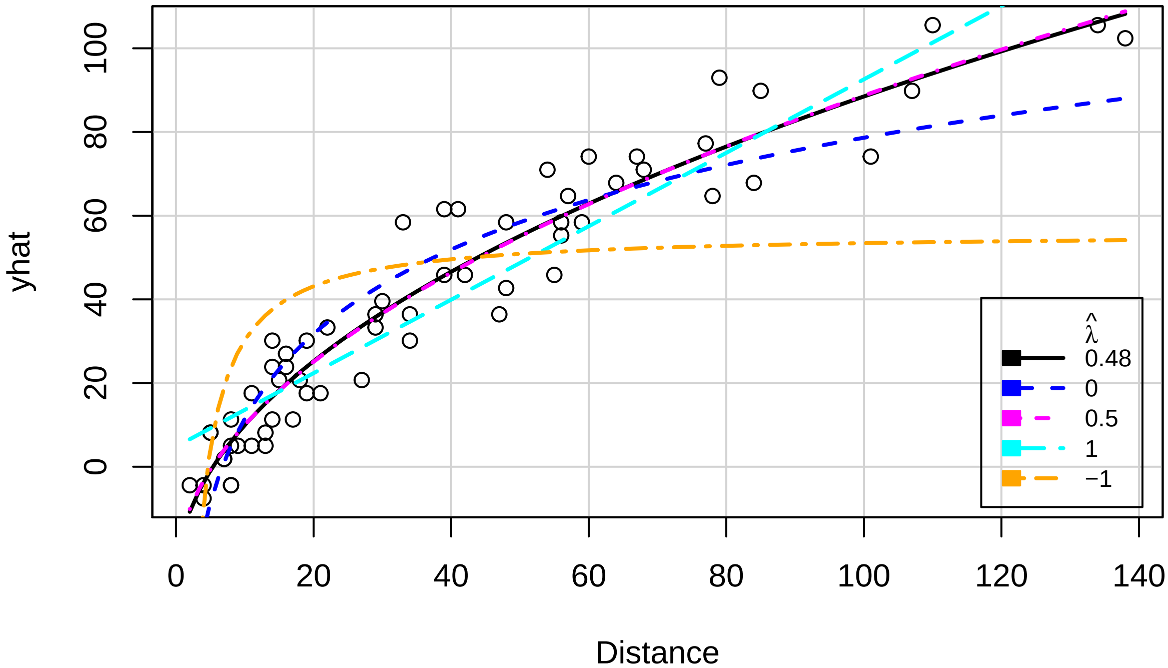
S631 HW8

Shibi He

ALR 8.2.1

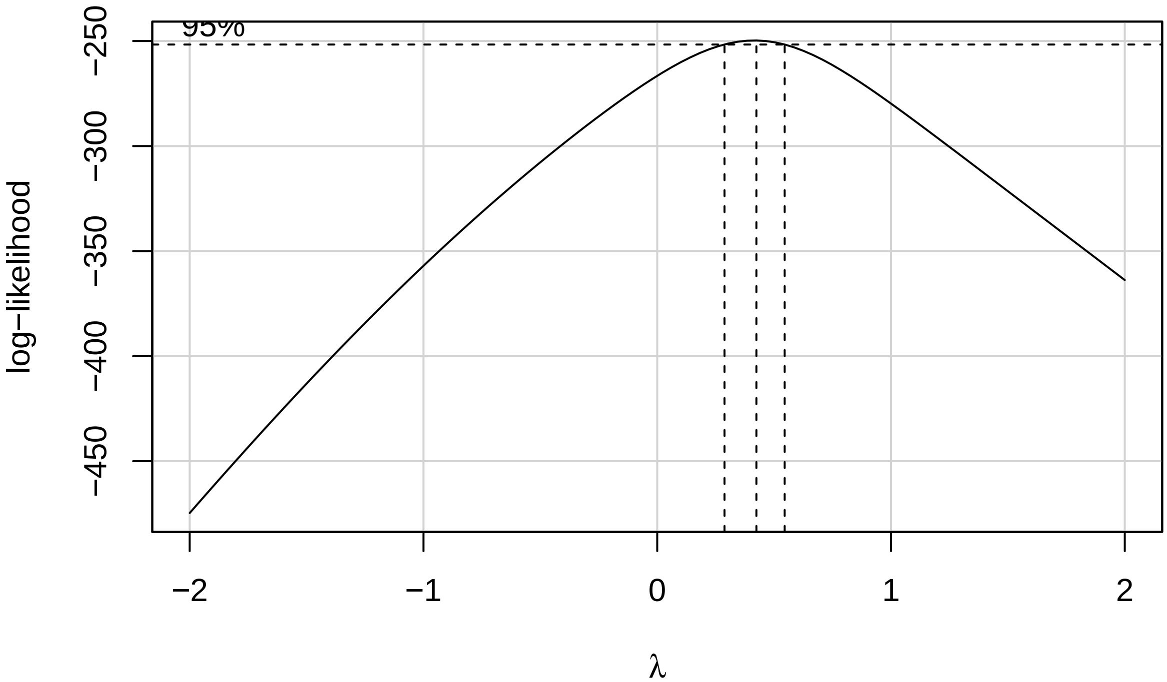
```r
m1 = lm(Distance ~ Speed, data=stopping)
p1=inverseResponsePlot(m1, c(0, 0.5, 1, -1))
```




```r
p1
```

```
##       lambda       RSS
## 1  0.4849737  4463.944
## 2  0.0000000  7890.434
## 3  0.5000000  4466.851
## 4  1.0000000  7293.835
## 5 -1.0000000 33149.061
```

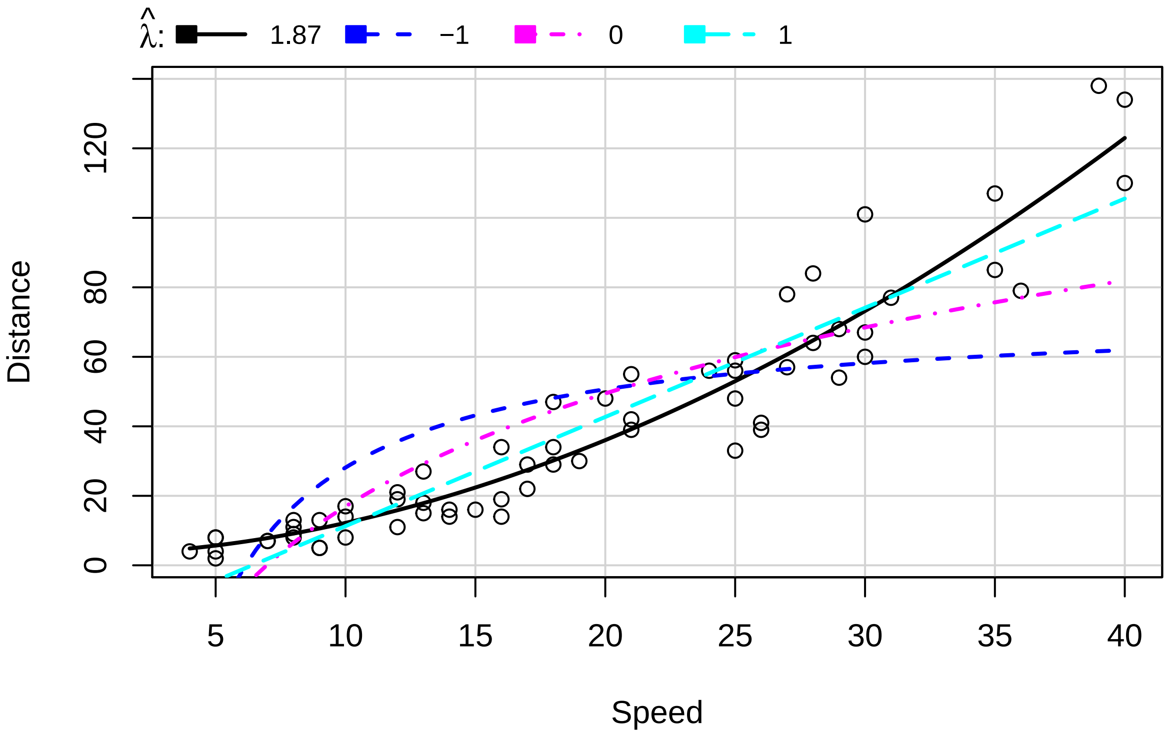
```r
boxCox(m1)
```



In the inverse fitted value plot, the best-fitting curve with $\hat{\lambda} = 0.48$ and the square root curve for $\lambda = 0.5$ is nearly identical, and so a square root transformation of Distance is suggested. Similarly, using the Box-Cox method, the square root transformation is in the 95% confidene interval for $\hat{\lambda} = 0.48$, agreeing with the inverse fitted value plot. So the approporiate transformation for Distance is a square root transformation.

ALR 8.2.2

```
with(stopping, invTranPlot(Speed, Distance))
```

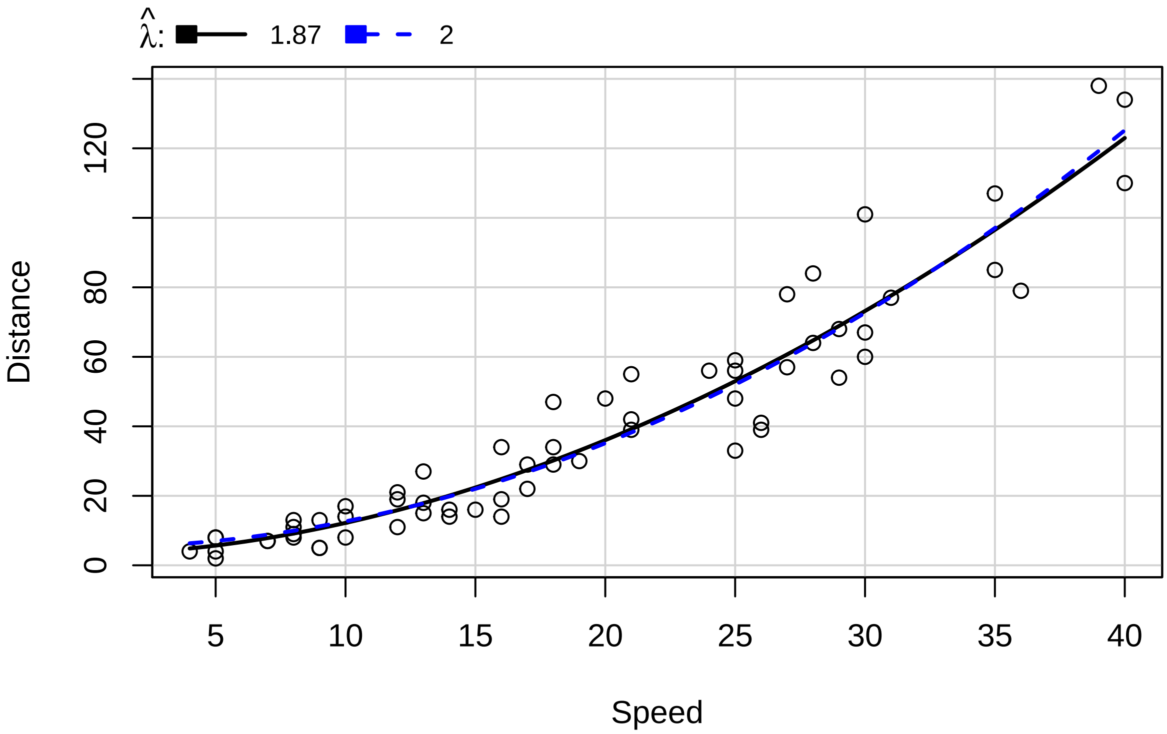



```
##      lambda       RSS
## 1  1.868443  5823.372
## 2 -1.000000 34951.108
## 3  0.000000 18844.172
## 4  1.000000  8310.166
```

The above graph shows that none of the fitted lines for $\lambda \in \{-1, 0, 1\}$ is close to the best-fitting line (i.e. $\hat{\lambda} = 1.87$) and the RSS(1.87) is much lower than the RSS for the other three values. Therefore, none of these transformation is adequate.

ALR 8.2.3

```
with(stopping, invTranPlot(Speed, Distance, 2))
```




```
##     lambda      RSS
## 1 1.868443 5823.372
## 2 2.000000 5869.232
```

Using $\lambda = 2$ is almost identical to the best-fitting line ($\hat{\lambda} = 1.87$) and RSS(2) is very close ot RSS(1.87), suggesting using a quadratic polynomial for the regressors.

```r
m2 = lm(Distance ~ Speed + I(Speed^2), data=stopping)
summary(m2)
```

```
##
## Call:
## lm(formula = Distance ~ Speed + I(Speed^2), data = stopping)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22.5192  -5.4527  -0.5519   3.8442  27.9373
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.58036    5.10266   0.310    0.758
## Speed        0.41607    0.55641   0.748    0.458
## I(Speed^2)   0.06556    0.01303   5.033 4.83e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.927 on 59 degrees of freedom
## Multiple R-squared:  0.9144, Adjusted R-squared:  0.9115
## F-statistic: 315.3 on 2 and 59 DF,  p-value: < 2.2e-16
```

ALR 8.2.4

Hald (1960)'s model:

```
m3 = lm(Distance ~ Speed + I(Speed^2), data=stopping, weights= 1/Speed^2 )
summary(m3)$coefficients
```

```
##                Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 1.50605213 2.03543726 0.7399158 4.622853e-01
## Speed       0.41967996 0.34326252 1.2226210 2.263345e-01
## I(Speed^2)  0.06556898 0.01056769 6.2046646 5.900848e-08
```

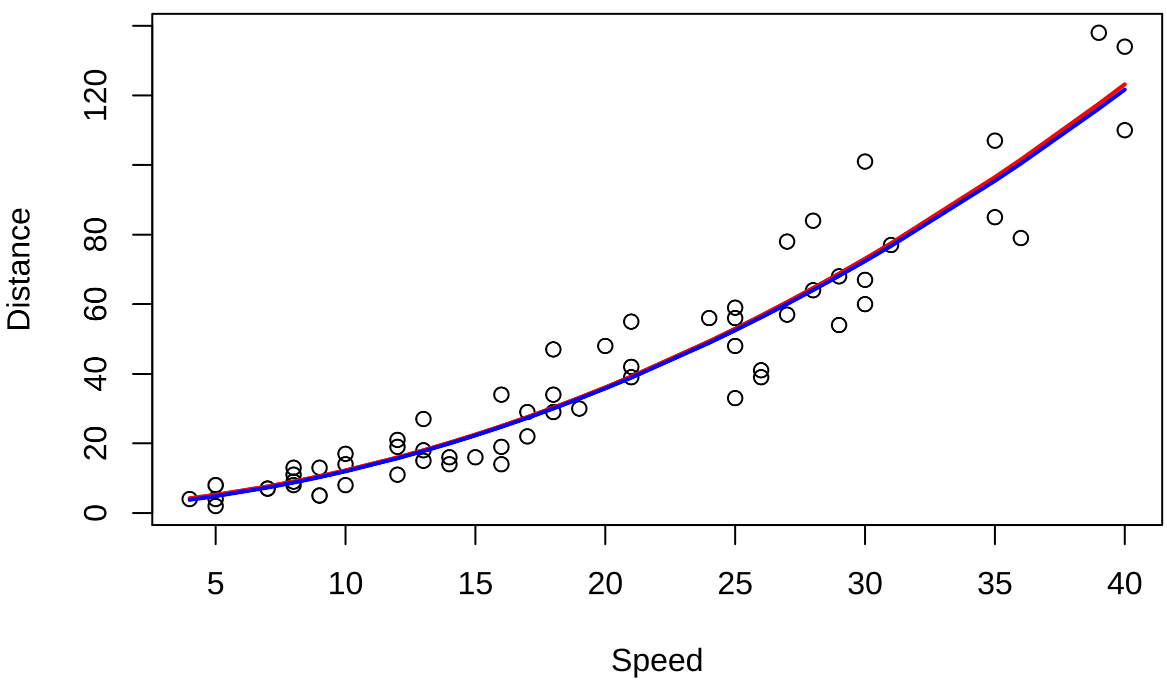
Square root transformation of Distance: (model in 8.2.1)

```
m4 = lm(sqrt(Distance) ~ Speed, data=stopping)
summary(m4)$coefficients
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 0.9323957 0.19790924  4.711229 1.503661e-05
## Speed       0.2524660 0.00927402 27.222931 1.826107e-35
```

Plot the fitted values:

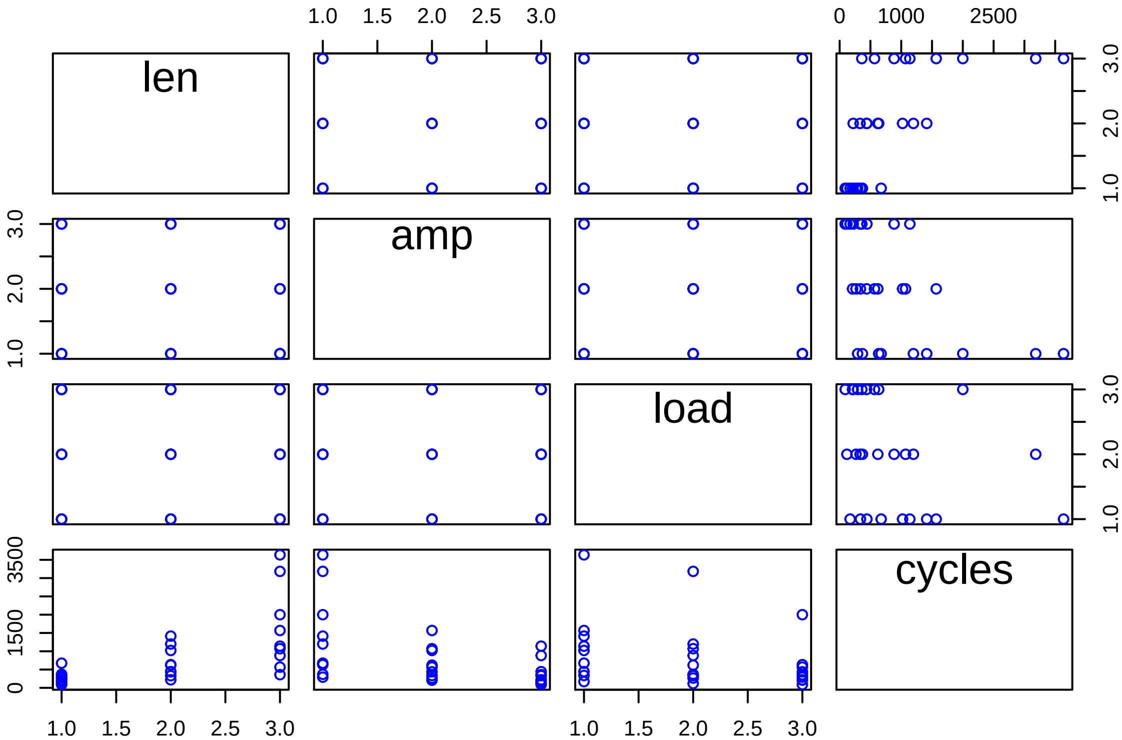
```
# Plot fitted curve from Hald's model
newdata = data.frame(Speed=stopping$Speed, fitted.dist=fitted(m3))
newdata = newdata[order(newdata$Speed), ]

# Plot fitted value from square root transformation of distance
newdata2 = data.frame(Speed=stopping$Speed, fitted.dist=fitted(m4)^2)
newdata2 = newdata2[order(newdata2$Speed), ]

plot(Distance ~ Speed, stopping)
lines(newdata, col="red", lwd=2)
lines(newdata2, col="blue", lwd=2)
```



The fit of these two models are almost identical as these two models are essentially the same. They both fit the data very well.

ALR 8.6.1

```
Wool$len=factor(Wool$len)
Wool$amp=factor(Wool$amp)
Wool$load=factor(Wool$load)
scatterplotMatrix(Wool, smooth = F,
                  regLine = F, diagonal = F)
```



These scatter plots show that as *len* increases (i.e.moving from level 1 to 3), *cycles* increases. As *amp* and *load* increases (i.e. moving from level 1 to 3), *cycles* decreases. Moreover, the variation in *cycles* also increases with *len* and decreases with *amp* and *load*.

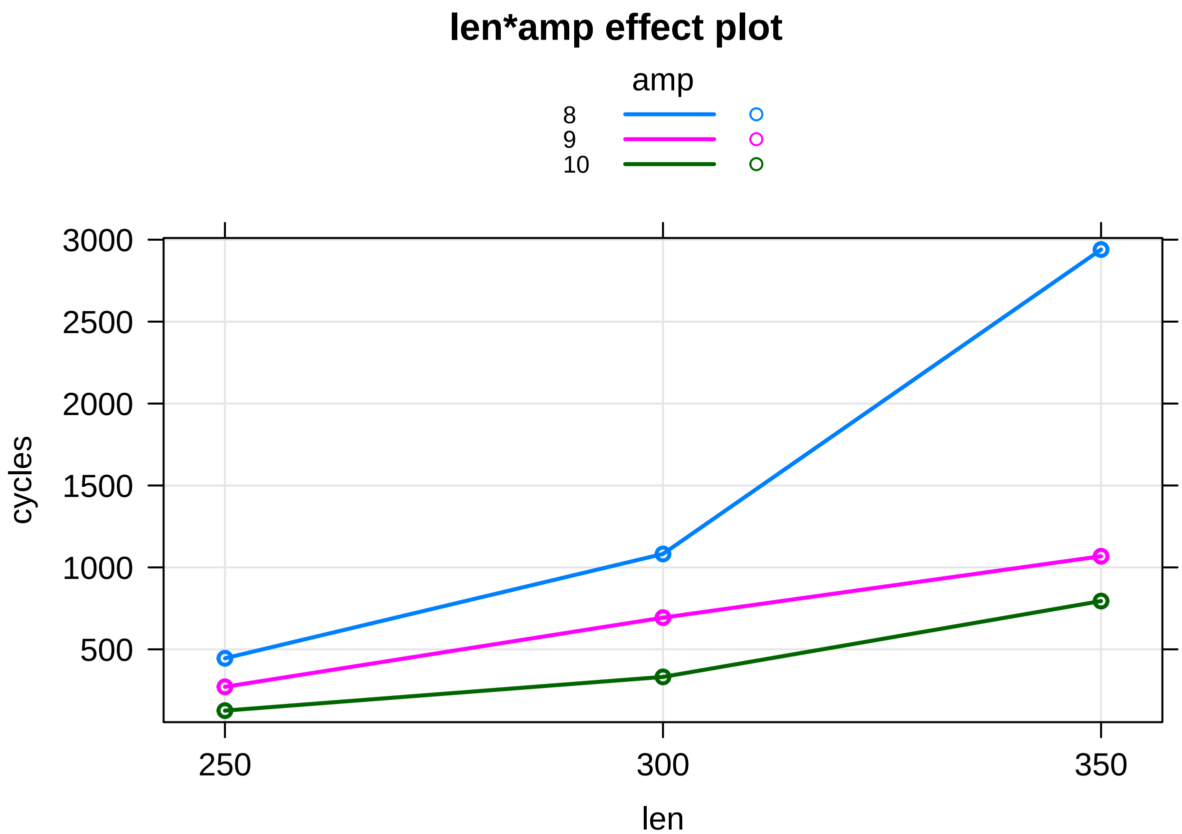
ALR 8.6.2

```
m1 = lm(cycles ~ len + amp + load + len:amp + len:load + amp:load, data=Wool)
summary(m1)$coefficients
```

```
##                  Estimate Std. Error       t value     Pr(>|t|)
## (Intercept)    6.825926e+02   92.36715  7.389993e+00 7.692870e-05
## len300         7.808889e+02  116.06503  6.728029e+00 1.483628e-04
## len350         2.895333e+03  116.06503  2.494579e+01 7.132916e-09
## amp9          -2.944444e+02  116.06503 -2.536892e+00 3.487913e-02
## amp10         -5.713333e+02  116.06503 -4.922528e+00 1.160310e-03
## load45        -2.041111e+02  116.06503 -1.758593e+00 1.166966e-01
## load50        -5.076667e+02  116.06503 -4.373985e+00 2.367799e-03
## len300:amp9   -2.146667e+02  127.14287 -1.688389e+00 1.298127e-01
## len350:amp9   -1.698000e+03  127.14287 -1.335506e+01 9.449907e-07
```

```
## len300:amp10  -4.310000e+02  127.14287 -3.389887e+00 9.501544e-03
## len350:amp10  -1.826000e+03  127.14287 -1.436180e+01 5.395545e-07
## len300:load45 -1.003333e+02  127.14287 -7.891385e-01 4.527816e-01
## len350:load45 -2.593333e+02  127.14287 -2.039700e+00 7.570906e-02
## len300:load50 -3.323333e+02  127.14287 -2.613857e+00 3.094431e-02
## len350:load50 -9.426667e+02  127.14287 -7.414232e+00 7.516557e-05
## amp9:load45    5.907341e-13  127.14287  4.646223e-15 1.000000e+00
## amp10:load45   1.843333e+02  127.14287  1.449813e+00 1.851551e-01
## amp9:load50    3.613333e+02  127.14287  2.841947e+00 2.174672e-02
## amp10:load50   5.716667e+02  127.14287  4.496254e+00 2.012033e-03
```

```
plot(Effect(c("len", "amp"), m1), rug=FALSE, grid=TRUE, multiline=TRUE)
```



The effects plot suggests that as *len* moves from level 1 to level 3 (i.e. increases from 250mm to 350mm), the expected *cycles* for all three different *amp* increases. Moreover, for each specific *len*, as *amp* moves from level 1 to level 3 (i.e. increases from 8mm to 10mm), the expected *cycles* decreases.

ALR 8.6.3

Fit the first-order mean function:

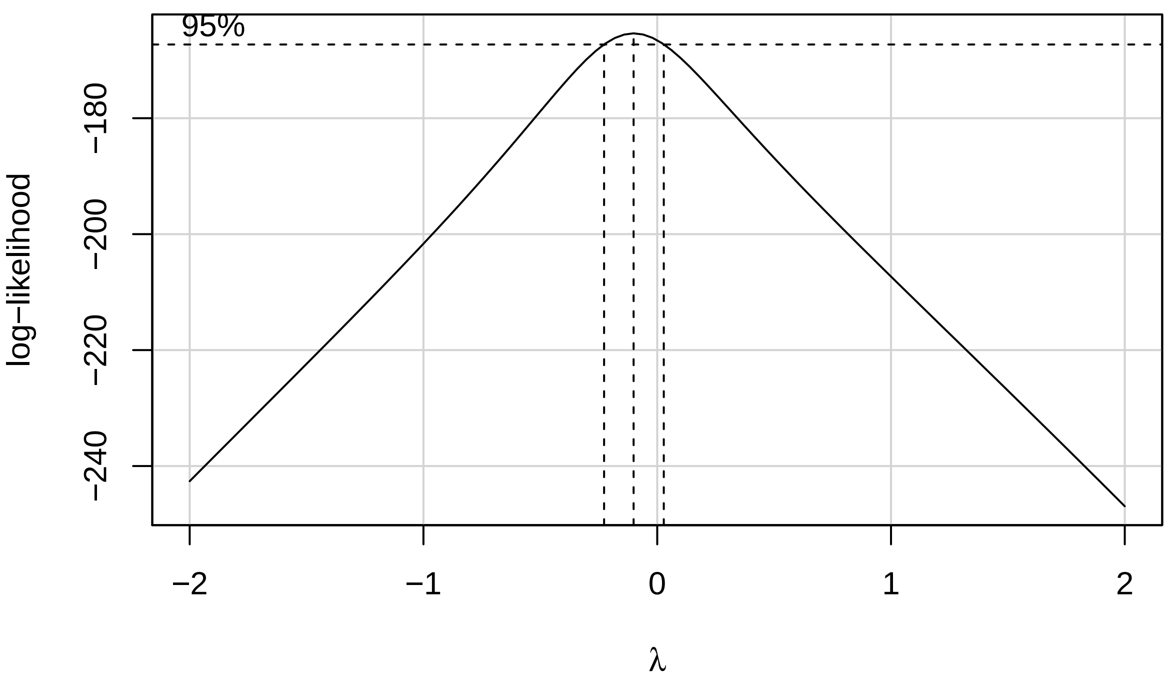
```
m2 = lm(cycles ~ len + amp + load, data=Wool)
summary(m2)$coefficients
```

```
##               Estimate Std. Error  t value     Pr(>|t|)
## (Intercept) 1203.3704   246.0169 4.891413 8.825572e-05
## len300       421.4444   227.7674 1.850328 7.909626e-02
## len350      1320.0000   227.7674 5.795385 1.137842e-05
```

```
## amp9          -811.5556   227.7674 -3.563089 1.948342e-03
## amp10        -1071.6667   227.7674 -4.705092 1.358079e-04
## load45        -262.5556   227.7674 -1.152735 2.626111e-01
## load50        -621.6667   227.7674 -2.729392 1.291836e-02
```

Use Box-Cox method to select a transformation for *cycles*:

```r
boxCox(m2)
```



The above graph shows that $\lambda = 0$ is within the 95% confidence interval of the $\hat{\lambda}$, suggesting that a log transformation of the response *cycles* might be adequate.

ALR 8.6.4

In the transformed scale, fit both the first-order and second-order model:

```r
# first-order mean model
model1 = lm(log(cycles) ~ len + amp + load, data=Wool)
summary(model1)$coefficients
```

```
##               Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)  6.4828712 0.09643609  67.224532 4.868964e-25
## len300       0.9183326 0.08928247  10.285699 1.965988e-09
## len350       1.6647683 0.08928247  18.646082 4.098808e-14
## amp9        -0.6552099 0.08928247  -7.338617 4.305151e-07
## amp10       -1.2617320 0.08928247 -14.131912 7.187036e-12
## load45      -0.3252896 0.08928247  -3.643376 1.616841e-03
## load50      -0.7852390 0.08928247  -8.794996 2.617057e-08
```

```r
# second-order model
model2 = lm(log(cycles) ~ len + amp + load + len:amp + len:load + amp:load, data=Wool)
summary(model2)$coefficients
```

```
##                   Estimate Std. Error      t value     Pr(>|t|)
## (Intercept)    6.362917021  0.1208067 52.670245023 1.871514e-11
```

```
## len300         0.913780242  0.1518010  6.019591433 3.164168e-04
## len350         1.963516108  0.1518010 12.934800080 1.208053e-06
## amp9          -0.413378703  0.1518010 -2.723161201 2.612066e-02
## amp10         -1.203297916  0.1518010 -7.926809420 4.665224e-05
## load45        -0.375587937  0.1518010 -2.474211877 3.845709e-02
## load50        -0.609676207  0.1518010 -4.016284777 3.861088e-03
## len300:amp9   -0.001114412  0.1662897 -0.006701631 9.948170e-01
## len350:amp9   -0.614678024  0.1662897 -3.696428546 6.073737e-03
## len300:amp10   0.064963631  0.1662897  0.390665374 7.062419e-01
## len350:amp10  -0.152965546  0.1662897 -0.919873805 3.845366e-01
## len300:load45  0.083463114  0.1662897  0.501913890 6.292480e-01
## len350:load45  0.145058615  0.1662897  0.872324670 4.084485e-01
## len300:load50 -0.133655179  0.1662897 -0.803748953 4.447658e-01
## len350:load50 -0.273658396  0.1662897 -1.645672479 1.384496e-01
## amp9:load45   -0.074415690  0.1662897 -0.447506291 6.663786e-01
## amp10:load45  -0.003211004  0.1662897 -0.019309696 9.850670e-01
## amp9:load50   -0.035285384  0.1662897 -0.212192229 8.372636e-01
## amp10:load50  -0.084089435  0.1662897 -0.505680331 6.267170e-01
```

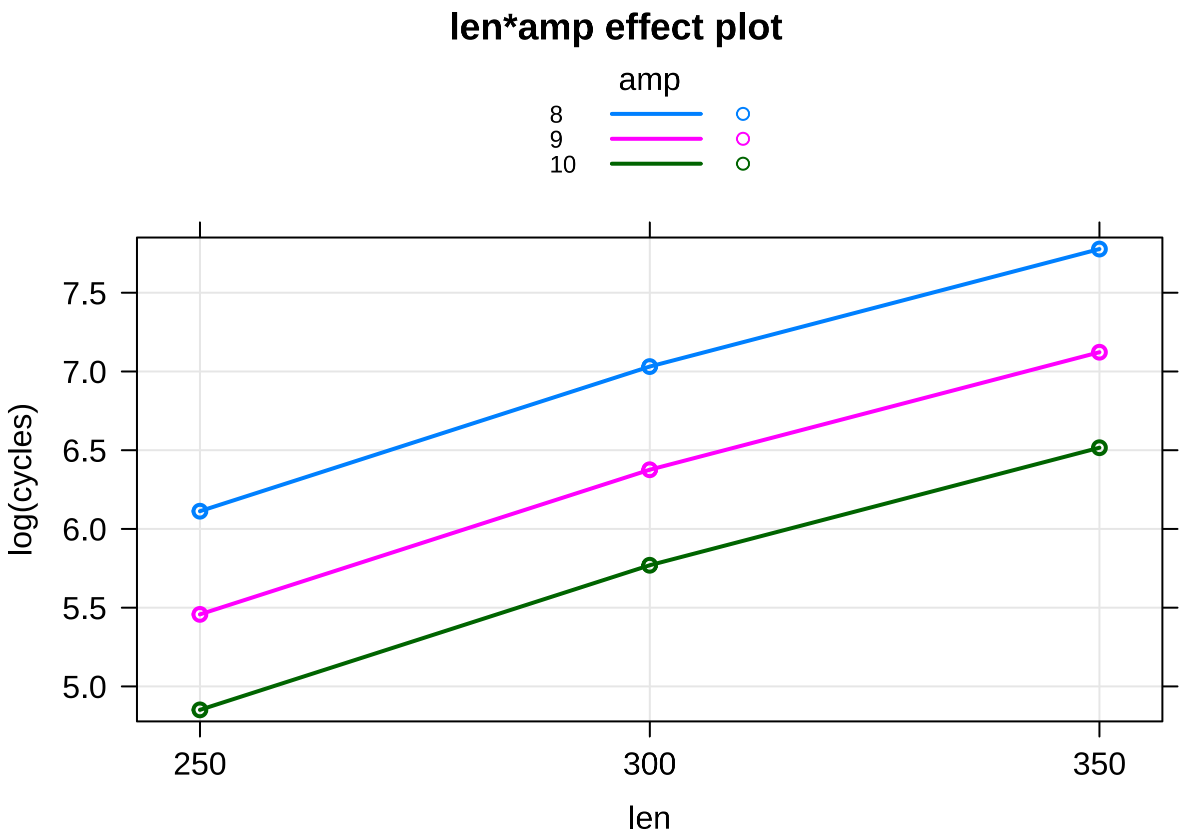
Compute an F-test comparing these two models:

```
anova(model1, model2)
```

```
## Analysis of Variance Table
##
## Model 1: log(cycles) ~ len + amp + load
## Model 2: log(cycles) ~ len + amp + load + len:amp + len:load + amp:load
##   Res.Df     RSS Df Sum of Sq     F Pr(>F)
## 1     20 0.71742
## 2      8 0.16591 12   0.55151 2.216 0.1325
```

The F-statistic=2.216 and the pvalue=0.1325, suggesting that we cannot reject the null hypothesis that model1 (i.e. the first-order model) is adequate.

```
plot(Effect(c("len", "amp"), model1), rug=FALSE, grid=TRUE, multiline=TRUE)
```

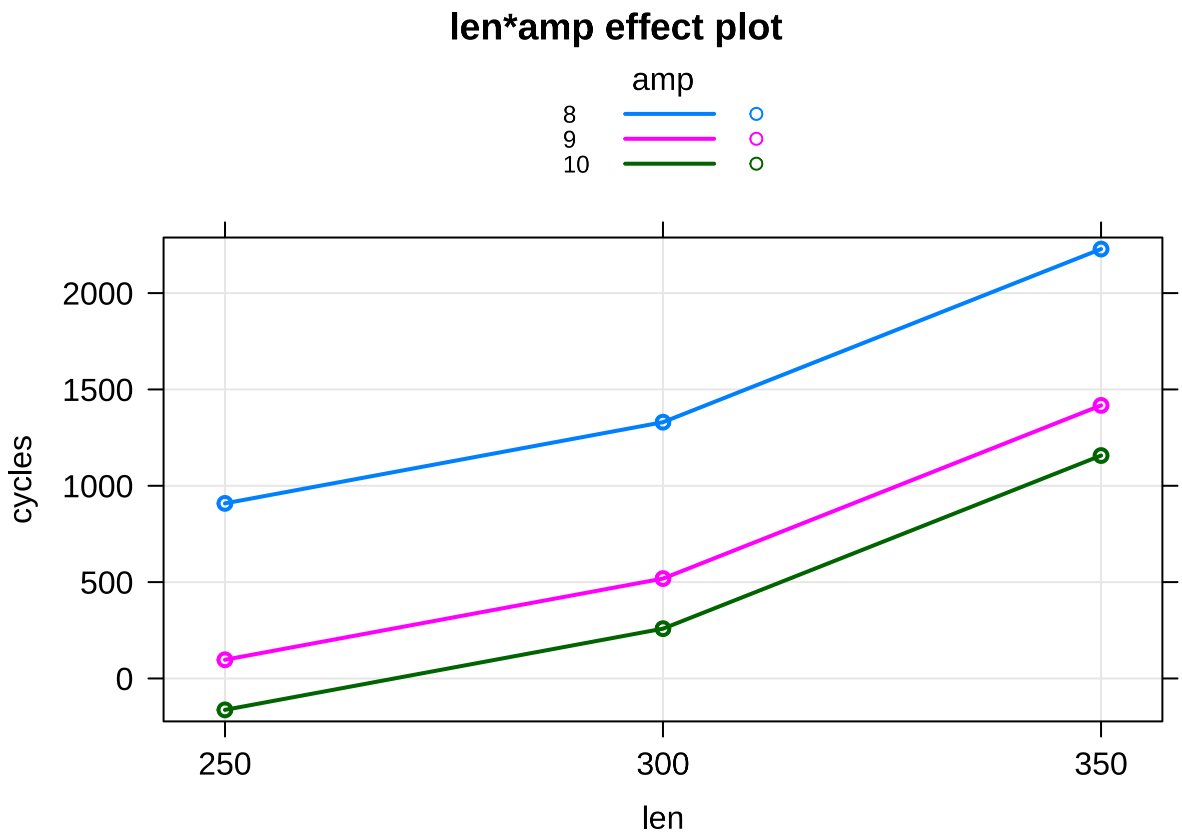


Redraw this effects plot with cycles rather than log(cycles):

```r
model3 = lm(cycles ~ len + amp + load, data=Wool)
summary(model3)$coefficients
```

```
##                Estimate Std. Error   t value     Pr(>|t|)
## (Intercept)   1203.3704   246.0169  4.891413 8.825572e-05
## len300         421.4444   227.7674  1.850328 7.909626e-02
## len350        1320.0000   227.7674  5.795385 1.137842e-05
## amp9          -811.5556   227.7674 -3.563089 1.948342e-03
## amp10        -1071.6667   227.7674 -4.705092 1.358079e-04
## load45        -262.5556   227.7674 -1.152735 2.626111e-01
## load50        -621.6667   227.7674 -2.729392 1.291836e-02
```

```r
plot(Effect(c("len", "amp"), model3), rug=FALSE, grid=TRUE, multiline=TRUE)
```



In the effects plot of Problem 8.6.2, the lines are not parallel because when we include the interaction term between *len* and *amp* into the model, when *amp* moves from one level to another, its effects on expected *cycles* is different for different velue of *len*.

However, in this model without interaction terms, the lines in the effects plot are completely parallel. This is because the effects of *amp* moving from one level to another are constant, reglardess of the value of *len*.