Workshop

Session 2: Linear Models and Their Allies

Bill Venables, CSIRO, Australia

UseR! 2012

Nashville

11 June, 2012

Contents

1 A simple example: the Janka Hardness data

Example taken from Williams (1959).

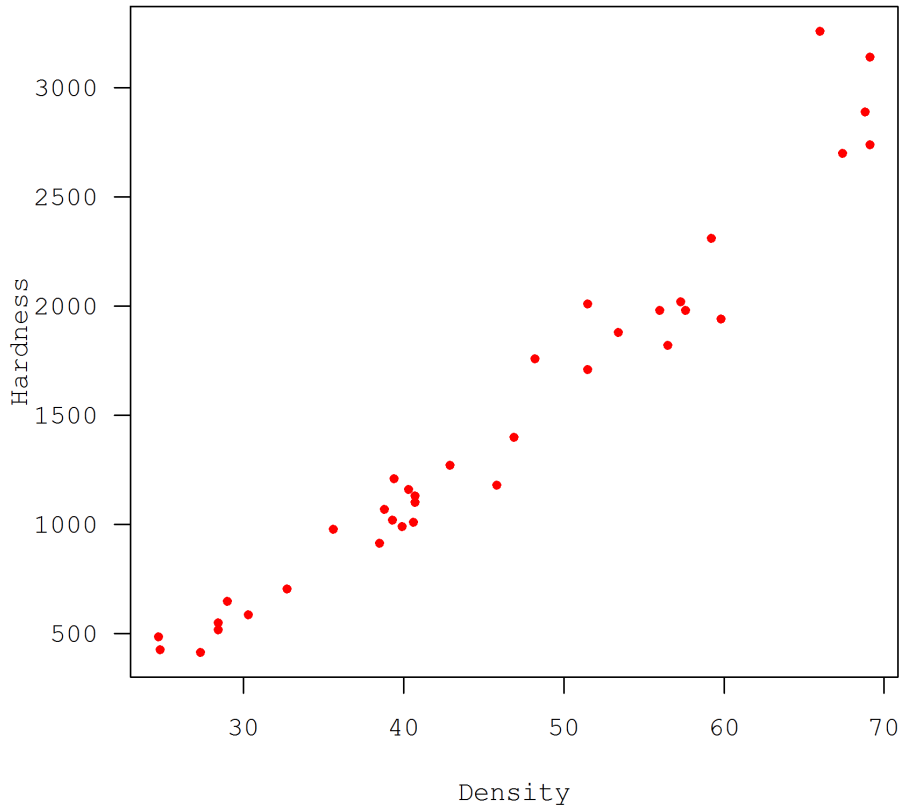
Two variables:

Hardness : The Janka hardness of a sample of timbers, (in lbs). (See here for a discussion.)

Density : The density of the sample, (in lbs/ft^3)

The problem: *Build a predictor of Harndess from Density*.

```
> data(janka)
> library(SOAR)
> Store(janka)   ## for convenience
> plot(Hardness ~ Density, janka, pch=20, col="red")
```



- Clearly strong dependence of Hardness on Density;
- Possibly curvilinear – try polynomials first;
- Possibly with unequal variance?

```
> m1 <- lm(Hardness ~ Density, janka)
> m2 <- update(m1, . ~ . + I(Density^2))
> m3 <- update(m2, . ~ . + I(Density^3))
> round(summary(m3)$coef, 4)

             Estimate Std. Error t value Pr(>|t|)
(Intercept)  -641.4379  1235.6550 -0.5191   0.6073
Density        46.8637    86.3018  0.5430   0.5909
I(Density^2)   -0.3312     1.9140 -0.1730   0.8637
I(Density^3)    0.0060     0.0135  0.4405   0.6625
```

1.1 Alternative parametrizations

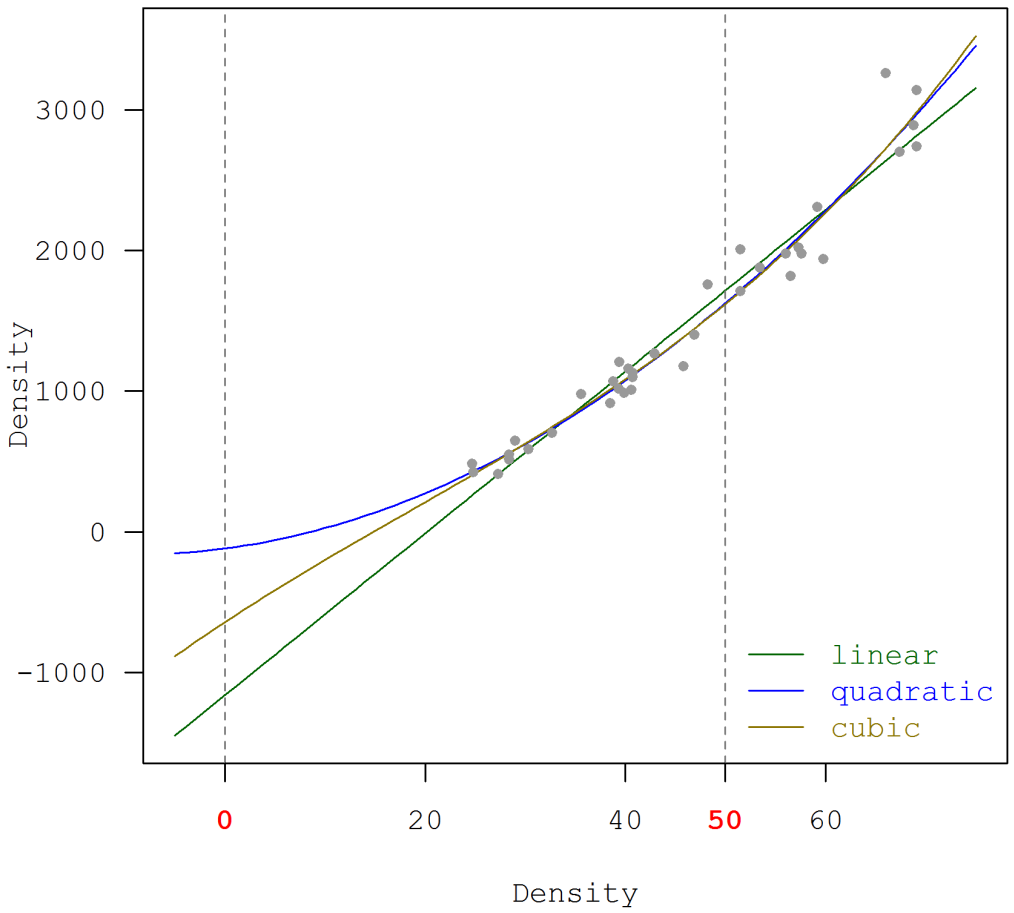
Nothing is "significant"! Why?

A modification:

```
> m1a <- lm(Hardness ~ I(Density - 50), janka)
> m2a <- update(m1a, . ~ . + I((Density - 50)^2))
> m3a <- update(m2a, . ~ . + I((Density - 50)^3))
> round(summary(m3a)$coef, 4)
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)         1618.6529    43.4597 37.2449   0.0000
I(Density - 50)       58.4367     4.8619 12.0193   0.0000
I((Density - 50)^2)    0.5626     0.1999  2.8147   0.0083
I((Density - 50)^3)    0.0060     0.0135  0.4405   0.6625
```

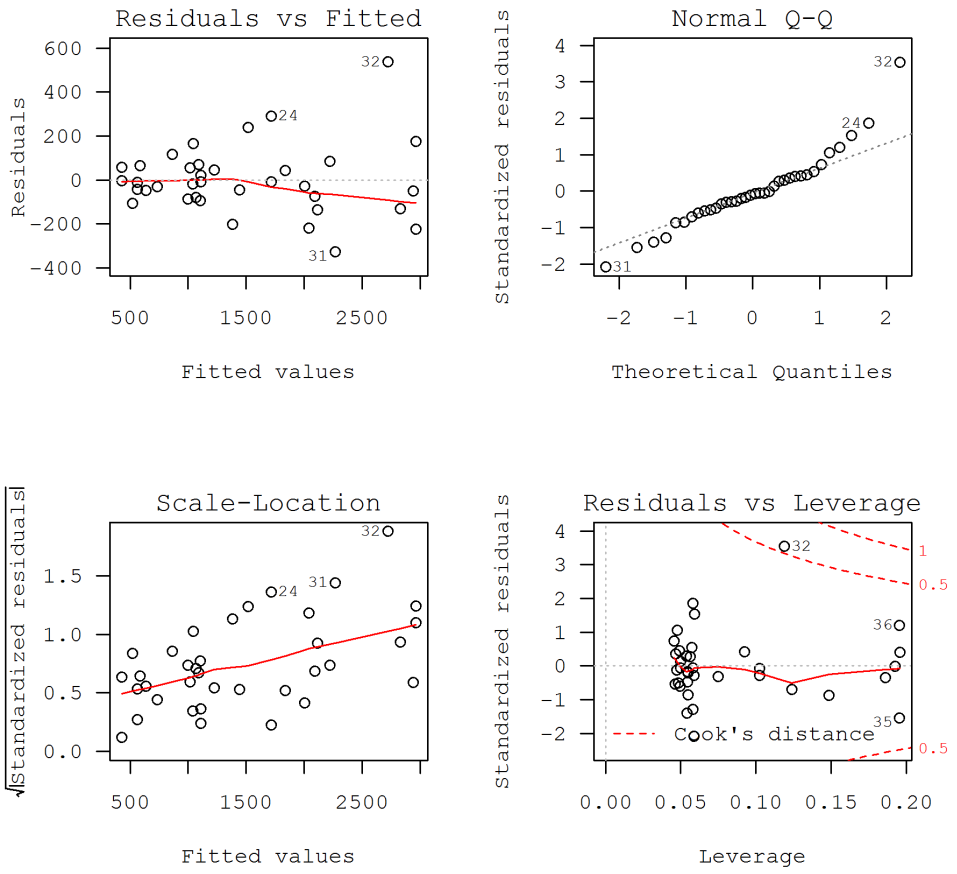
The models are fully equivalent, but the coefficients refer to different things

```
> pJanka <- with(janka, data.frame(Density = seq(-5, 75)))
> pJanka <- within(pJanka, {
    pM1 <- predict(m1a, pJanka, type = "response")
    pM2 <- predict(m2a, pJanka, type = "resp")
    pM3 <- predict(m3a, pJanka, type = "resp")
  })
> r0 <- with(pJanka, with(janka, range(Hardness, pM1, pM2, pM3)))
> plot(pM1 ~ Density, pJanka, ylim = r0, ylab = "Density",
       type = "l", col="darkgreen")
> lines(pM2 ~ Density, pJanka, col = "blue")
> lines(pM3 ~ Density, pJanka, col = "gold4")
> points(Hardness ~ Density, janka, pch = 20, col = grey(0.6))
> abline(v = c(0, 50), col = grey(0.5), lty = "dashed")
> axis(1, at = c(0, 50), font = 2, col.axis = "red")
> legend("bottomright", c("linear", "quadratic", "cubic"),
         lty = "solid", col = c("darkgreen", "blue", "gold4"),
         text.col = c("darkgreen", "blue", "gold4"), bty="n")
```


3000
2000
1000
0
-1000
Density
0
20
40
50
60
Density
linear
quadratic
cubic

1.2 Diagnostics and corrective action

```
> layout(matrix(1:4, 2,2, byrow = TRUE))
> plot(m2a)
```



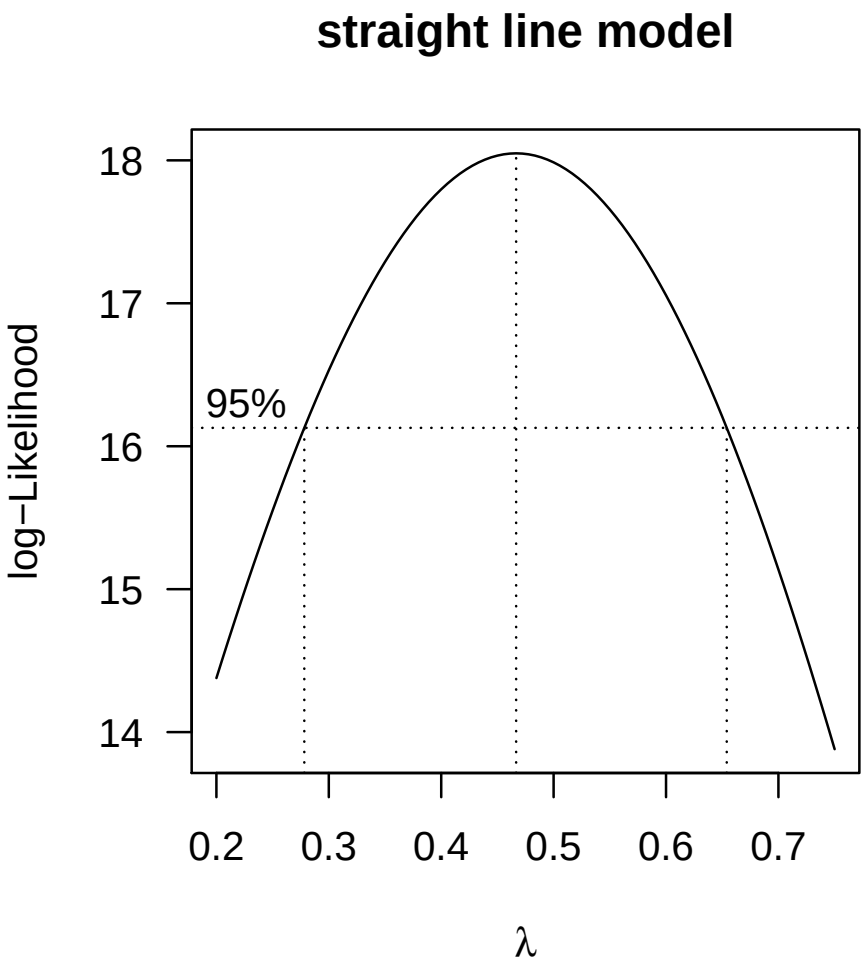
1.3 A transformation?

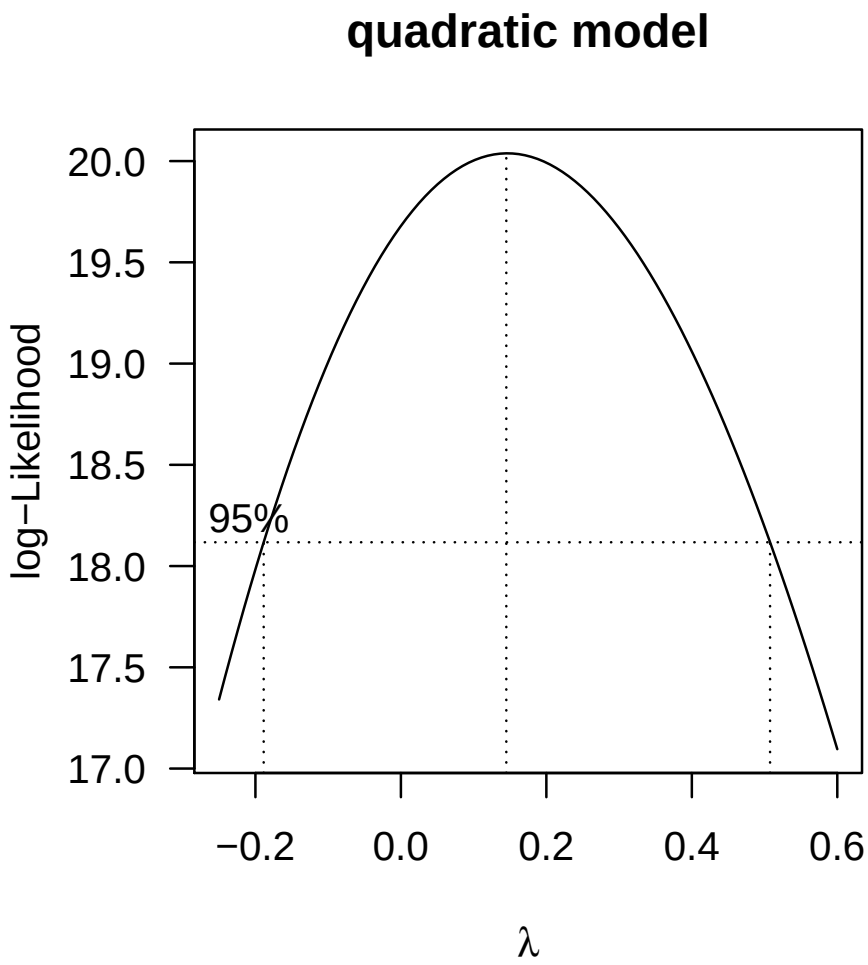
The main purpose of a transformation is to provide a scale in which the response is homoscedastic — but no necessarily the only purpose.

Transforming the response will affect both the mean structure and the variance.

Consider a Box-Cox transformation on a) the straight line model and b) the quadratic model.

```
> library(MASS)
> layout(matrix(1:2, 1))
> boxcox(m1a, lambda = seq( 0.2, 0.75, len=10))
> title(main = "straight line model")
> boxcox(m2a, lambda = seq(-0.25, 0.6, len=10))
> title(main = "quadratic model")
```


straight line model
log−Likelihood
18
17
16
15
14
95%
0.2
0.3
0.4
0.5
0.6
0.7
λ

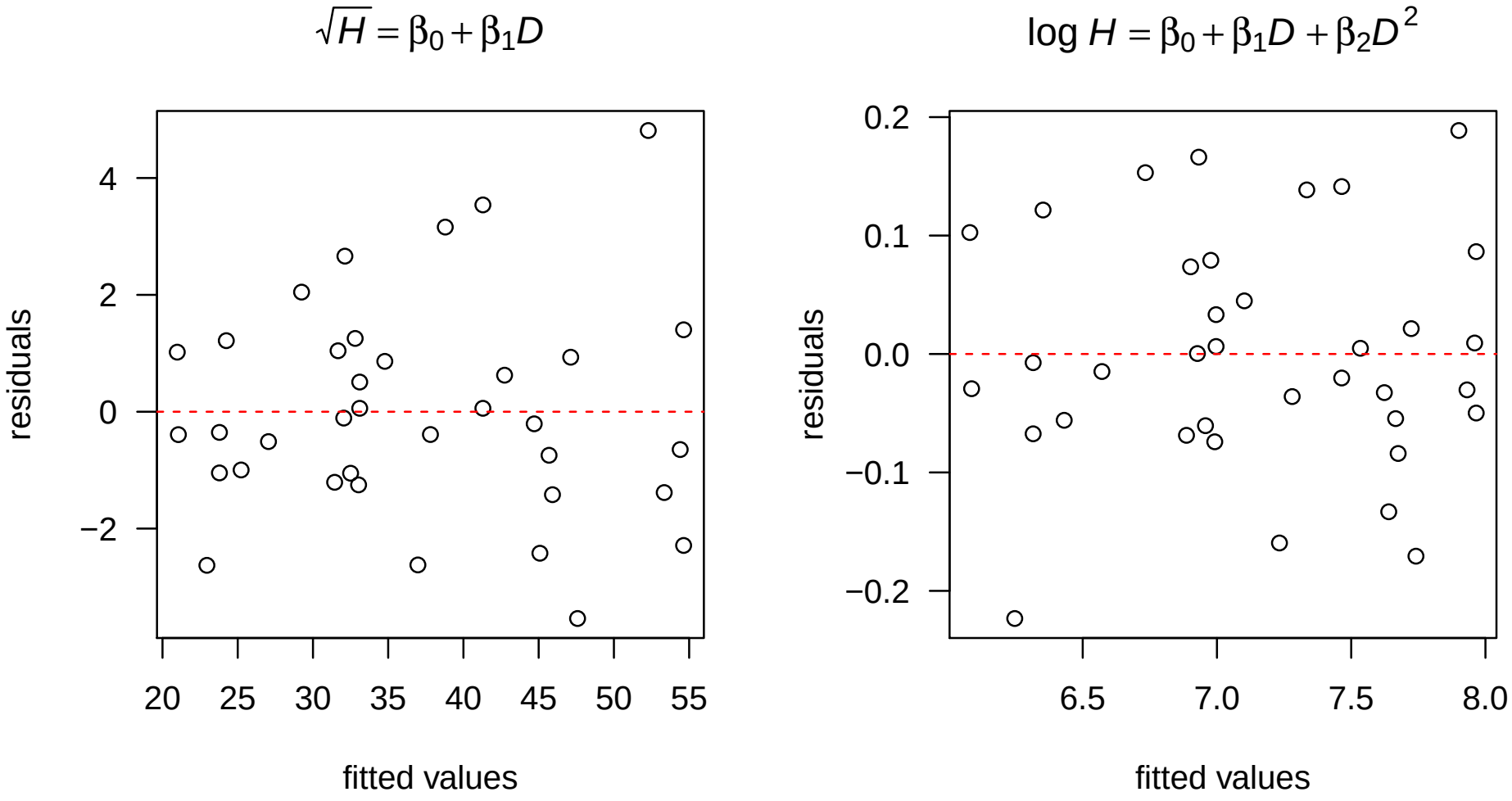

quadratic model
log−Likelihood
20.0
19.5
19.0
18.5
18.0
17.5
17.0
95%
−0.2
0.0
0.2
0.4
0.6
λ

Assuming a straight line — a square root transformation?

Allowing for a quadratic response — a log transformation?

Consider the effect of the transformation on residuals:

```
> mSL <- update(m1a, sqrt(.) ~ .)
> mQD <- update(m2a, log(.) ~ .)
> r1 <- resid(mSL); f1 <- fitted(mSL)
> r2 <- resid(mQD); f2 <- fitted(mQD)
> layout(matrix(1:2, 1))
> plot(f1, r1, xlab = "fitted values", ylab = "residuals",
      main = expression(sqrt(italic(H)) == beta[0] +
          beta[1]*italic(D)))
> abline(h=0, col="red", lty="dashed")
> plot(f2, r2, xlab = "fitted values", ylab = "residuals",
      main = expression(log~italic(H) == beta[0] +
          beta[1]*italic(D) + beta[2]*italic(D)^2))
> abline(h=0, col="red", lty="dashed")
```


$\sqrt{H} = \beta_0 + \beta_1 D$
residuals
4
2
0
−2
20
25
30
35
40
45
50
55
fitted values
$\log H = \beta_0 + \beta_1 D + \beta_2 D^2$
residuals
0.2
0.1
0.0
−0.1
−0.2
6.5
7.0
7.5
8.0
fitted values

The message so far:

- A square root transformation straightens out the regression line, but
- A log transformation is necessary to even out the variance.

This suggests a generalized linear model:

- with a `sqrt` link, to give a scale in which the response is straight line,
- with a variance function $\propto \mu^2$ to allow for variance heterogeneity.

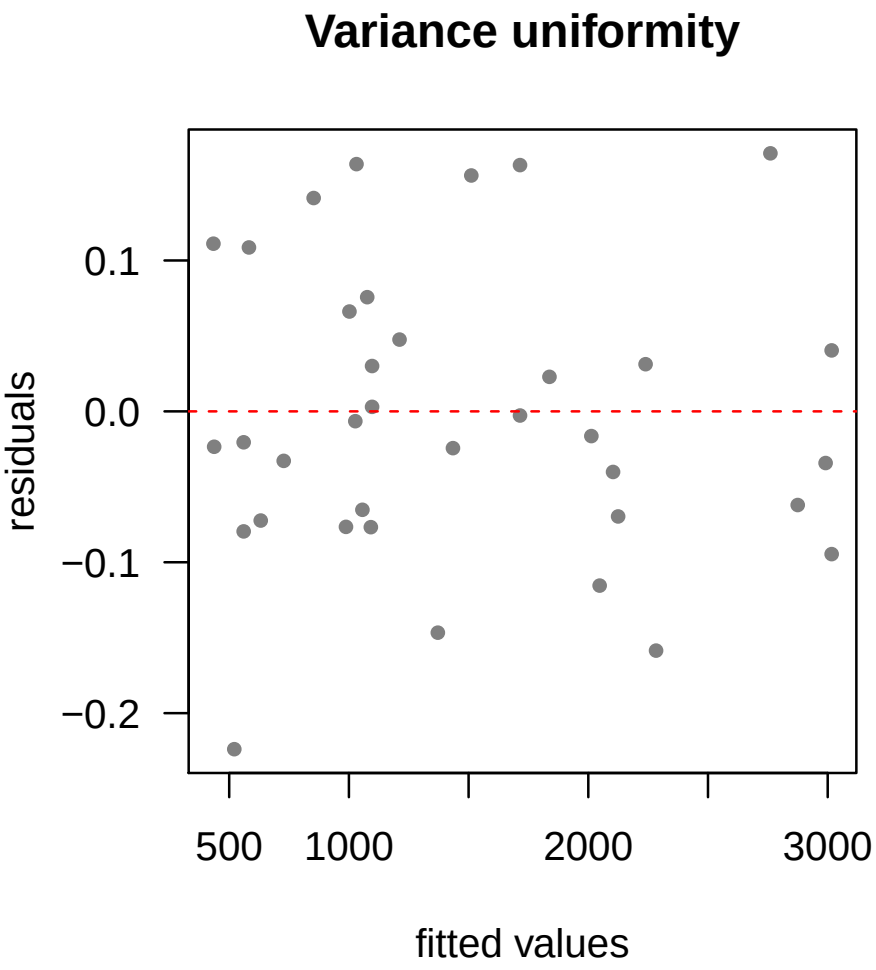
Within the GLM family this suggests a Gamma model, but the `sqrt` link is non-standard. Oh well...

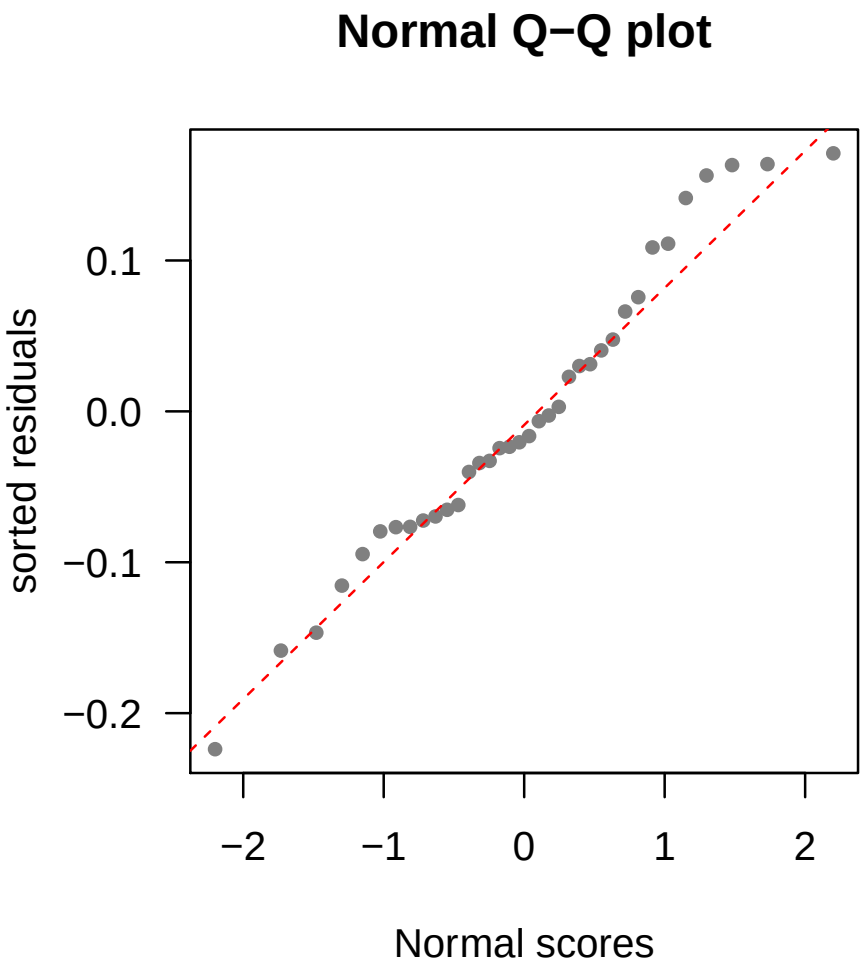
```
> library(glm2)
> mGLM <- glm2(Hardness ~ I(Density-50),
               family = Gamma(link = make.link("sqrt")),
               data = janka)
> mGLM2 <- update(mGLM, .~.+I((Density-50)^2))
> round(summary(mGLM2)$coef, 4)
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          40.4191     0.4382 92.2296   0.0000
I(Density - 50)       0.7516     0.0313 24.0035   0.0000
I((Density - 50)^2)  -0.0013     0.0017 -0.7373   0.4661
```

The glm2 library, (Marschner, 2011), offers the function *glm2*, a drop-in replacement for *glm* with better convergence properties, particularly for non-natural links.

The straight line model seems adequate - residual checks:

```
> rs <- resid(mGLM)
> fv <- fitted(mGLM)
> layout(matrix(1:2, 1))
> plot(fv, rs, xlab = "fitted values", ylab = "residuals",
      pch = 20, col=grey(0.5), main = "Variance uniformity")
> abline(h = 0, lty = "dashed", col="red")
> qqnorm(rs, pch = 20, col = grey(0.5), xlab = "Normal scores",
        ylab = "sorted residuals", main = "Normal Q-Q plot")
> qqline(rs, col="red", lty = "dashed")
```


Variance uniformity
residuals
0.1
0.0
−0.1
−0.2
500
1000
2000
3000
fitted values


Normal Q−Q plot
sorted residuals
0.1
0.0
−0.1
−0.2
−2
−1
0
1
2
Normal scores

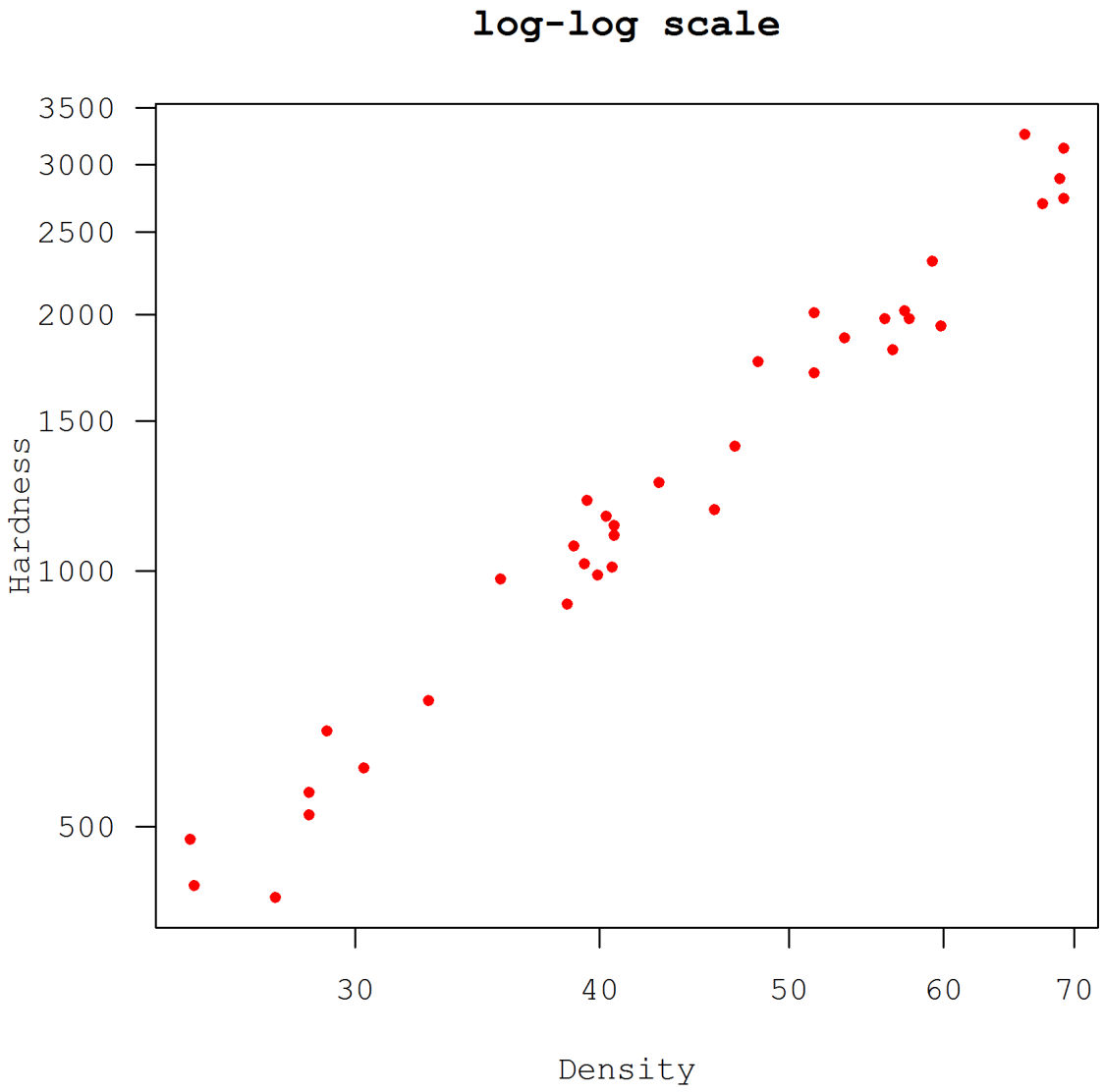
1.4 A simpler approach

- What happens if we transform *both* response and predictor?
- A mulitplicative model seems heuristically reasonable:

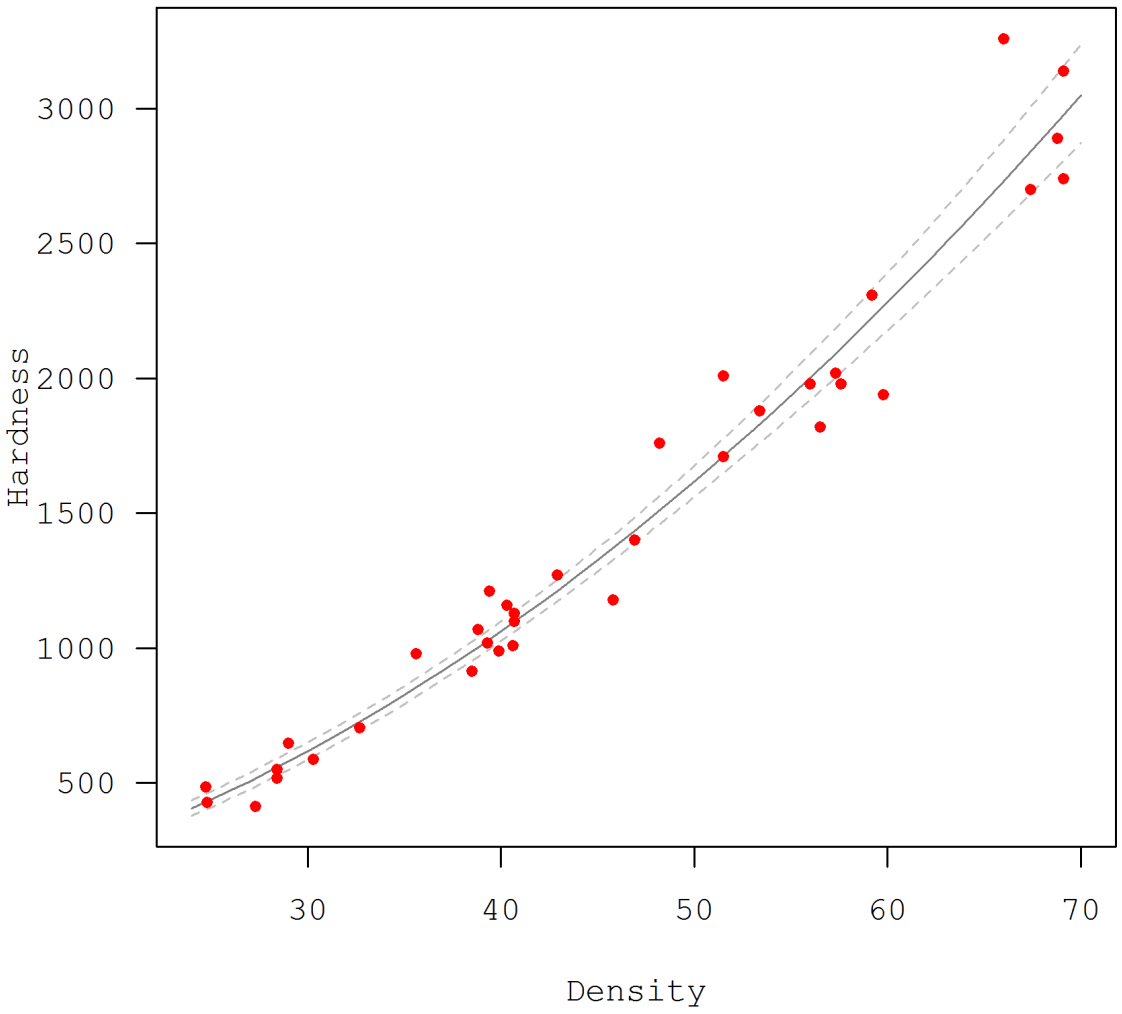
$$H = \alpha D^{\beta} \exp E \qquad \Longrightarrow \qquad \log H = \alpha^{\star} + \beta \log D + E$$

- Simple back transformation will give an estimate of the *median*. In some cases an estimate of the mean is more appropriate. See, e.g. Shen and Zhu (2008).

```
> plot(Hardness ~ Density, janka, log = "xy", pch=20, col="red",
       main = "log-log scale")
```


log-log scale
3500
3000
2500
2000
1500
1000
500
Hardness
30
40
50
60
70
Density

```
> LogM <- lm(log(Hardness) ~ poly(log(Density), 3), janka)
> round(summary(LogM)$coef, 4)
                       Estimate Std. Error  t value Pr(>|t|)
(Intercept)              7.1362     0.0167 426.5375   0.0000
poly(log(Density), 3)1   3.4304     0.1004  34.1734   0.0000
poly(log(Density), 3)2  -0.0482     0.1004  -0.4807   0.6340
poly(log(Density), 3)3  -0.0121     0.1004  -0.1208   0.9046
> LogM1 <- update(LogM, . ~ log(Density))
> pJanka <- data.frame(Density = 24:70)  ## just covers actual range
> prVal <- with(predict(LogM1, pJanka, type = "resp", se.fit = TRUE),
        exp(cbind(fitted=fit, lower=fit-2*se.fit, upper=fit+2*se.fit)))
> pJanka <- cbind(pJanka, prVal)
> ry <- with(pJanka, with(janka, range(Hardness, fitted, upper, lower)))
> plot(fitted ~ Density, pJanka, ylab="Hardness", type="l",
      ylim=ry, col=grey(0.5))
> lines(upper ~ Density, pJanka, lty="dashed", col=grey(0.75))
> lines(lower ~ Density, pJanka, lty="dashed", col=grey(0.75))
> points(Hardness ~ Density, janka, pch=20, col="red")
```


Hardness
3000
2500
2000
1500
1000
500
30
40
50
60
70
Density

1.5 Bootstrap confidence intervals

To illustrate direct bootstrap computations, we choose the Bayesian Bootstrap idea of Rubin (1981).

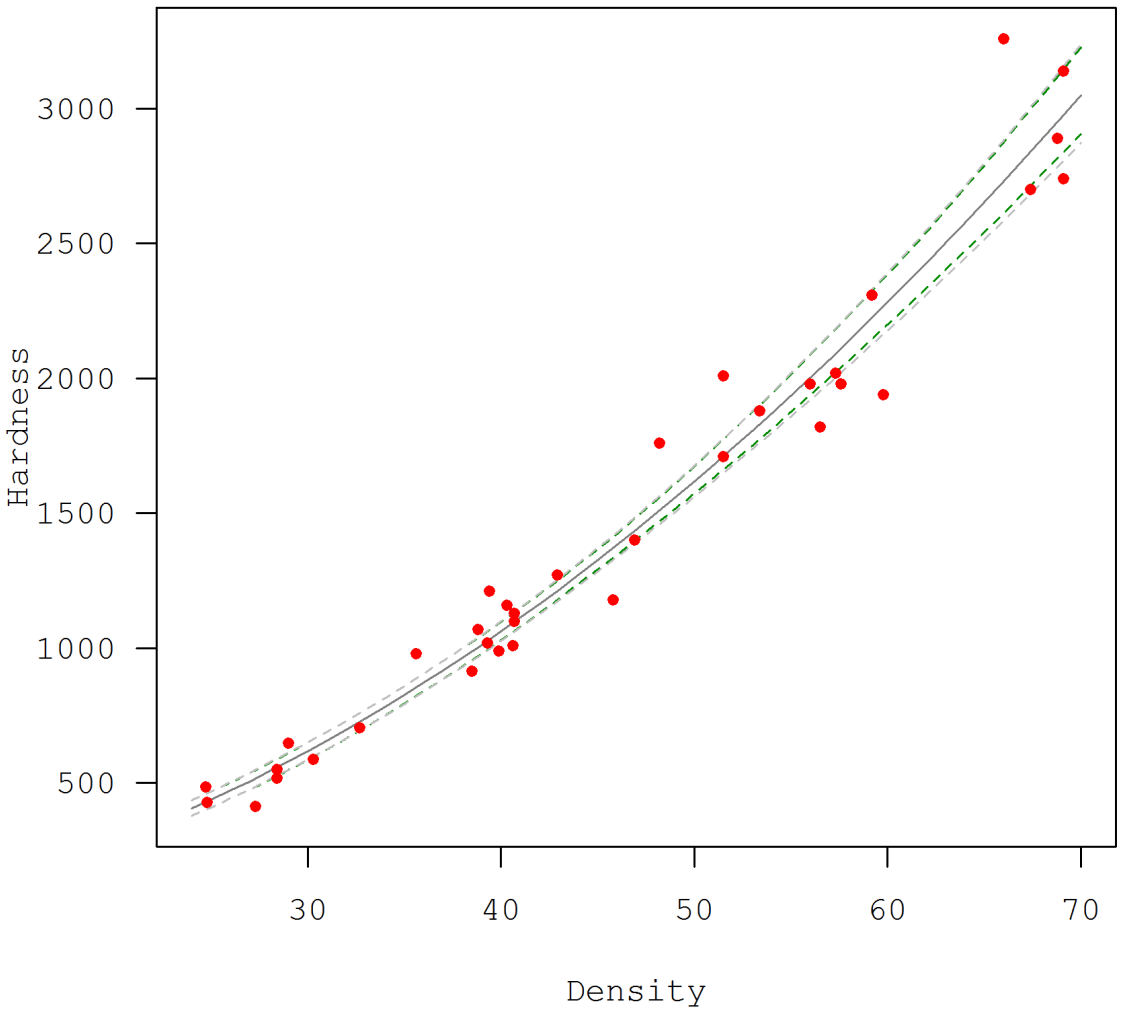
- Re-fit the model with *random weights* for the observations.
- Choosing $W \sim \text{Exp}(1)$ gives $\text{E}\, W = 1 = \text{Var}\, W$, the same as for the normal bootstrap. (Rubin gives a theoretical justification.)

```
> W <- rexp(10000)
> c(mean = mean(W), variance = var(W))

    mean variance
1.001832 1.013044
```

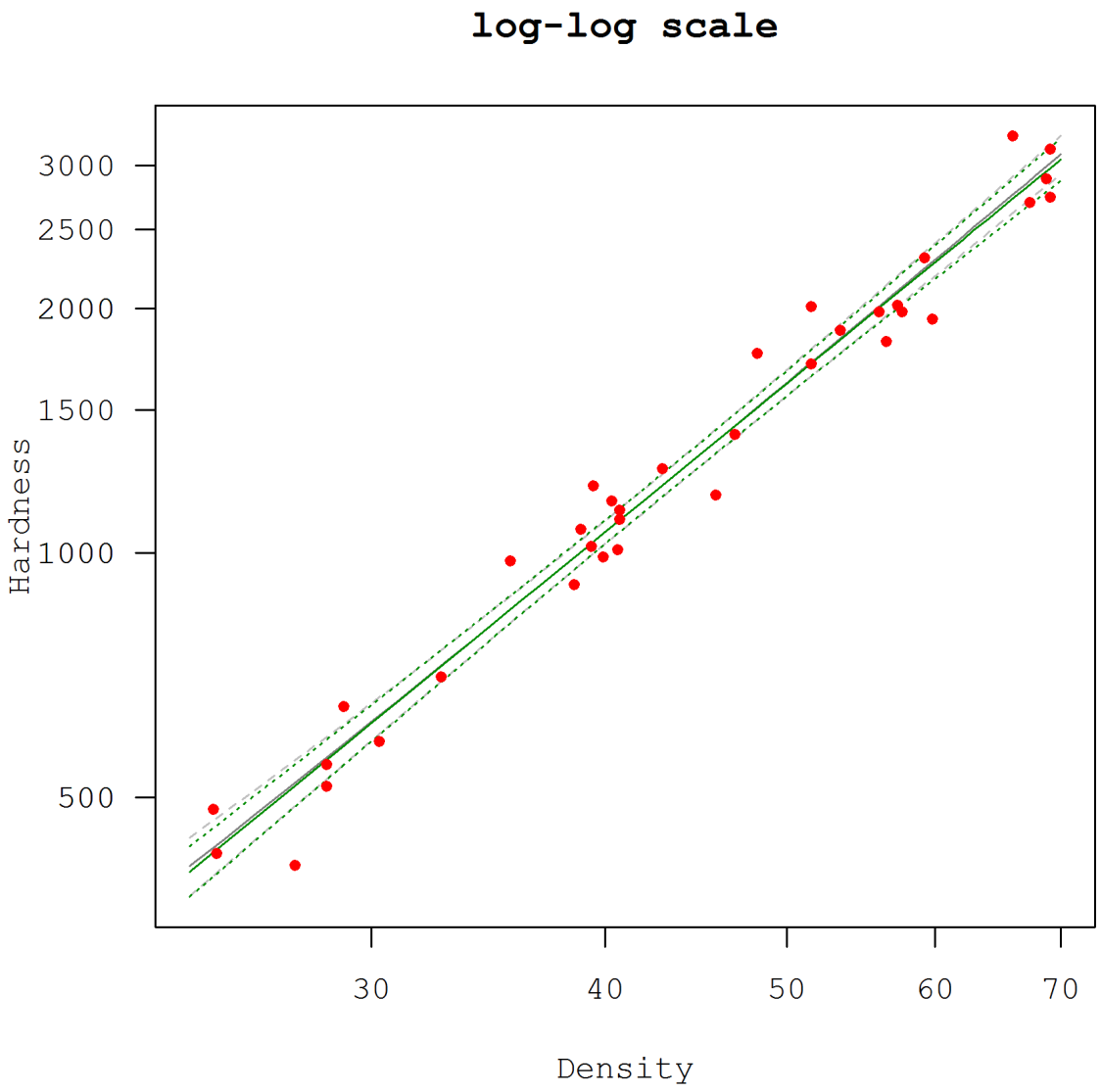
We do this for the log-log model only. (For the GLM — easy exercise!)

```
> set.seed(1234)
> X <- replicate(500, {
    tmp <- update(LogM1, weights = rexp(nrow(janka)))
    predict(tmp, pJanka, type = "resp")  ## predicts in log space
  })
> ci <- t(apply(X, 1, quantile, prob = c(1,39)/40))
> pJanka <- cbind(pJanka, lowerBB = exp(ci[, 1]), upperBB = exp(ci[,2]))
> ry <- with(pJanka, with(janka, range(Hardness, fitted, upper, lower)))
> plot(fitted ~ Density, pJanka, ylab = "Hardness", type = "l",
       ylim = ry, col=grey(0.5))
> lines(upperBB ~ Density, pJanka, lty = "dashed", col = "green4")
> lines(lowerBB ~ Density, pJanka, lty = "dashed", col = "green4")
> lines(upper ~ Density, pJanka, lty = "dashed", col = grey(0.75))
> lines(lower ~ Density, pJanka, lty = "dashed", col = grey(0.75))
> points(Hardness ~ Density, janka, pch=20, col="red")
> rm(X)
```


Hardness
3000
2500
2000
1500
1000
500
30
40
50
60
70
Density

Compare with the Gamma model:

```
> prVal <- with(predict(mGLM, pJanka, type = "link", se.fit = TRUE),
                cbind(fittedGLM = fit, lowerGLM = fit-2*se.fit,
                      upperGLM = fit+2*se.fit)^2)  ## inverse of sqrt
> pJanka <- cbind(pJanka, prVal)
> ry <- with(pJanka, range(ry, fittedGLM, upperGLM, lowerGLM))
> plot(fittedGLM ~ Density, pJanka, ylab = "Hardness", type = "l",
       ylim = ry, col=grey(0.5), log = "xy", main = "log-log scale")
> axis(2, at = 500*(1:6))
> lines(fitted ~ Density, pJanka, col = "green4")
> lines(upperGLM ~ Density, pJanka, lty = "dashed", col = grey(0.75))
> lines(lowerGLM ~ Density, pJanka, lty = "dashed", col = grey(0.75))
> lines(upper ~ Density, pJanka, lty = "dotted", col = "green4")
> lines(lower ~ Density, pJanka, lty = "dotted", col = "green4")
> points(Hardness ~ Density, janka, pch=20, col="red")
```


log-log scale
Hardness
3000
2500
2000
1500
1000
500
30
40
50
60
70
Density

1.6 Technical highlights

- Slide 3: using *data* to load a local "`.csv`" file;

  The `SOAR` package.

- Slide 6: *with* for local access to an object;

  within for modification of a data frame, with local access as well.

- Slide 8: *plot* methods for linear model objects.

- Slide 9: The *layout* function for multi-frame plotting.

- Slide 11: Mathematical expressions in plot annotation.

- Slide 14: The *make.link* function for generating non-standard links for GLMs;

  The `glm2` package for potentially troublesome GLM fits.

- Slide 18: Use of log scales in traditional graphics.
- Slide 19: Quick tests in polynomial regression using *poly*.

  Nested use of *with* for local access to two data frames simultaneously.
- Slide 22: The *replicate* convenience function for direct bootstrapping.

  The Bayesian Bootstrap idea of Rubin.

References

Marschner, I. C. (2011, December). glm2: Fitting generalized linear models with convergence problems. *The* **R** *Journal 3*(2), 12–15.

Rubin, D. B. (1981). The bayesian bootstrap. *The Annals of Statistics 9*, 130–134.

Shen, H. and Z. Zhu (2008). Efficient mean estimation in log-normal lineal modeld. *Journal of Statistical Planning and Inference 138*, 552–567.

Venables, W. N. and B. D. Ripley (2002). *Modern Applied Statistics with* **S** (Fourth ed.). New York: Springer. ISBN 0-387-95457-0.

Williams, E. J. (1959). *Regression Analysis*. New York: Wiley.

Session information

- R version 2.15.0 (2012-03-30), `i386-pc-mingw32`
- Locale: `LC_COLLATE=English_Australia.1252`, `LC_CTYPE=English_Australia.1252`, `LC_MONETARY=English_Australia.1252`, `LC_NUMERIC=C`, `LC_TIME=English_Australia.1252`
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: glm2 1.0, MASS 7.3-18, SOAR 0.99-10
- Loaded via a namespace (and not attached): tools 2.15.0