters.lm )
bodywt  lsize
 11.33  11.33
```

The variance inflation factors are large. It appears that there is a strong linear relationship between the explanatory variables bodyweight and litter size, inflating the standard error estimates of the model coefficients.

```
# examine lambda values from 0 to 1
> litters.lmr <- lm.ridge( brainwt ~ bodywt + lsize , data = litters
                           , lambda = seq(0,1,0.01) )
# plot the GCV criterion versus lambda
> plot( litters.lmr$lambda, litters.lmr$GCV, type = "l"
      , xlab = expression(lambda), ylab = "GCV" )
> abline( v = litters.lmr$lambda[ which.min( litters.lmr$GCV ) ] )
# the optimal lambda is 0.12
> litters.lmr$lambda[ which.min( litters.lmr$GCV ) ]
[1] 0.12
> coef( litters.lmr )[ which.min( litters.lmr$GCV ), ]
                  bodywt        lsize
0.203819676 0.022016497 0.005646201
> coef( litters.lm )
(Intercept)      bodywt       lsize
0.178246962 0.024306344 0.006690331
```



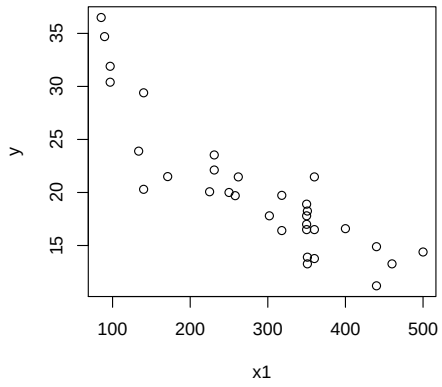
GCV is minimized at λ=0.12. We see that the coefficients have been shrunk towards 0 since $\lambda > 0$.

```
# Compute mean brain size at body weight = 7 and litter size = 10
> coef( litters.lmr )[ which.min( litters.lmr$GCV ), ]%*%c(1,7,10)
          [,1]
[1,] 0.4143972
# Compute mean brain size at body weight = 7 and litter size = 10
> predict( litters.lm, newdata = data.frame( bodywt = 7, lsize = 10 ) )
        1
0.4152947
# Set up function for bootstrapping a confidence interval on ridge fit
> litt.pred.lmr <- function(x, i) coef( lm.ridge( brainwt ~ bodywt + lsize
                  , data = x, subset = i, lambda = 0.12 ) )%*%c(1,7,10)
> boot.ci(boot( litters , litt.pred.lmr, 1000 ), type = "basic")
Level      Basic
95%   ( 0.4064,  0.4244 )
# Set up function for bootstrapping a confidence interval on lm fit
> litt.pred.lm <- function( x, i ) predict( lm( brainwt ~ bodywt + lsize
                  , data = x, subset = i )
                  , newdata = data.frame( bodywt = 7, lsize = 10 ) )
> boot.ci(boot( litters , litt.pred.lm, 1000 ), type = "basic")
Level      Basic
95%   ( 0.4070,  0.4245 )
# Get original lm fit confidence interval
> predict( litters.lm, newdata = data.frame( bodywt = 7, lsize = 10 )
         , interval = "confidence" )
        fit       lwr       upr
1 0.4152947 0.4062582 0.4243312
```

These methods all give approximately the same result. This is likely due to the good contrast and linearity of the data - a small λ value means that there was only a small amount of penalty applied to model coefficients.

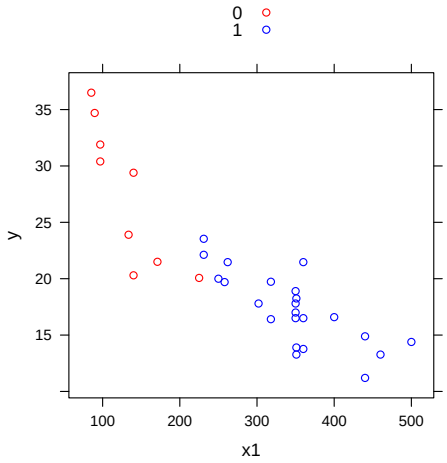
6.10

```
> plot( y ~ x1, data = table.b3 )
```



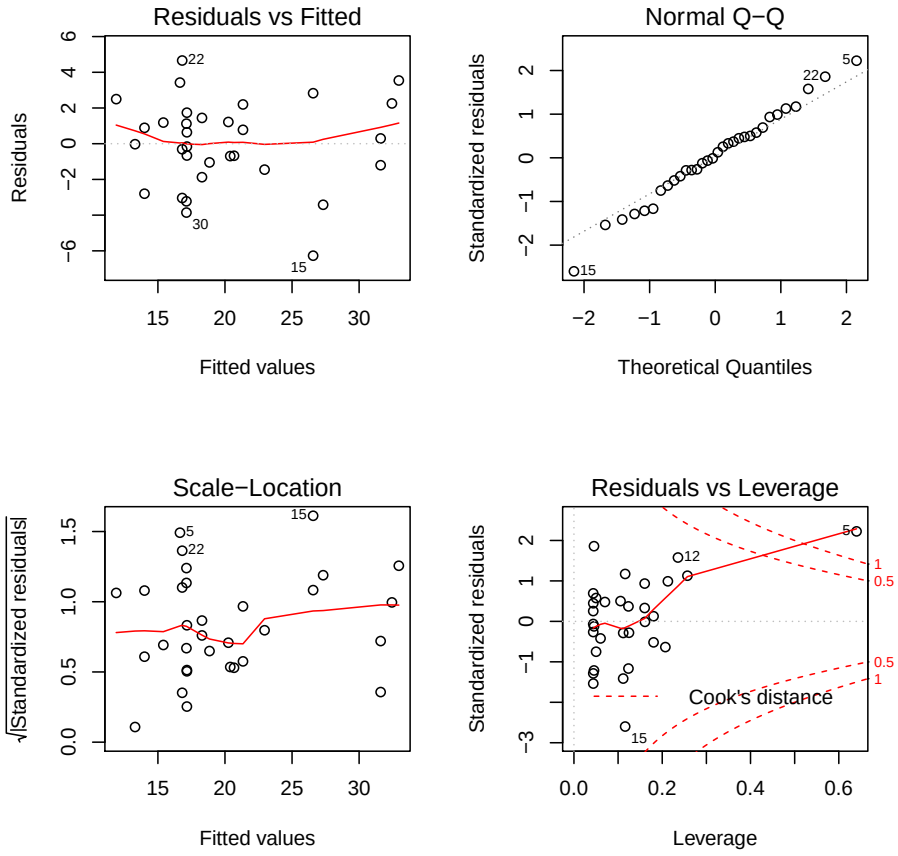
The relationship between mpg and displacement doesn't appear to be linear – more curved.

```
> xyplot( y ~ x1, data = table.b3, groups = x11, auto.key = TRUE )
```



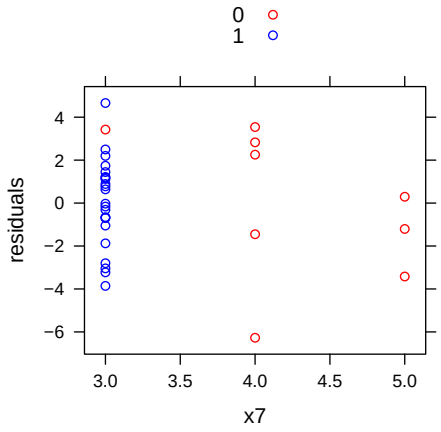
The type of transmission seems to take care of the curvy relationship – if we include type of transmission as a covariate, we should be able to model the relationship between mpg and displacement as linear conditional on type of transmission.

```
> b3.fit <- lm( y ~ x1*x11, data = table.b3 )
> par(mfrow = c(2,2))
> plot( b3.fit )
```



Point 5 has high leverage but is not an outlier. Still, it has a large amount of influence on the fit.

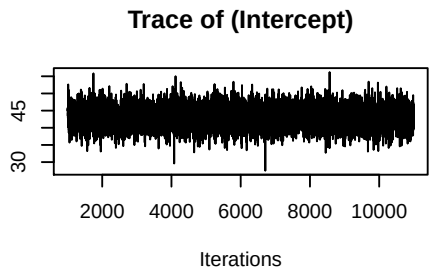
```
> xyplot(resid(b3.fit) ~ x7, group=x11, data=table.b3
        , ylab = "residuals", auto.key = TRUE )
```

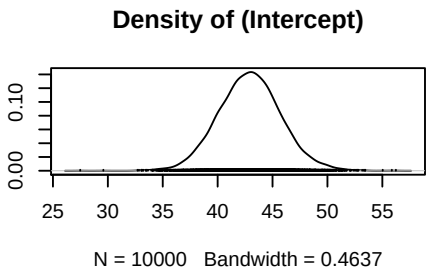


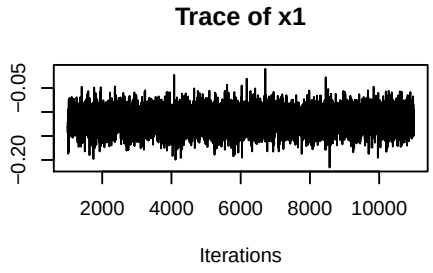
There is only one car with a 3-speed manual transmission. This turns out to be car 5, our influential point from the previous part.

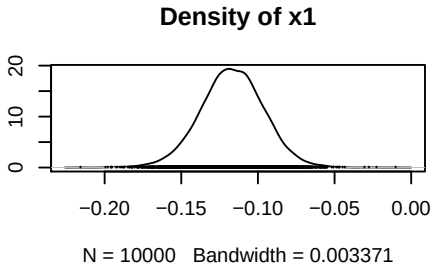
Stat 862 additional problem

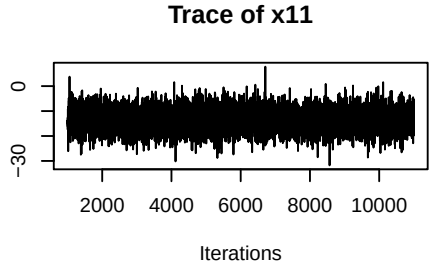
```
> library(MCMCpack)
# The next line uses an improper flat prior for model parameters
# Your choice might differ
> b3.mcmc <- MCMCregress( y ~ x1*x11, data = table.b3 )
> summary( b3.mcmc )
Iterations = 1001:11000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 10000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

                Mean      SD  Naive SE Time-series SE
(Intercept)  42.92382 2.83963 0.0283963      0.0283963
x1           -0.11680 0.02065 0.0002065      0.0002065
x11         -13.48131 3.97279 0.0397279      0.0397279
x1:x11        0.08172 0.02219 0.0002219      0.0002219
sigma2        7.05906 2.01691 0.0201691      0.0231913

2. Quantiles for each variable:
                2.5%       25%       50%       75%     97.5%
(Intercept)  37.48018  41.05054  42.93496  44.74877 48.64999
x1           -0.15816  -0.13012  -0.11678  -0.10323 -0.07676
x11         -21.45776 -16.03205 -13.44209 -10.90172 -5.67675
x1:x11        0.03868   0.06718   0.08152   0.09607  0.12607
sigma2        4.14535   5.62024   6.73214   8.14290 11.93661

> confint( b3.fit )
                  2.5 %      97.5 %
(Intercept)  37.31737558 48.52188798
x1           -0.15741527 -0.07613308
x11         -21.33806527 -5.58936167
x1:x11        0.03807661  0.12521715

# The confidence intervals are predictably similar.

> par(mfrow = c(5,2))
> plot( b3.mcmc, auto.layout = FALSE )
```

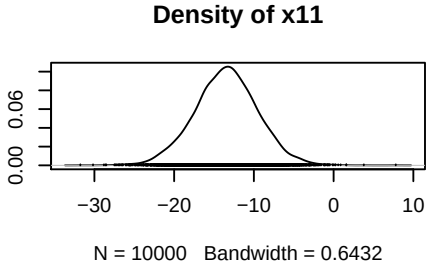

Trace of (Intercept)
30
45
2000
4000
6000
8000
10000
Iterations

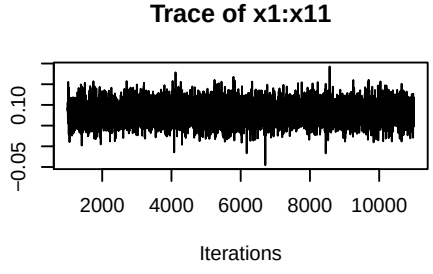

Density of (Intercept)
0.00
0.10
25
30
35
40
45
50
55
N = 10000 Bandwidth = 0.4637

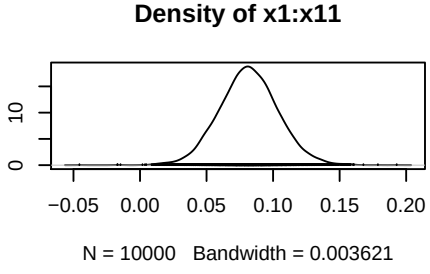

Trace of x1
−0.20
−0.05
2000
4000
6000
8000
10000
Iterations

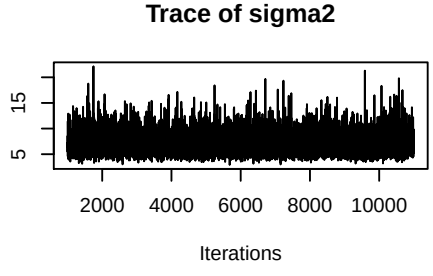

Density of x1
0
10
20
−0.20
−0.15
−0.10
−0.05
0.00
N = 10000 Bandwidth = 0.003371

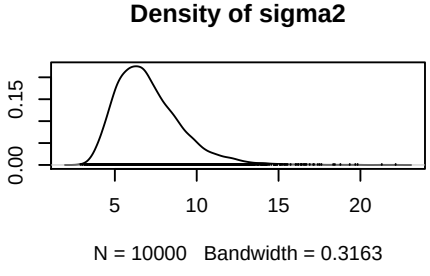

Trace of x11
−30
0
2000
4000
6000
8000
10000
Iterations


Density of x11
0.00
0.06
−30
−20
−10
0
10
N = 10000 Bandwidth = 0.6432


Trace of x1:x11
−0.05
0.10
2000
4000
6000
8000
10000
Iterations


Density of x1:x11
0
10
−0.05
0.00
0.05
0.10
0.15
0.20
N = 10000 Bandwidth = 0.003621


Trace of sigma2
5
15
2000
4000
6000
8000
10000
Iterations


Density of sigma2
0.00
0.15
5
10
15
20
N = 10000 Bandwidth = 0.3163