Challenges and Remediation for Least Squares

Debopriya Ghosh

2019-07-16

Background

This lecture reviews tools for detecting departures from basic assumptions for building and using predictive linear regression models.Suppose we have the following linear model in predictors:

$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots + \beta_{p-1} x_{i,p-1} + \epsilon_i$, where $i = 1, 2, \ldots, n$

Ordinary Least Squares

Here we write some advanced formulas in matrix algebra for your enlightenment. You can just ignore this if you care to. The oridinary least squares (OLS) estimate of the p vector of parameters β is the formula:

$\hat{\beta} = (X^T X)^{-1} X^T Y$, Where Y is the n vector of the output variable and X is the $n \times p$ design matrix of input variables.

Standard Assumptions

Constant Variance and Zero Covariance

$E(\epsilon_i) = 0$

$VAR(\epsilon_i) = \sigma^2$

$COV(\epsilon_i, \epsilon_j) = 0$, where $i, j = 1, 2, \ldots, n$

Normal Errors

$\epsilon_i \ N(0, \sigma^2) \ i = 1, 2, \ldots, n$

Correct Model Specification

$E(y_i) = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots + \beta_{p-1} x_{i,p-1}$ where $i = 1, 2, \ldots, n$

In the following sections we will present several diagnostic tools for detecting departures from standard assumptions.

Anscombe's Quartet Data.

Here we will use *Anscombe's Quartet Data* which precisely lays out the problems encountered with ordinary least square method.

Checking for non-constant variance

We will use *residual vs. fitted plot* and *scale-location* plot to detect non-constant error variance. We will also use F-test for detecting non-constant variance.

Checking for non-normal errors

Here, we introduce the *Normal Q-Q plot* and the *Shapiro-Wilk test*, to detect the non-normality in the errors. We present the *Box-Cox Power Transformation* of the output variable for making the errors

normal with constant variance.

Checking for influential outliers

Many times outliers are influential. They destory the normality of errors and distort the estimate of the model. Here, we will visualize the outliers in the data by using 3D graphics. For data with higher dimension, we present *leverage index* and *Cook's Distance* for identifying influential outliers.

Checking for correct model specification

We will use *added variable plot*, *partial residual plot*, and the *CERES plot* for identifying model miss-specification.

Interaction Models

The *CERES plot* reveal data subgroups which can be handled with a interaction model. Here we present interaction models for fitting subgroups of data.

Checking for collinearity in predictors

R Libraries

```
library("faraway")
library("car")
library("ggplot2")
library("gridExtra")
library("scatterplot3d")
# library('rgl')
```

Anscombe's Quartet Data

Anscombe configured for artificial datasets in the 1960's to illustrate the problems with traditional regression methods. The datasets consits of one regular behaving dataset and three ill-behaving datasets, that except upon visual inspection, look perfectly normal.

```
data(Quartet)
names(Quartet)
```

```
## [1] "x"  "y1" "y2" "y3" "x4" "y4"
```

- y1 is "nice" data, but y2, y3, and y4 yield exactly the same estimate and standard error as y1 when fitting a simple linear model.

We will observe

- y2 comes from a quadratic model, an example of model mispeification
- y3 has an outlier

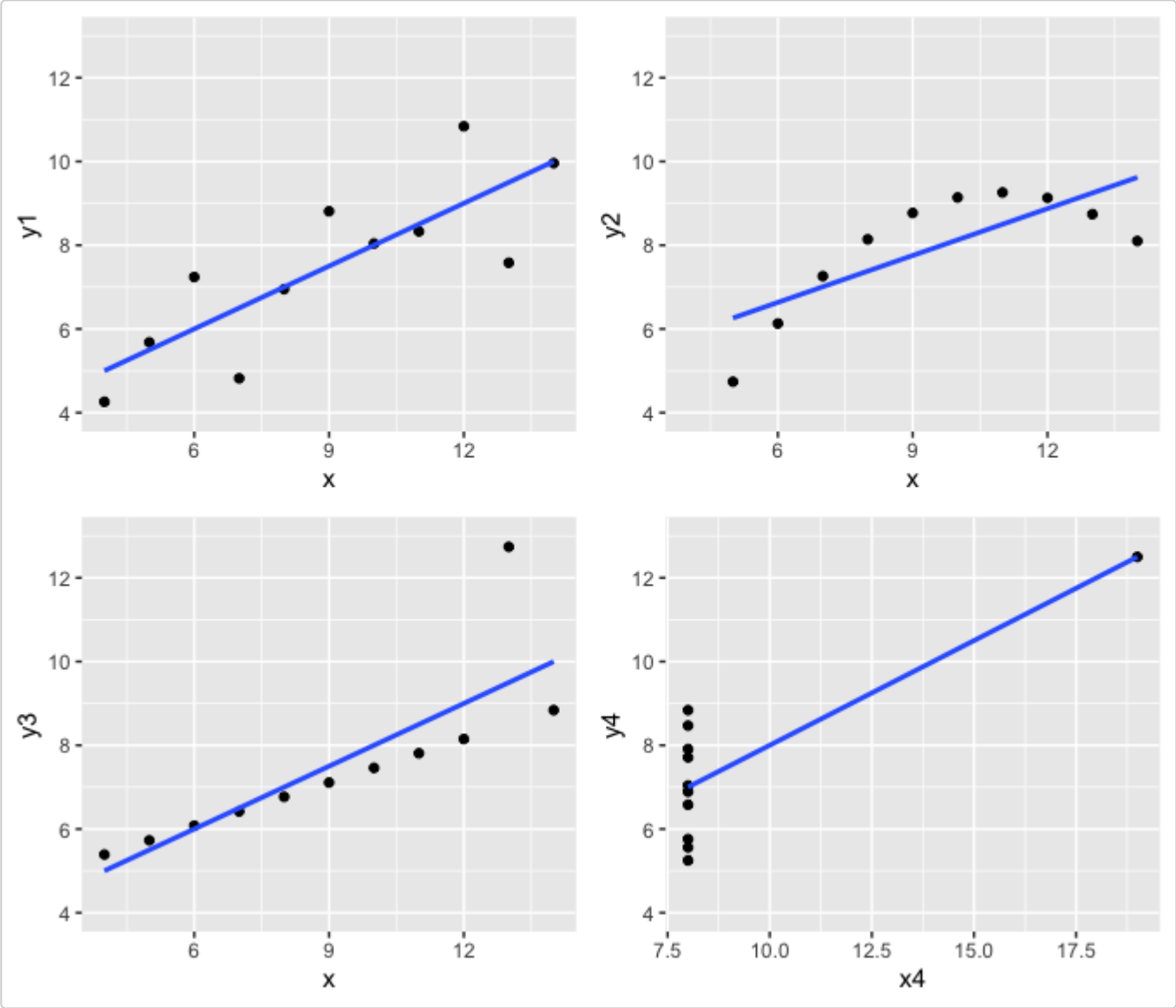
- y4 has a highly leveraged point

Fit a simple linear model in all four datasets

```
g1 = lm(y1 ~ x, Quartet)
g2 = lm(y2 ~ x, Quartet)
g3 = lm(y3 ~ x, Quartet)
g4 = lm(y4 ~ x4, Quartet)
```

Plot the datasets with the fitted line.

```
library(gridExtra)
p1 = qplot(x, y1, data = Quartet) + geom_smooth(method = "lm", se = FALSE) +
    ylim(4, 13)
p2 = qplot(x, y2, data = Quartet) + geom_smooth(method = "lm", se = FALSE) +
    ylim(4, 13)
p3 = qplot(x, y3, data = Quartet) + geom_smooth(method = "lm", se = FALSE) +
    ylim(4, 13)
p4 = qplot(x4, y4, data = Quartet) + geom_smooth(method = "lm", se = FALSE) +
    ylim(4, 13)
grid.arrange(p1, p2, p3, p4, nrow = 2)
```



Compare LS estimates and the standard error of estimate.

```
compareCoefs(g1, g2, g3, g4)
```

```
## Calls:
## 1: lm(formula = y1 ~ x, data = Quartet)
## 2: lm(formula = y2 ~ x, data = Quartet)
## 3: lm(formula = y3 ~ x, data = Quartet)
## 4: lm(formula = y4 ~ x4, data = Quartet)
##
##             Model 1 Model 2 Model 3 Model 4
## (Intercept)    3.00    3.00    3.00    3.00
## SE             1.12    1.13    1.12    1.12
##
## x             0.500   0.500   0.500
## SE            0.118   0.118   0.118
##
## x4                                    0.500
## SE                                    0.118
##
```

Conclusion:

- The fitted lines have exactly the same intercept and slope for each of the four datasets
- The standard errors of estimate are also the same
- Plots of the datasets are quite different

Key Takeaways:

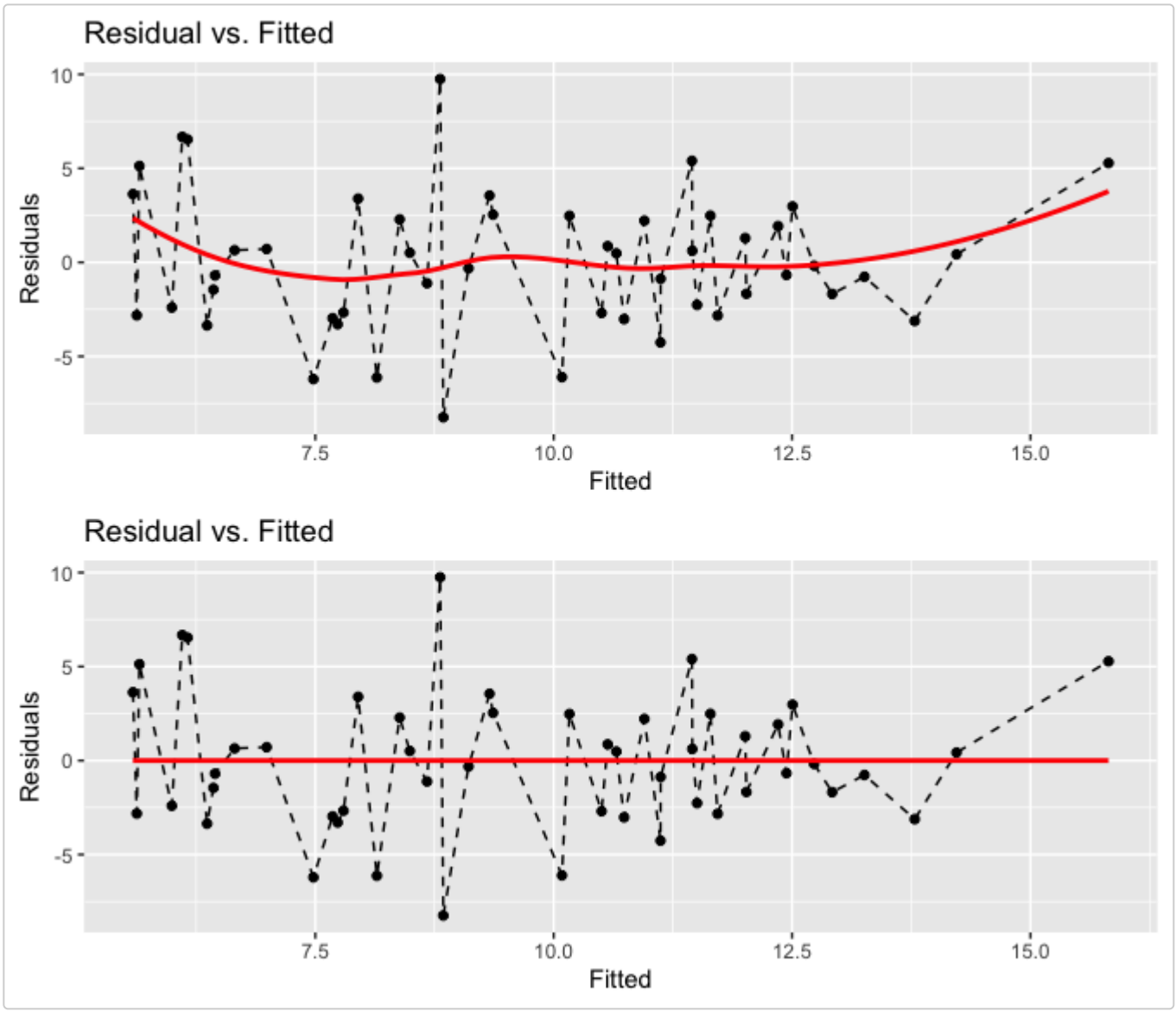
- Never go on the estimates and standard errors of least squares results alone, must also visualize the data
- In processing large datasets, or many datasets, or datasets with large number of variables, it may not be efficient to visualize the data, therefore there is need to replace least squares with a robust method that will not be influenced by outliers or be restricted by a linear model.

Checking for non-constant variance

Residual Plots

- We plot residuals and absolute values of residuals versus predicted values in the savings dataset.

```
data(savings)
g = lm(sr ~ pop15 + pop75 + dpi + ddpi, savings)

mod = fortify(g)

p1 = qplot(.fitted, .resid, data = mod) + geom_line(yintercept = 0, linetype = "dashed") +
    labs(title = "Residual vs. Fitted", x = "Fitted", y = "Residuals") + geom_smooth(color = "red",
    se = F)

p2 = qplot(.fitted, .resid, data = mod) + geom_line(yintercept = 0, linetype = "dashed") +
    labs(title = "Residual vs. Fitted", x = "Fitted", y = "Residuals") + geom_smooth(method = "lm",
    color = "red", se = F)

grid.arrange(p1, p2, nrow = 2)
```



- We have seen the Residual vs Fitted plot before where we have used it to detect outliers, here we use it also to detect patterns in residuals that would indicate nonconstant error variance.
- The second plot is called the Scale-Location plot, which strengthens the pattern in the residuals by ploting the absolute values.
- In both plys, we some evidence of heterskedasticity, in other words nonconstant error variance.
- In the Residual vs Fitted plot, we have added a “nonparametric” fitted line called loess, which stands for locally weighted scatterplot smoothing.
- Other ggplot method (function) available are lm, glm, gam, loess, rlm. For datasets with n < 1000 default is loess. For datasets with 1000 or more observations defaults to gam, see gam for more details.
- We used the lm method for the Scale-Location plot. We see that the downward sloping line strengthens our suspicion of nonconstant error variance.

An approximate test of non-constant error variance.

```
summary(lm(abs(residuals(g)) ~ fitted(g)))
```

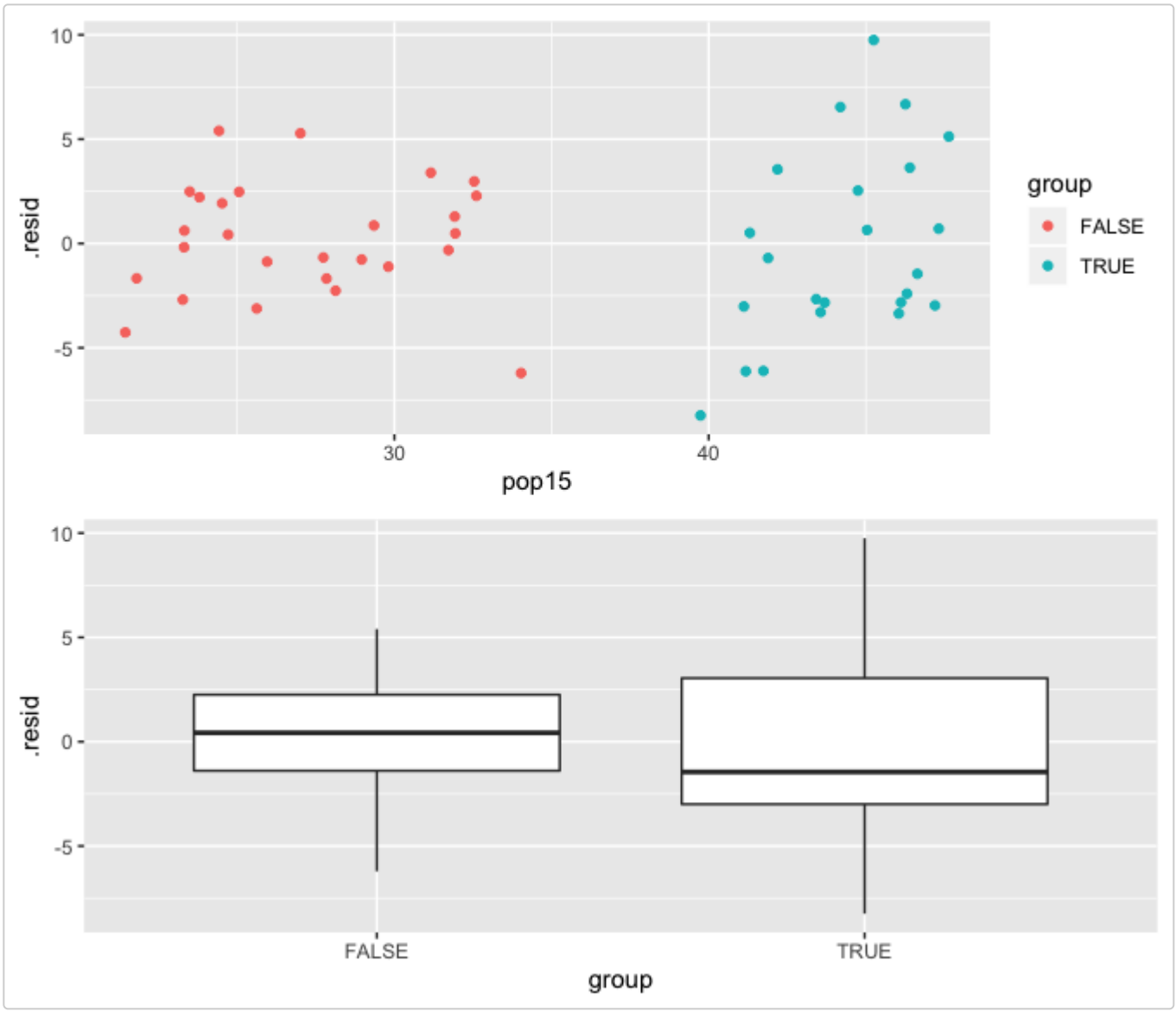
```
##
## Call:
## lm(formula = abs(residuals(g)) ~ fitted(g))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.8395 -1.6078 -0.3493  0.6625  6.7036
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.8398     1.1865   4.079  0.00017 ***
## fitted(g)    -0.2035     0.1185  -1.717  0.09250 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.163 on 48 degrees of freedom
## Multiple R-squared:  0.05784,    Adjusted R-squared:  0.03821
## F-statistic: 2.947 on 1 and 48 DF,  p-value: 0.0925
```

- We look at the t-test for the slope coefficient with null hypthesis that the slope is zero. At the 10% level of significance, we conclude that the slope is not zero since the p-value, 0.09250, is less than 0.10.
- This test is only approximate as the degrees of freedom number for the t-distribution, 48, is theorectically too large.

An F-test for non-constant error variance between two groups defined by a predictor

- This test is similar to the Breusch-Pagan test of heteroskedasticity
- We divide the residuals into two groups: pop15 > 35 and pop15 < 35.

```
group = savings$pop15 > 35
p1 = qplot(pop15, .resid, data = mod, color = group)
p2 = qplot(group, .resid, data = mod, geom = "boxplot")
grid.arrange(p1, p2, nrow = 2)
```




```
var.test(residuals(g)[savings$pop15 > 35], residuals(g)[savings$pop15 < 35])
```

```
##
##  F test to compare two variances
##
## data:  residuals(g)[savings$pop15 > 35] and residuals(g)[savings$pop15 < 35]
## F = 2.7851, num df = 22, denom df = 26, p-value = 0.01358
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  1.240967 6.430238
## sample estimates:
## ratio of variances
##           2.785067
```

- The boxplot clearly indicates that the residuals' variance of group pop15 < 35 is larger than the variance of group pop15 > 35
- The F-test compares the sample variances of the residuals of the two groups, with null hypothesis that the two variances are equal.

- We conclude that there is difference in the variance between these two groups with level of significance 10% since the p-value, 0.01, is less than 0.10

A variance stabilizing transformation

Remediate non-constant variance in two ways:

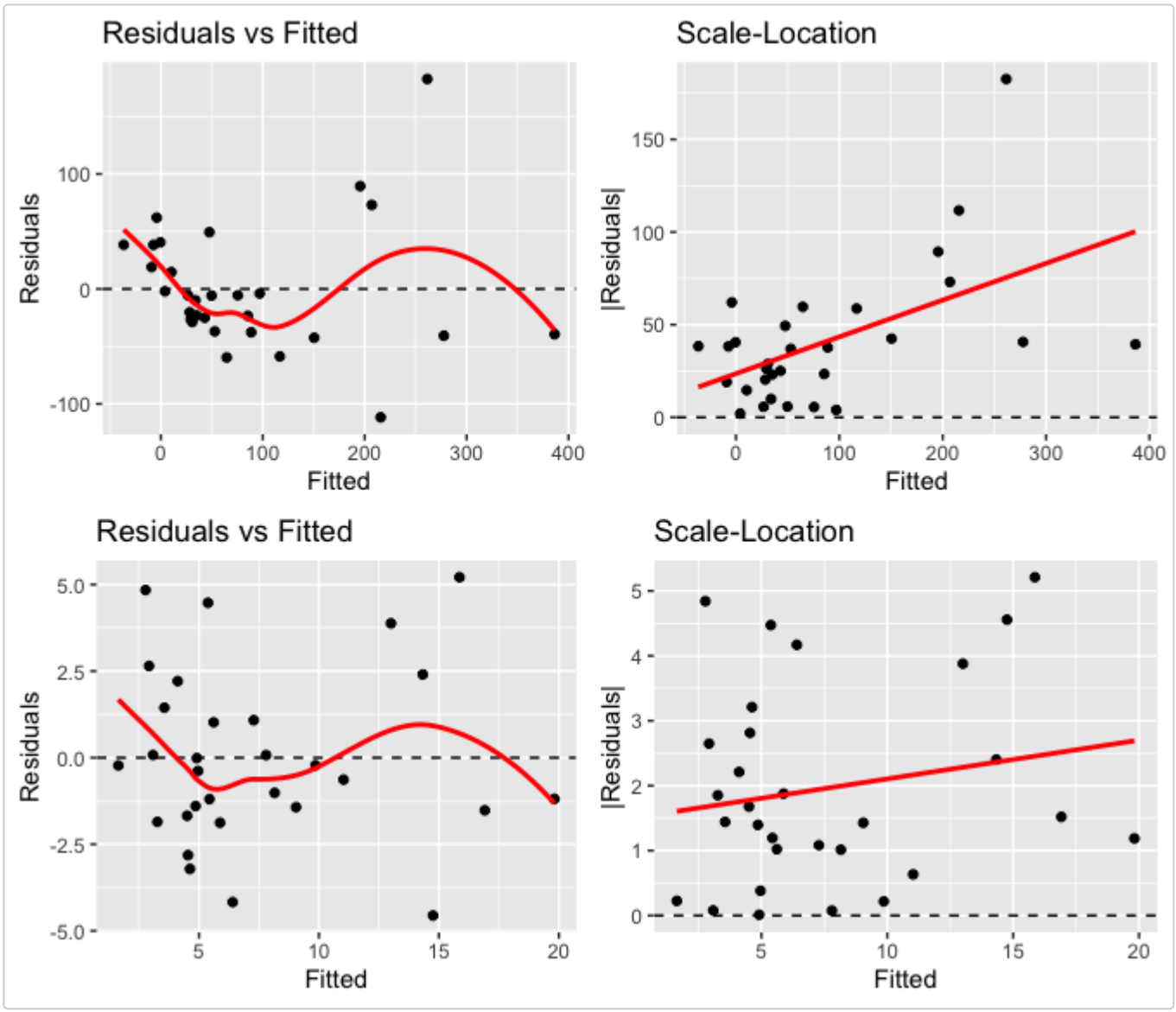
$y \leftarrow log(y)$

$y \leftarrow \sqrt{y}$

Sometimes it is difficult to determine which transformation to use. Try one if it is not effective try the other. Add a constant in the transform to make all values positive.

Using the gala data

```
data(gala)

gg = lm(Species ~ Area + Elevation + Scruz + Nearest + Adjacent, gala)
gs = lm(sqrt(Species) ~ Area + Elevation + Scruz + Nearest + Adjacent, gala)
modgg = fortify(gg)
modgs = fortify(gs)

p1 = qplot(.fitted, .resid, data = modgg) + geom_hline(yintercept = 0, linetype = "dashed") +
    labs(title = "Residuals vs Fitted", x = "Fitted", y = "Residuals") + geom_smooth(color = "red",
    se = F)

p2 = qplot(.fitted, abs(.resid), data = modgg) + geom_hline(yintercept = 0,
    linetype = "dashed") + labs(title = "Scale-Location", x = "Fitted", y = "|Residuals|") +
    geom_smooth(method = "lm", color = "red", se = F)

p3 = qplot(.fitted, .resid, data = modgs) + geom_hline(yintercept = 0, linetype = "dashed") +
    labs(title = "Residuals vs Fitted", x = "Fitted", y = "Residuals") + geom_smooth(color = "red",
    se = F)

p4 = qplot(.fitted, abs(.resid), data = modgs) + geom_hline(yintercept = 0,
    linetype = "dashed") + labs(title = "Scale-Location", x = "Fitted", y = "|Residuals|") +
    geom_smooth(method = "lm", color = "red", se = F)

grid.arrange(p1, p2, p3, p4, nrow = 2)
```



We perform the approximate test of heteroskedasticity on the transformed data.

```
summary(lm(abs(residuals(gs)) ~ fitted(gs)))
```

```
##
## Call:
## lm(formula = abs(residuals(gs)) ~ fitted(gs))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.8988 -1.2848 -0.3396  1.0177  3.1682
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.50779    0.53439   2.822  0.00869 **
## fitted(gs)   0.05968    0.06009   0.993  0.32915
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.55 on 28 degrees of freedom
```

```
## Multiple R-squared:  0.03403,    Adjusted R-squared:  -0.0004723
## F-statistic: 0.9863 on 1 and 28 DF,  p-value: 0.3292
```

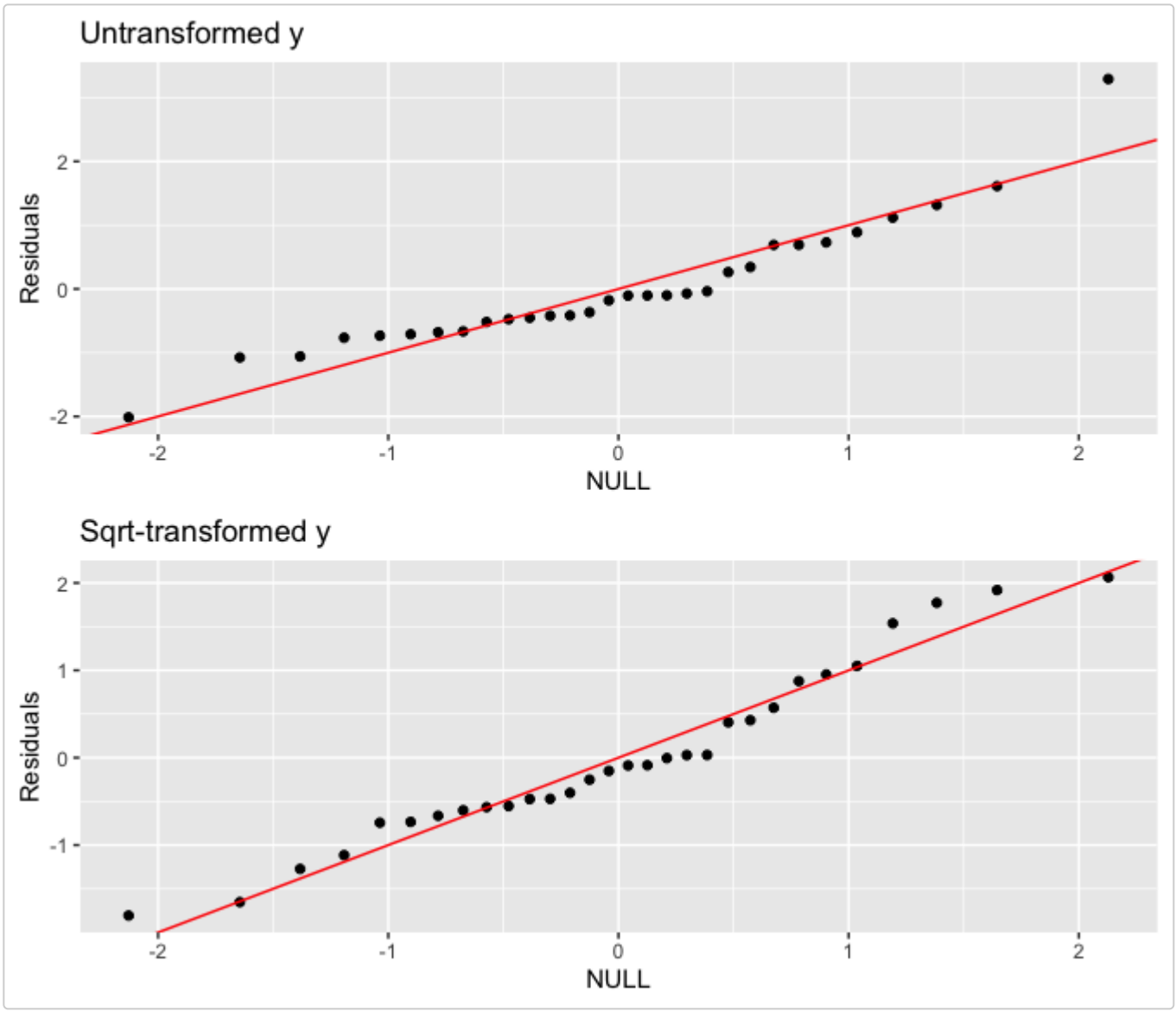
Conclusion: - The t-test does not reject constant error variance with a level of significance 10%, since the p-value, 0.3292, is greater than 0.10

Checking for non-normal errors

Normal Q-Q plots for detecting non-normality

- Keeping with the gala data we check the residuals for normality using the Normal Q-Q plot first on the untransformed model gg and then transformed model gs.

```
p1 = qplot(sample = scale(.resid), data = modgg) + geom_abline(intercept = 0,
    slope = 1, color = "red") + labs(title = "Untransformed y", y = "Residuals")

p2 = qplot(sample = scale(.resid), data = modgs) + geom_abline(intercept = 0,
    slope = 1, color = "red") + labs(title = "Sqrt-transformed y", y = "Residuals")

grid.arrange(p1, p2, nrow = 2)
```

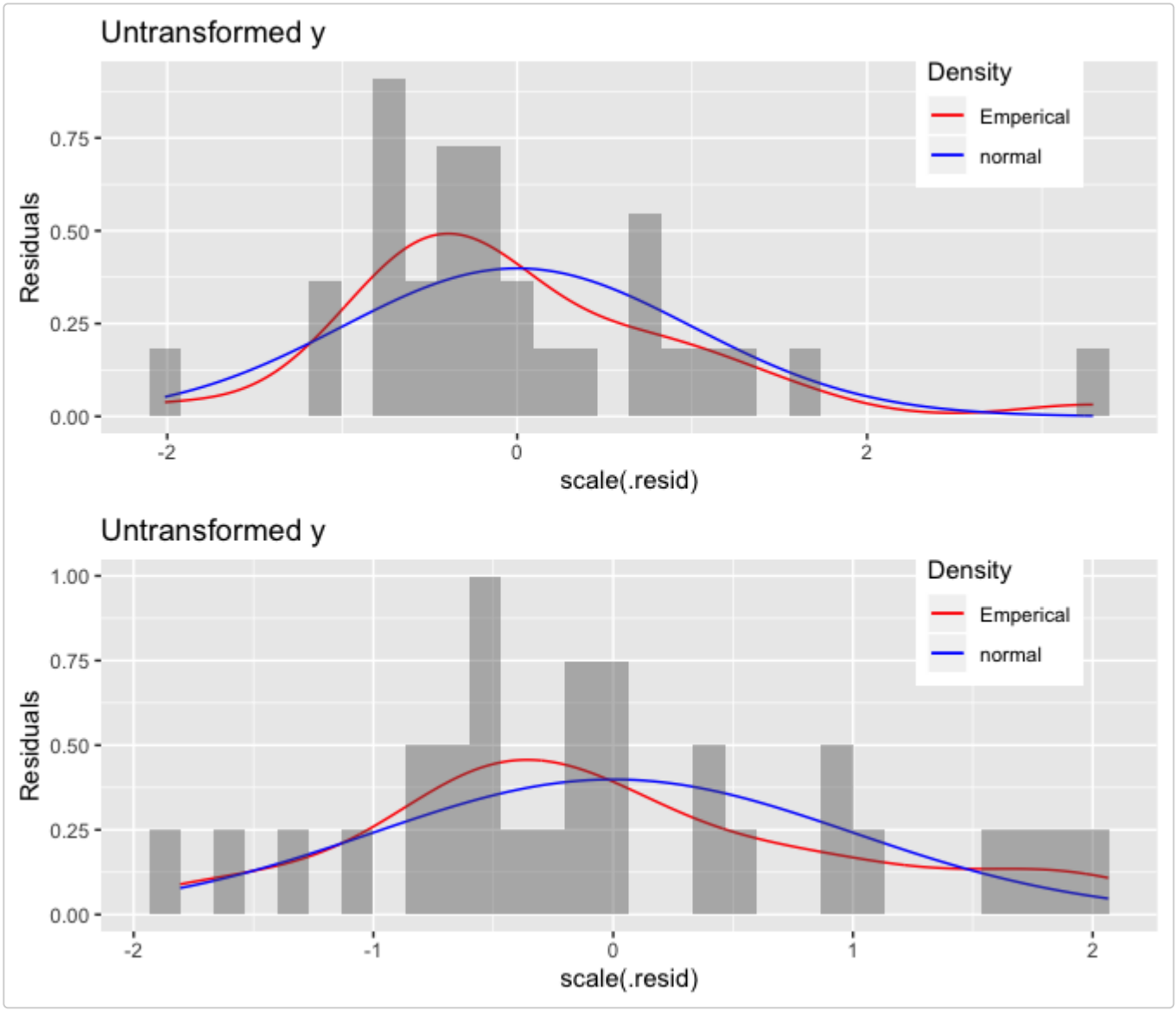


Histograms, kernel density plots

- We can also have a look at the histogram of the residuals with overlays of the kernel density estimator and the standard normal density.
- The histogram of the residuals alone is not suitable for detecting non-normality.
- However, the kernel density estimator compared to the normal density indicates that the residuals could be non-normal.

```
p1 = qplot(scale(.resid), data = modgg, geom = "blank") + geom_line(aes(y = ..density..,
    color = "Emperical"), stat = "density") + stat_function(fun = dnorm, aes(color = "normal")) +

geom_histogram(aes(y = ..density..), alpha = 0.4) + scale_color_manual(name = "Density",
    values = c("red", "blue")) + theme(legend.position = c(0.85, 0.85)) + labs(title = "Untransformed
y",
    y = "Residuals")


p2 = qplot(scale(.resid), data = modgs, geom = "blank") + geom_line(aes(y = ..density..,
    color = "Emperical"), stat = "density") + stat_function(fun = dnorm, aes(color = "normal")) +
```

```
geom_histogram(aes(y = ..density..), alpha = 0.4) + scale_color_manual(name = "Density",
    values = c("red", "blue")) + theme(legend.position = c(0.85, 0.85)) + labs(title = "Untransformed
y",
    y = "Residuals")

grid.arrange(p1, p2, nrow = 2)
```



- Clearly the residuals of the untransformed model are not normal
- The residuals of the sqrt-transformed model look better, but may not be normal

The Shapiro-Wilk test of normality

- Here we test the normality of residuals for model using the gala datset
- First we test the untransformed model and then the srt-transformed model

```
shapiro.test(residuals(gg))
shapiro.test(residuals(gs))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(gg)
## W = 0.91351, p-value = 0.01826
##
##
##  Shapiro-Wilk normality test
##
## data:  residuals(gs)
## W = 0.95869, p-value = 0.2867
```

- We reject the null hypothesis of normality for the residuals of untransformed model with level of significance 10% since the p-value is less than 0.10
- We fail to reject the null hypothesis of normality for the residuals of sqrt-transformed model with level of significance 10% since the p-value is greater than 0.10
- The *Shapiro-Wilk test* is essentially based the Pearson correlation between the residuals and the normal quantile, called W, which is equal to 0.914 and 0.959 for the untransformed and sqrt-transformed models respectively.

Box-Cox Power Transform

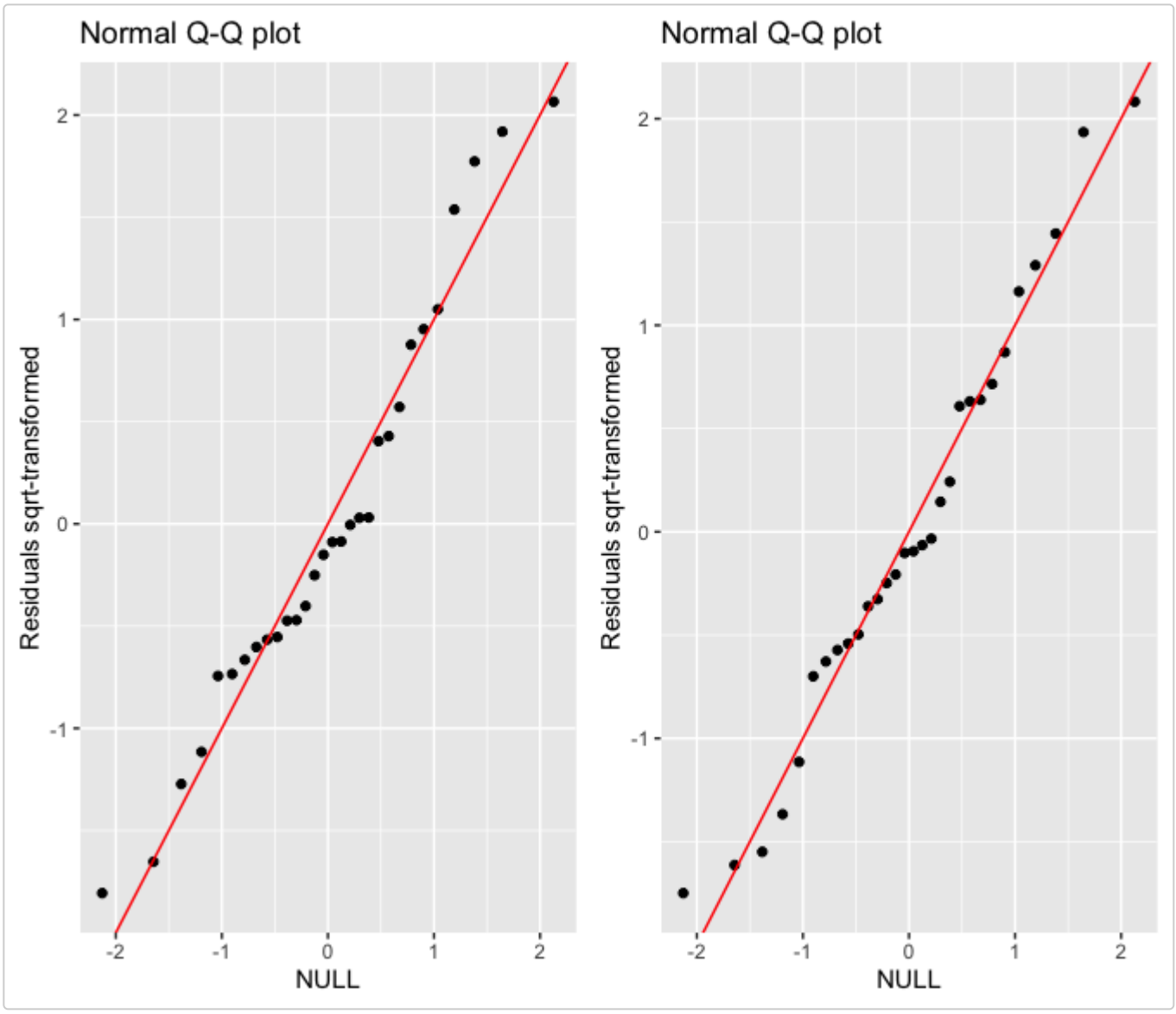
- The sqrt-transform is a member of the Box-Cox power tranforms. The transform is $y \leftarrow \frac{y^\lambda - 1}{\lambda}$ with limit $log(y)$ as $\lambda \rightarrow 0$
- We can find the power that minimizes the distance between the residuls and Q-Q line in *Q-Q plot* using an R function from the car package.

```
library(car)
(lambda = powerTransform(gg))
```

```
## Estimated transformation parameter
##        Y1
## 0.3153982
```

- Here, we use the Box-Cox power transform on the gala dataset

```
lam = lambda$lambda
glam = lm(Species^lam ~ Area + Elevation + Scruz + Nearest + Adjacent, gala)
modlam = fortify(glam)

p1 = qplot(sample = scale(.resid), data = modgs) + geom_abline(intercept = 0,
    slope = 1, color = "red") + labs(title = "Normal Q-Q plot", y = "Residuals sqrt-transformed")

p2 = qplot(sample = scale(.resid), data = modlam) + geom_abline(intercept = 0,
    slope = 1, color = "red") + labs(title = "Normal Q-Q plot", y = "Residuals sqrt-transformed")

grid.arrange(p1, p2, nrow = 1)
```



- The Shapiro-Wilk test concludes that the errors are normal for the Box-Cox Transform of Species with level of significance 10% since the p-value, 0.65 is greater than 0.10.

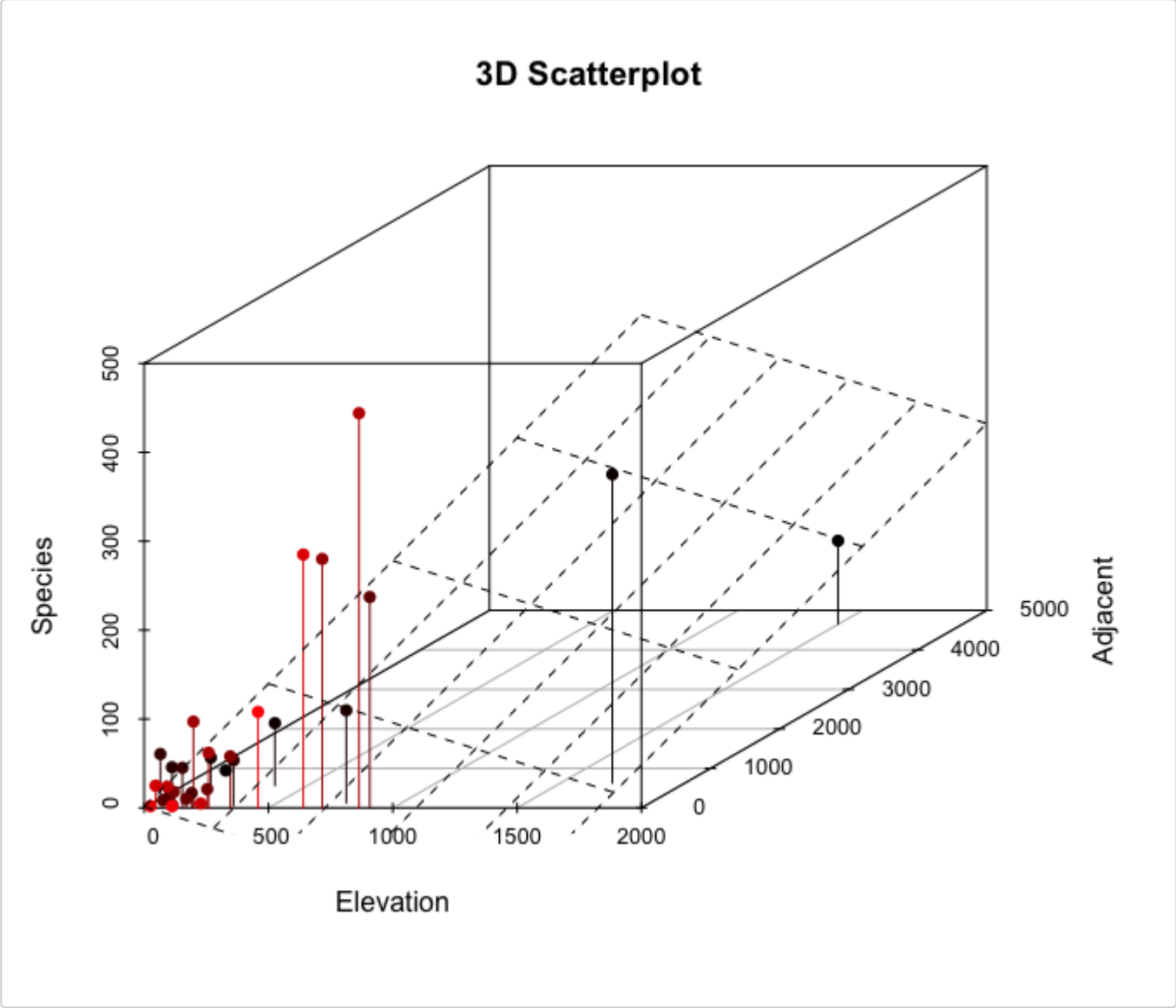
```
shapiro.test(residuals(glam))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(glam)
## W = 0.97378, p-value = 0.6469
```

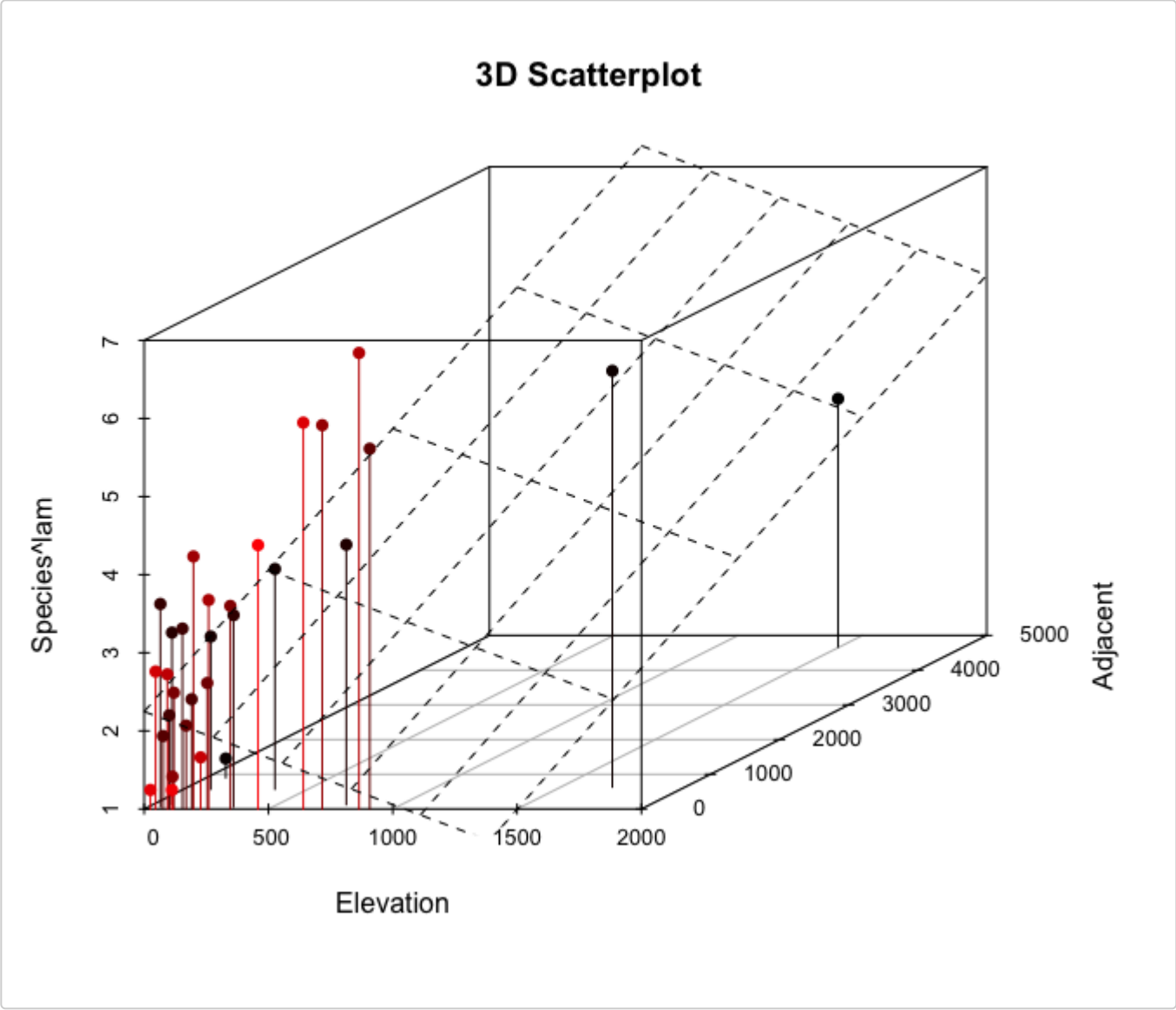
Checking for influential outliers

3D Scatterplot

```
library("scatterplot3d")
attach(gala)

s3d = scatterplot3d(Elevation, Adjacent, Species, pch = 16, highlight.3d = TRUE,
    type = "h", main = "3D Scatterplot")

fit = lm(Species ~ Elevation + Adjacent, gala)
s3d$plane3d(fit)
```




```
s3d = scatterplot3d(Elevation, Adjacent, Species^lam, pch = 16, highlight.3d = TRUE,
    type = "h", main = "3D Scatterplot")

fit = lm(Species^lam ~ Elevation + Adjacent, gala)
s3d$plane3d(fit)
```



- Clearly the Box-Cox transformed Species reveals the data better for small values of Elevation and Adjacent
- Notice the datapoint far away with the largest value of Adjacent, it looks like it is pulling the fitted plan toward it

```
detach(gala)
```

3D Spin plot

- Spinning the 3D Scatterplot is better for detecting outliers and influential points than the static scatterplots alone
- Pay attention to the data point with largest *Adjacent* value
- What is your conclusion?

```
# library(faraway) library(rgl) data(gala) attach(gala) plot3d(Elevation,
# Adjacent, Species^lam, col = 'red', size = 3) play3d(spin3d(axis =
```

```
# c(0,0,1))) detach(gala)
```

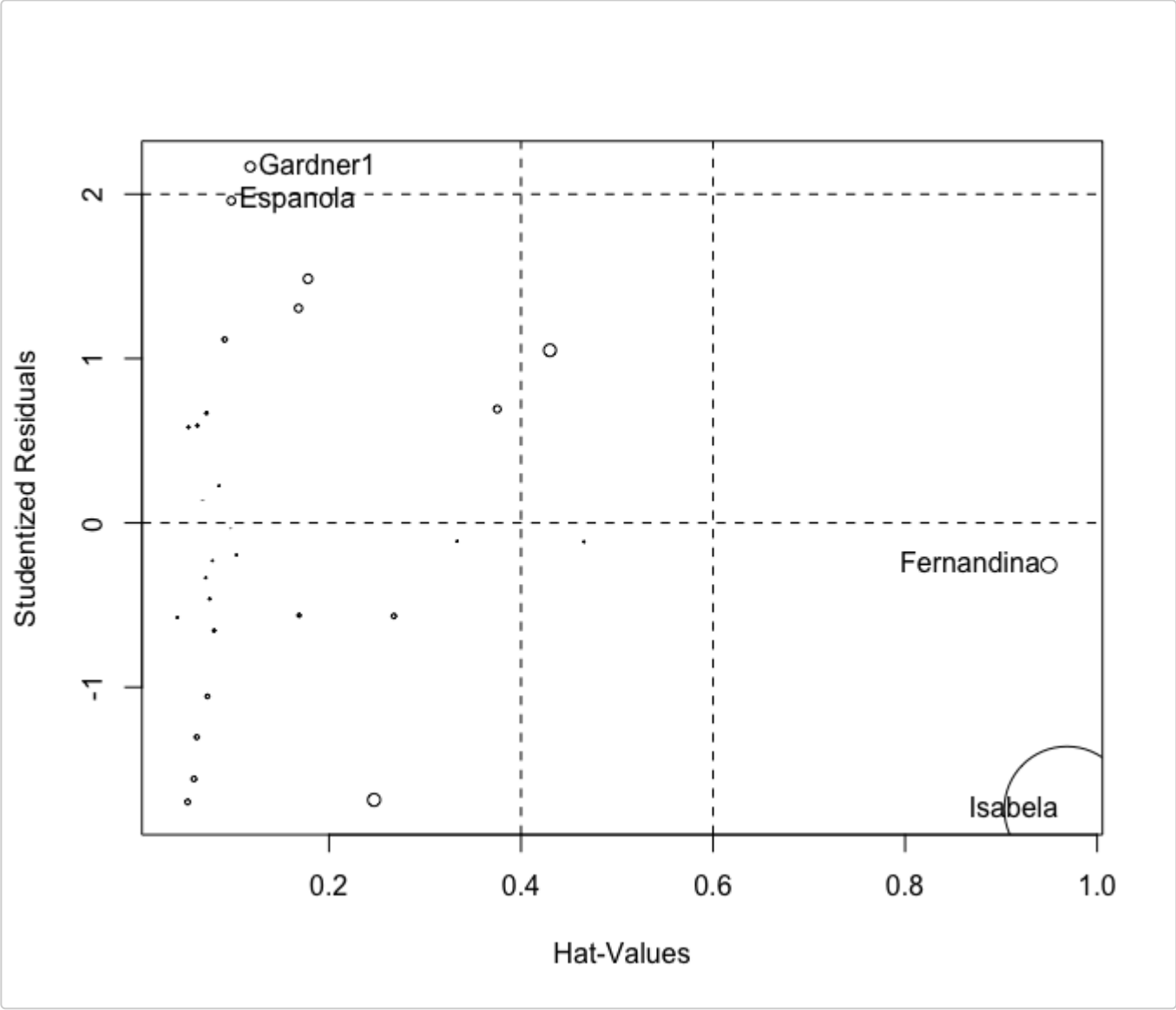
The leverage measure for detecting influential outliers

- The data visualization methods that we have seen so far are inefficient for production methods. Therefore, some statistics need to be employed that allow automatic detection of data anamolies
- Suppose X is the decision matrix
- The leverage measure, denoted by h_{ii} for the i^{th} observation comes from the i^{th} diagonal of the matrix, H

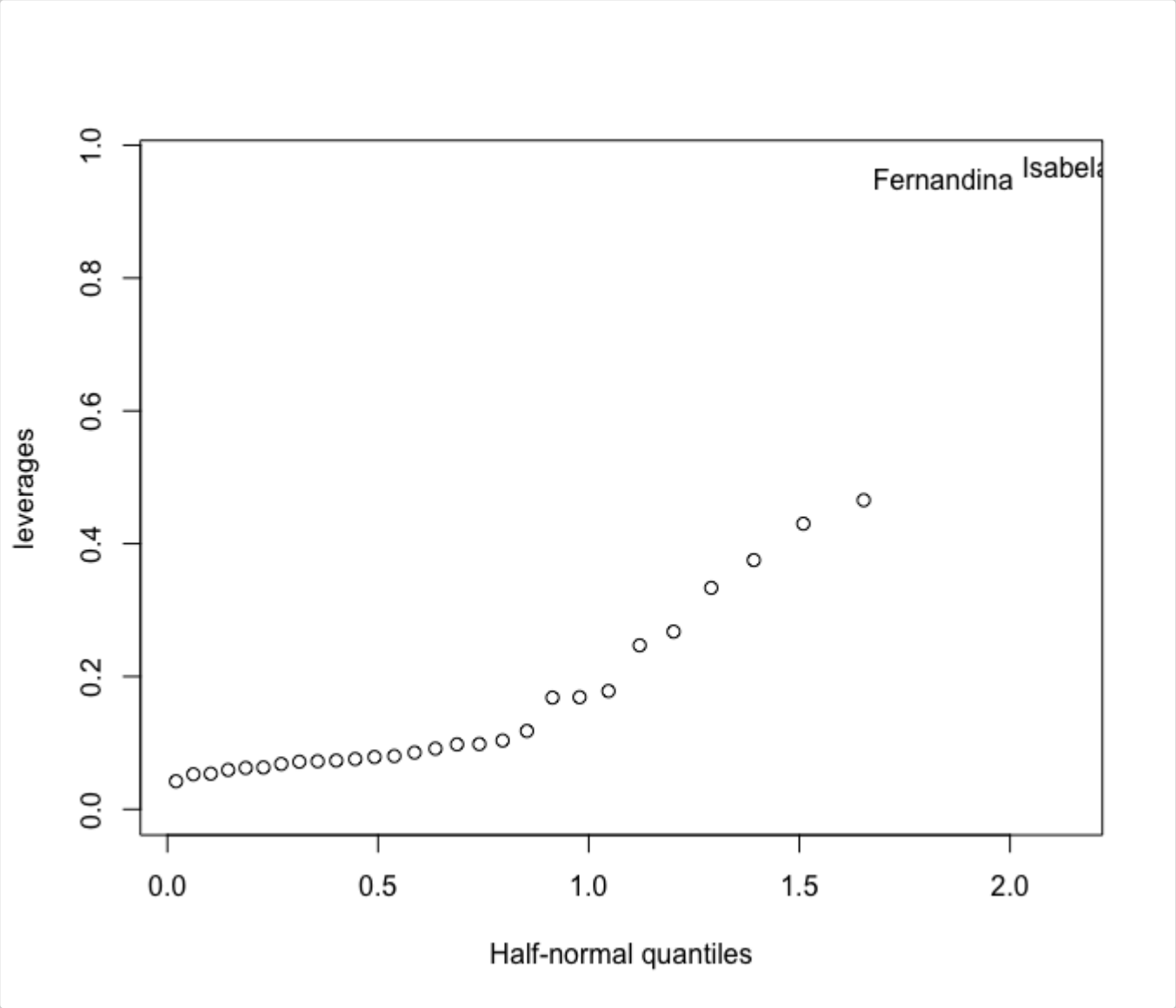
$$H = X(X^T X)^{-1} X^T$$

- We have several graphical displays using leverage and the *gala* dataset as example

```
influencePlot(glam)
```



```
islands = row.names(gala)
halfnorm(lm.influence(glam)$hat, labs = islands, ylab = "leverages")
```

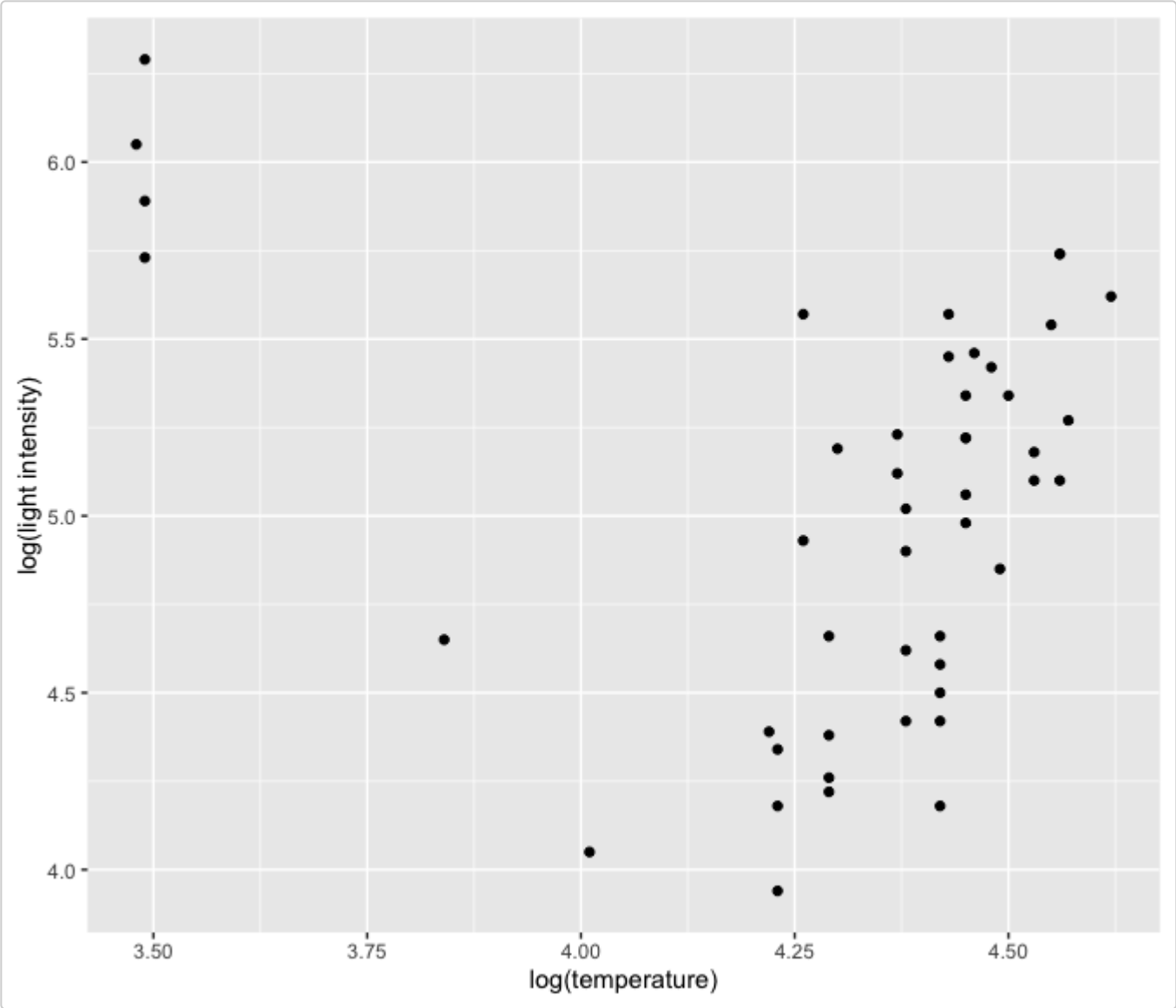


	StudRes	Hat	CookD
Espanola	1.9605832	0.0982458	0.0624038
Fernandina	-0.2559164	0.9496732	0.2143231
Gardner1	2.1667406	0.1179569	0.0906796
Isabela	-1.7407291	0.9685321	14.3315039

Using dataset star

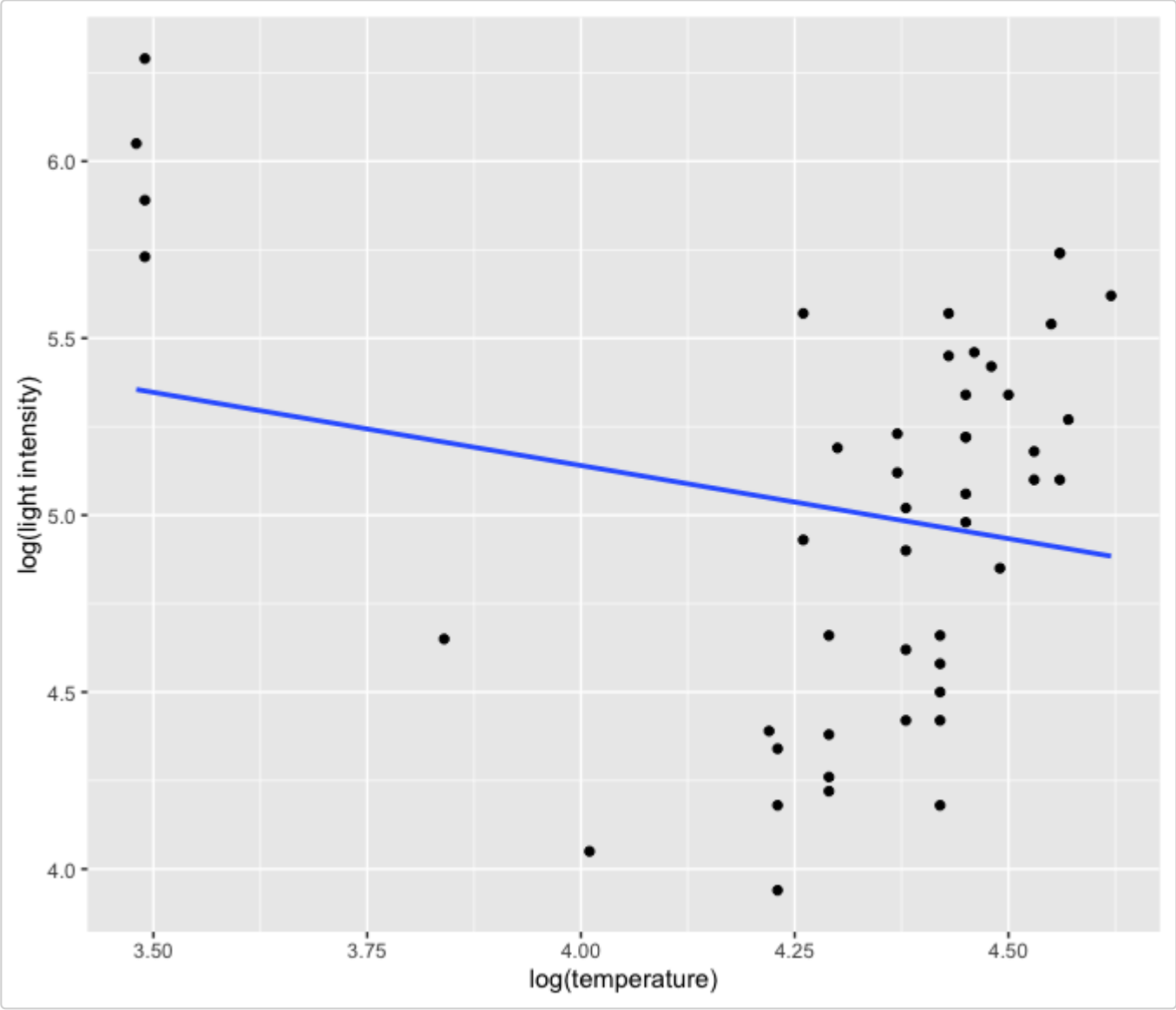
- To see how high leverage observation can distort OLS, we use a new dataset *star* in the *faraway* package
- The light and temperature of stars from a cluster in the direction of the Cygnus constellation
- These data have significant outliers visible with just a scatterplot

```
data(star)
p = qplot(star$temp, star$light, xlab = "log(temperature)", ylab = "log(light intensity)")
p
```



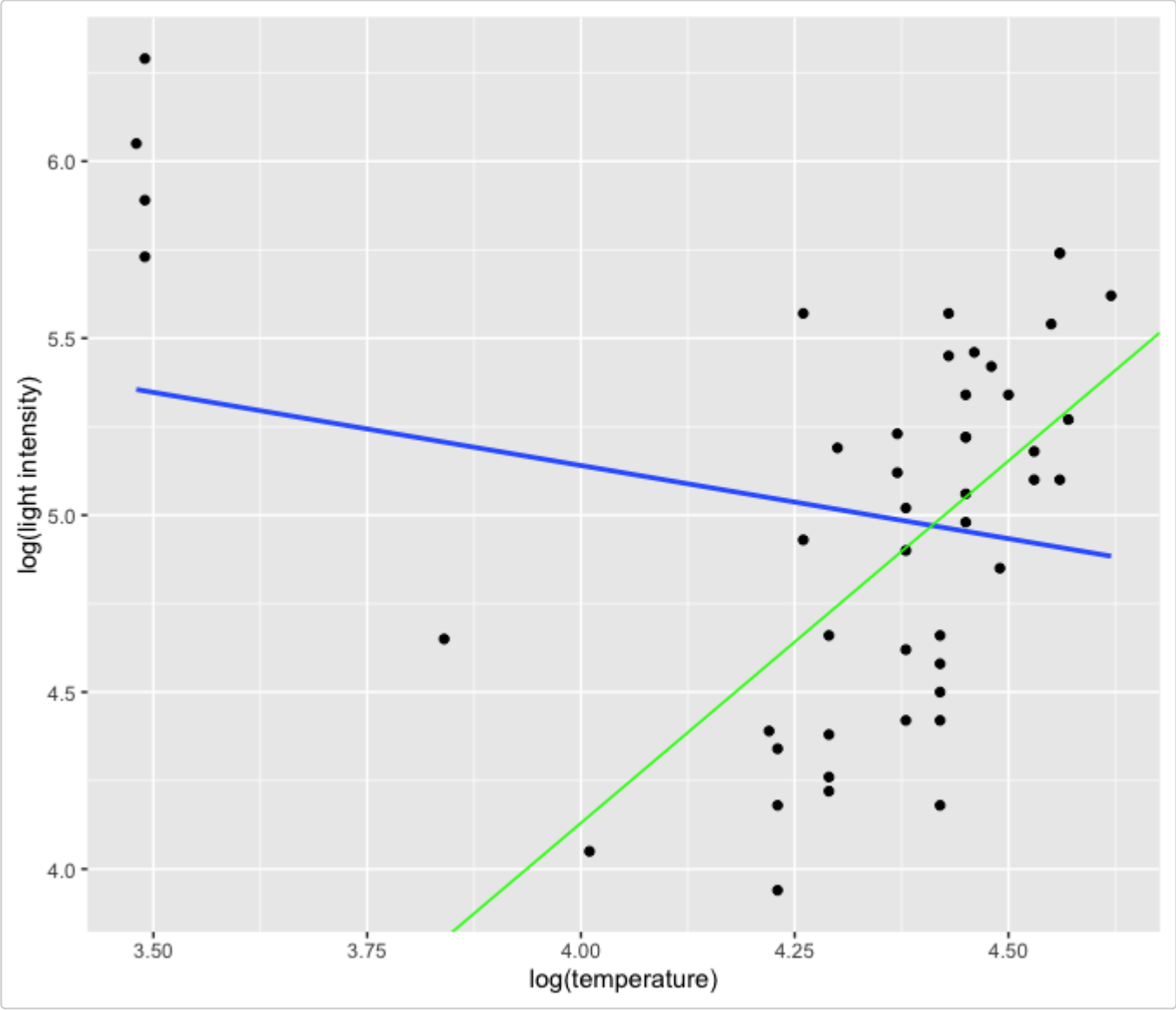
- The LSfitted line with outliers included in the data

```
ga = lm(light ~ temp, star)
p = p + geom_smooth(method = "lm", se = F)
p
```



- The LS fitted line with outliers excluded from the data
- Clearly, the outliers pull the OLS line toward them

```
gb = lm(light ~ temp, star, subset = (temp > 3.6))
p + geom_abline(intercept = coef(gb)[1], slope = coef(gb)[2], color = "green")
```



Cook's Distance for detecting influential outliers

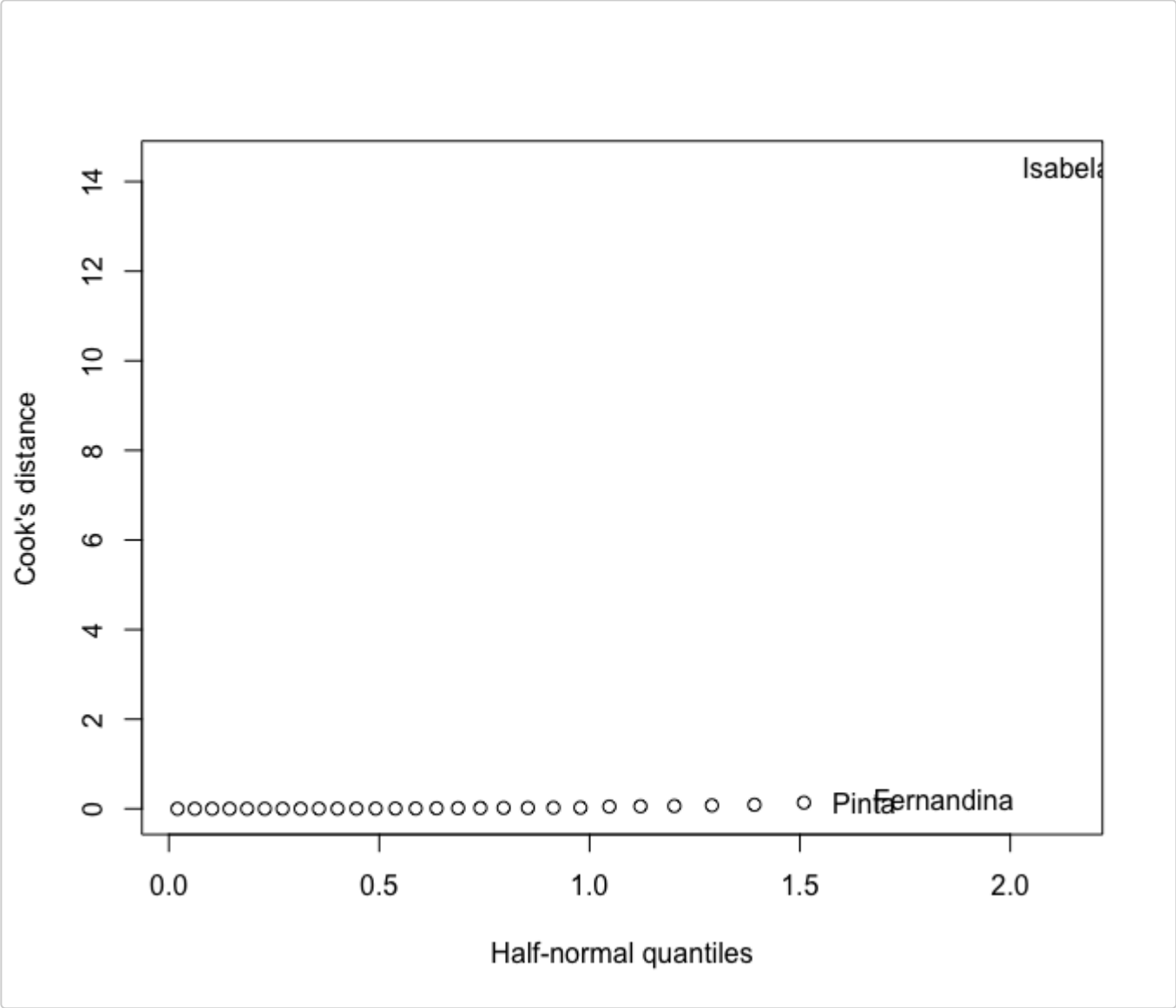
- Cook's distance measures how much each data point changes the fitted value if it were to be excluded from the dataset. It is defined as -

$$D_i = \frac{\Sigma_{j=1}^{n}(\hat{y}_j - \hat{y}_{j(i)})^2}{pMSE}$$

where, $\hat{y}_{j(i)}$ is the fitted value after removing data point i

- It appears that calculations require refitting the model n times as the removed data point transits over the entire dataset. However, Cook's distance can be simply computed from the Hat matrix and the residuals.
- Using the savings data, we calculate the Cook's distance for each data point
- Half normal plot of Cook's distance with labels of three largest values

```
cook = cooks.distance(glam)
halfnorm(cook, 3, labs = islands, ylab = "Cook's distance")
```



- model fit excluding observation with largest Cook's distance

```
glam1 = lm(Species^lam ~ Area + Elevation + Scruz + Nearest + Adjacent, data = gala,
    subset = (cook < max(cook)))
```

- Comparison of model fitted with and without the worst influential observation

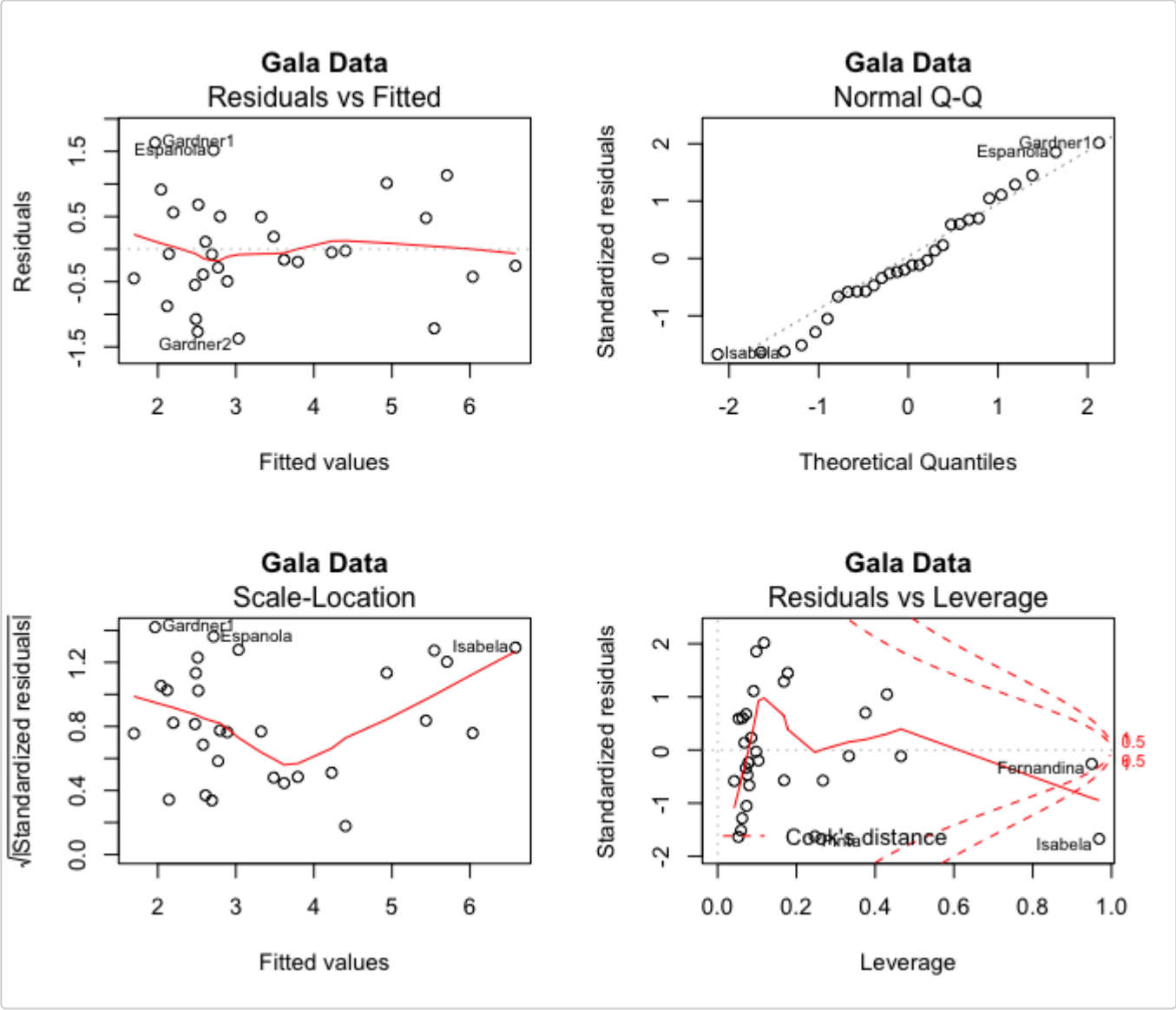
```
compareCoefs(glam, glam1)
```

```
## Calls:
## 1: lm(formula = Species^lam ~ Area + Elevation + Scruz + Nearest +
##   Adjacent, data = gala)
## 2: lm(formula = Species^lam ~ Area + Elevation + Scruz + Nearest +
##   Adjacent, data = gala, subset = (cook < max(cook)))
##
##               Model 1   Model 2
## (Intercept)     2.150     2.250
## SE              0.271     0.267
##
```

```
## Area        -0.000654  0.001422
## SE           0.000317  0.001231
##
## Elevation    0.004793  0.003630
## SE           0.000759  0.000989
##
## Scruz        -0.00405  -0.00308
## SE            0.00305   0.00298
##
## Nearest       0.01075   0.00903
## SE            0.01492   0.01436
##
## Adjacent    -0.000927 -0.000863
## SE           0.000250  0.000243
##
```

The omnibus diagnostic plot function

```
oldpar = par(mfrow = c(2, 2))
plot(glam, main = "Gala Data")
```

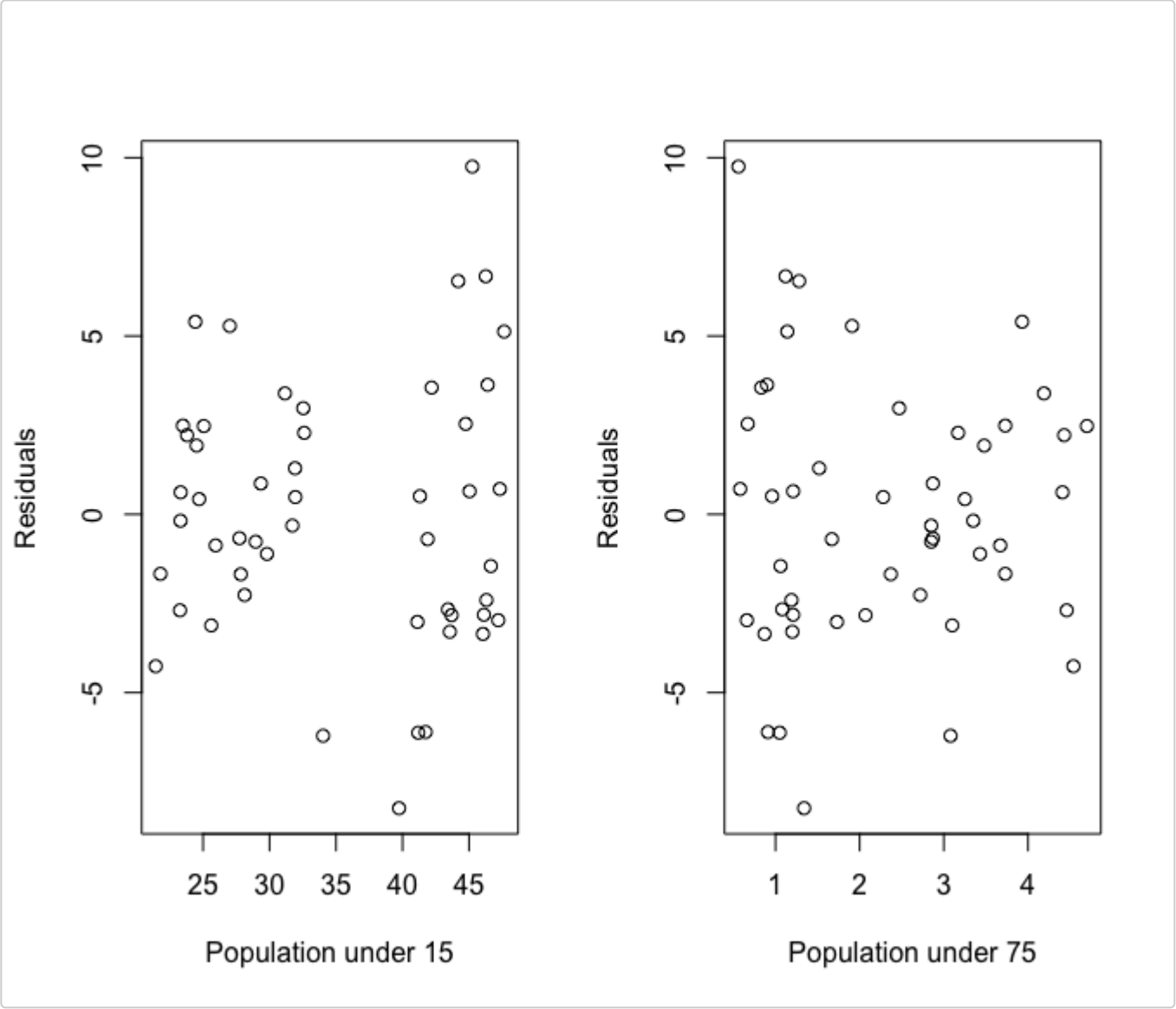


Checking for correct model specification

Checking if non-constant variance is related to a predictor

- Plots of residuals versus predictors are called linear residual plots

```
par(mfrow = c(1, 2))
plot(savings$pop15, residuals(g), xlab = "Population under 15", ylab = "Residuals")
plot(savings$pop75, residuals(g), xlab = "Population under 75", ylab = "Residuals")
```

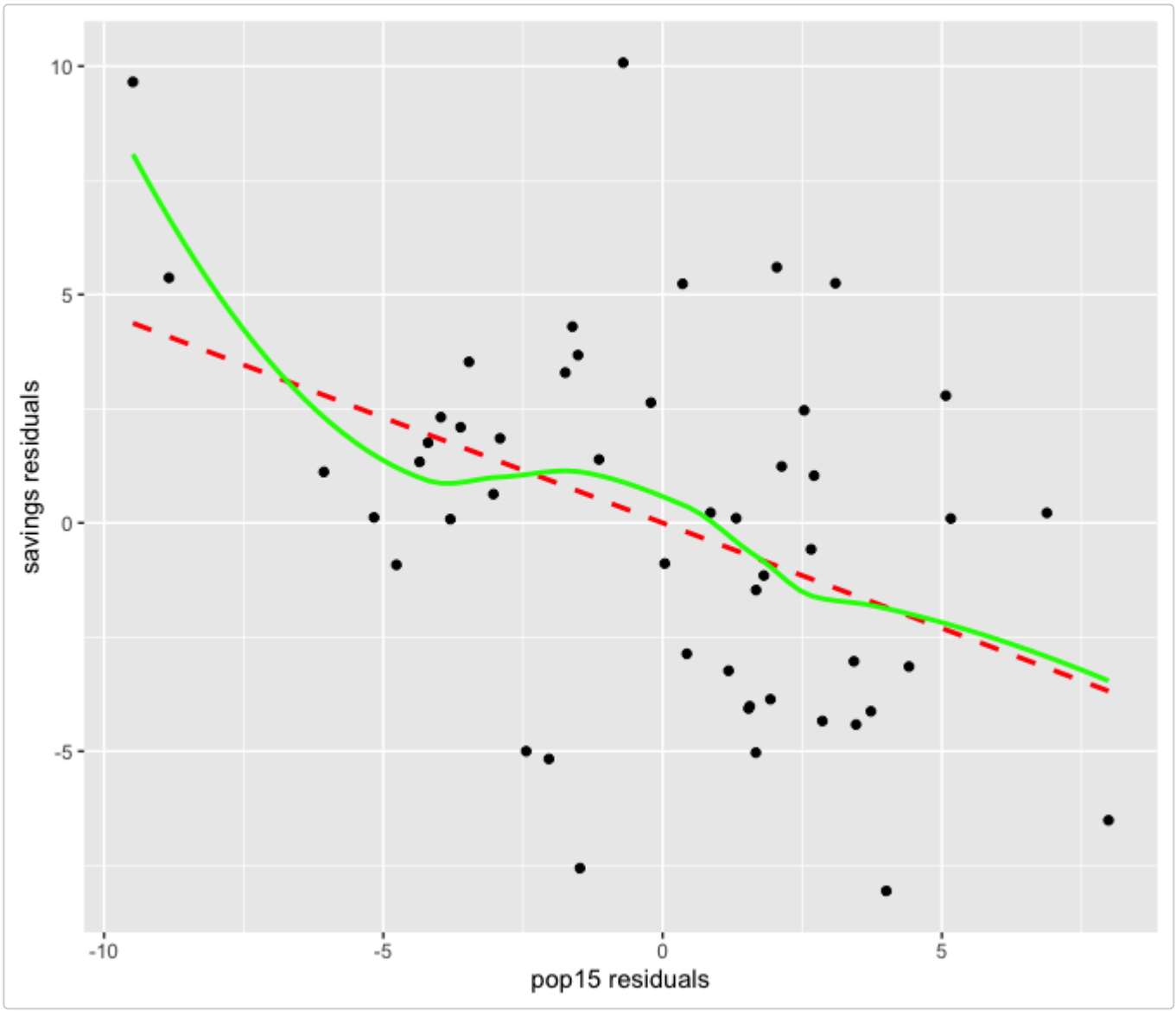


- We clearly see two clusters of data in “Population under 15” plot. We shall see how we can fit this data using an interaction model
- Dennis Cook showed examples where linear residual plots fail to detect heteroskedasticity and nonlinearity
- We will use Dennis Cook’s CERES plots

Added variable plot for checking model structure

- Model structure concerns the type of model or the transformation of predictors. We have discussed the transformation of predictors. Other types of models could include ploynomial models involving continuous predictors.
- For example, in the *savings* data, we can check if *sr* and *pop15* are related by some other relation than a straight line.
- To do this we use the special residuals from these fits:
- Fit *sr* on all variables except *pop15*
- Fit *pop15* on all other predictors
- Then we examine the relation between these two sets of residuals

```
d = residuals(lm(sr ~ pop75 + dpi + ddpi, savings))
m = residuals(lm(pop15 ~ pop75 + dpi + ddpi, savings))

qplot(m, d, xlab = "pop15 residuals", ylab = "savings residuals") + geom_smooth(method = "lm",
    color = "red", linetype = 2, se = F) + geom_smooth(method = "loess", color = "green",
    se = F)
```



- The "green" curve is a model-free (nonparametric) fit
- Since it is close to the straight line, it indicates that *pop15* does not need to be transformed or that its square does not need to be added to the model
- The slope of the straight line is the same as the coefficient in the full model

```
compareCoefs(lm(d ~ m), g)
```

```
## Calls:
## 1: lm(formula = d ~ m)
## 2: lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings)
##
```

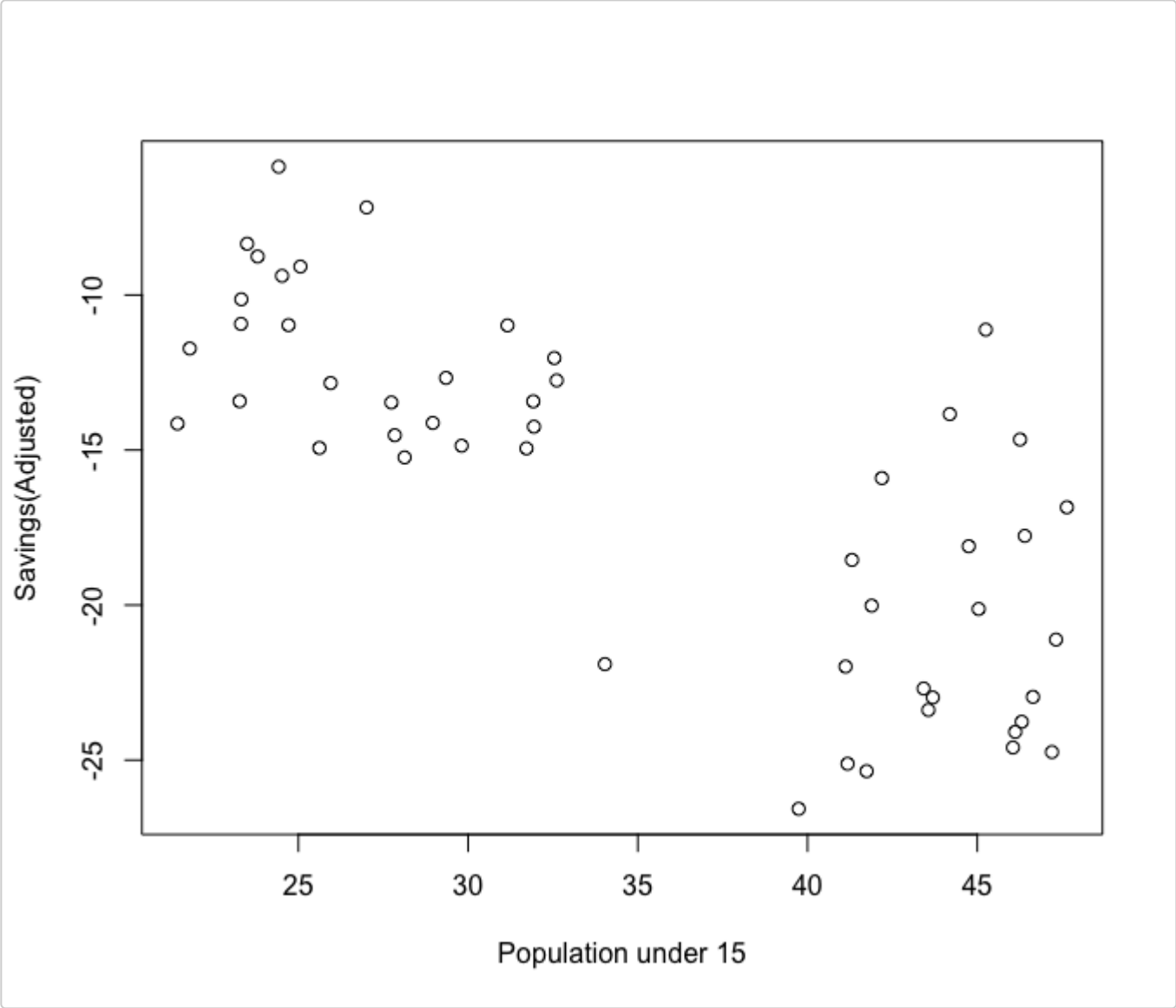
```
##              Model 1   Model 2
## (Intercept) 9.91e-17  2.86e+01
## SE          5.21e-01  7.35e+00
##
## m             -0.461
## SE             0.140
##
## pop15                   -0.461
## SE                       0.145
##
## pop75                    -1.69
## SE                        1.08
##
## dpi                  -0.000337
## SE                    0.000931
##
## ddpi                     0.410
## SE                       0.196
##
```

- LOESS and LOWESS (locally weighted scatter plot) are two strongly related non-parametric regression methods that combine multiple regression models in a k-nearest neighbor based meta-model

Partial residual plot for checking model structure

- The *partial-residual plot* is prefered to the added variable plot
- It uses *adjusted* response
- The partial-residual plot is sometimes called *component + residual* plot

```
plot(savings$pop15, coef(g)["pop15"] * savings$pop15 + residuals(g), xlab = "Population under 15",
    ylab = "Savings(Adjusted)")
```

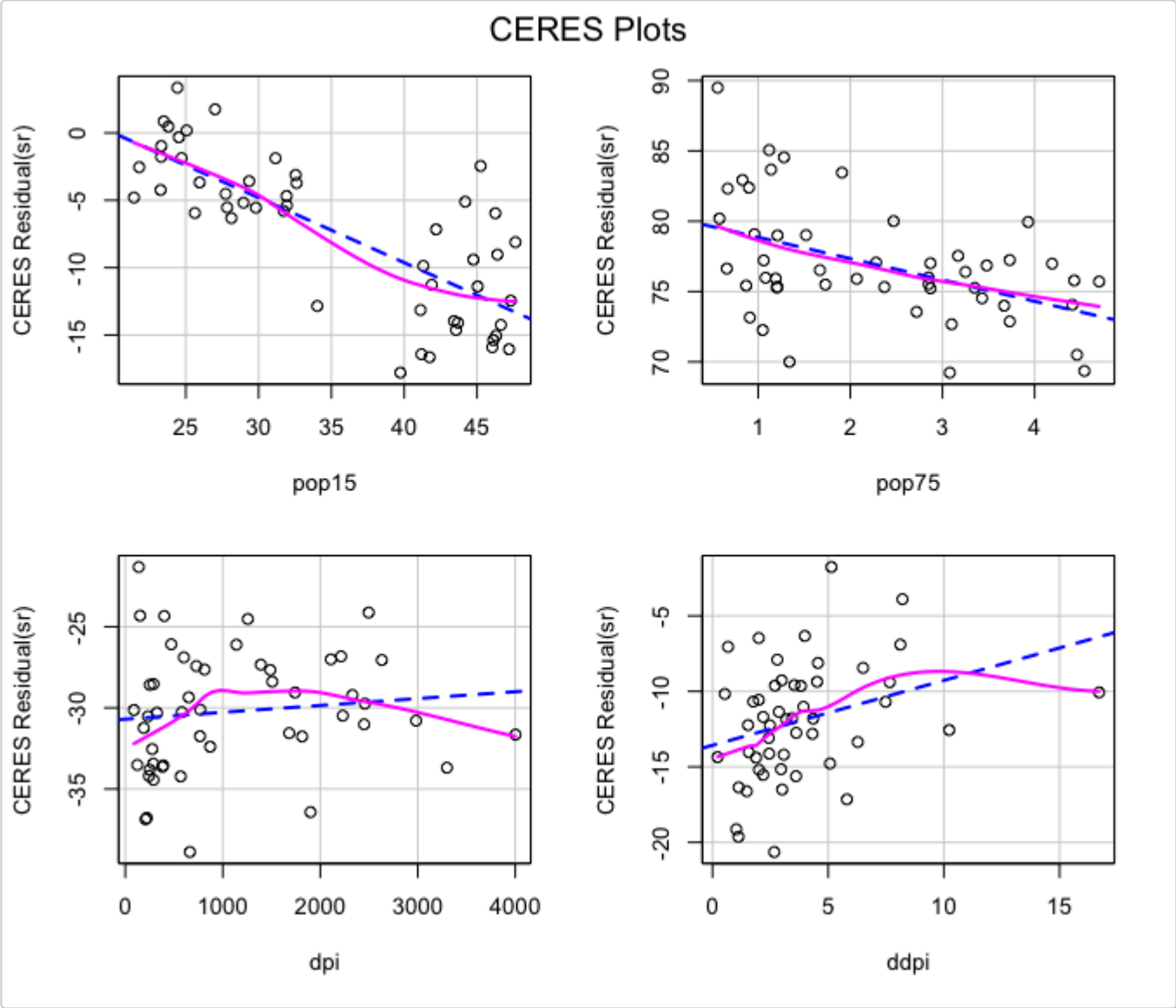


- Two easy ways to get the added variable plot

1. The partial-residual plot
2. CERES plot (Combined Conditional Expectatins and REsiduals plot)

- CERES plots are a generalization of component + residual (partial residual) plots that are less prone to leakage of nonlinearity among the predictors

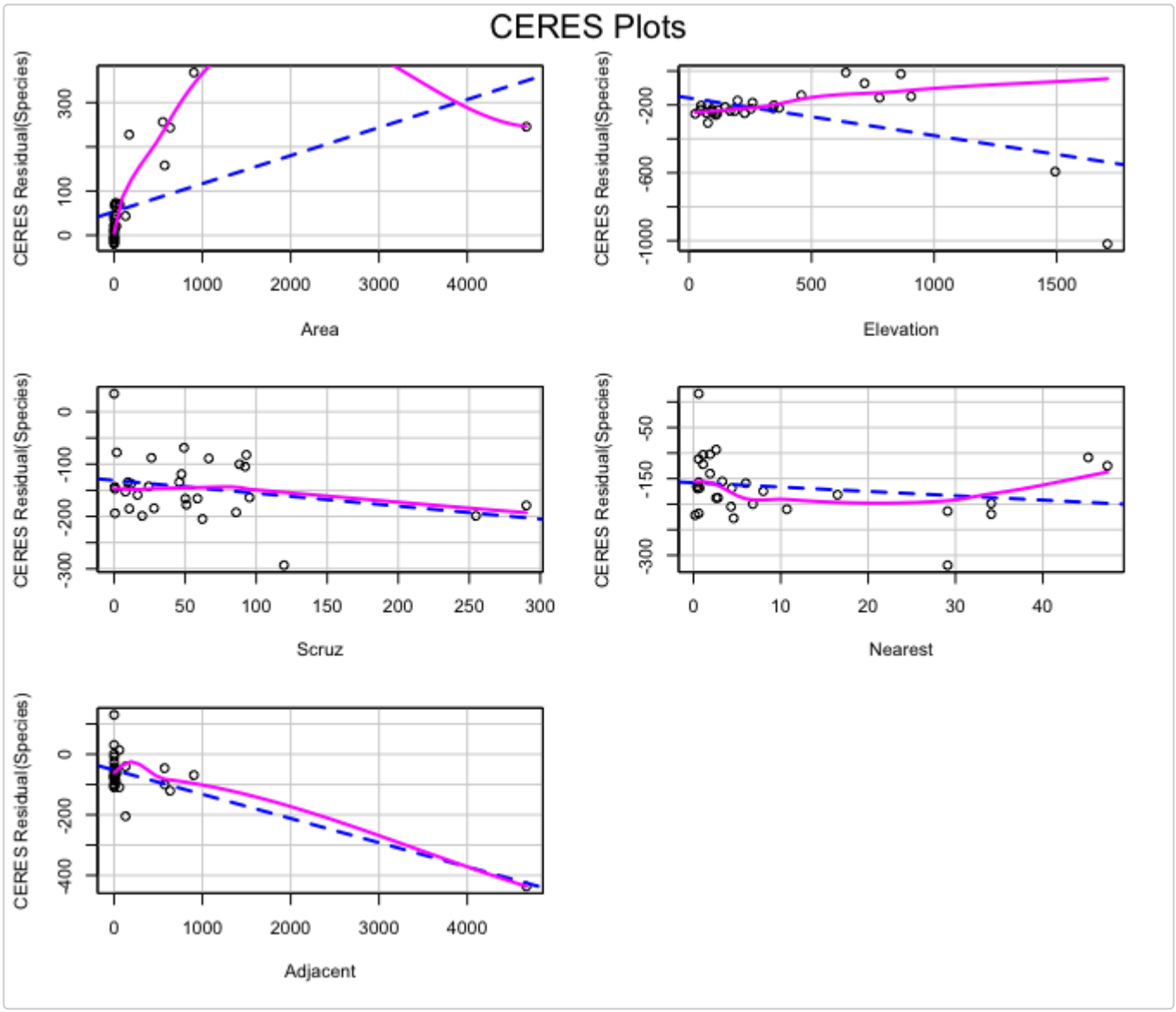
```
library(car)
ceresPlots(g, terms = ~.)
```



- Once again two clusters are evident in the savings data
- No evidence for transforms or higher order terms

Let's try CERES plot on the gala data

```
summary(gg)
ceresPlots(gg, terms = ~.)
```




```
##
## Call:
## lm(formula = Species ~ Area + Elevation + Scruz + Nearest + Adjacent,
##     data = gala)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -111.679  -34.898   -7.862   33.460  182.584
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  7.068221  19.154198   0.369 0.715351
## Area        -0.023938   0.022422  -1.068 0.296318
## Elevation    0.319465   0.053663   5.953 3.82e-06 ***
## Scruz       -0.240524   0.215402  -1.117 0.275208
## Nearest      0.009144   1.054136   0.009 0.993151
## Adjacent    -0.074805   0.017700  -4.226 0.000297 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 60.98 on 24 degrees of freedom
## Multiple R-squared:  0.7658, Adjusted R-squared:  0.7171
## F-statistic:  15.7 on 5 and 24 DF,  p-value: 6.838e-07
```

- Lets try adding *Elevation^2* to the model
- Since we have an influential observation, we shall fit with and without the observation. This is important for quadratic and other polynomial models

```
gg2 = lm(Species ~ Area + Elevation + I(Elevation^2) + Scruz + Nearest + Adjacent,
    data = gala)
cook = cooks.distance(gg2)
gg2c = lm(Species ~ Area + Elevation + I(Elevation^2) + Scruz + Nearest + Adjacent,
    data = gala, subset = (cook < max(cook)))
compareCoefs(gg2, gg2c)
```

```
## Calls:
## 1: lm(formula = Species ~ Area + Elevation + I(Elevation^2) + Scruz +
##   Nearest + Adjacent, data = gala)
## 2: lm(formula = Species ~ Area + Elevation + I(Elevation^2) + Scruz +
##   Nearest + Adjacent, data = gala, subset = (cook < max(cook)))
##
##                   Model 1   Model 2
## (Intercept)         -32.3      -4.8
## SE                   26.0      18.6
##
## Area               0.1012    0.3575
## SE                 0.0635    0.0659
##
## Elevation           0.582     0.328
## SE                  0.135     0.105
##
## I(Elevation^2) -0.000359 -0.000240
## SE              0.000172  0.000120
##
## Scruz             -0.1308   -0.0271
## SE                 0.2085    0.1444
##
## Nearest            -0.509    -0.583
## SE                  1.018     0.698
##
## Adjacent       -0.000918 -0.016322
## SE              0.039057  0.026960
##
```

Interaction Models

Dealing with clusters

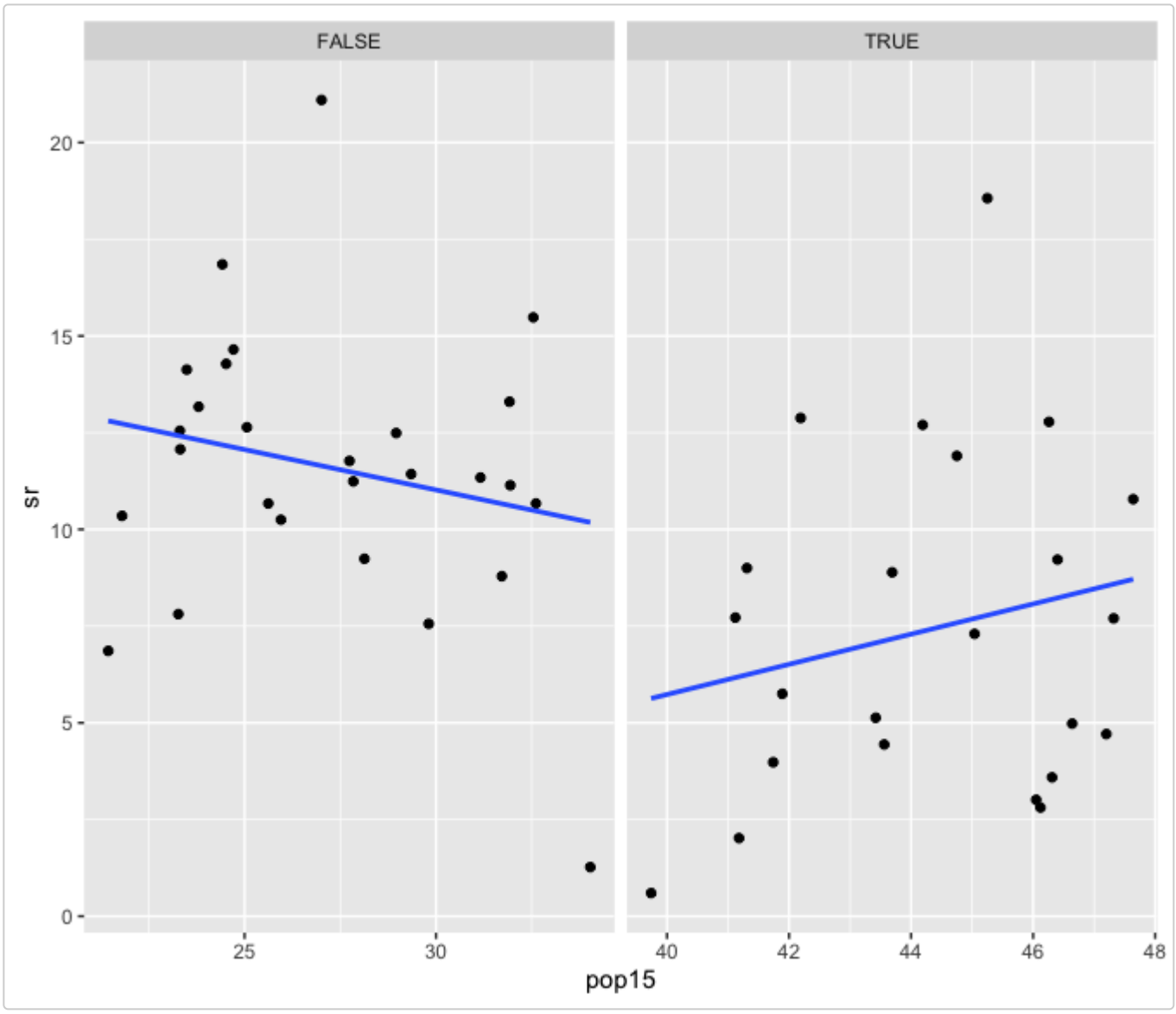
- We observed that the graphical tools revealed a cluster structure in the savings data, therefore, we introduce interaction model to deal with such cluster
- Lets consider the savings data

```
g1 = lm(sr ~ pop15 + pop75 + dpi + ddpi, savings, subset = (pop15 > 35))
g2 = lm(sr ~ pop15 + pop75 + dpi + ddpi, savings, subset = (pop15 < 35))
compareCoefs(g1, g2)
```

```
## Calls:
## 1: lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings,
##   subset = (pop15 > 35))
## 2: lm(formula = sr ~ pop15 + pop75 + dpi + ddpi, data = savings,
##   subset = (pop15 < 35))
##
##               Model 1   Model 2
## (Intercept)     -2.43     23.96
## SE              21.16      8.08
##
## pop15           0.274    -0.386
## SE              0.439     0.195
##
## pop75          -3.548    -1.328
## SE              3.033     0.926
##
## dpi          0.000421 -0.000459
## SE           0.005000  0.000724
##
## ddpi            0.395     0.884
## SE              0.290     0.295
##
```

- Clearly, we see the slope of the coefficient for *pop15* are of opposite sign between the two groups

```
savings$group = (savings$pop15 > 35)
p = qplot(pop15, sr, data = savings)
p = p + facet_grid(. ~ group, scales = "free")
p + geom_smooth(method = "lm", se = F)
```



- We fit models on two separate groups in the above example, fitting models for the separate group is not a good idea when the dataset sizes are small
- The proper way is to create a new dummy variable which is a factor identifying the group
- The advantage over separate group fitting is that the error variance is better estimated with the full dataset

An example of interaction model

An interaction model includes the main effects and the product of at least two predictors, one of which is a factor:

- Suppose we have a two predictor model, with predictors x (continuous) and t (factor with levels A, B, C, say)
- We create more variables that are the products $(x \times t_A)$, $(x \times t_B)$, and $(x \times t_C)$, where t_A, t_B, t_C are the indicator(0/1) variables for each level of t
- The interaction model would be -
  $y = \beta_0 + \beta_1 x + \beta_2 t_B + \beta_3 (x \times t_B) + \beta_4 t_C + \beta_5 (x \times t_C) + \epsilon$

- The interaction model creates a seperate regression line of $y \sim x$ for each group of data labeled by the levels of t
- Note that factor level A is the base group with intercept and slope β_0 and β_1, respectively

Interaction model for savings data

- The model without any interactions is called a main effects model

$$sr = \beta_0 + \beta_{pop15} pop15 + \beta_{pop75} pop75 + \beta_{dpi} dpi + \beta_{ddpi} ddpi$$

- We will fit this model and use it later for comparing it to a larger model

```
g_main_effects = lm(sr ~ pop15 + pop75 + dpi + ddpi, savings)
```

- The larger model is the interaction model
- Interaction models are useful for fitting separate models to subgroups of data
- We will create a factor for the two groups: *(pop15 > 35)*, *(pop15 < 35)*, we call it *lt35* and add it to the *savings* dataframe

```
library(faraway)
data(savings)
savings$lt35 = factor(savings$pop15 < 35)
levels(savings$lt35) = c(0, 1)
```

- The interaction model adds a term with two coefficients ($\beta_{lt35} lt35 + \beta_{lt35.pop15} lt35.pop15$) to the main effects model

$$sr = \beta_0 + \beta_{pop15} pop15 + \beta_{lt35} lt35 + \beta_{lt35.pop15} lt35.pop15 + \beta_{pop75} pop75 + \beta_{dpi} dpi + \beta_{ddpi} ddpi$$

```
g_interaction = lm(sr ~ lt35 * pop15 + pop75 + dpi + ddpi, savings)
```

- Does the interaction model do a better job predicting the savings rate?
- We run an $anova()$ function comparing the main effects and the interaction models

```
anova(g_main_effects, g_interaction)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
45	650.7130	NA	NA	NA	NA
43	554.2503	2	96.46266	3.741896	0.0317544

- The answer is yes since the F-test rejects the main effects model with the p-value less than 0.05
- Note the difference in the residual degrees of freedom between the main effects and interaction model is 2, which corresponds to the coefficients β_{lt35} and $\beta_{lt35.pop15}$

Checking for collinearity in predictors

We will use seatpos dataset

```
library(faraway)
data(seatpos)
`?`(seatpos)
g = lm(hipcenter ~ ., seatpos)
summary(g)
```

seatpos R Documentation

Car seat position depending driver size

Description

Car drivers like to adjust the seat position for their own comfort. Car designers would find it helpful to know where different drivers will position the seat depending on their size and age. Researchers at the HuMoSim laboratory at the University of Michigan collected data on 38 drivers.

Usage

```
data(seatpos)
```

Format

The dataset contains the following variables

Age

Age in years

Weight

Weight in lbs

HtShoes

Height in shoes in cm

Ht

Height bare foot in cm

Seated

Seated height in cm

Arm

lower arm length in cm

Thigh

Thigh length in cm

Leg

Lower leg length in cm

hipcenter

horizontal distance of the midpoint of the hips from a fixed location in the car in mm

Source

"Linear Models in R" by Julian Faraway, CRC Press, 2004

```
##
## Call:
## lm(formula = hipcenter ~ ., data = seatpos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -73.827 -22.833  -3.678  25.017  62.337
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 436.43213  166.57162   2.620   0.0138 *
## Age           0.77572    0.57033   1.360   0.1843
## Weight        0.02631    0.33097   0.080   0.9372
## HtShoes      -2.69241    9.75304  -0.276   0.7845
## Ht            0.60134   10.12987   0.059   0.9531
## Seated        0.53375    3.76189   0.142   0.8882
## Arm          -1.32807    3.90020  -0.341   0.7359
## Thigh        -1.14312    2.66002  -0.430   0.6706
## Leg          -6.43905    4.71386  -1.366   0.1824
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 37.72 on 29 degrees of freedom
## Multiple R-squared:  0.6866, Adjusted R-squared:  0.6001
## F-statistic:  7.94 on 8 and 29 DF,  p-value: 1.306e-05
```

- The correlation matrix detects pairwise collinearity

```
round(cor(seatpos), 1)
```

	Age	Weight	HtShoes	Ht	Seated	Arm	Thigh	Leg	hipcenter
Age	1.0	0.1	-0.1	-0.1	-0.2	0.4	0.1	0.0	0.2
Weight	0.1	1.0	0.8	0.8	0.8	0.7	0.6	0.8	-0.6
HtShoes	-0.1	0.8	1.0	1.0	0.9	0.8	0.7	0.9	-0.8
Ht	-0.1	0.8	1.0	1.0	0.9	0.8	0.7	0.9	-0.8

	Age	Weight	HtShoes	Ht	Seated	Arm	Thigh	Leg	hipcenter
Seated	-0.2	0.8	0.9	0.9	1.0	0.6	0.6	0.8	-0.7
Arm	0.4	0.7	0.8	0.8	0.6	1.0	0.7	0.8	-0.6
Thigh	0.1	0.6	0.7	0.7	0.6	0.7	1.0	0.6	-0.6
Leg	0.0	0.8	0.9	0.9	0.8	0.8	0.6	1.0	-0.8
hipcenter	0.2	-0.6	-0.8	-0.8	-0.7	-0.6	-0.6	-0.8	1.0

The variance inflation factor (VIF)

```
library(car)
vif(g)
```

```
##        Age     Weight    HtShoes         Ht     Seated        Arm
##   1.997931   3.647030 307.429378 333.137832   8.951054   4.496368
##      Thigh        Leg
##   2.762886   6.694291
```

- look for vif > 10
- Ampute some predictors from the model

```
g1 = lm(hipcenter ~ Age + Weight + Ht, seatpos)
summary(g1)
anova(g1, g)
```

```
##
## Call:
## lm(formula = hipcenter ~ Age + Weight + Ht, data = seatpos)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -91.526 -23.005   2.164  24.950  53.982
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 528.297729 135.312947   3.904 0.000426 ***
## Age           0.519504   0.408039   1.273 0.211593
## Weight        0.004271   0.311720   0.014 0.989149
## Ht           -4.211905   0.999056  -4.216 0.000174 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.49 on 34 degrees of freedom
## Multiple R-squared:  0.6562, Adjusted R-squared:  0.6258
## F-statistic: 21.63 on 3 and 34 DF,  p-value: 5.125e-08
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
34	45262.04	NA	NA	NA	NA
29	41261.78	5	4000.252	0.562299	0.7279438

Conclusion

- The small model is not rejected with a significance level of 10% since the p-value, 0.73, is greater than 0.10
- The small model ($p = 4$) is more stable with an R^2 of 0.656 compared to R^2 of 0.687 from the fit of the big model ($p = 9$)

Exercises

uswages

Use the *uswages* data in the *faraway* package. Make sure you identify and eliminate the missing values

```
# load data
library(faraway)
data(uswages)

# manipulating the data getting rid of negative values for exper

uswages$exper[uswages$exper < 0] = NA


# convert race, smsa, and pt to factor variables
uswages$race = factor(uswages$race)
levels(uswages$race) = c("White", "Black")

uswages$smsa = factor(uswages$smsa)
levels(uswages$smsa) = c("No", "Yes")

uswages$pt = factor(uswages$pt)
levels(uswages$pt) = c("No", "Yes")


# create region, a factor variable based on the four regions ne, mw,so, we
uswages = data.frame(uswages, region = 1 * uswages$ne + 2 * uswages$mw + 3 *
    uswages$so + 4 * uswages$we)
uswages$region = factor(uswages$region)
levels(uswages$region) = c("ne", "mw", "so", "we")

# delete the four regions ne,mw, so, we
uswages = subset(uswages, select = -c(ne:we))

# Take care of NAs
uswages = na.omit(uswages)
```

1. Non-constant variance

a. Using the *uswage* data, fit the model(m):

$$wage \sim educ + exper + race + smsa + pt + region$$

b. Produce the *Residuals vs. Fitted plot*, and discuss if there are heteroskedasticity in the error variance.

c. Prodce the *Scale-location* plot, and discuss if there is any heteroskedasticity in the error variance

d. Perform the approximate test of non-constant error variance

2. Non-normal errors

a. Plot the Normal Q-Q plot and Histogram of the residuals from model(m). Do they indicate non-normal errors?

b. Perform the Shapiro-Wilk test of normality for the residuals of model(m). What is the p-value and what does it say about normality?

c. Find the optimal Box-Cox power transform and apply it to *wage*, refit model(m), replot Normal Q-Q Plot and perform the Shapiro-Wilk test of normality again. Did the Box-Cox Transform work?

3. Influential outliers

a. Produce the influence plot for model(m). Are there any really large Cook Distances?

b. Produce half-normal plot of the leverage values. Are there any high leverage data points?

c. Produce half-normal plot of the Cook's Distance. Are there any points which has large Cook's distance?

d. Fit model excluding observation with largest Cook's distance. Do the coefficients change? Are there any coefficients with notable changes?

e. Produce the omnibus diagnostic plot for the model(m). Which observation consistently stands out as as outlier-influential point in all four plots?

4. Model structure

a. Produce the CERES plots for model(m). Do the factor variables stop the plots from printing?

b. How many plots are there? Why these?

c. Do the plots indicate a plynomial model should be considered?

5. Interaction model

a. Fit an interaction model using the *region* and the two numeric variables. Is the model useful?

b. Test the interaction model versus model(m). What is the p-value and which model does it indicate?

Collinearity

a. Find the variance inflation factors for model(m)
b. Do they indicate collinearity in the predictors?