Diagnostics and Transformations – Part 3

Contents

1 Introduction **1**

2 Three Classes of Problem to Detect and Correct **1**
2.1 Introduction 1
2.2 Graphical Examination of Nonlinearity 2

3 Transformation to Linearity: Rules and Principles **9**

4 Evaluation of Outliers **14**
4.1 The Lessons of Anscombe's Quartet 14
4.2 Leverage 16

1 Introduction

Introduction

In this lecture, we continue our examination of techniques for examining and adjusting model fit via residual analysis. We look at some advanced tools and statistical tests for helping us to automate the process, then we examine some well known graphical and statistical procedures for identifying high-leverage and influential observations.

We will examine them here primarily in the context of bivariate regression, but many of the techniques and principles apply immediately to multiple regression as well.

2 Three Classes of Problem to Detect and Correct

2.1 Introduction

Three Problems to Detect and Correct

Putting matters into perspective, in our discussions so far, we have actually dealt with 3 distinctly different problems when fitting the linear regression model. All of them can arise at once, or we may encounter some combination of them.

Three Problems

- *Nonlinearity.* The fundamental nature of the relationship between the variables "as they arrive" is not linear.

- *Non-Constant Variance.* Residuals do not show a constant variance at various points on the conditional mean line.
- *Outliers.* Unusual observations may be exerting a high degree of influence on the regression function.

Residuals Patterns, Nonlinearity, and Non-Constant Variance

Weisberg discusses a number of common patterns shown in residual plots. These can be helpful in diagnosing nonlinearity and non-constant variance.

Residuals Patterns, Nonlinearity, and Non-Constant Variance

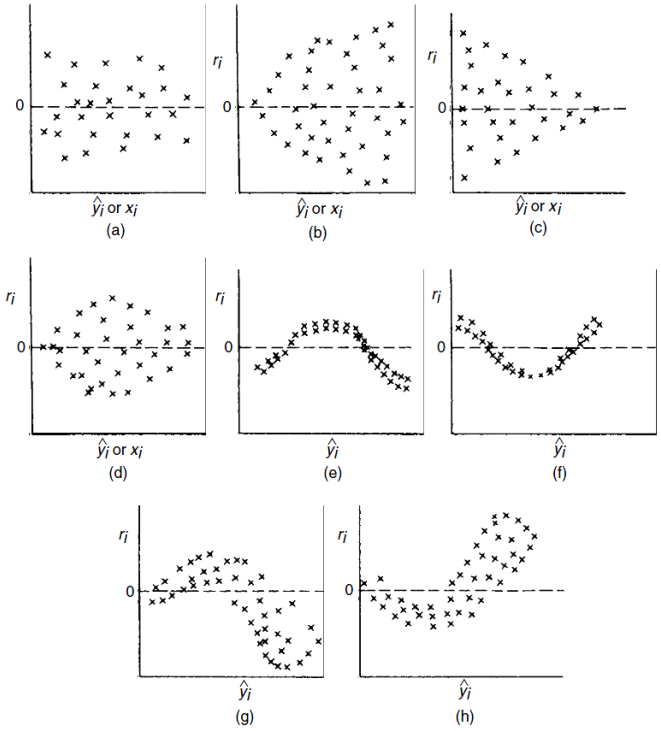


FIG. 8.2 Residual plots: (a) null plot; (b) right-opening megaphone; (c) left-opening megaphone; (d) double outward box; (e)–(f) nonlinearity; (g)–(h) combinations of nonlinearity and nonconstant variance function.

2.2 Graphical Examination of Nonlinearity

Graphical Examination of Nonlinearity

Often nonlinearity is obvious from the scatterplot.

However, as an aid to diagnosing the functional form underlying data, non-parametric smoothing is often useful as well.

The Loess Smoother

One of the best-known approaches to non-parametric regression is the loess smoother.

This works essentially by fitting a linear regression to a fraction of the points closest to a given x, doing that for many values of x. The smoother is obtained by joining the estimated values of $E(Y|X = x)$ for many values of x.

By fitting a straight line to the data, then adding the loess smoother, and looking for where the two diverge, we can often get a good visual indication of the nonlinearity in the data.

The Loess Smoother

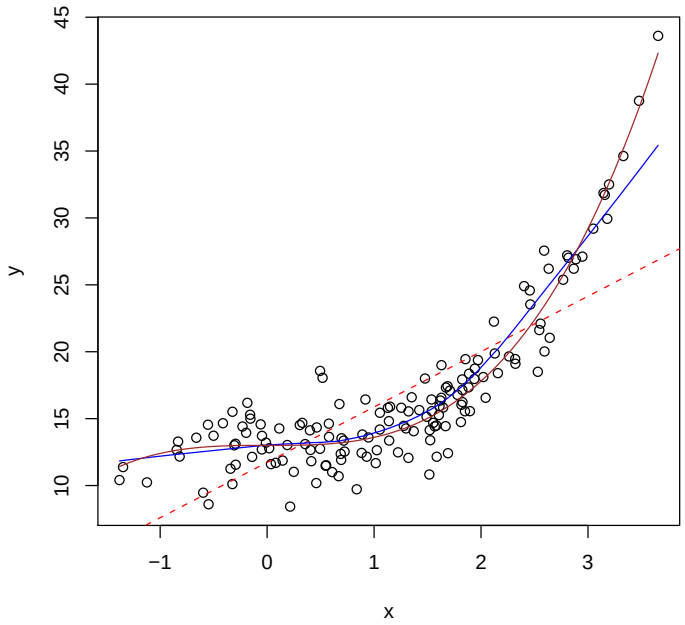
For example, in the last lecture, we created artificial data with a cubic component.

Let's recreate those data, then add

- add the linear fit line in dotted red
- the loess smooth line in blue
- the actual conditional mean function in brown

The Loess Smoother

```
> set.seed(12345)
> x <- rnorm(150,1,1)
> e <- rnorm(150,0,2)
> y <- .6 *x^3 + 13 + e
> fit.linear <- lm(y~x)
> plot(x,y)
> abline(fit.linear,lty=2,col='red')
> lines(lowess(y~x,f=6/10),col='blue')
> curve(.6 *x^3 + 13,col='brown',add=TRUE)
```



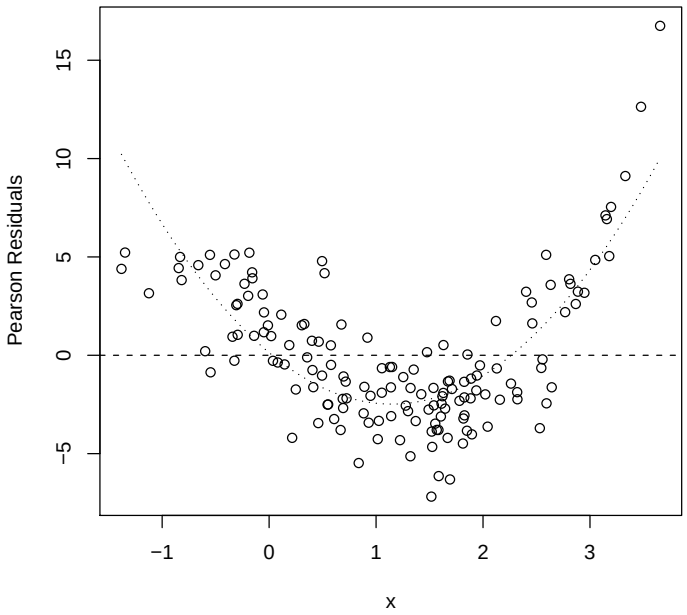
Automated Residual Plots

The function `residual.plots` automates the process of plotting residuals and computing significance tests for departure from linearity. It can produce a variety of plots, but in the case of bivariate regression, the key plots are the scatter plots of residuals vs. x, and residuals vs. fitted values. We'll just present the former here, but the latter becomes a vital tool in multiple regression.

The software also generates a statistical test of linearity, which is, of course, resoundingly rejected, and computes and plots a quadratic fit as an aid to visually detecting nonlinearity.

```
> residual.plots(fit.linear, fitted=FALSE)
```

```
  Test stat     Pr(>|t|)
x  15.71049 2.889014e-33
```



Weisberg discusses a statistical test of the null hypothesis of homogeneity of variance.

Departures from equality of variance will result in rejection of the null hypothesis.

Test of Constant Variance

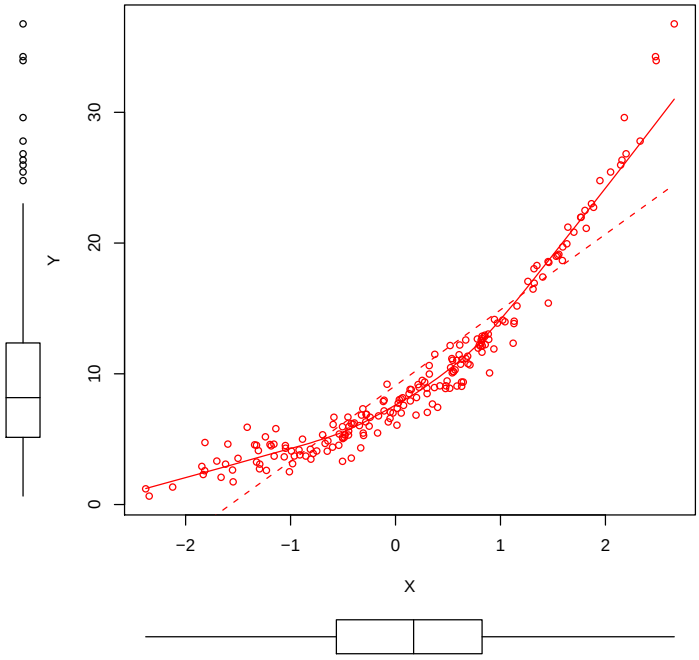
Below, we recreate some data from a previous lecture.

```
> set.seed(12345) ## seed the random generator
> X ← rnorm(200)
> epsilon ← rnorm(200)
> b1 ← .6
> b0 ← 2
> Y ← exp(b0 + b1 * X) + epsilon
```

Test of Constant Variance

If we have loaded the `car` library, we can create a useful plot of the data in one line with the `scatterplot` function. This gives you the data, the linear fit, the lowess fit, and boxplots on each margin.

```
> scatterplot(X,Y)
```

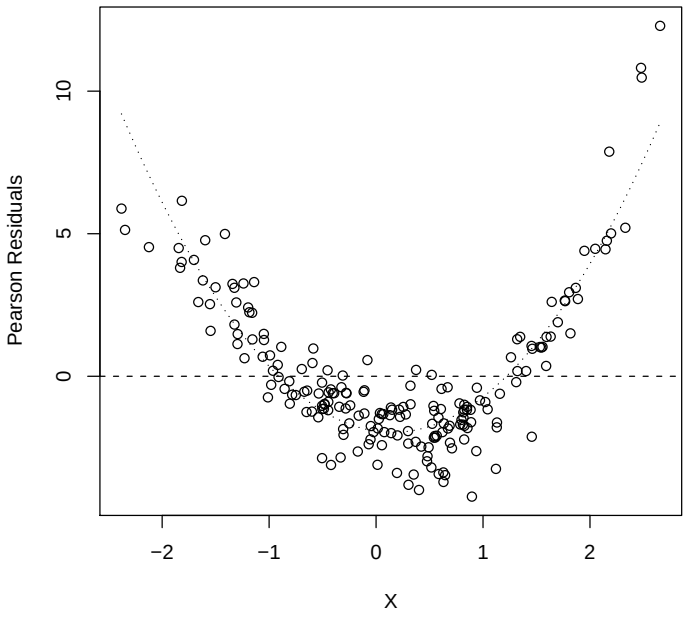


Test of Constant Variance

The nonlinearity is obvious in the residual plot:

```
> linear.fit ← lm(Y~X)
> residual.plots(linear.fit, fitted=F)
```

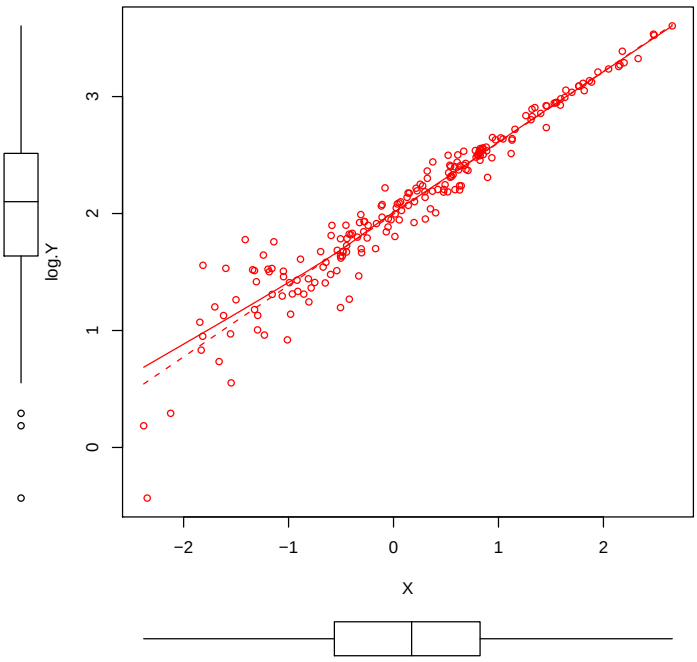
```
  Test stat     Pr(>|t|)
X  29.80535 6.282086e-75
```



Test of Constant Variance

As before, we transform Y to $\log(Y)$ and refit.

```
> log.Y ← log(Y)
> log.fit ← lm(log.Y ~ X)
> scatterplot(X, log.Y)
```

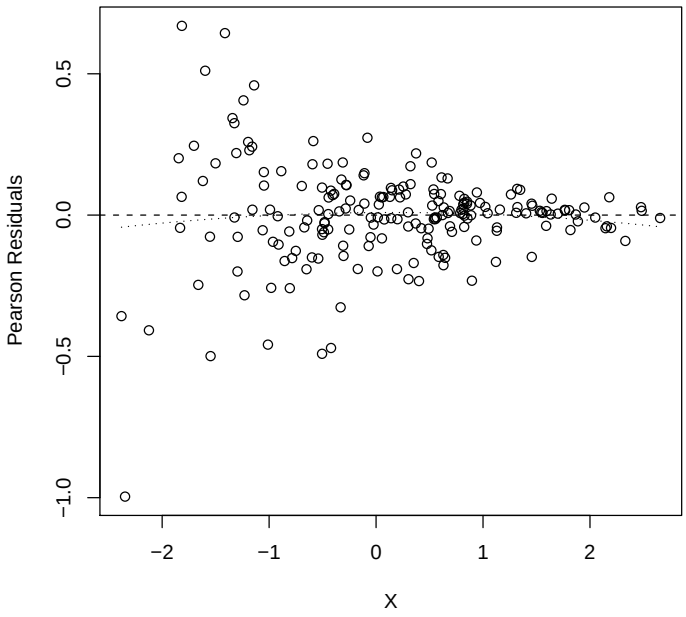


Test of Constant Variance

The residual plot and attached significance test shows that we have gotten rid of the nonlinearity, but the visual appearance strongly indicates non-constant variance.

```
> residual.plots(log.fit,fitted=FALSE)
```

```
  Test stat  Pr(>|t|)
X -0.8910355 0.3739971
```



This is confirmed by the test of constant variance.

```
> ncv.test(log.fit)

Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 147.5030    Df = 1     p = 0
```

3 Transformation to Linearity: Rules and Principles

Transformation Rules

Weisberg cites several rules and principles for transforming relationships to linearity, and in his ground-breaking work with Cook, he has provided a number of very useful tools for automating the transformation process.

In their book, *Applied Regression Including Computing and Graphics*, Cook and Weisberg present specialized free software for plotting data and linearizing the relationship by means of x-axis and y-axis "sliders," that allow you to move x and/or y up or down the transformation ladder.

Transformation Rules

In discussing transformation of variables, Weisberg mentions two rule, the *log rule* and the *range rule*.

The Log and Range Rules

- *The log rule.* If the values of a variable range over more than one order of magnitude and the variable is strictly positive, then replacing the variable by its logarithm is likely to be helpful.
- *The range rule.* If the range of a variable is considerably less than one order of magnitude, then any transformation of that variable is unlikely to be helpful.

Cook and Weisberg discuss applying Box-Cox transformations to either the y or x variable, or both. They mention two additional easy-to-remember rules that can make manipulating the value of λ more straightforward. Their rules are:

Spread Rules

- To spread the small values of a variable, make the power λ smaller.
- To spread the large values of a variable, make the power λ larger.

The Yeo-Johnson Family

The Box-Cox transformation family requires that data be positive. One approach to fixing non-positive data is simply to add a constant. We employed this approach earlier, but another more sophisticated approach is available, i.e., the Yeo-Johnson transformation family.

The Yeo-Johnson Family

The modified Box-Cox family $\psi_M(Y, \lambda_y)$ for a variable Y is a simple modification of the Box-Cox family:

$$\psi_M(Y, \lambda_y) = \begin{cases} \text{gm}(Y)^{1-\lambda_y} \times (Y^{\lambda_y} - 1)/\lambda & \lambda_y \neq 0 \\ \text{gm}(Y) \times \log y & \lambda = 0 \end{cases} \tag{1}$$

where gm is the *geometric mean*, $\text{gm}(Y) = \exp((\sum_i \log y_i)/N)$.

The Yeo-Johnson family is

$$\psi_{YJ}(U, \lambda) = \begin{cases} \psi_M(U + 1, \lambda) & U \geq 0 \\ \psi_M(-U + 1, 2 - \lambda) & U < 0 \end{cases} \tag{2}$$

Figure 7.9 from Weisberg shows some plots comparing the Box-Cox and Yeo-Johnson family transforms for some values of λ.

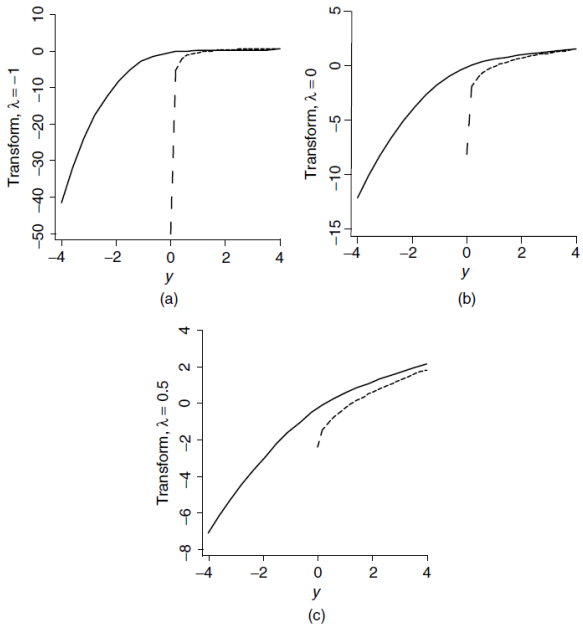


FIG. 7.9 Comparison of Box–Cox (dashed lines) and Yeo–Johnson (solid lines) power transformations for $\lambda = -1, 0, 0.5$. The Box–Cox transformations and Yeo–Johnson transformations behave differently for values of y close to zero.

Automated Transformation Software

The rules and principles discussed above can be very useful for arriving at a suitable transformation, especially when used in conjunction with the Arc freeware package.

Weisberg also discusses software for applying several classes of power transformations to both the independent and dependent variable.

Automated Transformation Software

As an example, consider the data that we just log-transformed, resulting in linearity, but a substantially non-constant variance.

In such situations, one has several options, with different authors taking somewhat different positions. For example, Weisberg mentions 4 options in his section 8.3.

These include use of a variance-stabilizing transformation and doing nothing.

In the latter case, estimates will still be unbiased, although somewhat less efficient. The standard error of estimate can no longer be used to construct confidence intervals, but bootstrapping can be employed.

Variance Stabilizing Transformations

Weisberg lists some common variance-stabilizing transformations in his Table 8.3.

TABLE 8.3 Common Variance Stabilizing Transformations

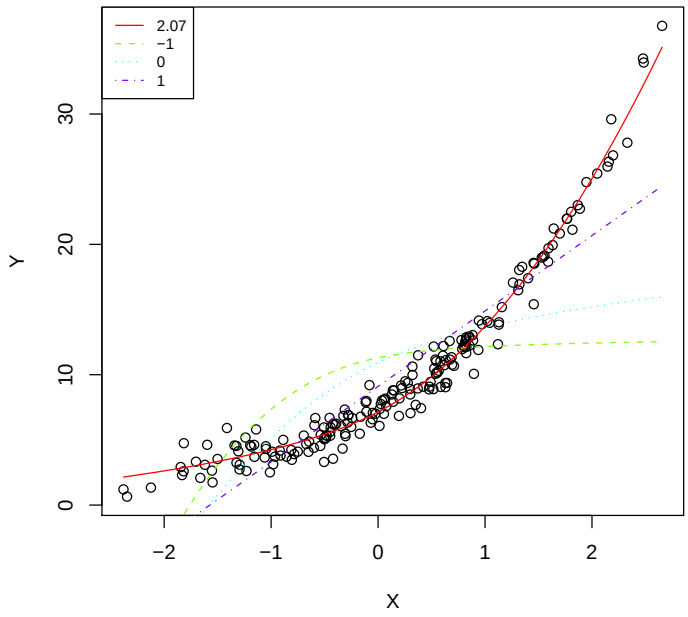
Y_T	Comments
$\sqrt{Y}$	Used when $\text{Var}(Y\|\text{X}) \propto \text{E}(Y\|\text{X})$, as for Poisson distributed data. $Y_T = \sqrt{Y} + \sqrt{Y+1}$ can be used if all the counts are small (Freeman and Tukey, 1950).
$\log(Y)$	Use if $\text{Var}(Y\|X) \propto [\text{E}(Y\|X)]^2$. In this case, the errors behave like a percentage of the response, $\pm 10\%$, rather than an absolute deviation, ± 10 units.
$1/Y$	The inverse transformation stabilizes variance when $\text{Var}(Y\|X) \propto [\text{E}(Y\|X)]^4$. It can be appropriate when responses are mostly close to 0, but occasional large values occur.
$\sin^{-1}(\sqrt{Y})$	The *arcsine square-root* transformation is used if Y is a proportion between 0 and 1, but it can be used more generally if y has a limited range by first transforming Y to the range (0, 1), and then applying the transformation.

In this case, however, an alternate transformation of X would have worked better than the log transform we employed. In the code below, we search for a Yeo-Johnson transformation. The code generates by default plots of $\lambda = -1, 0, 1$, and also finds and plots the best linearizing λ.

We apply the `inv.response.plot` function to the `linear.fit` object we obtained previously.

```
> inv.tran.plot(X,Y,family="yeo.johnson")
```

```
     lambda       RSS
1  2.066115  190.6161
2 -1.000000 6195.6576
3  0.000000 4231.4799
4  1.000000 1418.8249
```

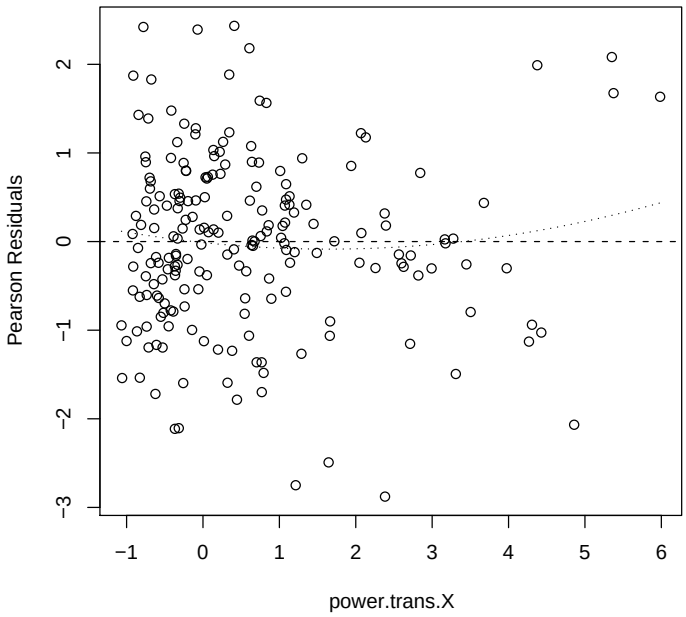


Automated Transformation Software

We can then use the `powtran` function to apply the transformation to X. The residual plot looks pretty good!

```
> power.trans.X ← powtran(X,lambda=2.066115,family="yeo.johnson")
> yeo.johnson.X.fit ← lm(Y ~ power.trans.X)
> residual.plots(yeo.johnson.X.fit,fitted=FALSE)
```

```
             Test stat  Pr(>|t|)
power.trans.X  1.057358 0.2916433
```



Automated Transformation Software

At least the non-constant variance test no longer rejects at the .05 level. (All the standard caveats about accepting the null apply here, of course.)

```
> ncv.test(yeo.johnson.X.fit)
```

```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 3.712164    Df = 1     p = 0.0540173
```

4 Evaluation of Outliers

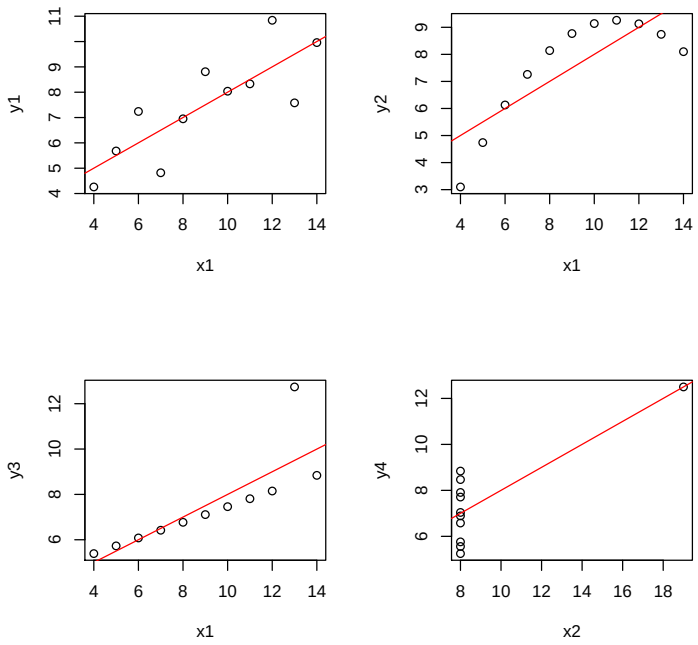
4.1 The Lessons of Anscombe's Quartet

Anscombe's Quartet

A famous example in the regression literature was provided by Anscombe, who presented 4 data sets with identical means, variances, and covariances, but very different looking scatterplots. These data came to be known as Anscombe's Quartet.

Anscombe's Quartet

```
> data(anscombe)
> attach(anscombe)
> par(mfrow=c(2,2))
> plot(x1,y1)
> abline(lm(y1~x1),col='red')
> plot(x1,y2)
> abline(lm(y2~x1),col='red')
> plot(x1,y3)
> abline(lm(y3~x1),col='red')
> plot(x2,y4)
> abline(lm(y4~x2),col='red')
```



Anscombe's Quartet

We see in the quartet some important aspects of regression. One point can have a powerful influence on a fit function, and data can have identical linear fit without being linear. All the data sets have identical R^2 values. For example:

```
> summary(lm(y1~x1))
```

```
Call:
lm(formula = y1 ~ x1)
```

```
Residuals:
     Min       1Q   Median       3Q      Max
-1.92127 -0.45577 -0.04136  0.70941  1.83882

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0001     1.1247   2.667  0.02573 *
x1            0.5001     0.1179   4.241  0.00217 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared: 0.6665,        Adjusted R-squared: 0.6295
F-statistic: 17.99 on 1 and 9 DF,  p-value: 0.002170
```

Anscombe's Quartet

In the following sections we shall briefly discuss aspects of outlier phenomena and outlier detection.

The discussion is relatively simple in two dimensions, but quickly becomes much more complicated in the context of multiple regression.

4.2 Leverage

Leverage

An observation can be unusual, but not have much or any effect on a linear regression fit line.

So we must distinguish between observations that are *unusual* and those that are *influential*.

Leverage

In general, to be influential, an observation has to be unusual.

However, an observation can be unusual without being influential.

Leverage

Leverage is a measure of how unusual an observation is.