Model Modifications

Bret Larget

Departments of Botany and of Statistics
University of Wisconsin—Madison

February 6, 2007

The Big Picture

- Residual plots can indicate lack of model fit.
- There are several possible remedies, including:
  1. *Transform one or both variables* and check if the standard assumptions are reasonable for the transformed variable(s).
     - Might be useful when residual plots indicate *non-linearity* and/or *heteroscedasticity*.
     - Conventional transformation include logarithms and square roots.
     - The Box-Cox family of transformations is also useful.
  2. *Use weighted least squares* when there is explainable *heteroscedasticity* but the linear model is otherwise fine.
  3. *Use polynomial regression* when there is non-linearity (curvature) but variances are close to constant.

Bacteria Count Example

- Data consist of number of surviving bacteria after exposure to X-rays for different time periods.
- `Time` denotes time measured in six-minute intervals.
- `N` denotes the number of survivors in hundreds.

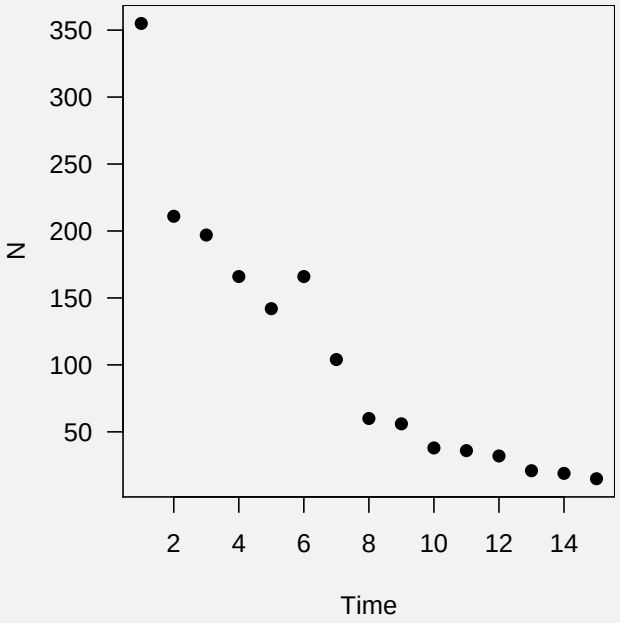
Time	1	2	3	4	5	6	7	8
N	355	211	197	166	142	166	104	60
Time	9	10	11	12	13	14	15	
N	56	38	36	32	21	19	15	

Example (cont.)

- Begin by plotting data.
- Fit a linear model.
- Assess fit informally with residual plots.

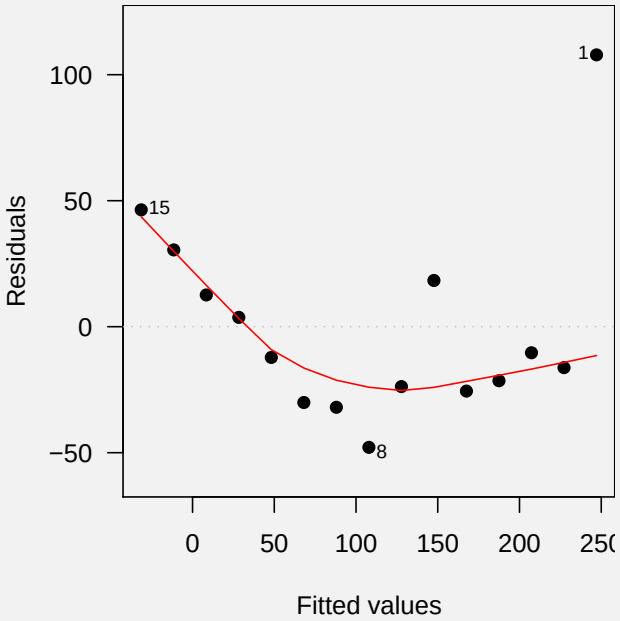
Scatterplot

```
> Time = 1:15
> N = c(355, 211, 197, 166, 142, 166, 104, 60, 56, 38,
+     36, 32, 21, 19, 15)
> par(las = 1, pch = 16)
> plot(Time, N)
> fit1 = lm(N ~ Time)
> plot(fit1, which = 1)
```



Residual Plot

- Scatterplot shows lack of linearity.
- Residual plot also shows increasing variance.
- Observation #1 is a bit of an outlier.
- Consider transforming variables to see if model fits better.



Review of Exponentiation and Logarithms

- The constant $\mathrm{e} \approx 2.718$ is the base of the natural logarithm.
- Recall from calculus, e is the unique base where the derivative equals the function, $\frac{\mathrm{d}}{\mathrm{d}x}(\mathrm{e}^x) = \mathrm{e}^x$.
- I will use log, not ln, to stand for the natural logarithm.
- Products of exponentials are exponentials of sums, $\mathrm{e}^a \times \mathrm{e}^b = \mathrm{e}^{a+b}$.
- The natural logarithm of e is one, $\log \mathrm{e} = 1$.
- Any logarithm of 1 is zero, $\log 1 = 0$.
- Rule for exponents, $\log a^b = b \log a$.
- Logarithms of products, $\log(ab) = \log(a) + \log(b)$.
- e^x *exists for all* x *and* $\mathrm{e}^x > 0$.
- $\log x$ *is defined only for* $x > 0$.

Exponential Model

- Here there is a theoretical model:

$$n_t = n_0 \mathrm{e}^{\beta_1 t} \times E,$$

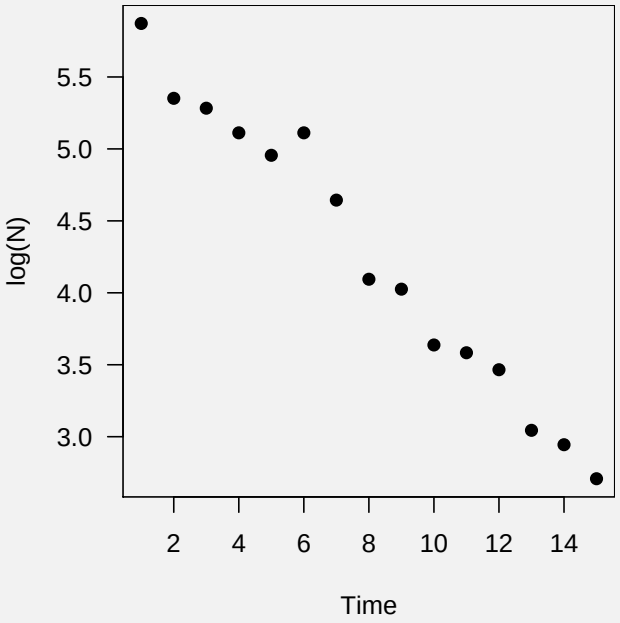
  where
  - t is time
  - n_t is the number of bacteria at time t
  - n_0 is the number of bacteria at time $t = 0$
  - $\beta_1 < 0$ is a decay rate
  - E is some multiplicative error.
- Take natural logs of both sides of the model:

$$\begin{aligned} \log(n_t) = \log(n_0 \mathrm{e}^{\beta_1 t} E) &= \log(n_0) + \log(\mathrm{e}^{\beta_1 t}) + \log(E) \\ &= \log(n_0) + \beta_1 t + \log(E) \\ &= \beta_0 + \beta_1 t + e, \end{aligned}$$

- That is, we log-transformed n_t and the result is a usual linear-line model, if the error E on the original scale is multiplicative and its logarithm is normally distributed.

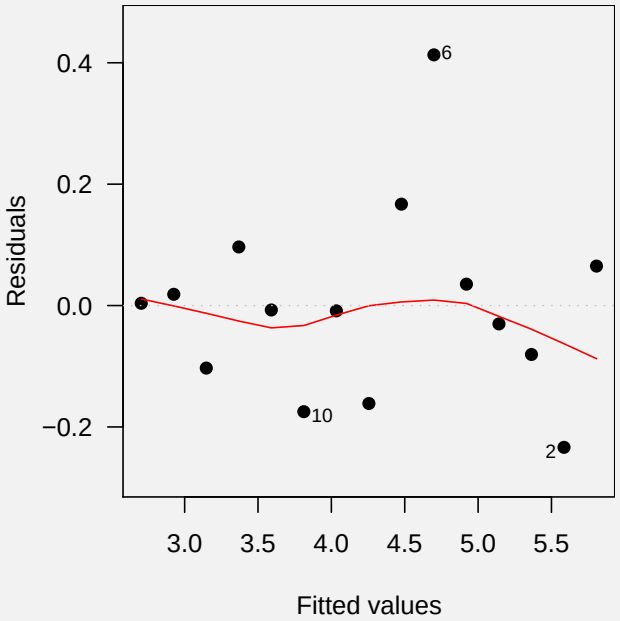
Scatterplot

```
> par(las = 1, pch = 16)
> plot(Time, log(N))
> fit2 = lm(log(N) ~ Time)
> plot(fit2, which = 1)
```



Residual Plot

- Diagnostics are consistent with a model that fits well.
- There is no obvious non-linearity.
- There is no obvious heteroscedasticity.
- Residual plot has no large deviations from random scatter.



Fitted Model for Log-Transformed Data

```
> summary(fit2)

Call:
lm(formula = log(N) ~ Time)

Residuals:
      Min        1Q    Median        3Q       Max
-0.233578 -0.091798 -0.007255  0.050165  0.413068

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.028695   0.088259   68.31  < 2e-16 ***
Time        -0.221629   0.009707  -22.83  7.1e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1624 on 13 degrees of freedom
Multiple R-Squared: 0.9757, Adjusted R-squared: 0.9738
F-statistic: 521.3 on 1 and 13 DF,  p-value: 7.103e-12
```

- $\hat{\beta}_0 = 6.03$, $\hat{\beta}_1 = -0.222$.
- On the original scale, $\exp(\hat{\beta}_0) = 415$.
- Fitted Model:

$$y = 415 \times \mathrm{e}^{-0.222x}$$

where:

- y = bacteria count in hundreds
- x = time in 6-minute intervals

Confidence and Prediction Intervals

```
> t0 = data.frame(Time = 10)
> predict(fit2, t0, interval = "c")

         fit     lwr      upr
[1,] 3.812403 3.71256 3.912246

> predict(fit2, t0, interval = "p")

         fit     lwr      upr
[1,] 3.812403 3.44756 4.177246

> exp(predict(fit2, t0, interval = "c"))

         fit      lwr      upr
[1,] 45.25907 40.95853 50.01116

> exp(predict(fit2, t0, interval = "p"))

         fit      lwr     upr
[1,] 45.25907 31.42363 65.1861
```

- The point estimate for the mean bacteria count in the 10th time interval is 45.3.
- A 95% confidence interval goes from 41 to 50.
- A 95% prediction interval goes from 31.4 to 65.2.

Other Transformations

- We could transform either y or x or both.
- Common transformations include:
  - natural log, ln or log
  - log base 10, $\log_{10}$
  - square root, $\sqrt{\cdot}$
  - reciprocal, $1/y$
- Less common transformations include:
  - Squaring, y^2
  - Reciprocal squaring, $1/y^2$
  - Cube root, $y^{1/3}$,
  - Arcsin transformation, $\arcsin\sqrt{y}$, useful when y is a proportion.

Box-Cox Transformations

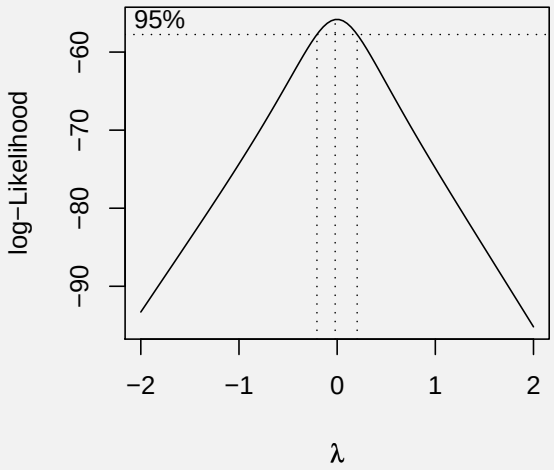
- *Box-Cox transformations* are a continuous family of power transformations.
- The log transformation corresponds to a power of 0.
- The Box-Cox transformation is:

$$\begin{aligned} y(\lambda) &= \frac{y^\lambda - 1}{\lambda}, \qquad \text{if } \lambda \neq 0 \\ y(\lambda) &= \log y, \qquad \text{if } \lambda = 0 \end{aligned}$$

- In R, this transfomration is in the library `MASS` in a function `boxcox`.
- In theory, we can estimate λ as a continuous parameter.
- In practice, we might select a common transformation power close to the continuous estimate.

Box-Cox Transformation on Bacteria

```
> Time = 1:15
> N = c(355, 211, 197, 166, 142, 166, 104, 60, 56,
+     38, 36, 32, 21, 19, 15)
> fit1 = lm(N ~ Time)
> library(MASS)
> boxcox(fit1)
```



Remarks

- Typically choose transformations emprically.
- Transformation of y can affect both linearity and variance homogeneity.
- Transformations of x only affect linearity.
- Sometimes, solving one problem causes another: for example, transforming y to stabilize the variance can introduce curvature.
- Start simple and experiment.

Weighted Regression

- *Weighted regression* is a method that can be used when linearity is correct, but variance is not constant.
- Sometimes there may be a theory justifying a formula for the variance as a function of x, such as $\sigma_x^2 = \sigma^2 \times x^2$ which would be he case if variance increased quadratically.
- Sometimes people use empirical methods to estimate variances.
- Differences in variances can result from having access to means but not individual values from different sample sizes.
- In weighted regression, *we use a weighted least squares criterion*.
- If observation y_i has variance σ_i^2, the appropriate least squares criterion is:

$$\sum_{i=1}^{n} w_i(y_i - (b_0 + b_1 x_i))^2$$

where $w_i = 1/\sigma_i^2$.

Example

- Here is an example of artificial data reflecting missing data.
- In a planned experiment, there were 10 seeds in each of five treatment groups.
- However, not all seeds germinated resulting in actual sample sizes of 10, 10, 7, 8, and 6.
- The response of height at a specific number of days after planting is reported as a mean per plant.
- The treatment is a concentration of fertilizer.
- Data.

conc	1.0	2.0	4.0	8.0	16.0
height	20.7	21.1	23.1	25.1	28.7
n	10	10	7	8	6

- We could use simple linear regression on the raw data before taking means. (There are real settings where individual data is unavailable.)

R

```
> conc = c(1, 2, 4, 8, 16)
> ht = c(20.7, 21.1, 23.1, 25.1, 28.7)
> n = c(10, 10, 7, 8, 6)
> fit.u = lm(ht ~ conc)
> fit.w = lm(ht ~ conc, weights = n)
> summary(fit.u)
> summary(fit.w)
> plot(fitted(fit.u), rstudent(fit.u))
> plot(fitted(fit.w), rstudent(fit.w))
```

```
> coef(fit.u)

(Intercept)        conc
 20.4333333   0.5333333

> sqrt(10) * summary(fit.u)$sigma

[1] 1.577621

> coef(fit.w)

(Intercept)        conc
 20.3433552   0.5441396

> summary(fit.w)$sigma

[1] 1.395566
```


Residual Plots