Working with R

A University of Queensland Advanced Workshop

Session 5: Elementary Linear Models

Bill Venables, CSIRO/Data61, Dutton Park

Rhetta Chappell, Griffith University

2–5 February, 2021

Contents

1 A simple example: the Janka Hardness data

Example taken from E. J. Williams (1959) *Regression Analysis*.

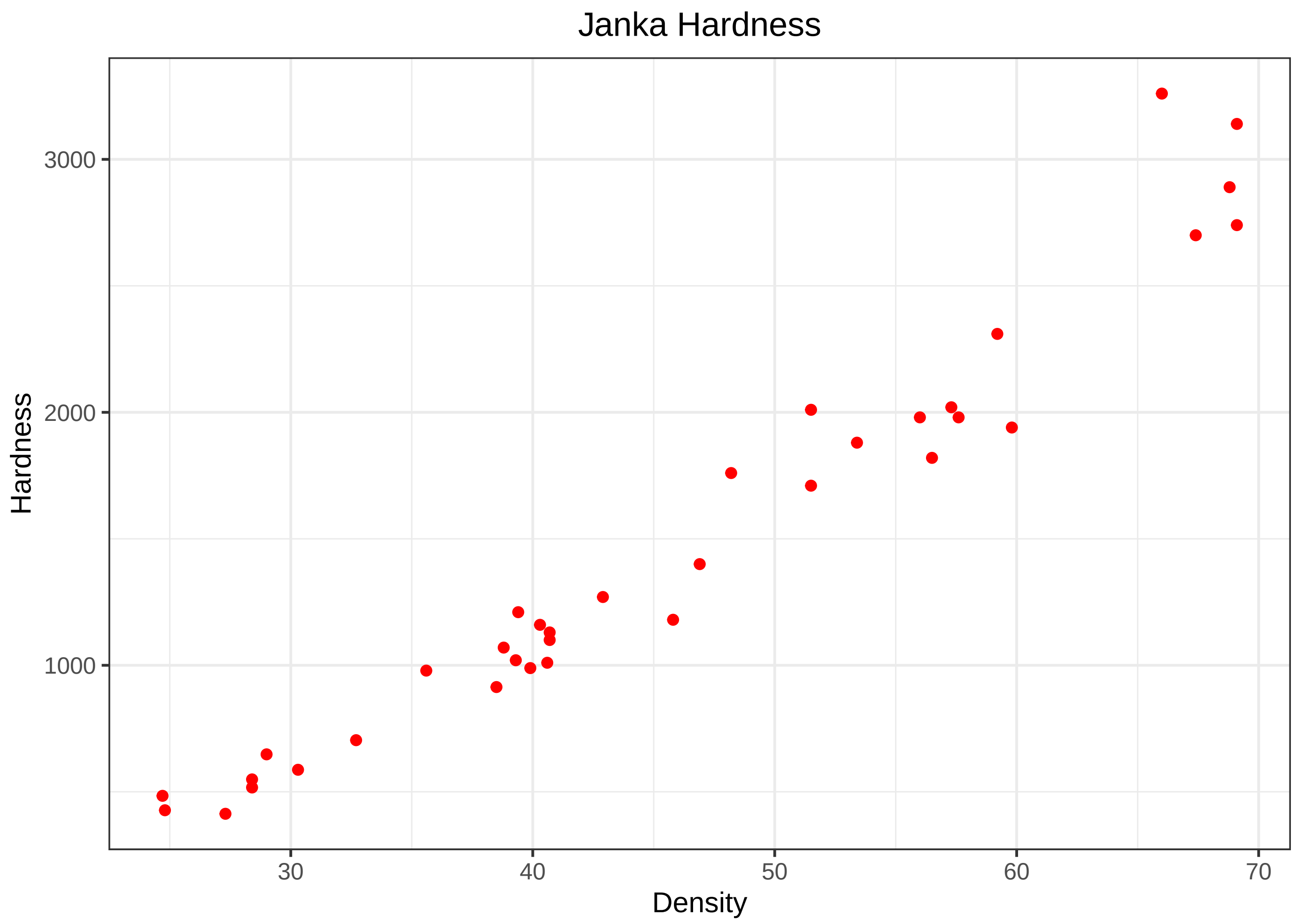
Two variables:

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 Hardness from Density.*

```r
theme_set(theme_bw() + theme(plot.title = element_text(hjust=0.5)))

ggplot(janka) + aes(x = Density, y = Hardness) +
  geom_point(colour = "red") +  labs(title = "Janka Hardness")
```



- 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:

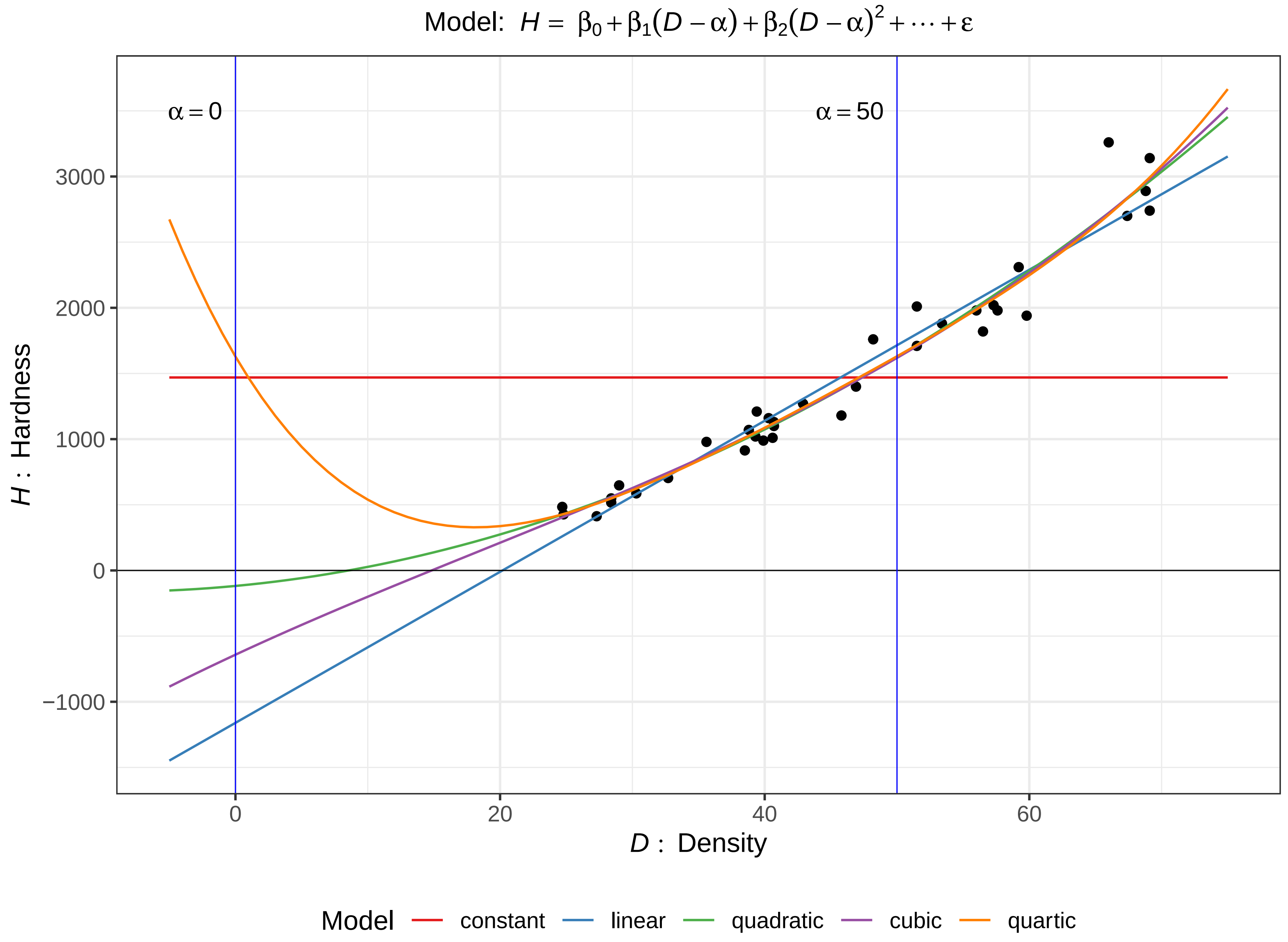
```
m1a <- lm(Hardness ~ I(Density - 50), janka)
m2a <- update(m1a, . ~ . + I((Density - 50)^2))
m3a <- update(m2a, . ~ . + I((Density - 50)^3))
round(summary(m3a)$coef, 4)

                      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

The explanation in graphical terms

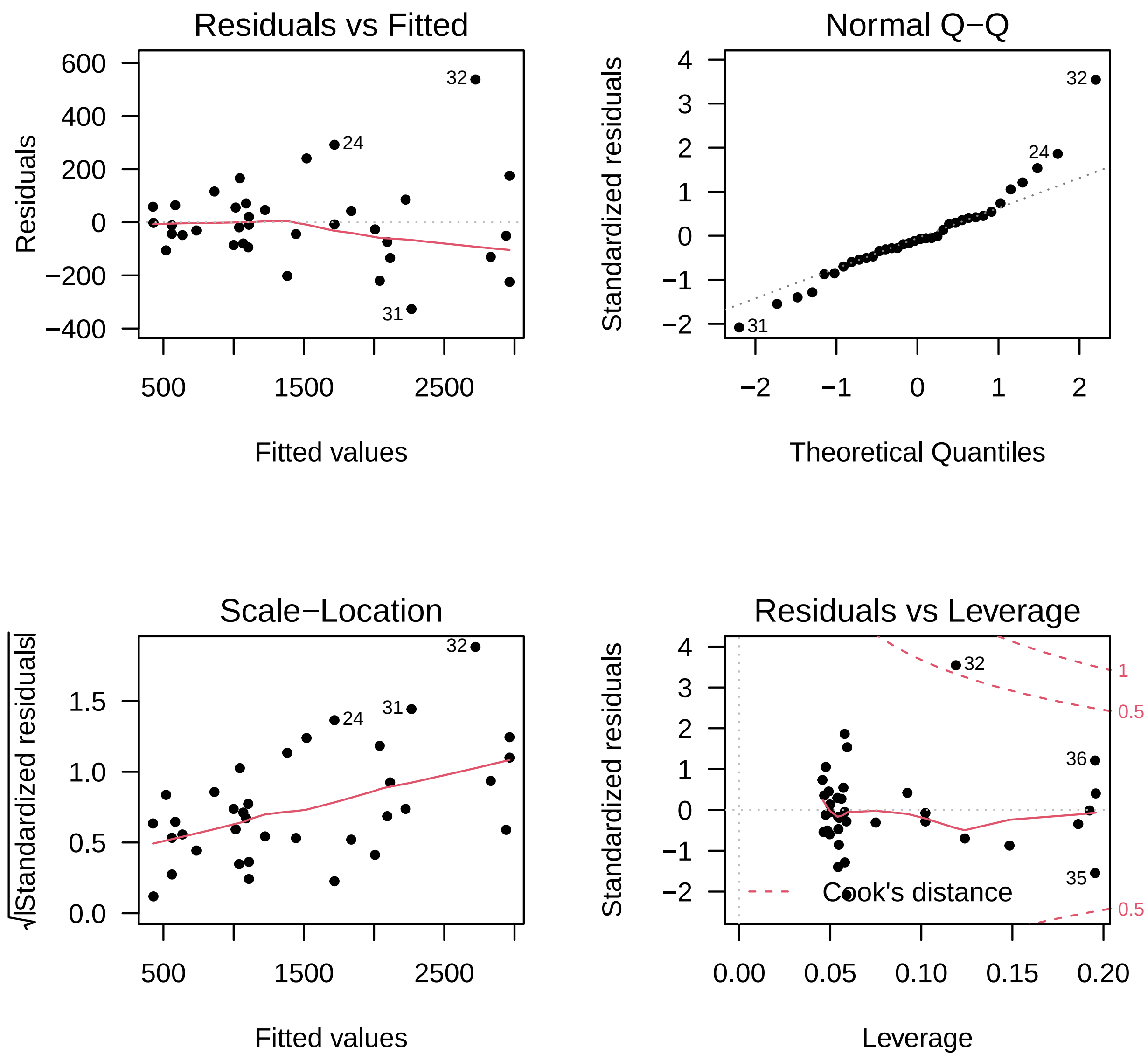
```r
m0a <- lm(Hardness ~ 1, janka)  ## constant predictor model
m4a <- update(m3a, .~.+I((Density-50)^4))

pJanka <- data.frame(Density = -5:75)
pJanka <- within(pJanka, {
  constant  <- predict(m0a, pJanka)
  linear    <- predict(m1a, pJanka)
  quadratic <- predict(m2a, pJanka)
  cubic     <- predict(m3a, pJanka)
  quartic   <- predict(m4a, pJanka)
})

pJankaLong <- pJanka %>%
  pivot_longer(names_to="Model", values_to="Hardness", constant:quartic) %>%
  mutate(Model = factor(Model, levels = cs(constant, linear, quadratic,
                                           cubic, quartic)))

titlex <- expression("Model:"*'  '*italic(H) ==' '*italic(beta[0] +
    beta[1]*(D - alpha) + beta[2]*(D - alpha)^2 + cdots + epsilon))
```

```r
p <- ggplot(janka) + aes(x = Density, y = Hardness) + geom_point()  +
  geom_line(aes(colour = Model), data = pJankaLong, size = 0.5) +
  geom_hline(yintercept = 0, linetype = "solid", size = 0.25) +
  geom_vline(xintercept = c(0,50), linetype = "solid",
             size = 0.25, colour = "blue") +
  annotate("text", x = c(0,50)-1, y = 3500, hjust = 1,
           label = paste("alpha ==", c(0,50)), parse = TRUE,
           size = 4) +
  labs(x = expression(italic(D):' Density'),
       y = expression(italic(H):' Hardness'),
       title = titlex) +
  theme(text = element_text(size = 12), legend.position = "bottom",
        plot.title = element_text(hjust = 0.5, vjust = 1, size = 12)) +
  scale_colour_brewer(palette = "Set1")
p
```


Model: $H = \beta_0 + \beta_1(D-\alpha) + \beta_2(D-\alpha)^2 + \cdots + \varepsilon$
$\alpha = 0$
$\alpha = 50$
H : Hardness
D : Density
3000
2000
1000
0
−1000
0
20
40
60
Model
constant
linear
quadratic
cubic
quartic

1.2 Diagnostics and corrective action

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



Residuals vs Fitted
Residuals
Fitted values
600
400
200
0
−200
−400
500
1500
2500
32
24
31
Normal Q−Q
Standardized residuals
Theoretical Quantiles
4
3
2
1
0
−1
−2
32
24
31
Scale−Location
√|Standardized residuals|
Fitted values
1.5
1.0
0.5
0.0
32
31
24
Residuals vs Leverage
Standardized residuals
Leverage
0.00
0.05
0.10
0.15
0.20
32
36
35
Cook's distance
1
0.5
0.5

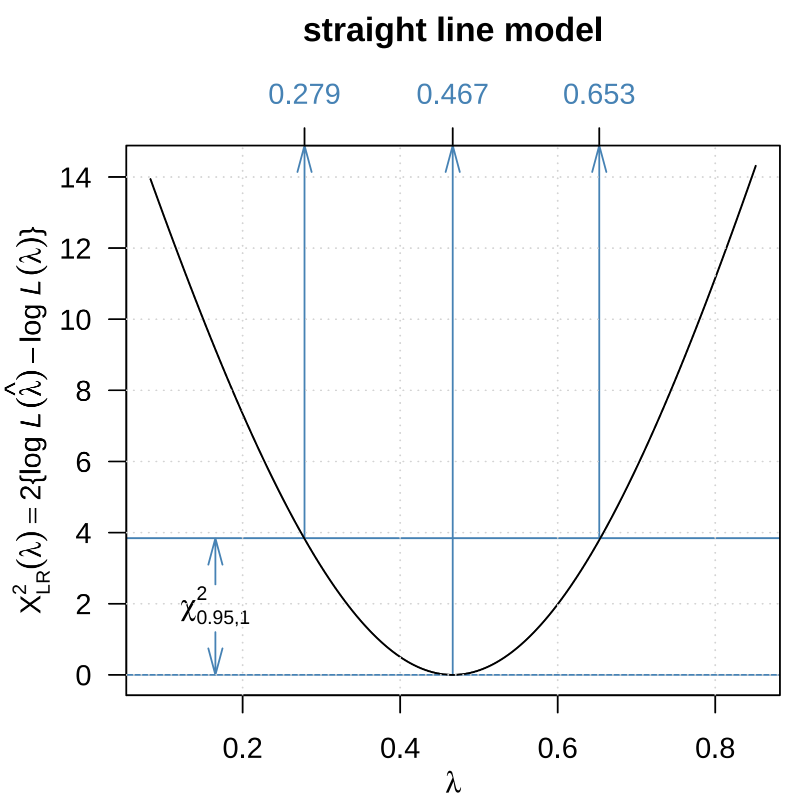
1.3 A transformation?

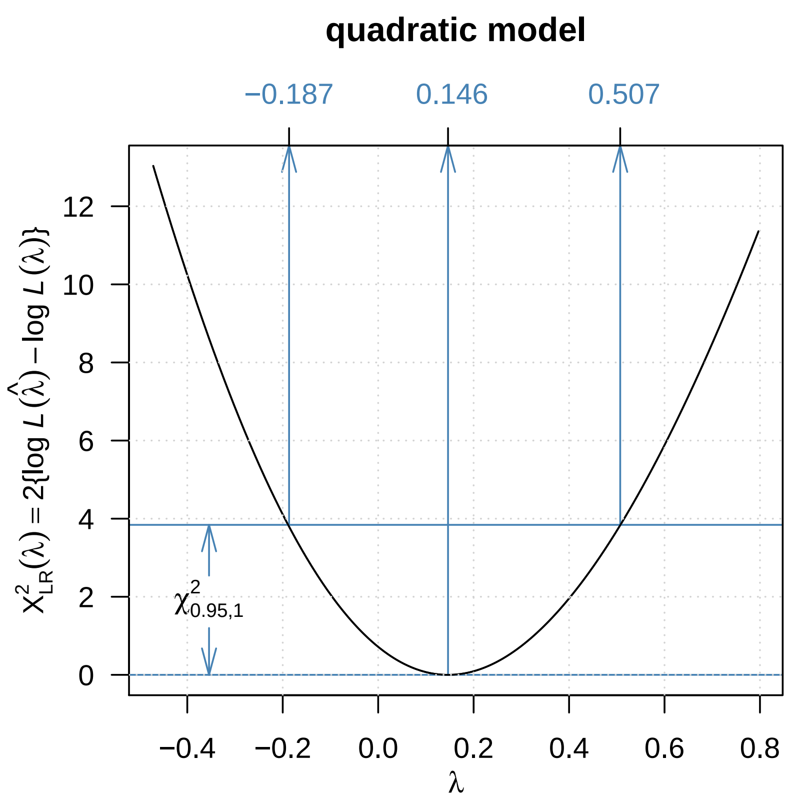
The main purpose of a transformation is to provide a scale in which the response is homoscedastic — but no necessarily the only purpose.

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.

```r
layout(matrix(1:2, 1))
box_cox(m1a)
title(main = "straight line model", line = 3)
box_cox(m2a)
title(main = "quadratic model", line = 3)
```


straight line model
0.279
0.467
0.653
$X^2_{LR}(\lambda) = 2\{\log L(\hat{\lambda}) - \log L(\lambda)\}$
$\chi^2_{0.95,1}$
0
2
4
6
8
10
12
14
0.2
0.4
0.6
0.8
λ

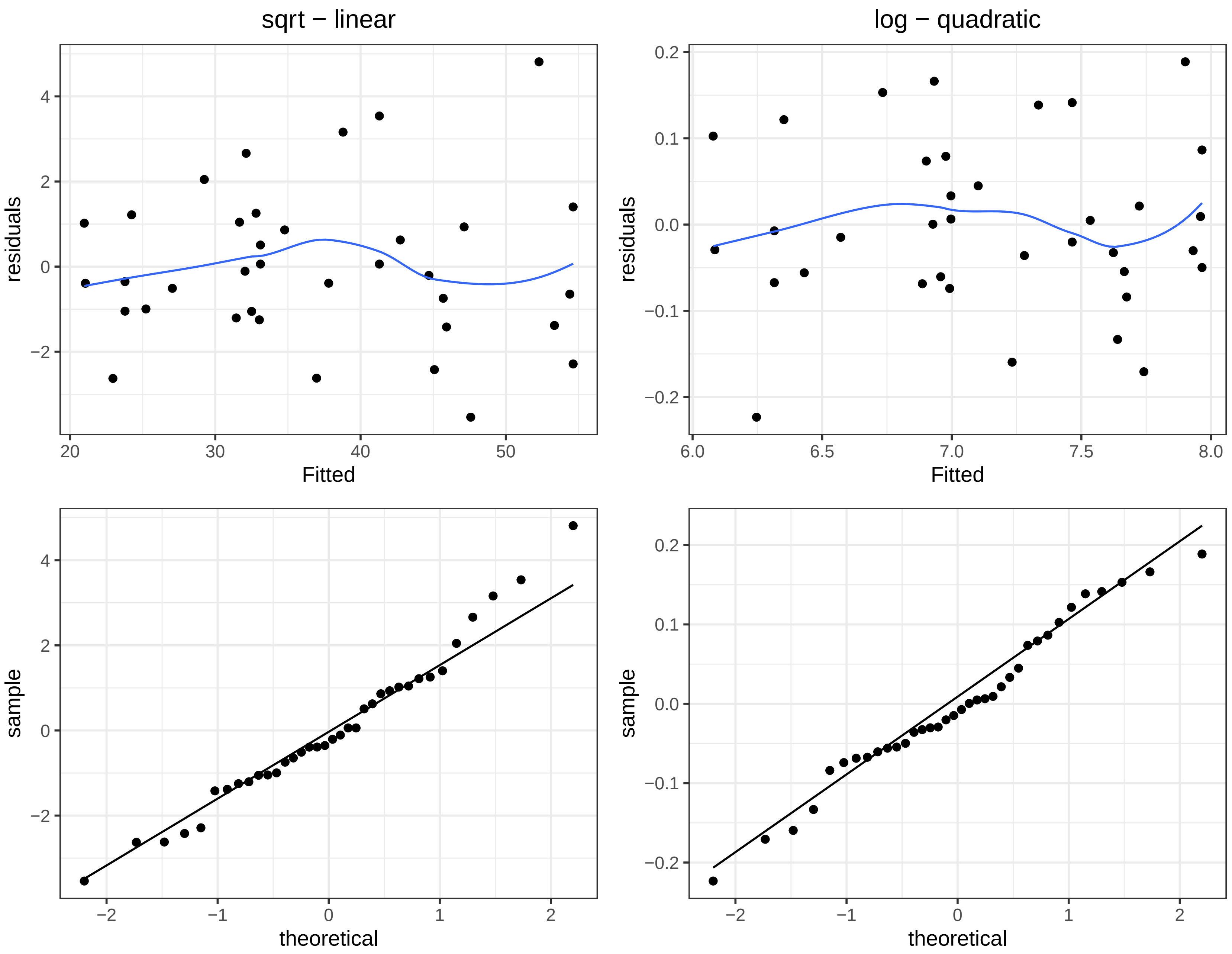

quadratic model
−0.187
0.146
0.507
$X^2_{LR}(\lambda) = 2\{\log L(\hat{\lambda}) - \log L(\lambda)\}$
$\chi^2_{0.95,1}$
0
2
4
6
8
10
12
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
λ

Assuming a straight line — a square root transformation?

Allowing for a quadratic response — a log transformation?

Consider the effect of the transformation on residuals:

```r
m1 <- lm(sqrt(Hardness) ~ poly(Density, 1), janka)
m2 <- lm(log(Hardness) ~ poly(Density, 2), janka)
Janka <- within(janka, {
  r1 <- resid(m1);   r2 <- resid(m2)
  f1 <- fitted(m1);  f2 <- fitted(m2)
})
p0 <- ggplot(Janka)
p1 <- p0 + aes(x = f1, y = r1) + geom_point() +
  labs(x = "Fitted", y = "residuals", title = "sqrt - linear") +
  geom_smooth(se = FALSE, method = "loess", size = 0.5,formula = y~x)
p2 <- p0 + aes(x = f2, y = r2) + geom_point() +
  labs(x = "Fitted", y = "residuals", title = "log - quadratic") +
  geom_smooth(se = FALSE, method = "loess", size = 0.5, formula = y~x)
p3 <- p0 + aes(sample = r1) + stat_qq() + stat_qq_line()
p4 <- p0 + aes(sample = r2) + stat_qq() + stat_qq_line()

(p1 + p2)/(p3 + p4)
```


sqrt − linear
log − quadratic
residuals
Fitted
sample
theoretical

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.

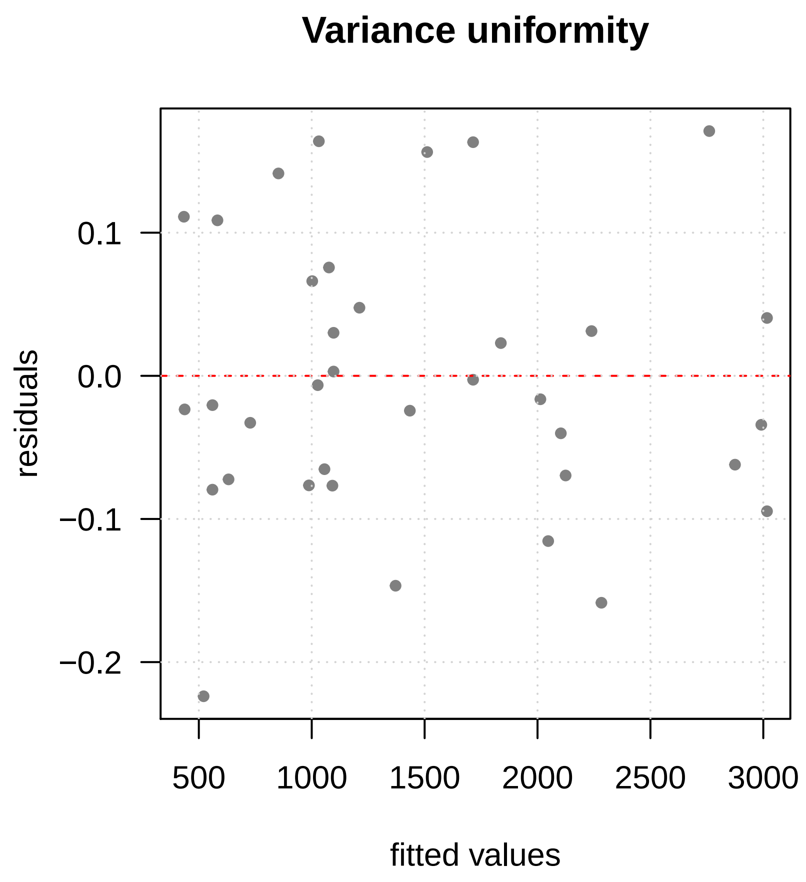
Within the GLM family this suggests a Gamma model. The $sqrt$ link is non-standard, which *used* to be a problem, but no longer.

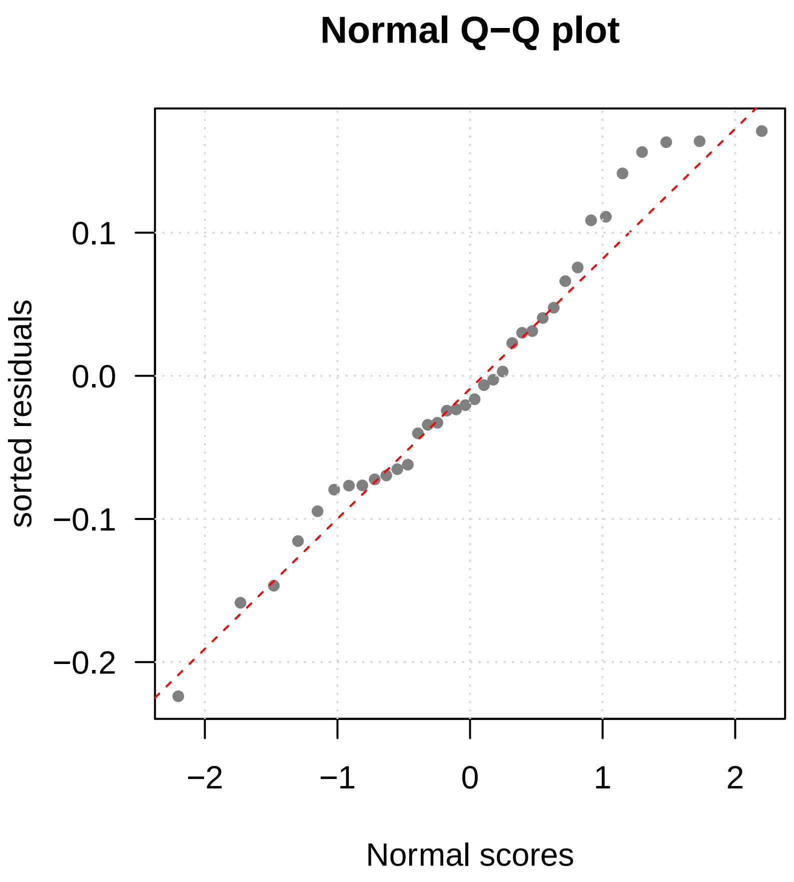
```r
mGLM <- glm(Hardness ~ I(Density-50),
             family = Gamma(link = "sqrt"),
             data = janka)
mGLM2 <- update(mGLM, .~.+I((Density-50)^2))
round(summary(mGLM2)$coef, 4)
```

```
                    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 straight line model seems adequate - residual checks:

```r
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"); grid()

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"); grid()
```


Variance uniformity
residuals
fitted values
0.1
0.0
−0.1
−0.2
500
1000
1500
2000
2500
3000


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

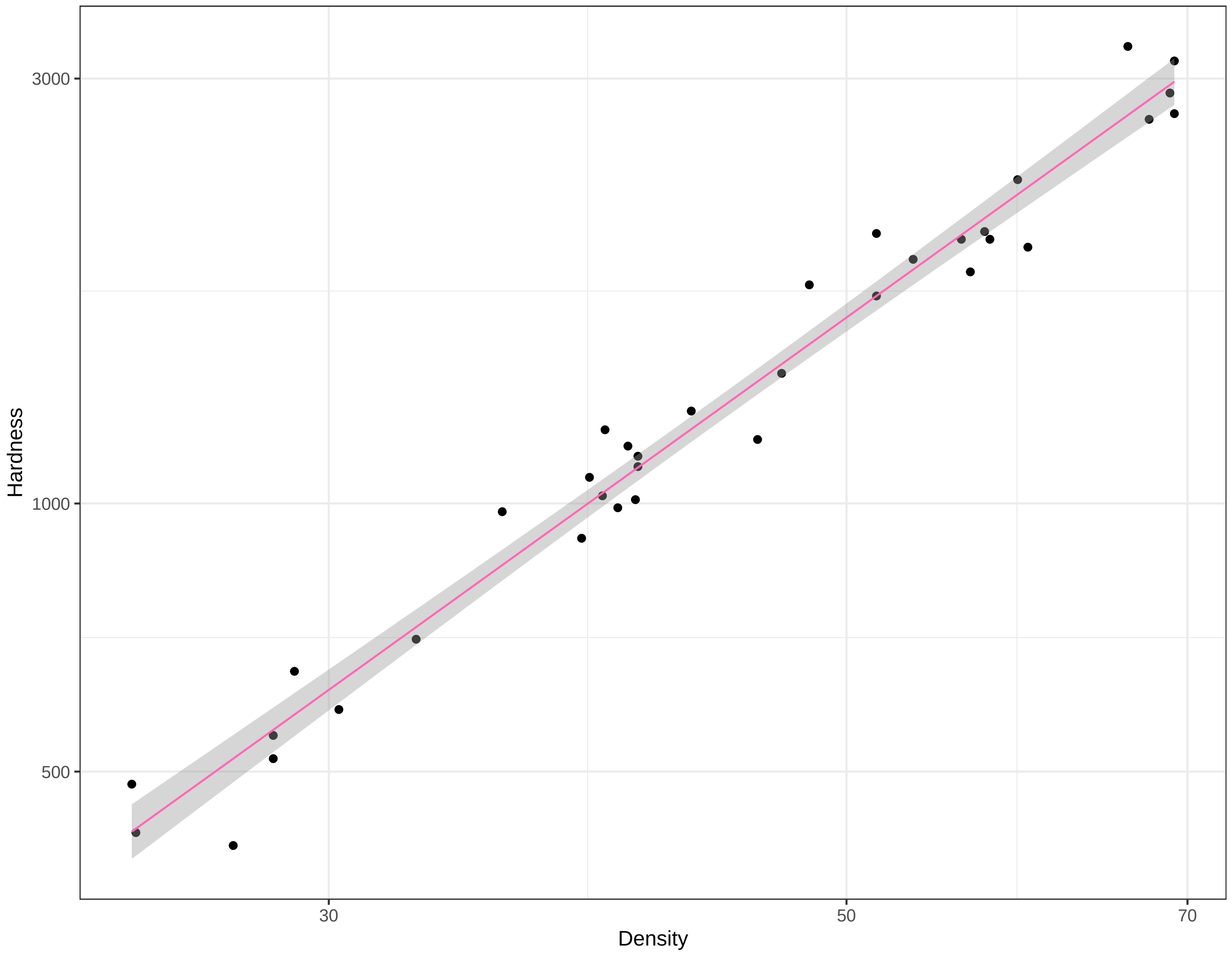
1.4 A simpler approach

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

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

```
ggplot(janka) + aes(x = Density, y = Hardness) +
  geom_point() + scale_x_log10() + scale_y_log10() +
  stat_smooth(method = "lm", size = 0.5, colour = "hot pink", formula = y~x)
```

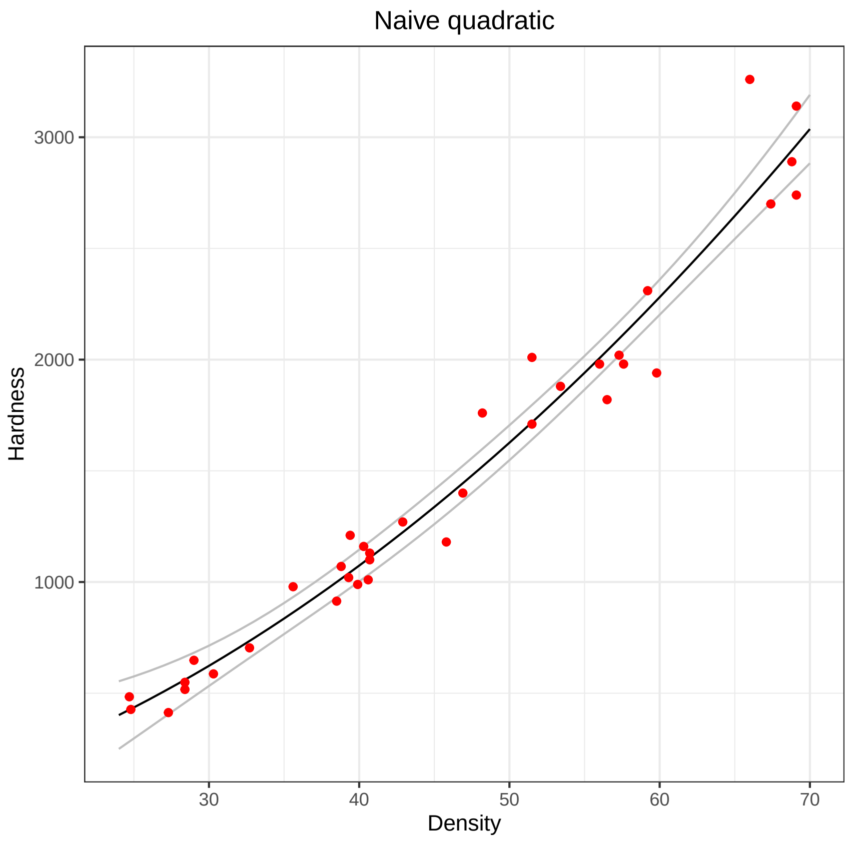
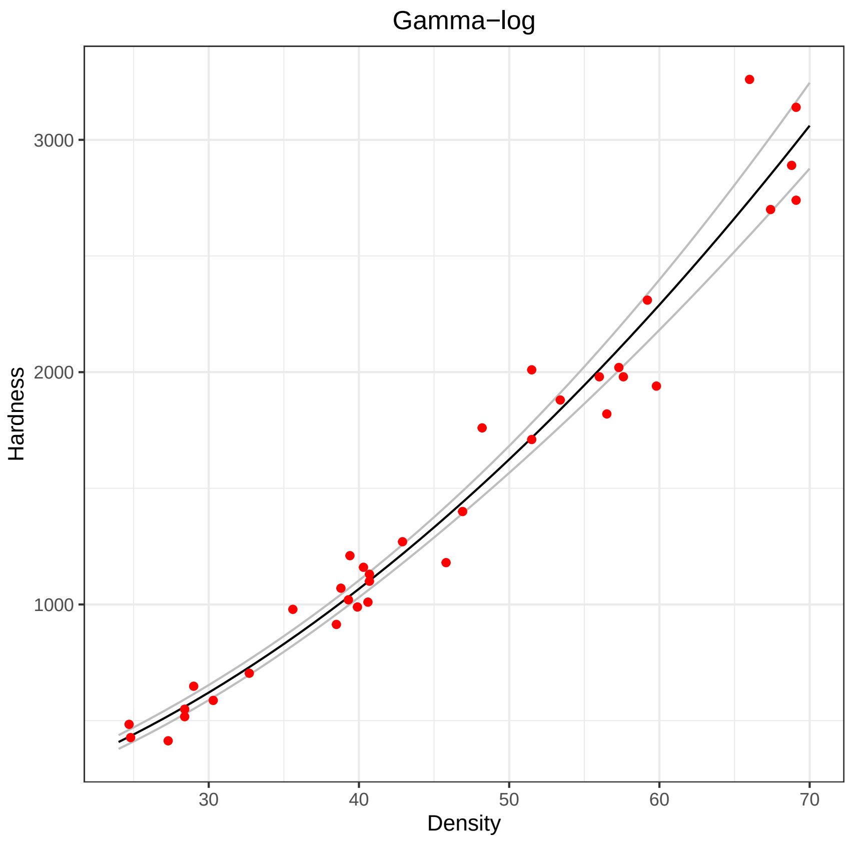
- This suggests a Gamma model with a *log* link and a log-transformed predictor, linear term only. So it proves to be.
- Using a GLM, leaving the response untransformed, allows us to predict on the natural scale, thus avoiding the complications associated with back transforming predictions.


3000
1000
500
Hardness
30
50
70
Density

Compare the final model with a simple quadratic regression, ignoring variance heterogeneity.

```r
jankaLM <- lm(Hardness ~ poly(Density, 2), janka)
pJanka <- data.frame(Density = 24:70)
ext <- predict(jankaLM, pJanka, type = "resp", se.fit = TRUE) %>%
  as.data.frame() %>%
  within({
    lower <- fit - 2*se.fit
    upper <- fit + 2*se.fit
  })
pJanka <- cbind(pJanka, ext)
pLM <- ggplot(pJanka) + aes(x = Density) +
  geom_line(aes(y = fit), colour = "black") +
  geom_line(aes(y = lower), colour = "grey") +
  geom_line(aes(y = upper), colour = "grey") +
  geom_point(data = janka, aes(y = Hardness), colour = "red") +
  xlab("Density") + ylab("Hardness") + labs(title = "Naive quadratic")
```

```r
jankaGLM <- glm(Hardness ~ log(Density), Gamma(link = log), janka)
pJanka <- data.frame(Density = 24:70)
ext <- predict(jankaGLM, pJanka, type = "resp", se.fit = TRUE) %>%
  as.data.frame() %>%
  within({
    lower <- fit - 2*se.fit
    upper <- fit + 2*se.fit
  })
pJanka <- cbind(pJanka, ext)

pGLM <- ggplot(pJanka) + aes(x = Density) +
  geom_line(aes(y = fit), colour = "black") +
  geom_line(aes(y = lower), colour = "grey") +
  geom_line(aes(y = upper), colour = "grey") +
  geom_point(data = janka, aes(y = Hardness), colour = "red") +
  xlab("Density") + ylab("Hardness") + labs(title = "Gamma-log")
#
# gridExtra::grid.arrange(pLM, pGLM, nrow = 1)
pLM + pGLM
```


Naive quadratic
Hardness
Density
3000
2000
1000
30
40
50
60
70

Gamma−log
Hardness
Density
3000
2000
1000
30
40
50
60
70

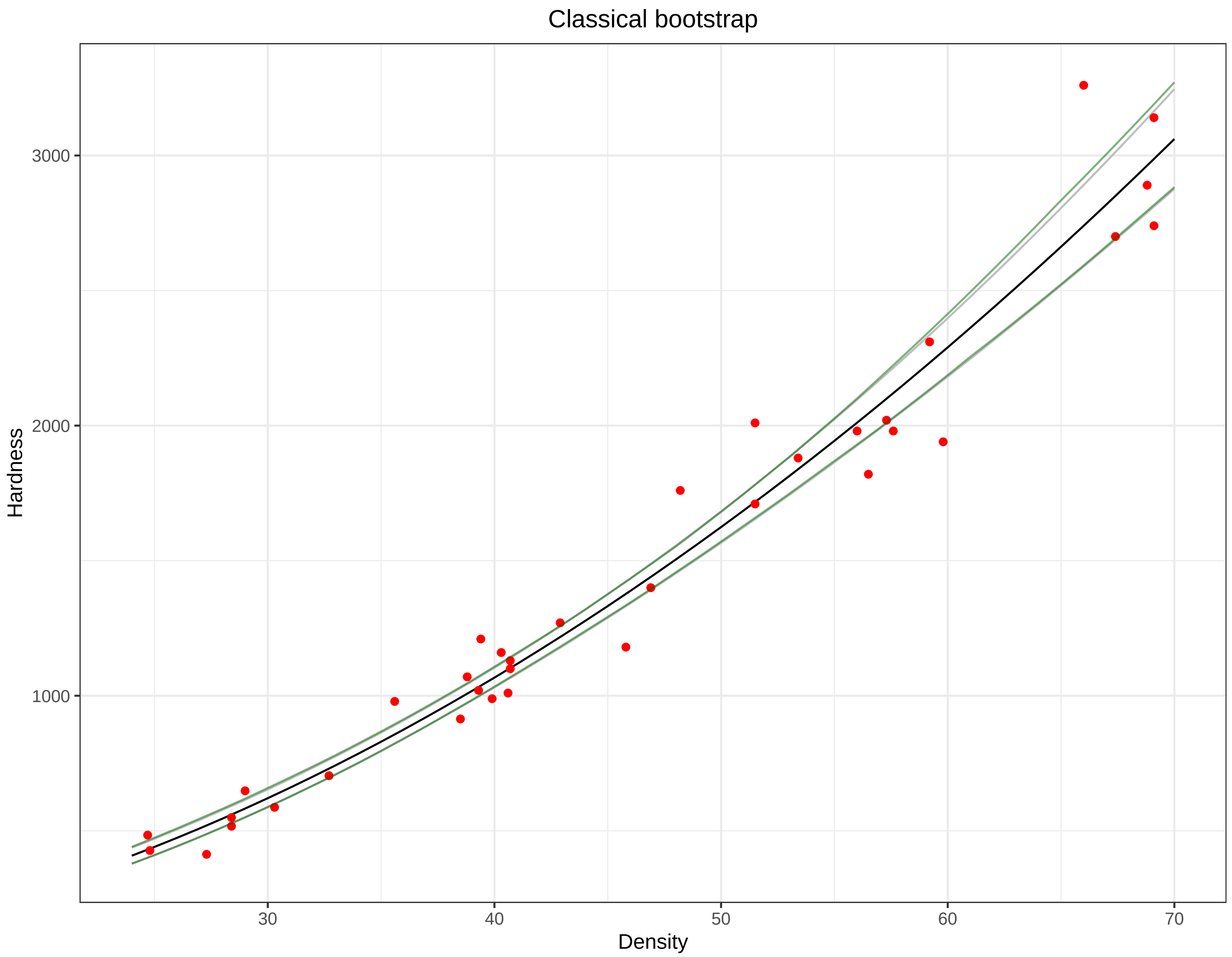
1.5 Bootstrap confidence intervals for the mean

1.5.1 Classical version

The idea is that we predict the mean for a large number of *bootstrap samples* of the original model, re-fitting the model as if the bootstrap sample were the real data. The quantiles of the bootstrap predictions provide the appropriate confidence interval.

The code is simple and explains the method more precisely.

```r
set.seed(20210202)
boot_sample <- function(data) data[sample(nrow(data), replace = TRUE), ]

ci <- replicate(500, {
  tmp <- update(jankaGLM, data = boot_sample(janka))  ## bootstrap data
  predict(tmp, pJanka, type = "resp")                 ## predictions for target
}) %>%
  apply(1, quantile, prob = c(0.025, 0.975))

pJanka <- pJanka %>%
  within({
    lowerBB <- ci[1, ]
    upperBB <- ci[2, ]
  })

greenish <- alpha("dark green", 0.5)
pGLM + labs(title = "Classical bootstrap") +
  geom_line(data = pJanka, aes(x = Density, y = lowerBB),
            colour = greenish, size = 0.5) +
  geom_line(data = pJanka, aes(x = Density, y = upperBB),
            colour = greenish, size = 0.5)
```


Classical bootstrap
Hardness
Density
3000
2000
1000
30
40
50
60
70

1.5.2 Bayesian version

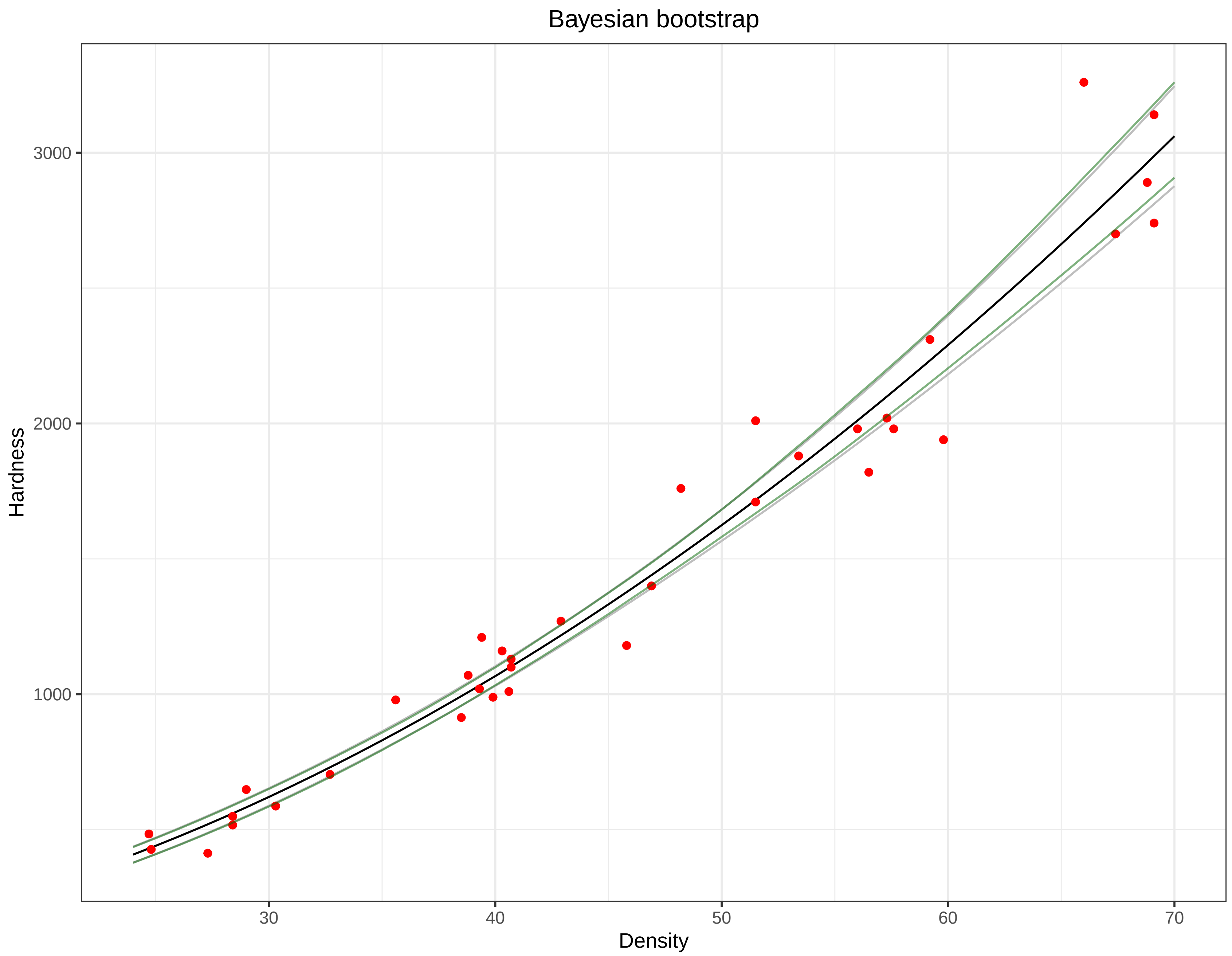
An alternative to the classical bootsrtap is the Bayesian Bootstrap idea of Rubin (1981).

- 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.)

```
X <- matrix(sample(100, size = 1000*100, replace = TRUE), ncol = 100) %>%
  apply(1, tabulate, nbins = 100)
CB <- c(mean     = mean(colMeans(X)),
        variance = mean(colMeans((X-1)^2)))
W <- rexp(10000)
BB <- c(mean = mean(W), variance = var(W))
rbind(Classical = CB, Bayesian = BB)

               mean variance
Classical 1.0000000 0.991460
Bayesian  0.9958835 1.005921
```

The code is nearly identical to the classical case.

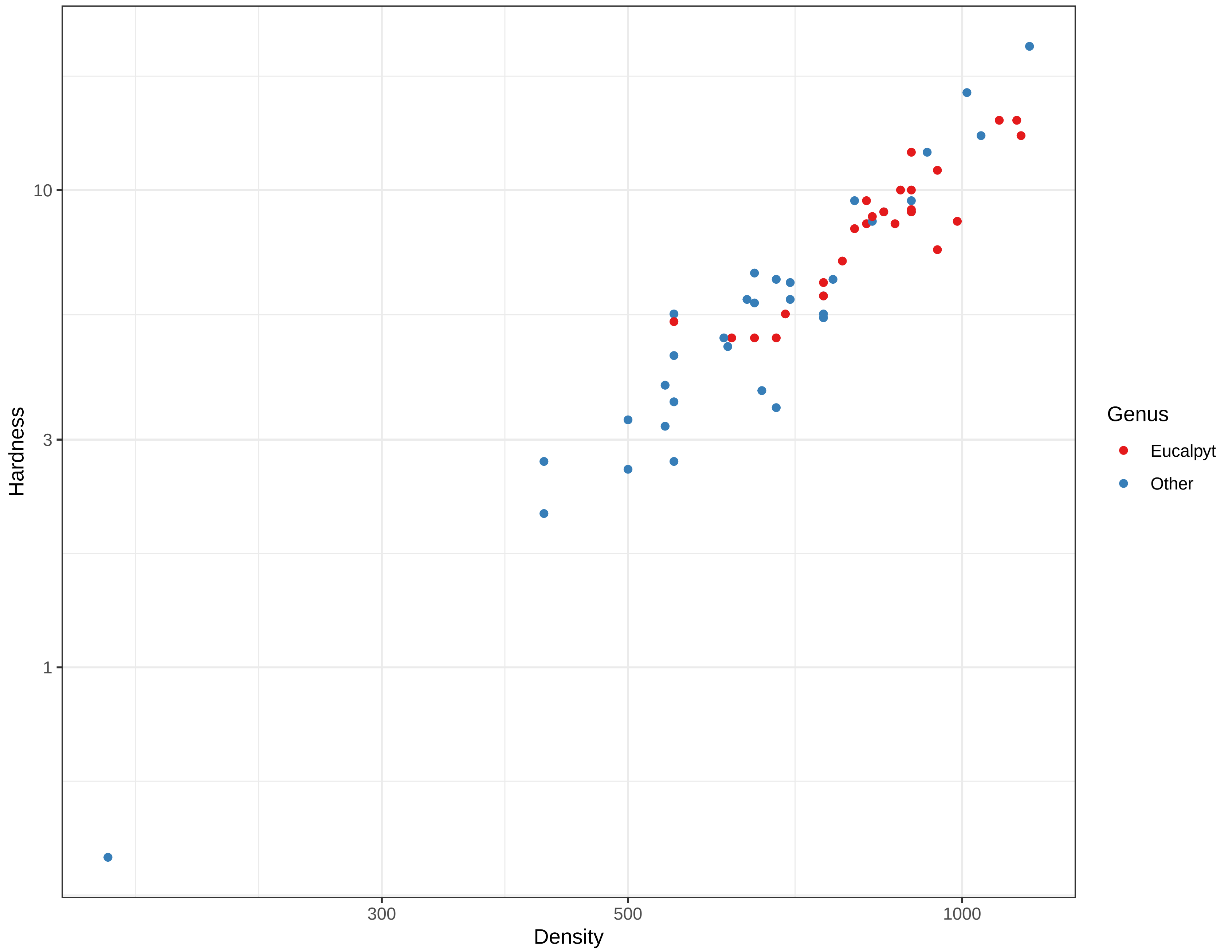
```r
set.seed(20210202)
Norm <- function(x) x/mean(x)

ci <- replicate(500, {
  tmp <- update(jankaGLM, weights = Norm(rexp(nrow(janka))))
  predict(tmp, pJanka, type = "resp")
}) %>%
  apply(1, quantile, prob = c(0.025, 0.975))

pJanka <- pJanka %>%
  within({
    lowerBB <- ci[1, ]
    upperBB <- ci[2, ]
  })

greenish <- alpha("dark green", 0.5)
pGLM + labs(title = "Bayesian bootstrap") +
  geom_line(data = pJanka, aes(x = Density, y = lowerBB),
            colour = greenish, size = 0.5) +
  geom_line(data = pJanka, aes(x = Density, y = upperBB),
            colour = greenish, size = 0.5)
```

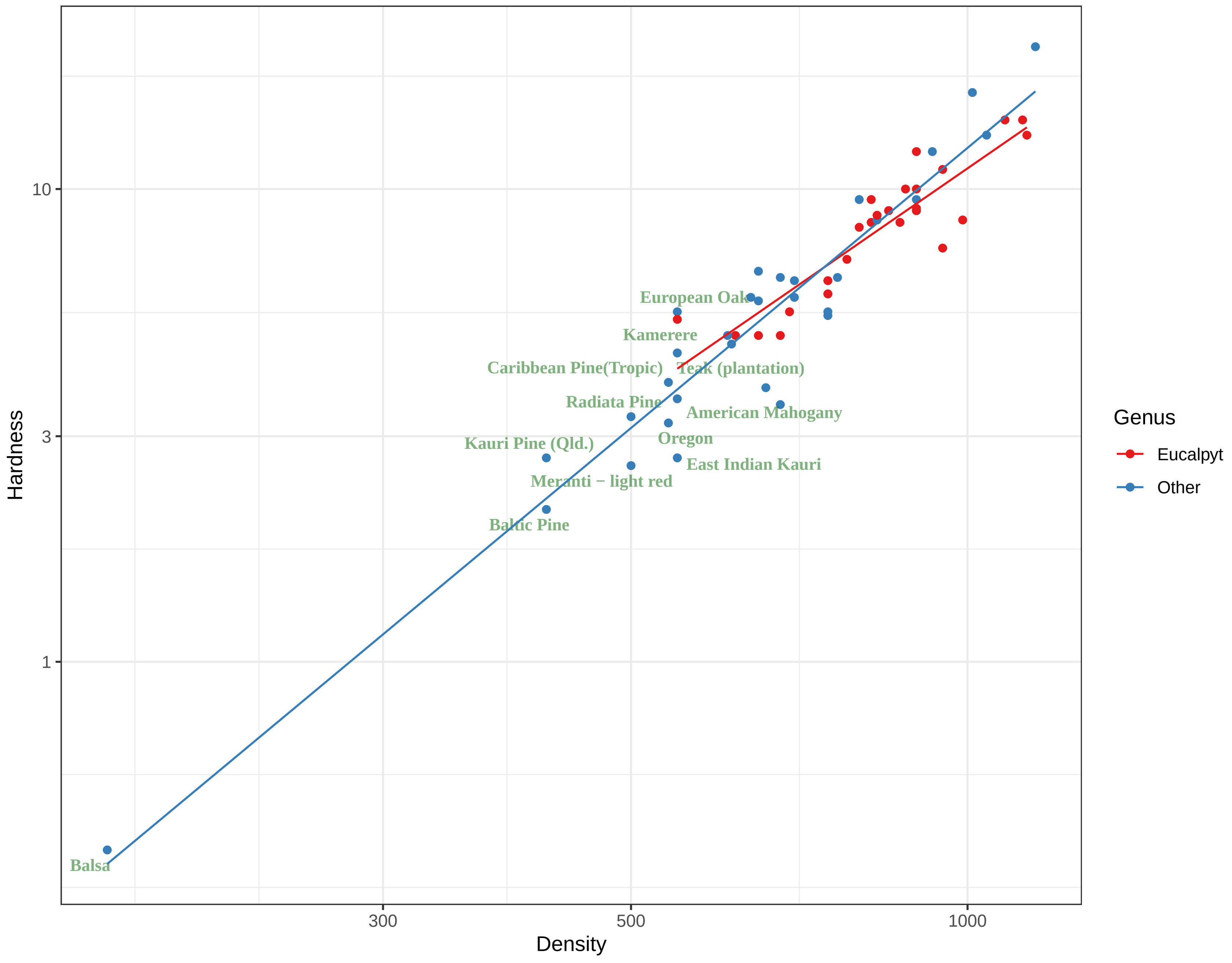

Bayesian bootstrap
Hardness
Density
3000
2000
1000
30
40
50
60
70

1.6 Some more recent data

These date from 2012. They were obtained from the internet, but the units were not given.

```r
Janka <- within(Janka2012, {
  Type <- ifelse(grepl("Eucalyptus", Binomial), "Eucalpyt", "Other")
})

jp <- ggplot(Janka) + aes(x = Density, y = Hardness) + geom_point() +
  aes(colour = Type) + # coord_trans("log10", "log10") +
  scale_x_log10() + scale_y_log10() +
  scale_colour_brewer(palette = "Set1", name = "Genus")
jp
jp  +  ## identify the light species
  ggrepel::geom_text_repel(data = filter(Janka, Density < 600),
                           aes(label = Name), size = 3,
                           colour = greenish,  ## some transparency
                           fontface = "bold", family = "serif") +
  geom_smooth(method = "lm", se = FALSE, formula = y ~ x, size = 0.5)
```


10
3
1
300
500
1000
Density
Hardness
Genus
Eucalpyt
Other



Hardness
Density
10
3
1
300
500
1000
Genus
Eucalpyt
Other
European Oak
Kamerere
Caribbean Pine(Tropic)
Teak (plantation)
Radiata Pine
American Mahogany
Oregon
Kauri Pine (Qld.)
East Indian Kauri
Meranti − light red
Baltic Pine
Balsa

Comparing model estimates:

```r
oldJanka <- glm(Hardness~log(Density), Gamma(link="log"), janka)
newJanka <- update(oldJanka, data = Janka)
format(cbind(old = coef(oldJanka),
             recent = coef(newJanka)),
       digits = 5) %>% booktabs()
```

	old	recent
(Intercept)	0.026034	-10.605491
log(Density)	1.883123	1.893879

The discrepancy in the intercept coefficients is due to a mismatch of units. The coefficient of log(Density) is unit free.

References

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

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

Session information

Date: 2021-01-29

- R version 4.0.3 (2020-10-10), `x86_64-pc-linux-gnu`
- Running under: `Ubuntu 20.04.1 LTS`
- Matrix products: default
- BLAS: `/usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0`
- LAPACK: `/usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0`
- Base packages: base, datasets, graphics, grDevices, methods, stats, utils
- Other packages: dplyr 1.0.3, english 1.2-5, forcats 0.5.1, ggplot2 3.3.3, ggthemes 4.2.4, gridExtra 2.3, knitr 1.31, lattice 0.20-41, patchwork 1.1.1, purrr 0.3.4, readr 1.4.0, scales 1.1.1, stringr 1.4.0, tibble 3.0.5, tidyr 1.1.2, tidyverse 1.3.0, WWRCourse 0.2.3, WWRData 0.1.0, WWRGraphics 0.1.2, WWRUtilities 0.1.2, xtable 1.8-4
- Loaded via a namespace (and not attached): assertthat 0.2.1, backports 1.2.1, broom 0.7.3, cellranger 1.1.0, cli 2.2.0, colorspace 2.0-0, compiler 4.0.3, crayon 1.3.4, DBI 1.1.1, dbplyr 2.0.0, digest 0.6.27, ellipsis 0.3.1, evaluate 0.14, fansi 0.4.2, farver 2.0.3, fractional 0.1.3, fs 1.5.0, generics 0.1.0, ggrepel 0.9.1, glue 1.4.2, grid 4.0.3, gtable 0.3.0, haven 2.3.1, highr 0.8, hms 1.0.0, httr 1.4.2, iterators 1.0.13, jsonlite 1.7.2, labeling 0.4.2, lazyData 1.1.0, lifecycle 0.2.0, lubridate 1.7.9.2, magrittr 2.0.1, MASS 7.3-53, Matrix 1.3-2, mgcv 1.8-33, modelr 0.1.8, munsell 0.5.0, nlme 3.1-151, parallel 4.0.3, PBSmapping 2.73.0, pillar 1.4.7, pkgconfig 2.0.3, R6 2.5.0, randomForest 4.6-14, RColorBrewer 1.1-2, Rcpp 1.0.6, readxl 1.3.1, reprex 1.0.0, rlang 0.4.10, rpart 4.1-15, rstudioapi 0.13, rvest 0.3.6, SOAR 0.99-11, splines 4.0.3, stringi 1.5.3, tidyselect 1.1.0, tools 4.0.3, vctrs 0.3.6, withr 2.4.1, xfun 0.20, xml2 1.3.2