StatPSET1

Peter Dunson

2023-09-04

```
#.5a: The observational unit in this study is how much ice cream each subject scooped themselves.

#.5b: This is a controlled experiment because they are not just observing how much ice
#cream is taken, but rather they are giving the subjects objects that may impact their decisions.

#.05c: A response variable could be number of ounces scooped by a subject, which is quantitative.

#0.5d: The explanatory variables are bowl size (large, 34oz, vs small, 17 oz) and spoon
#size(large, 3 oz, vs small, 2 oz)
```

```
#.14a: 12.69235 predicted wins
PF = 441/16
PA = 250/16
wins = 3.6 + .5*PF -.3*PA #+epsilon
wins
```

```
## [1] 12.69375
```

```
#.14b: The residual is 1.30625. This residual seems pretty small given the context,
#it is pretty impressive that the predicted was that accurate. The predicted value was
#only 1.3 wins off the true number of wins.
resids = 14-wins
resids
```

```
## [1] 1.30625
```

```
#.14c: This means that the actual number of wins was 3.48 wins lower than what was
#predicted by the model.
```

```
#.15a: For a wooden roller coaster the model predicts a top speed of 54mph.

#.15b: For a steel roller coaster the model predicts a top speed of 61.6mph
TypeCode1 = 1 #steel roller coasters
TypeCode0 = 0 #wooden roller coasters
TopSpeedsteel = 54 + 7.6*TypeCode1 #+ epsilon
TopSpeedwood = 54 + 7.6*TypeCode0 #+ epsilon
TopSpeedsteel
```

```
## [1] 61.6
```

```
TopSpeedwood
```

```
## [1] 54
```

```
#.15c: The difference in speeds between the coasters is 7.6 mph. This value appears in the
#equation as the slope value, which makes sense because the slope represents the change
#in the dependent variable (speed mph) for a one unit change in the independent
#variable (type of roller coaster)
TopSpeedDIFF = TopSpeedsteel - TopSpeedwood
TopSpeedDIFF
```

```
## [1] 7.6
```

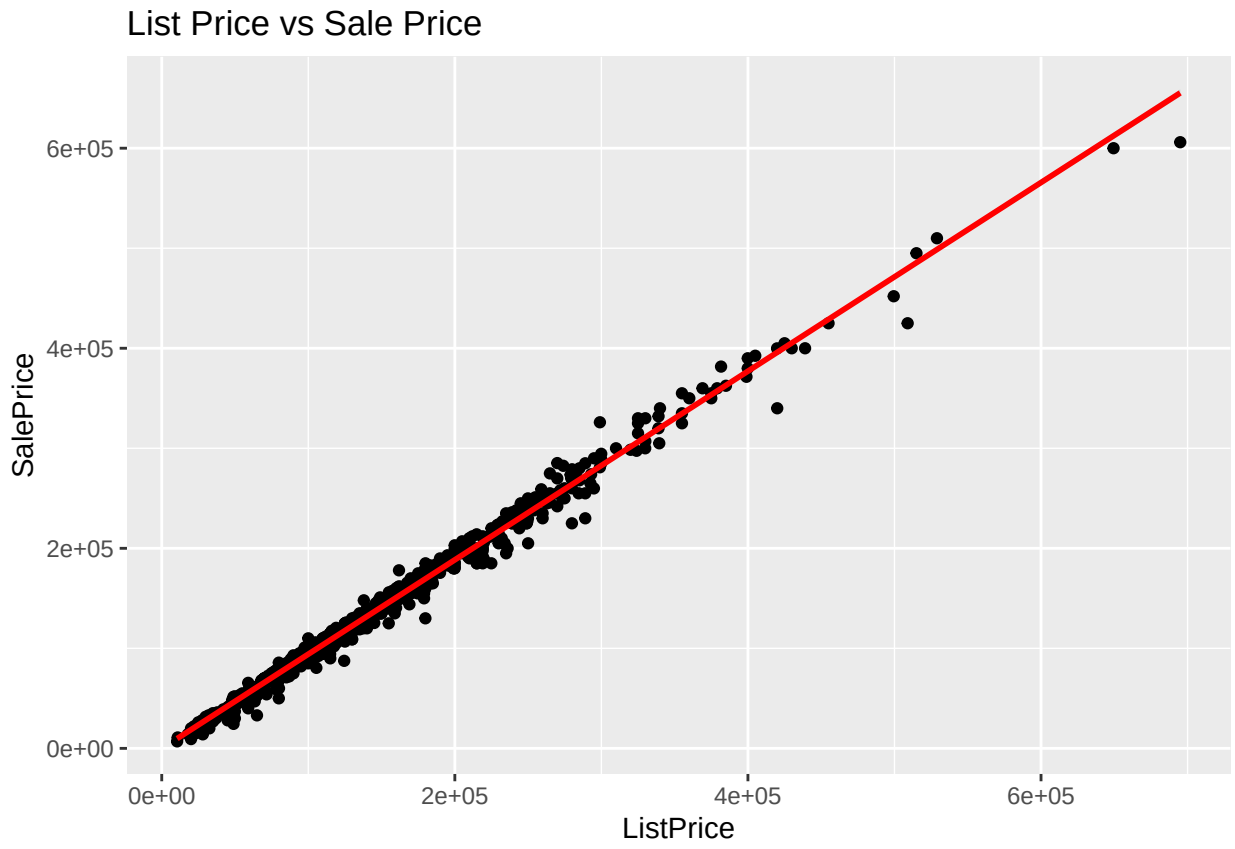
```
#.16a:
#Age: I expect the coefficient for age to be negative, because as the coaster gets older it
#will slow down due to rust and other such environmental factors.
#Total Length: I expect the coefficient for Length to be fairly neutral, I don't think that a
#longer coaster will have more speed. It may be slightly positive.
#Maximum Height: The coefficient for max height will definitely be positive, the higher the
#coaster, the more momentum/potential energy that is achievable and therefore more speed.
#Maximum Vertical Drop: I assume this is measured with an angle, but the steeper/higher the
#vertical drop, the faster the coaster will go, so positive coefficient.

#.16b: Aside from the type of roller coaster, the drop seems to be the biggest predictor.
#Drop has a slope of .11 which is higher than any other variable aside from TypeCode
#which we already know is the biggest predictor.

#.16c: The signs of the coefficients are exactly as I would expect. I predicted them
#perfectly, especially Length which I said would be neutral/slightly positive.

#.16d: 59.97mph is the top speed of this coaster.
Height = 150
Drop = 100
Length = 4000
Age = 10
Speed = 33.4 + .10*Height + .11*Drop + .0007*Length - .023*Age - 2.0*TypeCode1 #+epsilon
Speed
```

```
## [1] 59.97
```

```
library(ggplot2)
#1.20a: This pattern looks very linear, with a greater concentration in the lower
#left corner which means there are less expensive houses in this dataset.
library(Stat2Data)
data("GrinnellHouses")
#plot(GrinnellHouses$ListPrice, GrinnellHouses$SalePrice,
#    xlab = "List Price", ylab = "Sale Price",
#    main = "Sale Price vs. List Price")


ggplot(GrinnellHouses, aes(ListPrice, SalePrice)) +
```

```
  geom_point() +
  geom_smooth(method = "lm", color = "red") + #LSreg. line
  ggtitle("List Price vs Sale Price")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
#not using ggplot
#1.20b:
mymodel <- lm(SalePrice ~ ListPrice, data = GrinnellHouses)



#1.20c: our slope is  9.431e-01, this means that for every unit in price that the
#list price goes up, the sales price goes up 9.431e-01. This means there is a positive
#correlation and the coefficient is positive.
summary(mymodel)
```

```
##
## Call:
## lm(formula = SalePrice ~ ListPrice, data = GrinnellHouses)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -55942  -3275    846   4141  44168
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.448e+02  5.236e+02  -0.277    0.782
## ListPrice    9.431e-01  3.201e-03 294.578   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8019 on 927 degrees of freedom
## Multiple R-squared:  0.9894, Adjusted R-squared:  0.9894
## F-statistic: 8.678e+04 on 1 and 927 DF,  p-value: < 2.2e-16
```

```
#1.22a: For a house listed at $99,500, the model would predict it to sell for $93,693.65.
ListPrice99 = 99500
SalesPrice99 = -1.448e+02 + 9.431e-01*ListPrice99
SalesPrice99
```

```
## [1] 93693.65
```

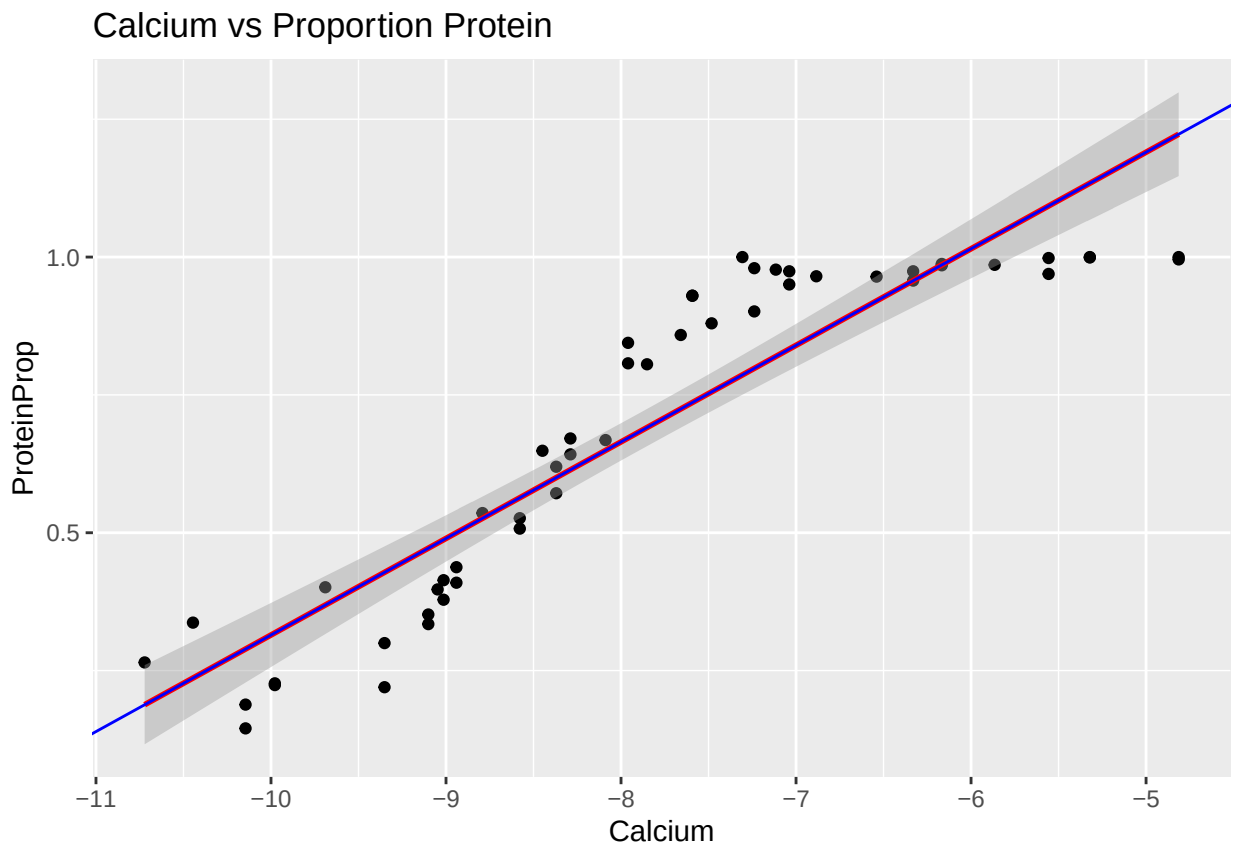
```
#1.22b: The residual is 306.35, which is very small, an excellent predictor.
resids122 = 94000 - SalesPrice99 #$94000, not $95000
resids122
```

```
## [1] 306.35
```

```
#1.22c: The linear regression does seem to fit the data very well in this circumstance.
#List prices and sales prices have a strong linear relationship.
```

```
data("Fluorescence")
#1.38a: protein prop = -11.1535 + 4.7750(calcium)
mymodel138 <- lm(ProteinProp ~ Calcium, data = Fluorescence)
m <- 0.17514 #slope
b <- 2.06586 #intercept
#This is a plot for Calcium vs Proportion of Protein, I have fit a model to the data and
#used abline to confirm the model is the correct regression line.
ggplot(Fluorescence, aes(Calcium, ProteinProp)) +
  geom_point() +
  geom_smooth(method = "lm", color = "red") + #i could do without SE, but I like to see it there
  geom_abline(intercept = b, slope = m, color = "blue") +
  ggtitle("Calcium vs Proportion Protein")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

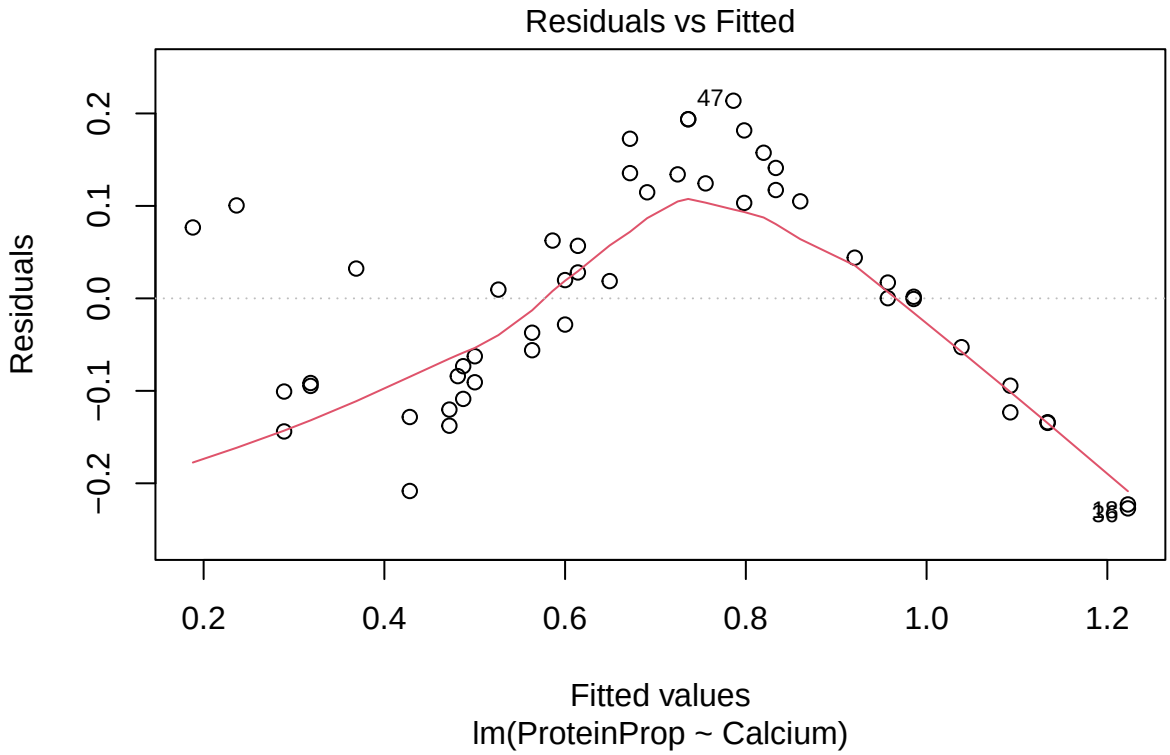


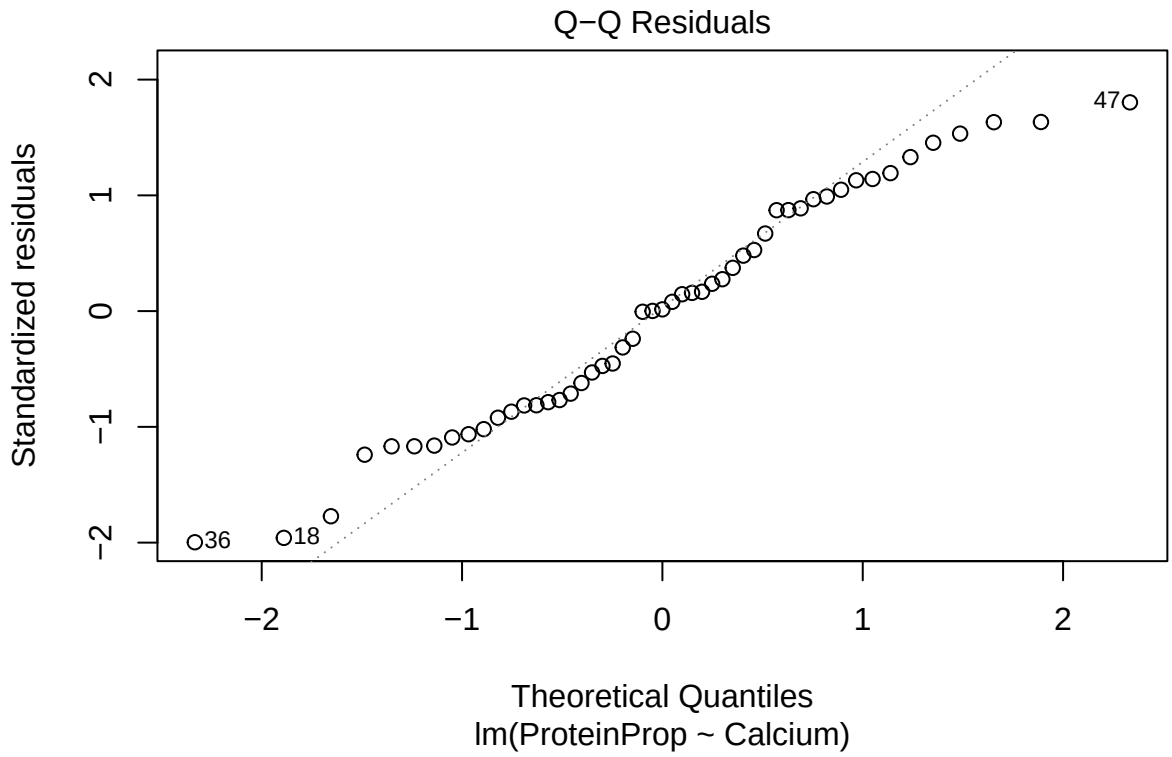

```
#1.38b: The standard error is 0.3018 for the regression.
summary(mymodel138)
```

```
##
## Call:
## lm(formula = ProteinProp ~ Calcium, data = Fluorescence)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.22712 -0.09454  0.00176  0.10410  0.21375
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.06586    0.08876   23.27   <2e-16 ***
## Calcium      0.17514    0.01107   15.82   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1199 on 49 degrees of freedom
## Multiple R-squared:  0.8363, Adjusted R-squared:  0.8329
## F-statistic: 250.3 on 1 and 49 DF,  p-value: < 2.2e-16
```

```
#1.38c: Regression line plotted above, twice to be sure that ggplots smooth function
#did it correctly. The residual line doesn't appear to be a good fit visually.
#The r squared value is fairly high though so it's probably good enough.
```

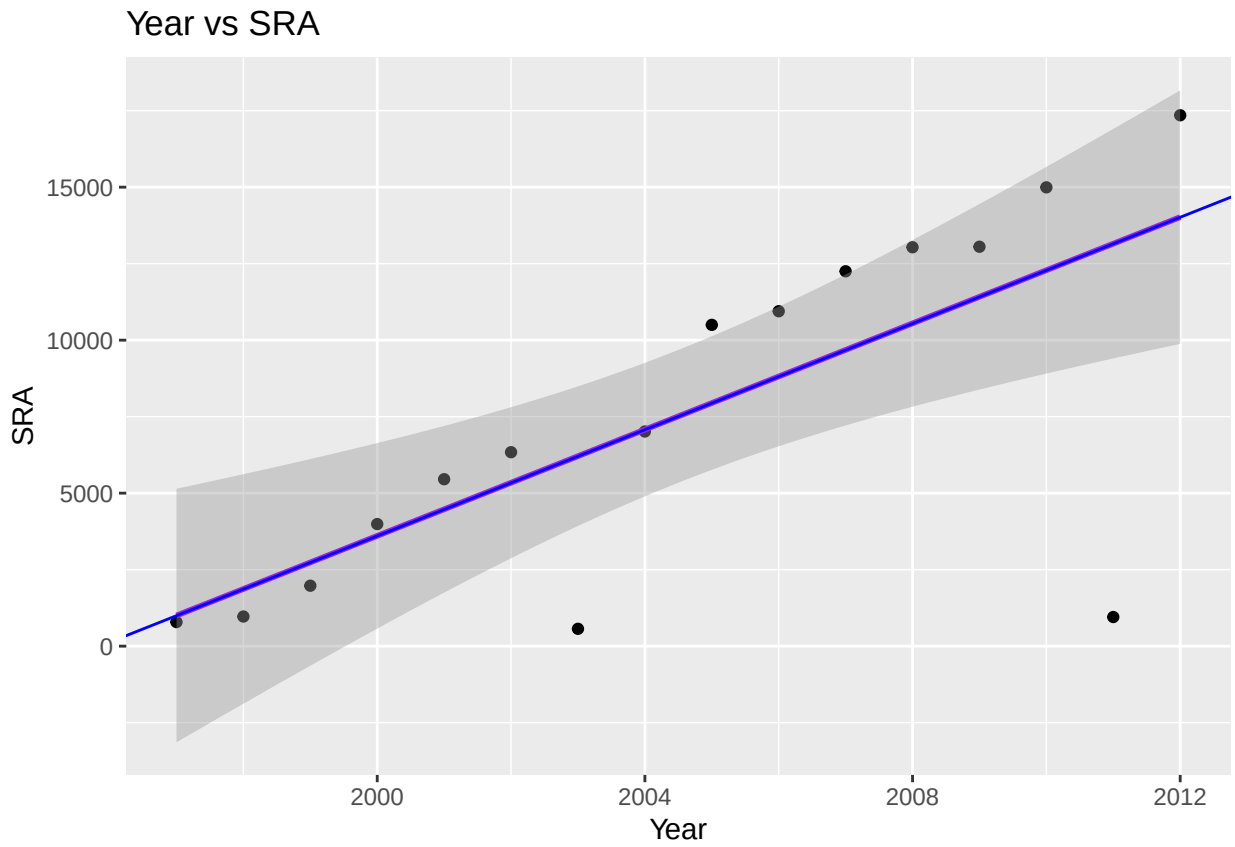
```r
#1.38d:
#Zero mean condition, the mean of the residuals seem to hover around 0, not
#perfectly but close enough.
#Constant Variance, this condition definitely isn't met, there is a upside-down U
#shaped curve in the Residuals vs Fitted graph.
#Independence: The residuals seem to be independent of one another based on the
#context of this question.
#Linearity, the data seems to be fairly linear in the Q-Q plot, not perfectly but close enough.
#Normality: the data follows the line well in the Q-Q residual plot.

#the two residual plots for residuals vs fitted and QQ Residuals are used to check
#conditions and find outliers.
plot(mymodel138, 1:2)
```






```
data("Retirement")
#1.47a:
#make sure that the int and slope are right from summary stats
m1 <- 868.0
b1 <- -1732400.2
#This is a plot of Year vs SRA, I am using this plot to find the regression line
#and to see how well the model fits the data.
ggplot(Retirement, aes(Year, SRA)) +
  geom_point() +
  geom_smooth(method = "lm", color = "purple") +
  geom_abline(intercept = b1, slope = m1, color = "blue") +
  ggtitle("Year vs SRA")
```

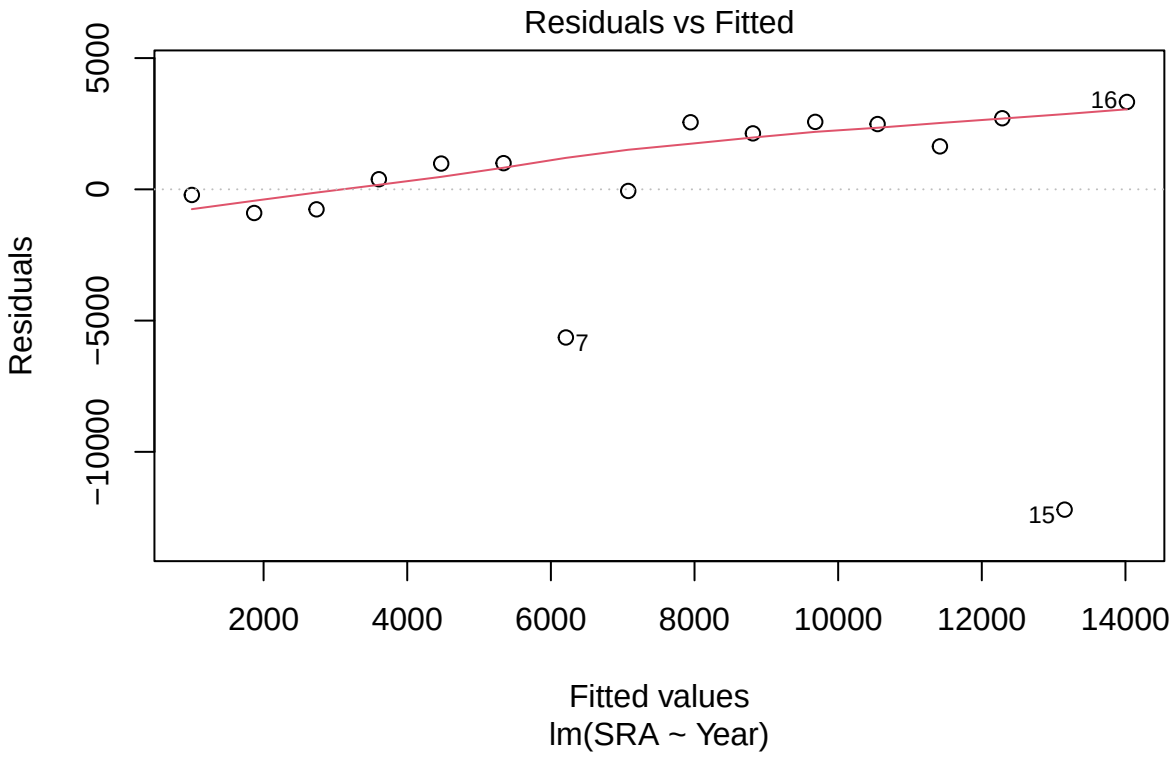
```
## `geom_smooth()` using formula = 'y ~ x'
```

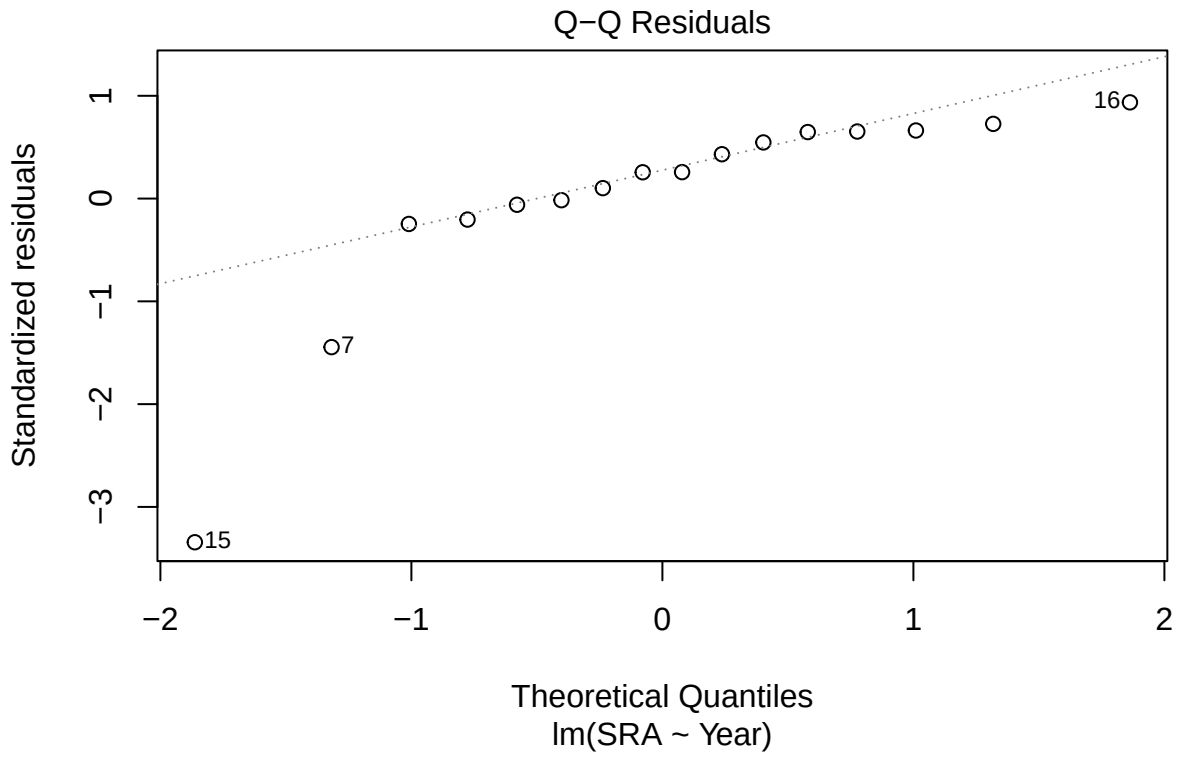



```
#linear model to use later for summary stats and residual plots
mymodel147 <- lm(SRA ~ Year, data = Retirement)
summary(mymodel147)
```

```
##
## Call:
## lm(formula = SRA ~ Year, data = Retirement)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -12201.0   -350.9    990.2   2503.6   3328.7
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1732400.2   439864.9  -3.938  0.00148 **
## Year              868.0      219.4   3.956  0.00144 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4046 on 14 degrees of freedom
## Multiple R-squared:  0.5278, Adjusted R-squared:  0.494
## F-statistic: 15.65 on 1 and 14 DF,  p-value: 0.001436
```

```
#these residual plots are used for checking conditions and finding residual outliers.
plot(mymodel147, 1:2)
```


Residuals vs Fitted
Residuals
5000
0
−5000
−10000
2000
4000
6000
8000
10000
12000
14000
16
7
15
Fitted values
lm(SRA ~ Year)




```
#computing residuals for sabbatical years
Year2003 = 2003
Year2011 = 2011
SRA2003 = 868.0*Year2003 -1732400.2 #+epsilon
SRA2011 = 868.0*Year2011 -1732400.2 #+epsilon
SRA2003 #6203.8
```

```
## [1] 6203.8
```

```
SRA2011 #13147.8
```

```
## [1] 13147.8
```

```
resid2003 = 566.25 - 6203.8
resid2011 = 952.04 - 13147.8
resid2003 #-5637.55
```

```
## [1] -5637.55
```

```
resid2011 #-12195.76
```

```
## [1] -12195.76
```

```r
#the residuals for 2003 and 2011 are massive outliers, this is apparent from the numerical
#evidence (the residual values themselves) which are way off the residuals of the other
#years. This is also apparent from the graphical evidence. When we look at the residuals
#vs fitted plot, points 7 (2003) and 15 (2011) are way below the other values which all
#hover around 0.
```

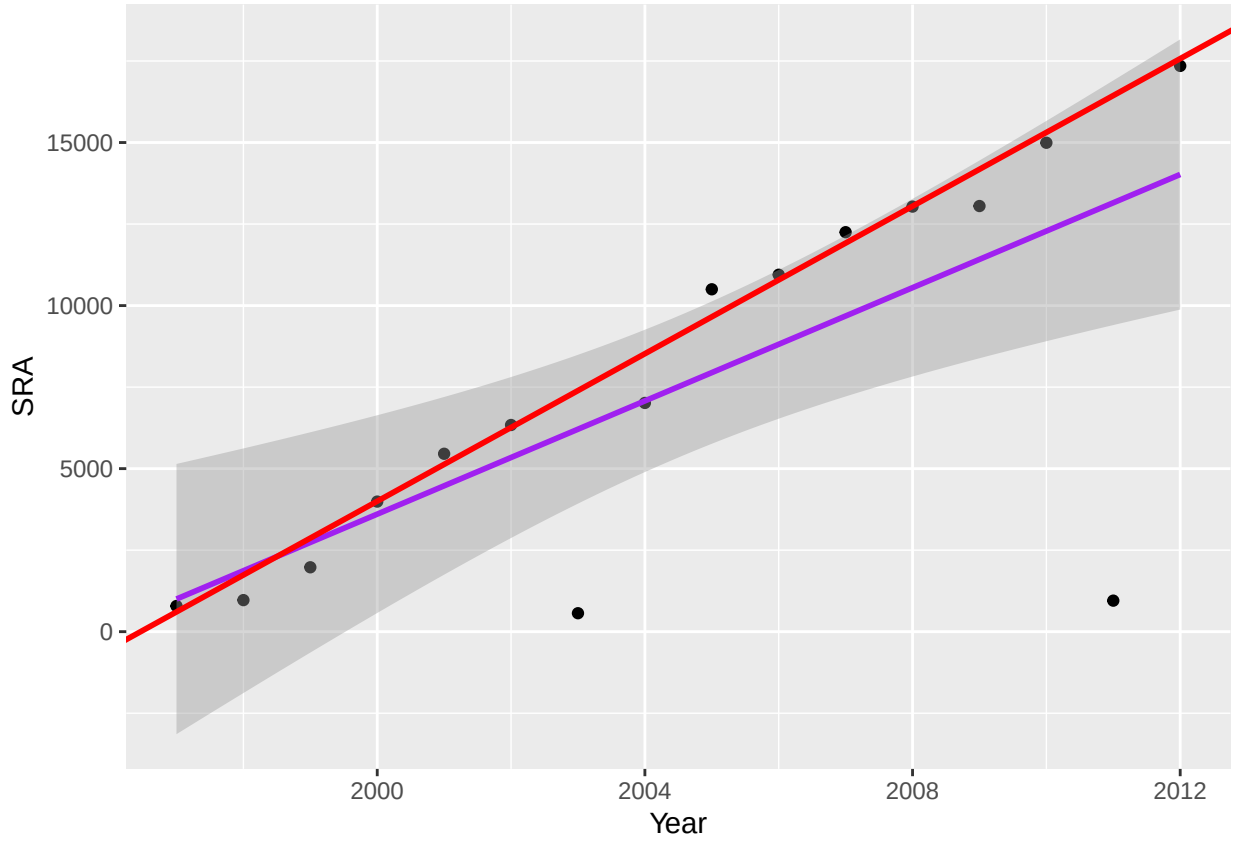
```r
#1.47b:
#removing points
mymodel147b <- lm(SRA ~ Year, data = Retirement [-c(7,15), ])
summary(mymodel147b)
```

```
##
## Call:
## lm(formula = SRA ~ Year, data = Retirement[-c(7, 15), ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1299.1  -446.8   198.8   384.4  1055.1
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.258e+06  7.891e+04  -28.62 2.06e-12 ***
## Year         1.131e+03  3.937e+01   28.72 1.97e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 674.7 on 12 degrees of freedom
## Multiple R-squared:  0.9857, Adjusted R-squared:  0.9845
## F-statistic: 825.1 on 1 and 12 DF,  p-value: 1.97e-12
```

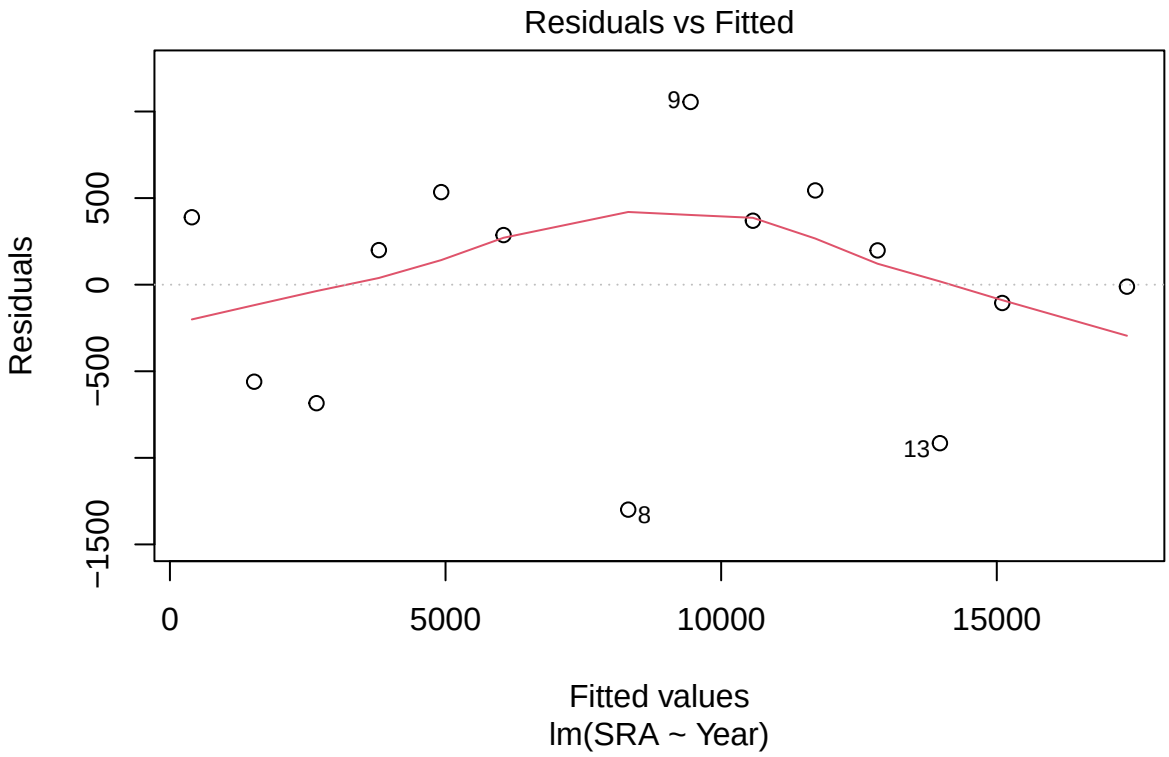
```r
#new slope and intercept to show difference
m2 <- 1.131e+03
b2 <- -2.258e+06

#plotted with the old regression line (purple) and new without the outliers (red).
ggplot(Retirement, aes(Year, SRA)) +
  geom_point() +
  geom_smooth(method = "lm", color = "purple") +
  geom_abline(intercept = b2, slope = m2, color = "red", size = 1)
```

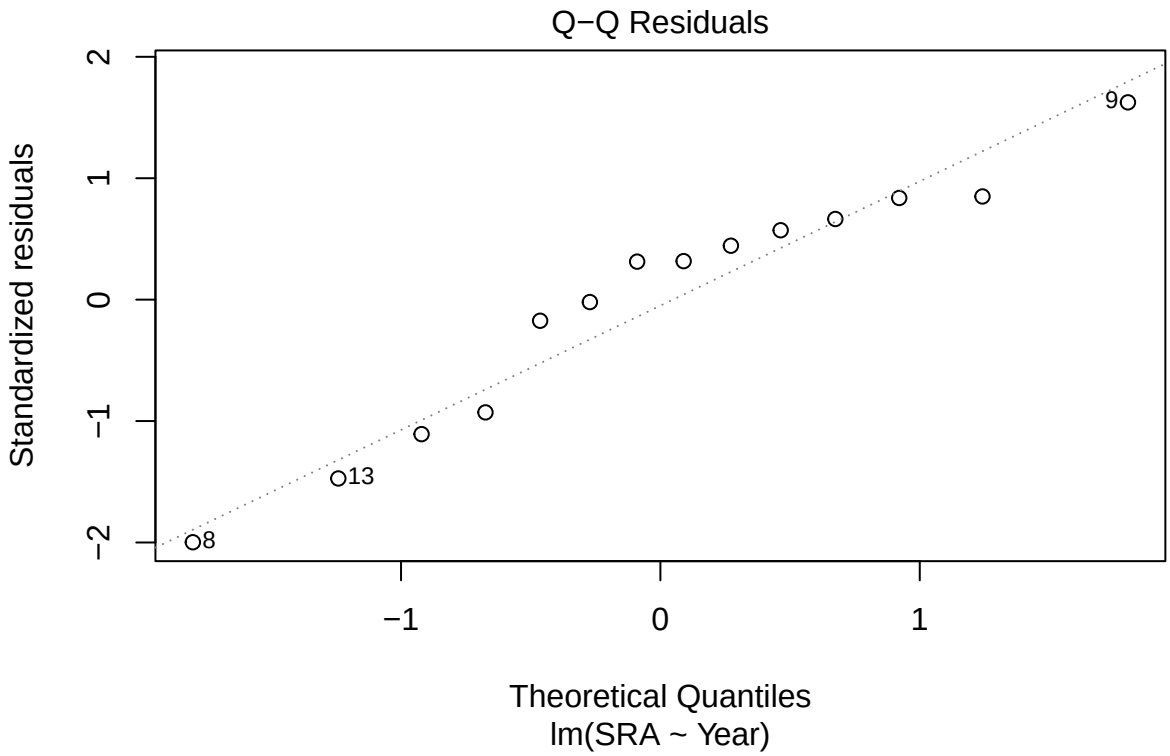
```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
plot(mymodel147b, 1:2)
```


Residuals vs Fitted
Residuals
500
0
−500
−1500
9
13
8
0
5000
10000
15000
Fitted values
lm(SRA ~ Year)




```
#I would say that the new model provides a better fit for the data because the R^2 value is much
#higher (.98 vs .52) without the outliers, and the line looks like a WAY better fit in the
#regression and the residual plots.
```