

# Chapter 8

# Polynomial Regression

## Contents

---

<b>8.1 Polynomial Models with One Predictor</b>	<b>243</b>
8.1.1 Example: Cloud point and percent I-8	248
<b>8.2 Polynomial Models with Two Predictors</b>	<b>255</b>
8.2.1 Example: Mooney viscosity	255
8.2.2 Example: Mooney viscosity on log scale	259

---

## 8.1 Polynomial Models with One Predictor

A  $p^{th}$  order polynomial model relating a dependent variable  $Y$  to a predictor  $X$  is given by

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \cdots + \beta_p X^p + \varepsilon.$$

This is a multiple regression model with predictors  $X, X^2, \dots, X^p$ . For  $p = 2, 3, 4,$  and  $5$  we have quadratic, cubic, quartic and quintic relationships, respectively.

A second order polynomial (quadratic) allows at most one local maximum or minimum (i.e., a point where trend changes direction from increasing to decreasing, or from decreasing to increasing). A third order polynomial (cubic) allows at most two local maxima or minima. In general, a  $p^{th}$  order polynomial

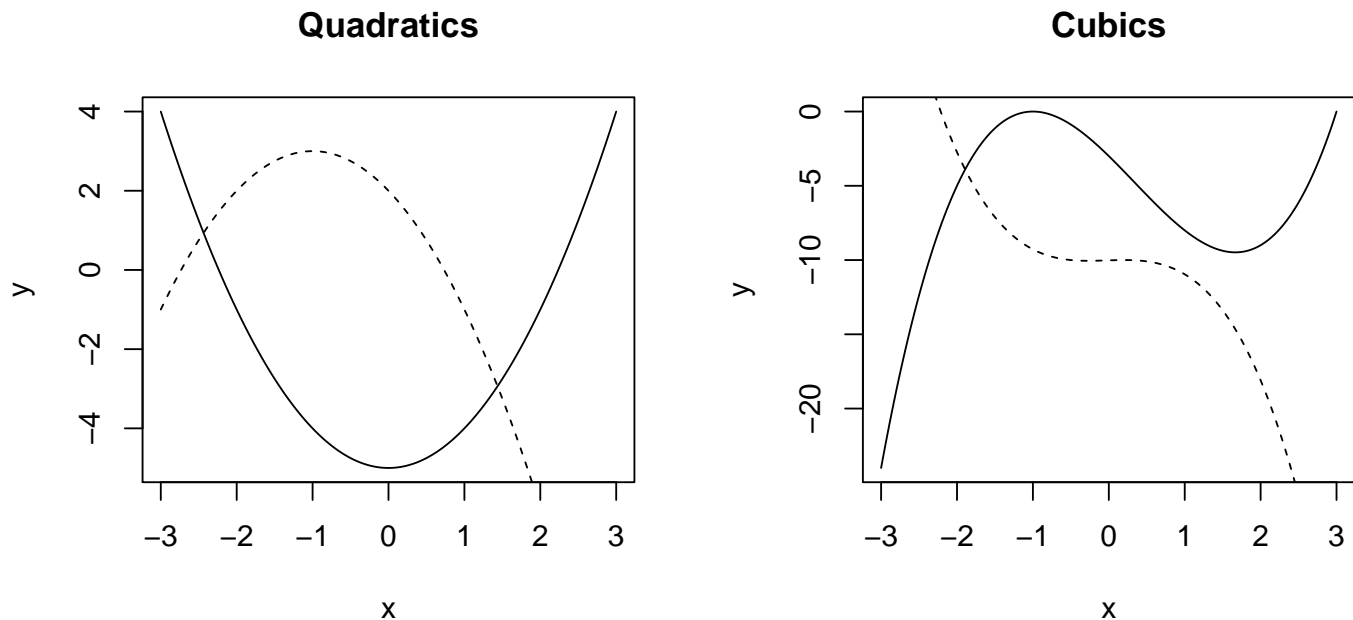
allows at most  $p - 1$  local maxima or minima. The two panels below illustrate different quadratic and cubic relationships.

```
library(tidyverse)

# load ada functions
source("ada_functions.R")

#### Creating polynomial plots
# R code for quadratic and cubic plots
x <- seq(-3, 3, 0.01)
y21 <- x^2 - 5
y22 <- -(x + 1)^2 + 3
y31 <- (x + 1)^2 * (x - 3)
y32 <- -(x - 0.2)^2 * (x + 0.5) - 10

plot(x, y21, type="l", main="Quadratics", ylab="y")
points(x, y22, type="l", lt=2)
plot(x, y31, type="l", main="Cubics", ylab="y")
points(x, y32, type="l", lt=2)
```



It is important to recognize that not all, or even none, of the “turning-points” in a polynomial may be observed if the range of  $X$  is suitably restricted.

Although polynomial models allow a rich class of non-linear relationships between  $Y$  and  $X$  (by virtue of Taylor’s Theorem in calculus), some caution is needed when fitting polynomials. In particular, the extreme  $X$ -values can be highly influential, numerical instabilities occur when fitting high order models, and predictions based on high order polynomial models can be woeful.

To illustrate the third concern, consider a data set  $(Y_i, X_i)$  for  $i = 1, 2, \dots, n$  where the  $X_i$ s are distinct. One can show mathematically that an  $(n - 1)^{st}$  degree polynomial will fit the observed data exactly. However, for a high order

polynomial to fit exactly, the fitted curve must oscillate wildly between data points. In the picture below, I show the  $10^{\text{th}}$  degree polynomial that fits exactly the 11 distinct data points. Although  $R^2 = 1$ , I would not use the fitted model to make predictions with new data. (If possible, models should always be validated using new data.) Intuitively, a quadratic or a lower order polynomial would likely be significantly better. In essence, the  $10^{\text{th}}$  degree polynomial is modelling the variability in the data, rather than the trend.

```
# R code for quadratic and cubic plots

X <- rnorm(11) # observed
Y <- rnorm(11) # observed
X1 <- X^1
X2 <- X^2
X3 <- X^3
X4 <- X^4
X5 <- X^5
X6 <- X^6
X7 <- X^7
X8 <- X^8
X9 <- X^9
X10 <- X^10

fit <- lm(Y~X + X2 + X3 + X4 + X5 + X6 + X7 + X8 + X9 + X10)
fit$coefficients

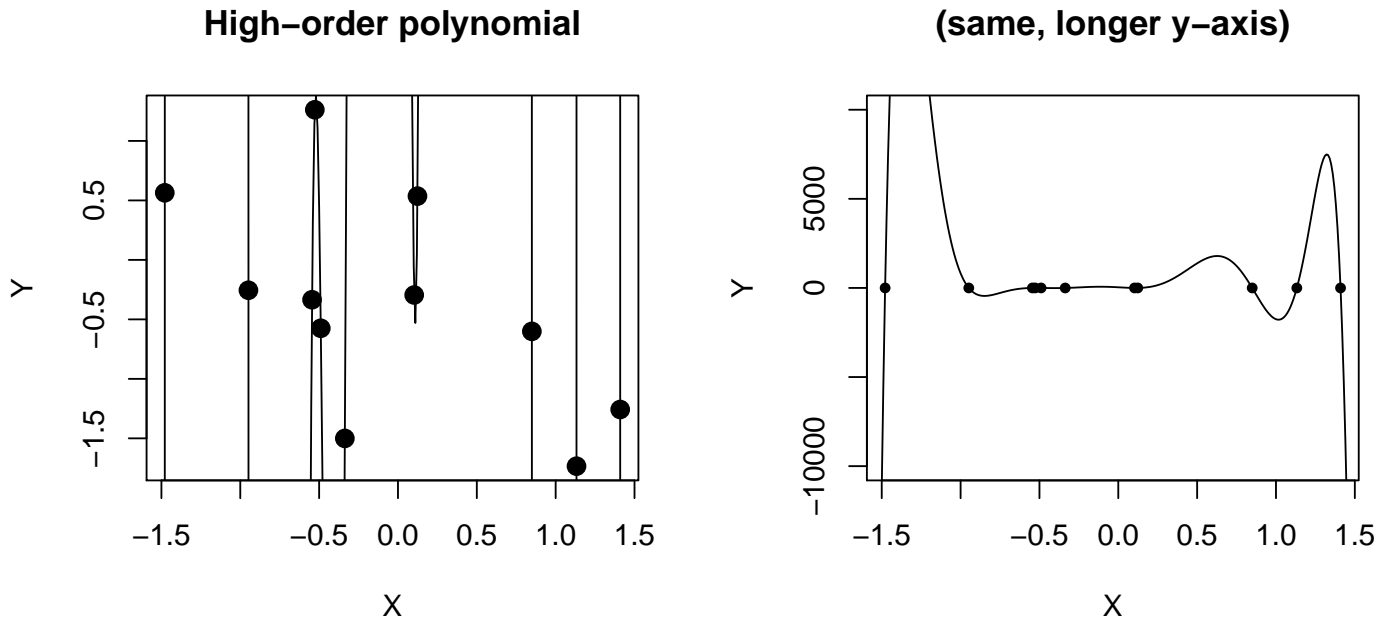
## (Intercept)      X          X2          X3          X4
##  36.70206    -461.55109   -620.55094  13030.85848  29149.14342
##           X5          X6          X7          X8          X9
## -26416.29553 -81282.20211  15955.10270  70539.53467  -3396.10960
##           X10
## -18290.46769

x <- seq(-2.5, 2.5, 0.01)
x1 <- x^1
x2 <- x^2
x3 <- x^3
x4 <- x^4
x5 <- x^5
x6 <- x^6
x7 <- x^7
x8 <- x^8
x9 <- x^9
x10 <- x^10

xx <- matrix(c(rep(1, length(x)), x1, x2, x3, x4, x5, x6, x7, x8, x9, x10), ncol=11)

y <- xx %*% fit$coefficients

plot( X, Y, main="High-order polynomial", pch=20, cex=2)
points(x, y, type="l", lt=1)
plot( X, Y, main="(same, longer y-axis)", pch=20, cex=1, ylim=c(-10000,10000))
points(x, y, type="l", lt=1)
```



Another concern is that the importance of lower order terms (i.e.,  $X$ ,  $X^2$ ,  $\dots$ ,  $X^{p-1}$ ) depends on the scale in which we measure  $X$ . For example, suppose for some chemical reaction,

$$\text{Time to reaction} = \beta_0 + \beta_1 \text{Temp} + \beta_2 \text{Temp}^2 + \varepsilon.$$

The significance level for the estimate of the Temp coefficient depends on whether we measure temperature in degrees Celsius or Fahrenheit.

To avoid these problems, I recommend the following:

1. Center the  $X$  data at  $\bar{X}$  and fit the model

$$Y = \beta_0 + \beta_1(X - \bar{X}) + \beta_2(X - \bar{X})^2 + \dots + \beta_p(X - \bar{X})^p + \varepsilon.$$

This is usually important only for cubic and higher order models.

2. Restrict attention to low order models, say quartic (fourth-degree) or less. If a fourth-order polynomial does not fit, a transformation may provide a more succinct summary.
3. Pay careful attention to diagnostics.
4. Add or delete variables using the natural hierarchy among powers of  $X$  and include all lower order terms if a higher order term is needed. For example, in a forward-selection type algorithm, add terms  $X$ ,  $X^2$ ,  $\dots$ ,

sequentially until no additional term is significant, but do not delete powers that were entered earlier. Similarly, with a backward-elimination type algorithm, start with the model of maximum acceptable order (for example a fourth or third order polynomial) and consider deleting terms in the order  $X^p, X^{p-1}, \dots$ , until no further terms can be omitted. The **select=backward** option in the **reg** procedure **does not allow** you to invoke the hierarchy principle with backward elimination. The backward option sequentially eliminates the least significant effect in the model, regardless of what other effects are included.

### 8.1.1 Example: Cloud point and percent I-8

The cloud point of a liquid is a measure of the degree of crystallization in a stock, and is measured by the refractive index <sup>1</sup>. It has been suggested that the percent of I-8 (variable “i8”) in the base stock is an excellent predictor of the cloud point using a second order (quadratic) model:

$$\text{Cloud point} = \beta_0 + \beta_1 \text{I8} + \beta_2 \text{I8}^2 + \varepsilon.$$

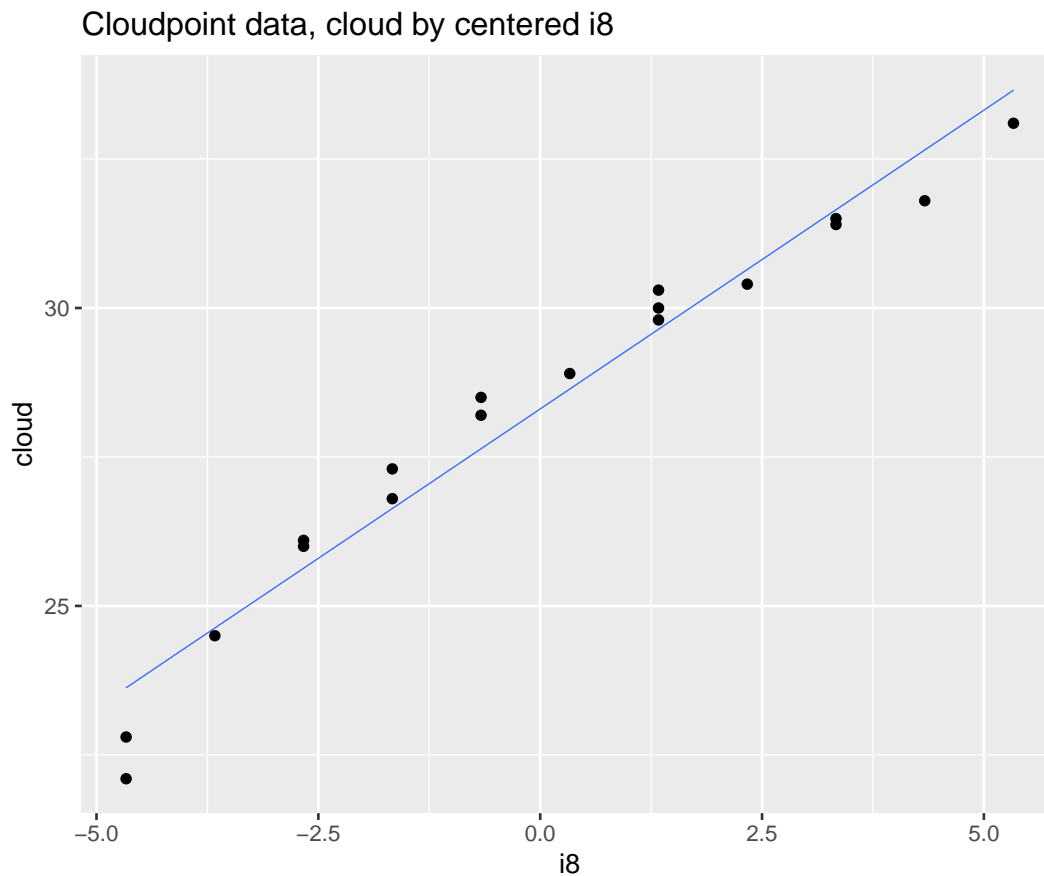
Data were collected to examine this model.

```
#### Example: Cloud point
dat_cloudpoint <-
  read_table2("http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch08_cloudpoint.dat") %>%
  mutate(
    # center i8 by subtracting the mean
    i8 = i8 - mean(i8)
  )
## Parsed with column specification:
## cols(
##   i8 = col_double(),
##   cloud = col_double()
## )
```

The plot of the data suggest a departure from a linear relationship.

```
library(ggplot2)
p <- ggplot(dat_cloudpoint, aes(x = i8, y = cloud))
p <- p + geom_smooth(method = lm, se = FALSE, size = 1/4)
p <- p + geom_point()
p <- p + labs(title="Cloudpoint data, cloud by centered i8")
print(p)
```

<sup>1</sup>Draper and Smith 1966, p. 162

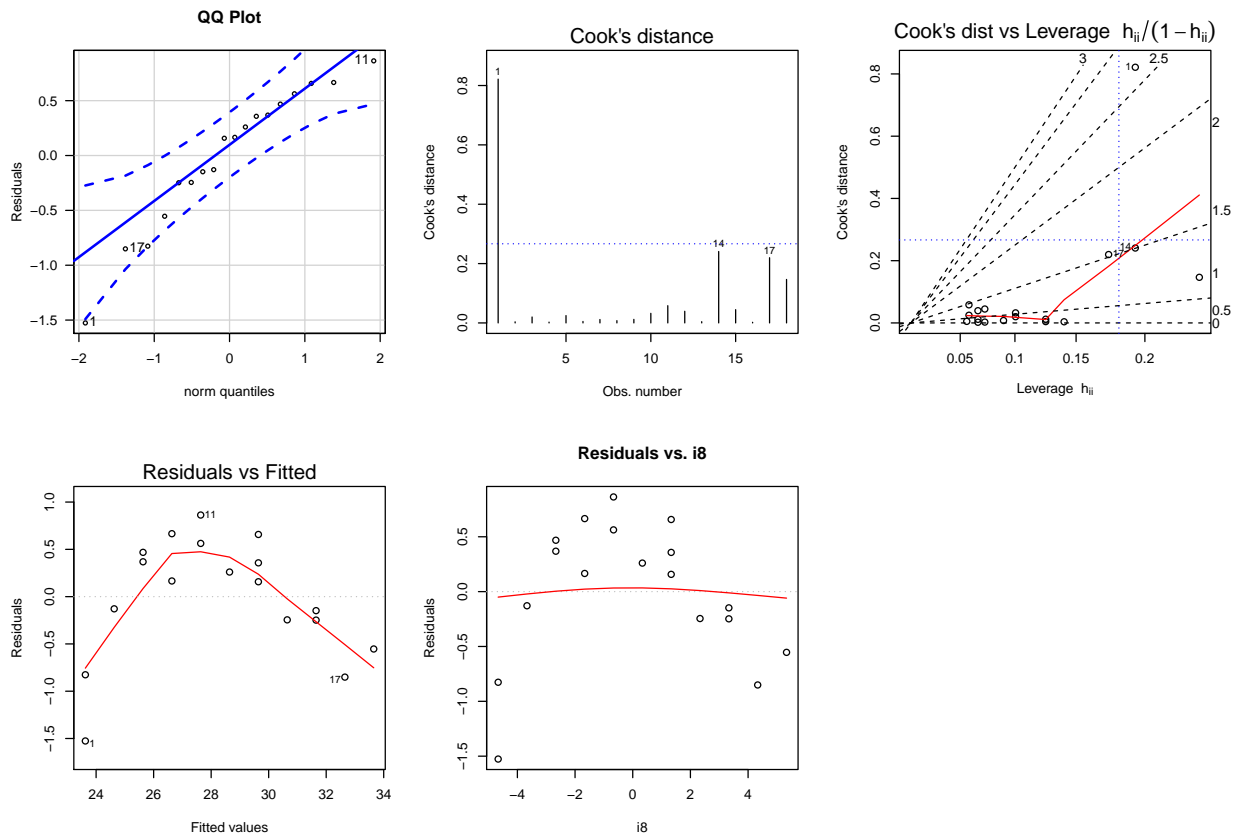


Fit the simple linear regression model and plot the residuals.

```
lm_c_i <-
  lm(
    cloud ~ i8
  , data = dat_cloudpoint
  )
#library(car)
#Anova(aov(lm_c_i), type=3)
#summary(lm_c_i)
```

The data plot is clearly nonlinear, suggesting that a simple linear regression model is inadequate. This is confirmed by a plot of the studentized residuals against the fitted values from a simple linear regression of Cloud point on i8. Also by the residuals against the i8 values. We do not see any local maxima or minima, so a second order model is likely to be adequate. To be sure, we will first fit a cubic model, and see whether the third order term is important.

```
# plot diagnostics
lm_diag_plots(lm_c_i, sw_plot_set = "simple")
```



The output below shows that the cubic term improves the fit of the quadratic model (i.e., the cubic term is important when added last to the model). The plot of the studentized residuals against the fitted values does not show any extreme abnormalities. Furthermore, no individual point is poorly fitted by the model. Case 1 has the largest studentized residual:  $r_1 = -1.997$ .

```
# I() is used to create an interpreted object treated "as is"
# so we can include quadratic and cubic terms in the formula
# without creating separate columns in the dataset of these terms
lm_c_i3 <-
  lm(
    cloud ~ i8 + I(i8^2) + I(i8^3)
    , data = dat_cloudpoint
  )
#library(car)
#Anova(aov(lm_c_i3), type=3)
summary(lm_c_i3)

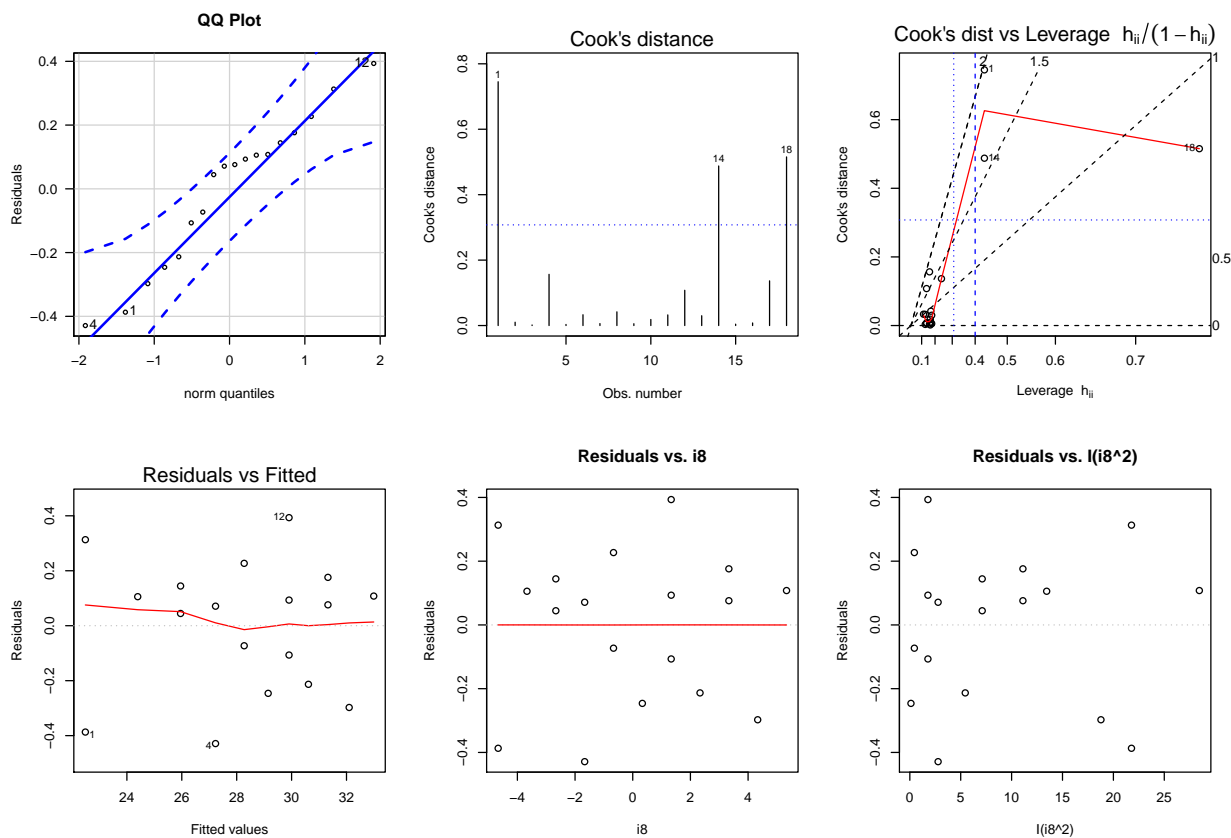
##
## Call:
## lm(formula = cloud ~ i8 + I(i8^2) + I(i8^3), data = dat_cloudpoint)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

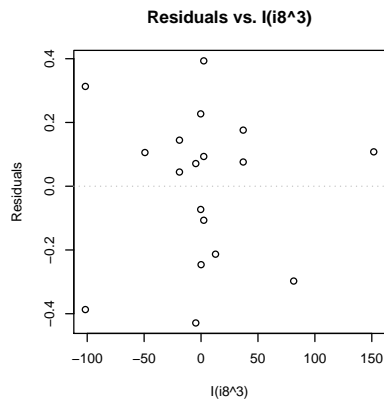


```
## -0.42890 -0.18658 0.07355 0.13536 0.39328
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 28.870451  0.088364 326.723 < 2e-16 ***
## i8           0.847889  0.048536  17.469 6.67e-11 ***
## I(i8^2)      -0.065998  0.007323  -9.012 3.33e-07 ***
## I(i8^3)       0.009735  0.002588   3.762 0.0021 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2599 on 14 degrees of freedom
## Multiple R-squared:  0.9943, Adjusted R-squared:  0.9931
## F-statistic: 812.9 on 3 and 14 DF,  p-value: 6.189e-16
```

Below are plots of the data and the studentized residuals.

```
# plot diagnostics
lm_diag_plots(lm_c_i3, sw_plot_set = "simple")
```





The first and last observations have the lowest and highest values of  $I8$ , given by 0 and 10, respectively. These cases are also the most influential points in the data set (largest Cook's  $D$ ). If we delete these cases and redo the analysis we find that the cubic term is no longer important ( $p$ -value=0.55) when added after the quadratic term. One may reasonably conclude that the significance of the cubic term in the original analysis is solely due to the two extreme  $I8$  values, and that the quadratic model appears to fit well over the smaller range of  $1 \leq I8 \leq 9$ .

```
# remove points for minimum and maximum i8 values
dat_cloudpoint2 <-
  dat_cloudpoint %>%
  filter(
    !(i8 == min(i8)) | (i8 == max(i8)) )
  )

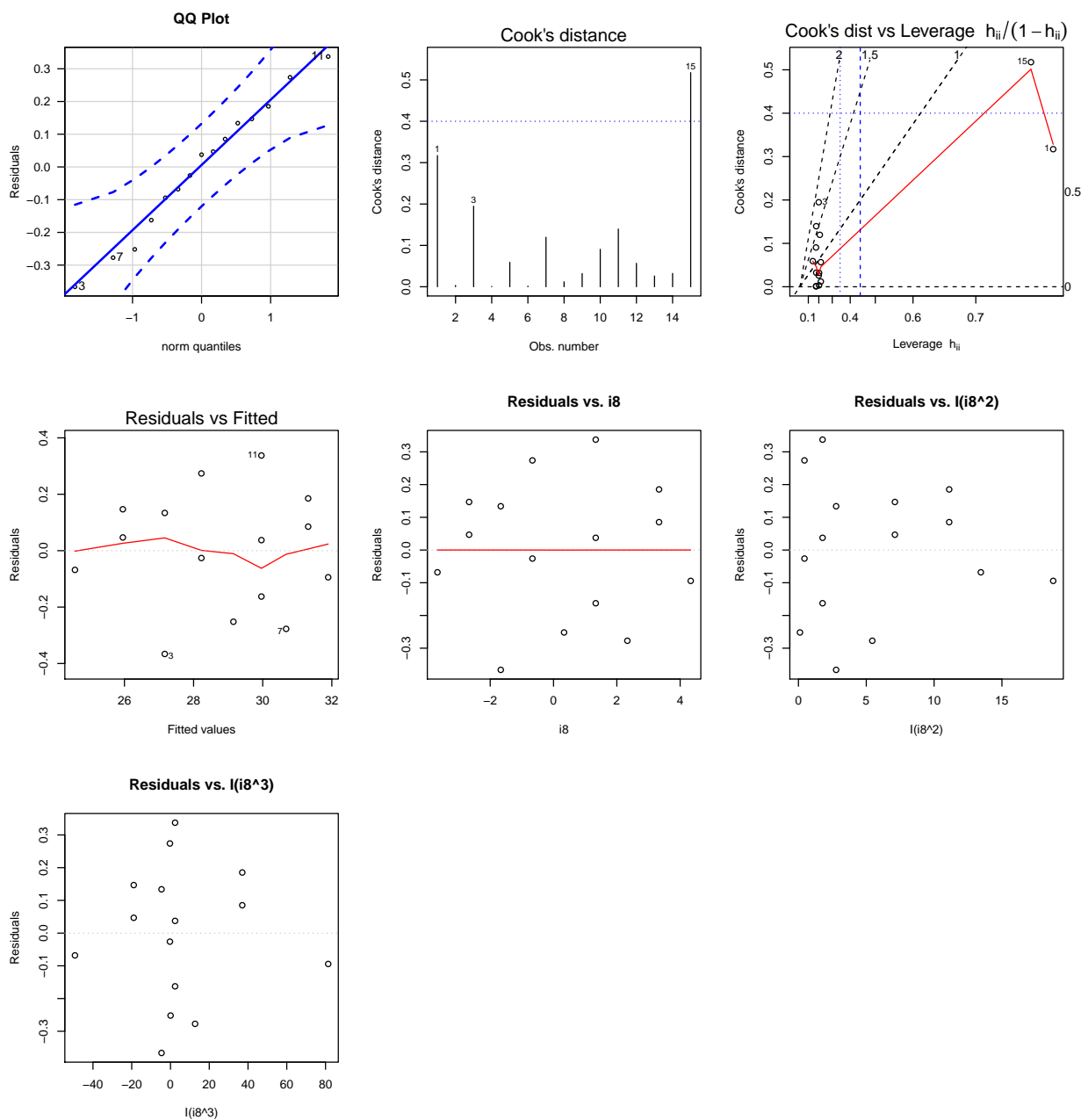
lm_c_i2 <-
  lm(
    cloud ~ i8 + I(i8^2) + I(i8^3)
    , data = dat_cloudpoint2
  )

#library(car)
#Anova(aov(lm_c_i2), type=3)
summary(lm_c_i2)

##
## Call:
## lm(formula = cloud ~ i8 + I(i8^2) + I(i8^3), data = dat_cloudpoint2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36620 -0.12845  0.03737  0.14031  0.33737
##
## Coefficients:
```

```
##           Estimate Std. Error t value Pr(>|t|)
## (Intercept) 28.857039  0.089465 322.551 < 2e-16 ***
## i8          0.904515  0.058338  15.505 8.04e-09 ***
## I(i8^2)     -0.060714  0.012692  -4.784 0.000568 ***
## I(i8^3)     0.003168  0.005166   0.613 0.552200
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2313 on 11 degrees of freedom
## Multiple R-squared:  0.9917, Adjusted R-squared:  0.9894
## F-statistic: 436.3 on 3 and 11 DF,  p-value: 1.032e-11
```

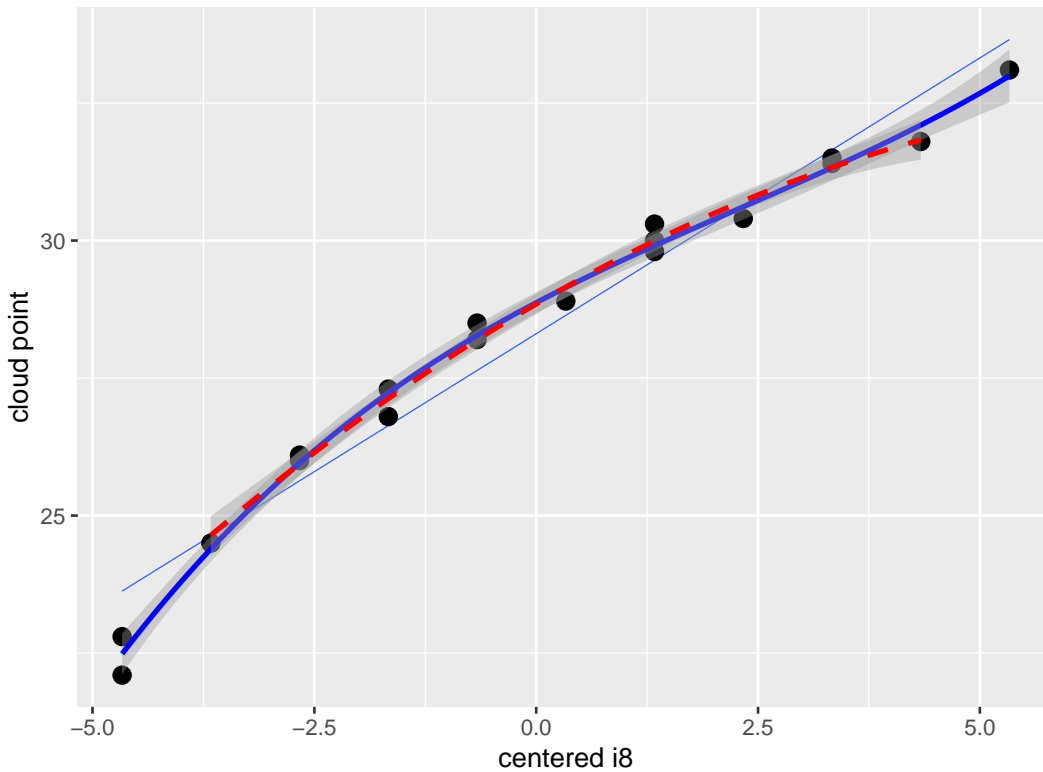
```
# plot diagnostics
lm_diag_plots(lm_c_i2, sw_plot_set = "simple")
```



These are the two fitted models to the data. The blue solid line is the cubic model with all of the observations and the red dashed line is the quadratic model excluding the minimum and maximum i8 observations.

```
library(ggplot2)
p <- ggplot(dat_cloudpoint, aes(x = i8, y = cloud))
p <- p + geom_smooth(method = lm, se = FALSE, size = 1/4)
p <- p + geom_point(size = 3)
p <- p + stat_smooth(method = "lm"
                    , formula = y ~ x + I(x^2) + I(x^3)
                    , colour = "blue", linetype = "solid")
p <- p + stat_smooth(data = dat_cloudpoint2
                    , method = "lm"
                    , formula = y ~ x + I(x^2)
                    , colour = "red", linetype = "dashed")
p <- p + labs( title = "Cloudpoint data, cloud by centered i8"
              , x = "centered i8"
              , y = "cloud point"
              , caption =
                paste0(
                  "Blue solid = Cubic model, all observations\n"
                  , "Red dashed = Quadratic model, excluding extreme i8 observations"
                )
              )
print(p)
```

Cloudpoint data, cloud by centered i8



Blue solid = Cubic model, all observations  
Red dashed = Quadratic model, excluding extreme i8 observations

## 8.2 Polynomial Models with Two Predictors

Polynomial models are sometimes fit to data collected from experiments with two or more predictors. For simplicity, consider the general quadratic model, with two predictors:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 + \varepsilon.$$

The model, which can be justified as a second order approximation to a smooth trend, includes quadratic terms in  $X_1$  and  $X_2$  and the product or interaction of  $X_1$  and  $X_2$ .

### 8.2.1 Example: Mooney viscosity

The data below give the Mooney viscosity at 100 degrees Celsius ( $Y$ ) as a function of the filler level ( $X_1$ ) and the naphthenic oil ( $X_2$ ) level for an experiment involving filled and plasticized elastomer compounds.

```
#### Example: Mooney viscosity
dat_mooney <-
  read_table2("http://statacumen.com/teach/ADA2/notes/ADA2_notes_Ch08_mooney.dat")
## Parsed with column specification:
## cols(
##   oil = col_double(),
##   filler = col_double(),
##   mooney = col_double()
## )
```

At each of the 4 oil levels, the relationship between the Mooney viscosity and filler level (with 6 levels) appears to be quadratic. Similarly, the relationship between the Mooney viscosity and oil level appears quadratic for each filler level (with 4 levels). This supports fitting the general quadratic model as a first step in the analysis.

The output below shows that each term is needed in the model. Although there are potentially influential points (cases 6 and 20), deleting either or both cases does not change the significance of the effects in the model (not shown).

```

# I create each term separately
lm_m_o2_f2 <-
  lm(
    mooney ~ oil + filler + I(oil^2) + I(filler^2) + I(oil * filler)
    , data = dat_mooney
  )
summary(lm_m_o2_f2)

##
## Call:
## lm(formula = mooney ~ oil + filler + I(oil^2) + I(filler^2) +
##     I(oil * filler), data = dat_mooney)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -6.3497 -2.2231 -0.1615  2.5424  5.2749
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   27.144582   2.616779  10.373 9.02e-09 ***
## oil           -1.271442   0.213533  -5.954 1.57e-05 ***
## filler         0.436984   0.152658   2.862  0.0108 *
## I(oil^2)       0.033611   0.004663   7.208 1.46e-06 ***
## I(filler^2)    0.027323   0.002410  11.339 2.38e-09 ***
## I(oil * filler) -0.038659   0.003187 -12.131 8.52e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.937 on 17 degrees of freedom
## (1 observation deleted due to missingness)
## Multiple R-squared:  0.9917, Adjusted R-squared:  0.9892
## F-statistic: 405.2 on 5 and 17 DF,  p-value: < 2.2e-16

## poly() will evaluate variables and give joint polynomial values
##      which is helpful when you have many predictors
# head(dat_mooney, 10)
# head(poly(dat_mooney$oil, dat_mooney$filler, degree = 2, raw = TRUE), 10)
## This model is equivalent to the one above
# lm_m_o2_f2 <- lm(mooney ~ poly(oil, filler, degree = 2, raw = TRUE), data = dat_mooney)
# summary(lm_m_o2_f2)

# Put predicted values into the dataset for plotting
# 1 missing value requires a method to put the predicted values into the correct indices
dat_mooney$pred <- NA
dat_mooney$pred[as.numeric(names(predict(lm_m_o2_f2)))] <- predict(lm_m_o2_f2)

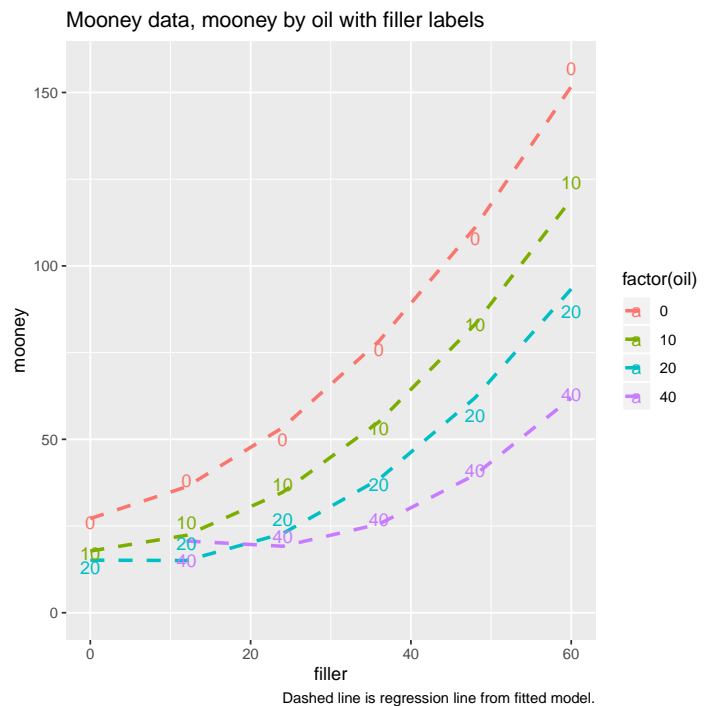
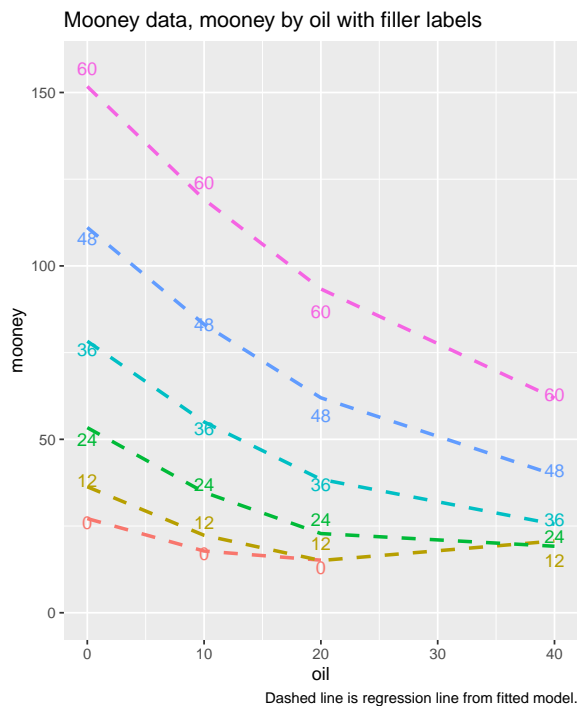
library(ggplot2)
p <- ggplot(dat_mooney, aes(x = oil, y = mooney, colour = factor(filler), label = filler))
p <- p + geom_line(aes(y = pred), linetype = 2, size = 1)
p <- p + geom_text()
p <- p + scale_y_continuous(limits = c(0, max(dat_mooney$mooney, na.rm=TRUE)))
p <- p + labs(
  title = "Mooney data, mooney by oil with filler labels"
)

```

```

, caption = paste0("Dashed line is regression line from fitted model.")
)
print(p)
## Warning: Removed 1 rows containing missing values (geom_path).
## Warning: Removed 1 rows containing missing values (geom_text).
library(ggplot2)
p <- ggplot(dat_mooney, aes(x = filler, y = mooney, colour = factor(oil), label = oil))
p <- p + geom_line(aes(y = pred), linetype = 2, size = 1)
p <- p + geom_text()
p <- p + scale_y_continuous(limits = c(0, max(dat_mooney$mooney, na.rm=TRUE)))
p <- p + labs(
  title = "Mooney data, mooney by oil with filler labels"
, caption = paste0("Dashed line is regression line from fitted model.")
)
print(p)
## Warning: Removed 1 rows containing missing values (geom_path).
## Warning: Removed 1 rows containing missing values (geom_text).

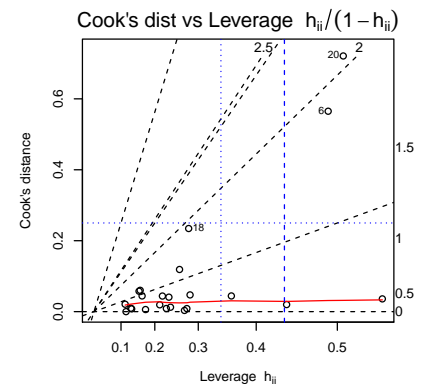
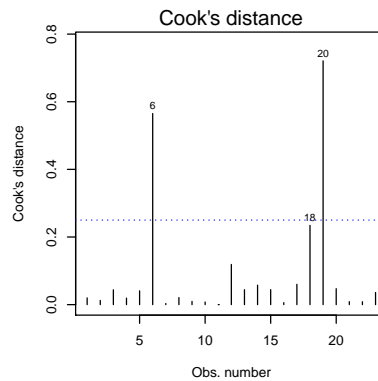
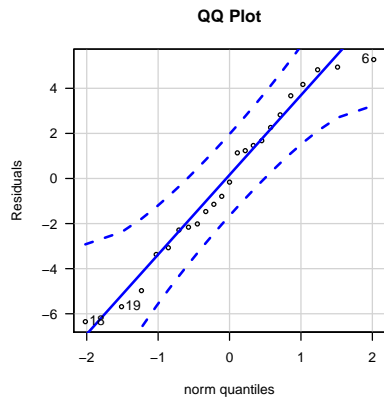
```

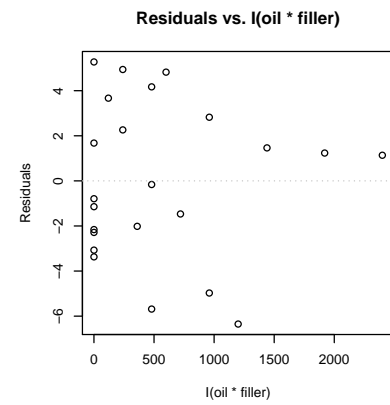
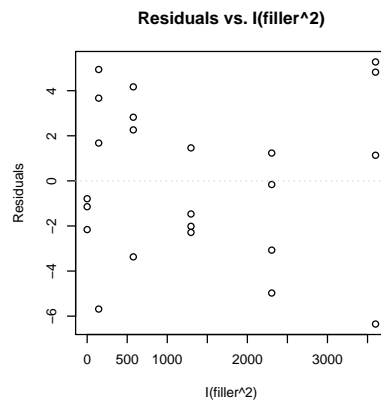
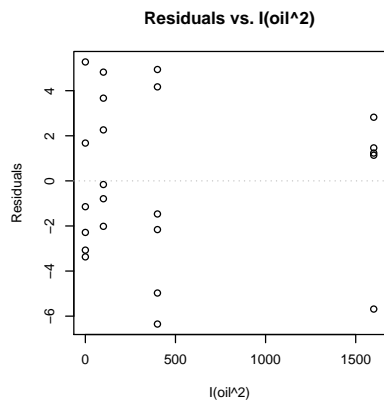
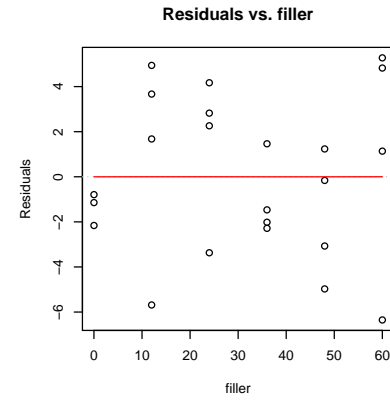
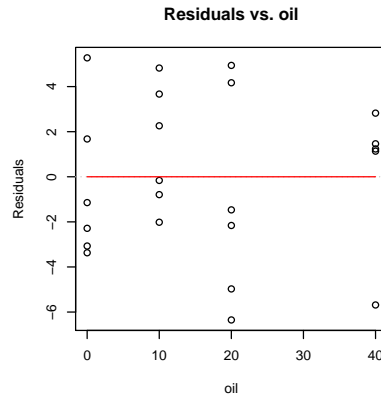
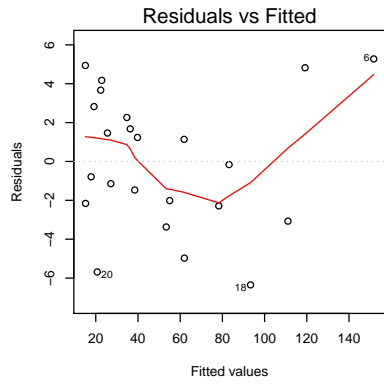


```

# plot diagnostics
lm_diag_plots(lm_m_o2_f2, sw_plot_set = "simple")

```







## 8.2.2 Example: Mooney viscosity on log scale

As noted earlier, transformations can often be used instead of polynomials. For example, the original data plots suggest transforming the Mooney viscosity to a log scale. If we make this transformation and replot the data, we see that the log Mooney viscosity is roughly linearly related to the filler level at each oil level, but is a quadratic function of oil at each filler level. The plots of the transformed data suggest that a simpler model will be appropriate.

```
# log transform the response
dat_mooney <-
  dat_mooney %>%
  mutate(
    logmooney = log(mooney)
  )
```

To see that a simpler model is appropriate, we fit the full quadratic model. The interaction term can be omitted here, without much loss of predictive ability (R-squared is similar). The p-value for the interaction term in the quadratic model is 0.34.

```
# I create each term separately
lm_lm_o2_f2 <-
  lm(
    logmooney ~ oil + filler + I(oil^2) + I(filler^2) + I(oil * filler)
    , data = dat_mooney
  )
summary(lm_lm_o2_f2)

##
## Call:
## lm(formula = logmooney ~ oil + filler + I(oil^2) + I(filler^2) +
##     I(oil * filler), data = dat_mooney)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.077261 -0.035795  0.009193  0.030641  0.075640
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.236e+00  3.557e-02  90.970 < 2e-16 ***
## oil           -3.921e-02  2.903e-03 -13.507 1.61e-10 ***
## filler        2.860e-02  2.075e-03  13.781 1.18e-10 ***
## I(oil^2)       4.227e-04  6.339e-05   6.668 3.96e-06 ***
## I(filler^2)    4.657e-05  3.276e-05   1.421  0.173
## I(oil * filler) -4.231e-05  4.332e-05  -0.977  0.342
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

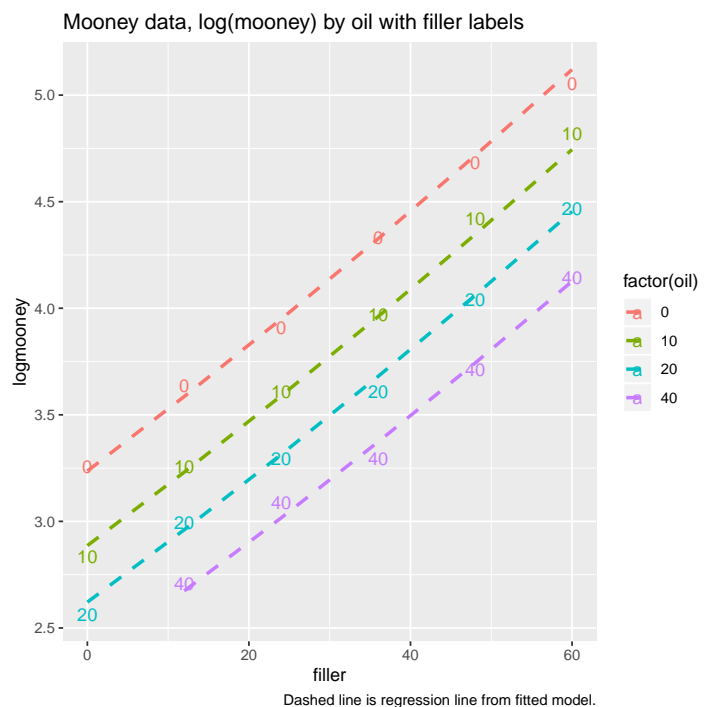
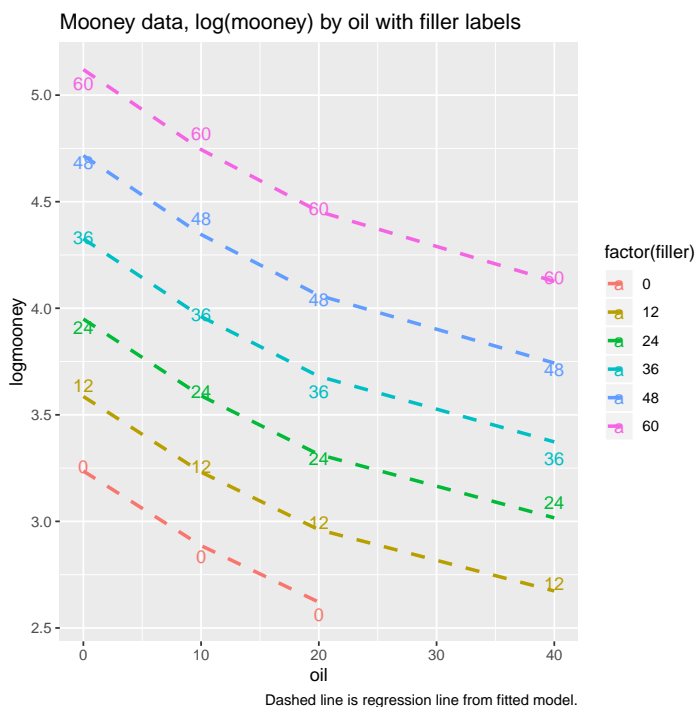
```
##
## Residual standard error: 0.05352 on 17 degrees of freedom
## (1 observation deleted due to missingness)
## Multiple R-squared: 0.9954, Adjusted R-squared: 0.9941
## F-statistic: 737 on 5 and 17 DF, p-value: < 2.2e-16

# Put predicted values into the dataset for plotting
# 1 missing value requires a method to put the predicted values into the correct indices
dat_mooney$pred <- NA
dat_mooney$pred[as.numeric(names(predict(lm_lm_o2_f2)))] <- predict(lm_lm_o2_f2)

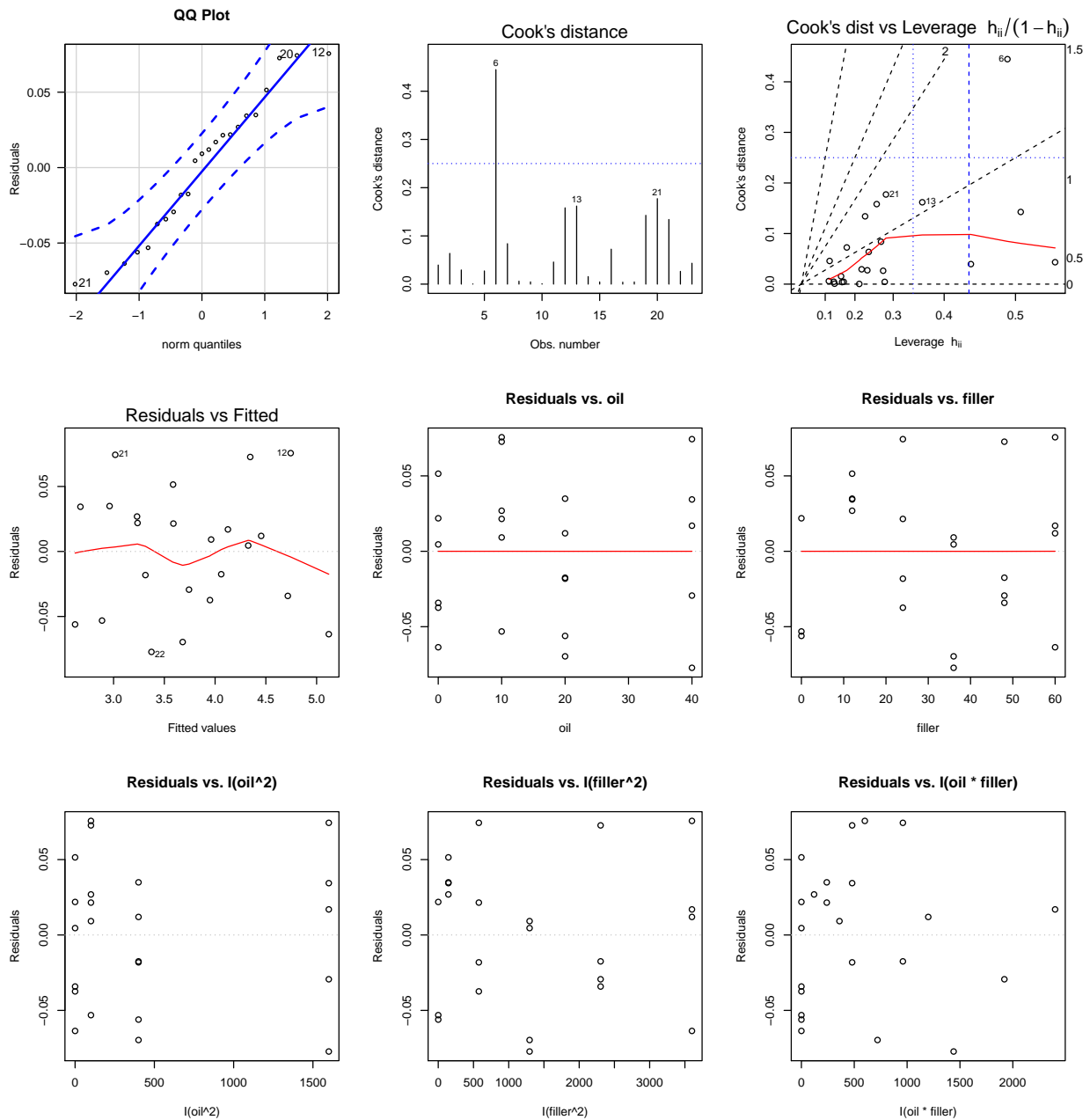
library(ggplot2)
p <- ggplot(dat_mooney, aes(x = oil, y = logmooney, colour = factor(filler), label = filler))
p <- p + geom_line(aes(y = pred), linetype = 2, size = 1)
p <- p + geom_text()
#p <- p + scale_y_continuous(limits = c(0, max(dat_mooney$logmooney, na.rm=TRUE)))
p <- p + labs(
  title = "Mooney data, log(mooney) by oil with filler labels"
  , caption = paste0("Dashed line is regression line from fitted model.")
)
print(p)

## Warning: Removed 1 rows containing missing values (geom_path).
## Warning: Removed 1 rows containing missing values (geom_text).
library(ggplot2)
p <- ggplot(dat_mooney, aes(x = filler, y = logmooney, colour = factor(oil), label = oil))
p <- p + geom_line(aes(y = pred), linetype = 2, size = 1)
p <- p + geom_text()
#p <- p + scale_y_continuous(limits = c(0, max(dat_mooney$logmooney, na.rm=TRUE)))
p <- p + labs(
  title = "Mooney data, log(mooney) by oil with filler labels"
  , caption = paste0("Dashed line is regression line from fitted model.")
)
print(p)

## Warning: Removed 1 rows containing missing values (geom_path).
## Warning: Removed 1 rows containing missing values (geom_text).
```



```
# plot diagnostics
lm_diag_plots(lm_lm_o2_f2, sw_plot_set = "simple")
```



After omitting the interaction term, the quadratic effect in filler is not needed in the model (output not given). Once these two effects are removed, each of the remaining effects is significant.

```
# I create each term separately
lm_lm_o2_f <-
  lm(
    logmooney ~ oil + filler + I(oil^2)
    , data = dat_mooney
  )
summary(lm_lm_o2_f)
##
## Call:
```

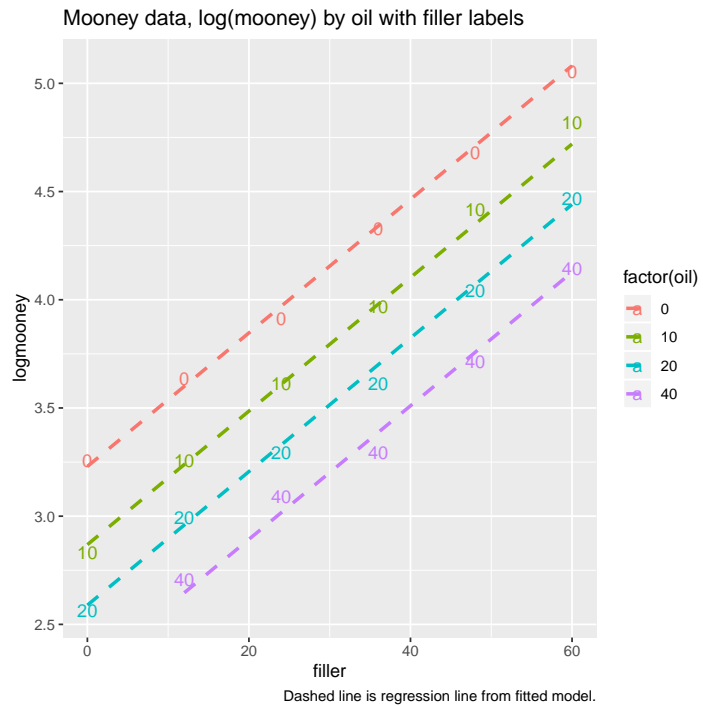
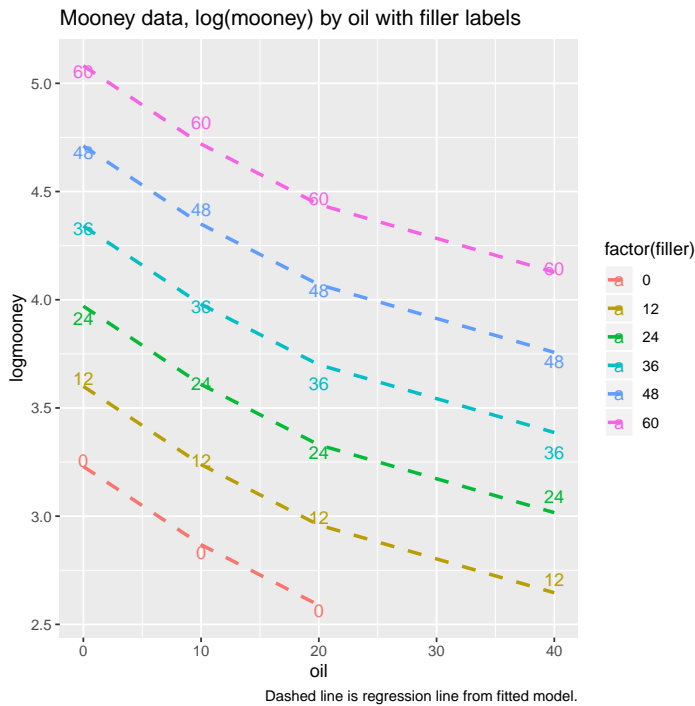
```
## lm(formula = logmooney ~ oil + filler + I(oil^2), data = dat_mooney)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.090796 -0.031113 -0.008831  0.032533  0.100587
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.230e+00  2.734e-02 118.139 < 2e-16 ***
## oil          -4.024e-02  2.702e-03 -14.890 6.26e-12 ***
## filler       3.086e-02  5.716e-04  53.986 < 2e-16 ***
## I(oil^2)     4.097e-04  6.356e-05   6.446 3.53e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05423 on 19 degrees of freedom
## (1 observation deleted due to missingness)
## Multiple R-squared:  0.9947, Adjusted R-squared:  0.9939
## F-statistic: 1195 on 3 and 19 DF,  p-value: < 2.2e-16

# Put predicted values into the dataset for plotting
# 1 missing value requires a method to put the predicted values into the correct indices
dat_mooney$pred <- NA
dat_mooney$pred[as.numeric(names(predict(lm_lm_o2_f)))] <- predict(lm_lm_o2_f)

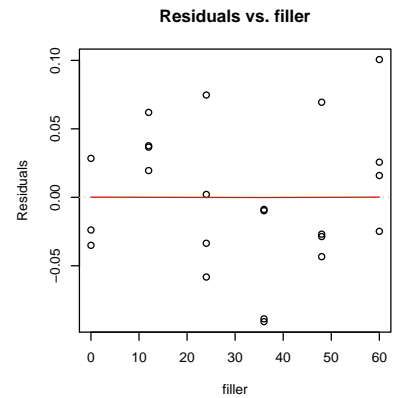
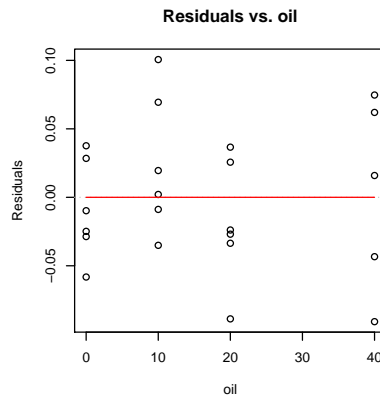
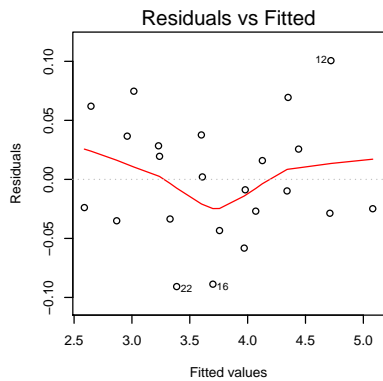
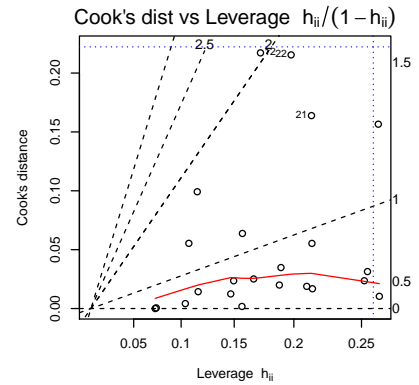
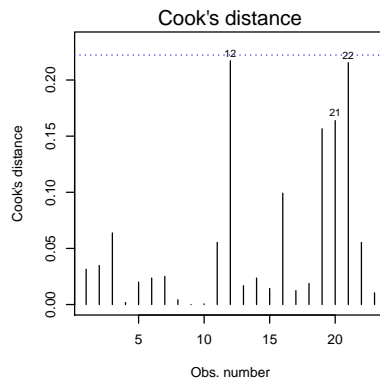
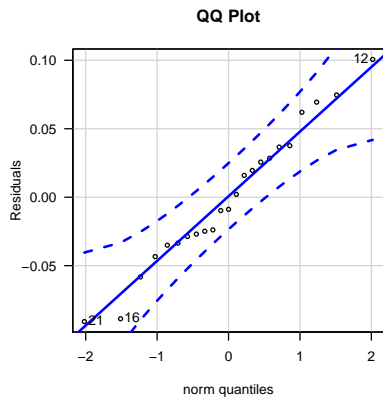
library(ggplot2)
p <- ggplot(dat_mooney, aes(x = oil, y = logmooney, colour = factor(filler), label = filler))
p <- p + geom_line(aes(y = pred), linetype = 2, size = 1)
p <- p + geom_text()
#p <- p + scale_y_continuous(limits = c(0, max(dat_mooney$logmooney, na.rm=TRUE)))
p <- p + labs(
  title = "Mooney data, log(mooney) by oil with filler labels"
  , caption = paste0("Dashed line is regression line from fitted model.")
)
print(p)

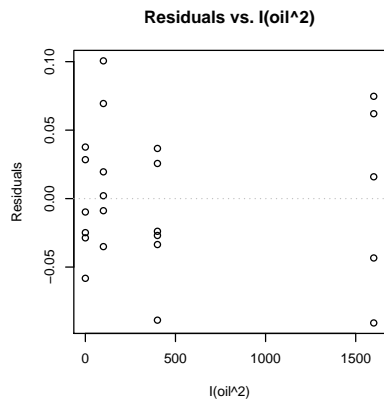
## Warning: Removed 1 rows containing missing values (geom.path).
## Warning: Removed 1 rows containing missing values (geom.text).
library(ggplot2)
p <- ggplot(dat_mooney, aes(x = filler, y = logmooney, colour = factor(oil), label = oil))
p <- p + geom_line(aes(y = pred), linetype = 2, size = 1)
p <- p + geom_text()
#p <- p + scale_y_continuous(limits = c(0, max(dat_mooney$logmooney, na.rm=TRUE)))
p <- p + labs(
  title = "Mooney data, log(mooney) by oil with filler labels"
  , caption = paste0("Dashed line is regression line from fitted model.")
)
print(p)

## Warning: Removed 1 rows containing missing values (geom.path).
## Warning: Removed 1 rows containing missing values (geom.text).
```



```
# plot diagnostics
lm_diag_plots(lm_lm_o2_f, sw_plot_set = "simple")
```





The model does not appear to have inadequacies. Assuming no difficulties, an important effect of the transformation is that the resulting model is simpler than the model selected on the original scale. This is satisfying to me, but you may not agree. Note that the selected model, with linear effects due to the oil and filler levels and a quadratic effect due to the oil level, agrees with our visual assessment of the data. Assuming no inadequacies, the predicted log Moody viscosity is given by

$$\widehat{\log(\text{Moody viscosity})} = 3.2297 - 0.0402 \text{ Oil} + 0.0004 \text{ Oil}^2 + 0.0309 \text{ Filler}.$$

Quadratic models with two or more predictors are often used in industrial experiments to estimate the optimal combination of predictor values to maximize or minimize the response, over the range of predictor variable values where the model is reasonable. (This strategy is called “response surface methodology”.) For example, we might wish to know what combination of oil level between 0 and 40 and filler level between 0 and 60 provides the lowest predicted Mooney viscosity (on the original or log scale). We can visually approximate the minimizer using the data plots, but one can do a more careful job of analysis using standard tools from calculus.