1 A non-linear regression: Hankinson's equation

This example comes from Williams (1959), Example 2.3, page 20, (with adaptations).[1]

Hankinson's formula relates the modulus of rupture, f_θ, for a timber plank to the angle of stress, θ:

$$f_\theta = \frac{f_0 f_{\pi/2}}{f_{\pi/2}\cos^2\theta + f_0 \sin^2\theta}$$

The formula may be "linearized" by taking reciprocals:

$$\frac{1}{f_\theta} = \frac{\cos^2\theta}{f_0} + \frac{\sin^2\theta}{f_{\pi/2}}$$

but the error structure implied by fitting a simple regression model to this formula may be unrealistic. It may be enough to provide a starting estimate, though.

A more realistic model might be to assume a multiplicative error structure, that is, errors in the measurement of f_θ tend to affect the result *proportionately* rather than additively:

$$Y = \frac{\alpha_1\alpha_2}{\alpha_2\cos^2\theta + \alpha_1\sin^2\theta} e^{\varepsilon}, \quad \varepsilon \sim \mathrm{N}(0, \sigma^2)$$

Taking logs converts the error structure to additive, but the parametric form for the regression model remains irreducibly non-linear:

$$\log Y = \log\alpha_1 + \log\alpha_2 - \log\left(\alpha_2\cos^2\theta + \alpha_1\sin^2\theta\right) + \varepsilon, \quad \varepsilon \sim \mathrm{N}(0, \sigma^2)$$

Data from a calibration experiement are given in Table 1, and are available as the data frame `hankinson`.

	θ°	f_θ
01	0.0	16880
02	2.5	14720
03	5.0	14340
04	7.5	12740
05	10.0	12390
06	15.0	7140
07	20.0	7170
08	30.0	4710
09	45.0	2280
10	60.0	1720
11	90.0	970

Table 1: Modulus of rupture data for calibration of Hankinson's formula

[1] For some background information see here, and links therein.

```
> hankinson <- within(read.csv("hankinson.csv"), {
    theta <- thetaDeg * base::pi/180
    MRupture <- MRupture/2240  ## change lbs to tons
  })
> StartM <- lm(1/MRupture ~ 0+I(cos(theta)^2) + I(sin(theta)^2), hankinson)
> (f <- structure(1/coef(StartM), names = c("a1", "a2")))
```

```
       a1        a2 
9.2648299 0.4892273 
```

We can now fit a non-linear model using these as starting values for the parameters:

```
> NonlinM <- nls(log(MRupture) ~ log(a1) + log(a2) -
                 log(a1*sin(theta)^2 + a2*cos(theta)^2), hankinson,
                 start = f, trace = TRUE)
> summary(NonlinM)
```

The main output is shown in Table 2.

	Estimate	Std. Error	t value	Pr(>|t|)
a1	7.0561	0.45	15.59	0.0000
a2	0.5396	0.04	15.32	0.0000

Table 2: Coefficient estimates from the non-linear regression

A generalization of the model allows the power to which the trigonometric terms are raised to be a third parameter, though such refinements are rarely necessary. To check this in the present case, we fit the extended model and test the simpler model as a sub-model of it.

```
> g <- c(coef(NonlinM), p = 0)
> NonlinMg <- nls(log(MRupture) ~ log(a1) + log(a2) -
                 log(a1*sin(theta)^(2+p) + a2*cos(theta)^(2+p)), hankinson,
                 start = g)
> summary(NonlinMg)
> anova(NonlinM, NonlinMg)
```

The main `summary` results are shown in Table 3 and the analysis of variance results in Table 4 on the next page.

	Estimate	Std. Error	t value	Pr(>|t|)
a1	6.8181	0.53	12.98	0.0000
a2	0.5158	0.05	10.78	0.0000
p	0.1237	0.17	0.72	0.4904

Table 3: Coefficient estimates from the extended non-linear regression

Now consider predictions from the standard model. First we set up a data frame with the angle θ on a one degree scale. We use the angle in degrees for plotting and the angle in radians for computations.

	Res.Df	Res.Sum Sq	Df	Sum Sq	F value	Pr(>F)
1	9	0.1503				
2	8	0.1417	1	0.00860	0.4858	0.5056

Table 4: Analysis of variance table testing the standard non-linear model, (1), within the extended model, (2)

```
> pHankinson <- within(data.frame(thetaDeg = 0:90),
                       theta <- thetaDeg*base::pi/180)
> pHankinson$MRupture <- exp(predict(NonlinM, pHankinson))
```

Note that simply back-transforming the predictions from the log scale gives an estimate of the *median* modulus of rupture on the natural scale, not the mean.

For standard errors we consider two forms of bootstrap estimates, first the so-called "Bayesian bootstrap", which re-fits the model using random exponential weights. These are chosen so that the weights have both mean and variance unity:

```
> nr <- nrow(hankinson)
> f <- coef(NonlinM)  ### update f to the LS estimates; names are OK
> Z <- replicate(500, {
    tmpM <- update(NonlinM, weights = rexp(nr), trace = FALSE)
    exp(predict(tmpM, newdata = pHankinson))
  })
> lims <- apply(Z, 1, quantile, probs = c(1, 39)/40)
> pHankinson <- within(pHankinson, {
    lowerBB <- lims[1,]  ### NB results taken from the ROWS of lims
    upperBB <- lims[2,]
  })
> rm(lims, Z)
```

Secondly, consider standard bootstrap estimates, where the model is re-fitted using bootstrap samples of the original data:

```
> bsample <- function(dfr) dfr[sample(nrow(dfr), replace = TRUE), ]
> Z <- replicate(500, {
    tmpM <- update(NonlinM, data = bsample(hankinson), trace = FALSE)
    exp(predict(tmpM, newdata = pHankinson))
  })
> lims <- apply(Z, 1, quantile, probs = c(1, 39)/40)
> pHankinson <- within(pHankinson, {
    lower <- lims[1,]
    upper <- lims[2,]
  })
> rm(lims, Z)
```

Finally we present the results in graphical form, and clean up the global environment:

```
> ylim <- with(pHankinson, range(upper, lower, upperBB, lowerBB))
```

```
> with(pHankinson, {
    plot(thetaDeg, MRupture, type = "l", ylim = ylim, las = 1,
         xlab = expression(theta * degree), ylab = "Modulus of rupture")
    lines(thetaDeg, lower, col="blue", lty = "dotted")
    lines(thetaDeg, upper, col="blue", lty = "dotted")
    lines(thetaDeg, lowerBB, col="red", lty = "dotdash")
    lines(thetaDeg, upperBB, col="red", lty = "dotdash")
    points(MRupture ~ thetaDeg, hankinson, pch = 4, col = "darkgreen")
  })
> library(SOAR)
> Store(hankinson, pHankinson)
> rm(f,g, NonlinM, NonlinMg, StartM, ylim, nr, bsample)
```

The result is shown in Figure 1 on the following page.

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: SOAR 0.99-10, xtable 1.7-0
- Loaded via a namespace (and not attached): tools 2.15.0

References

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

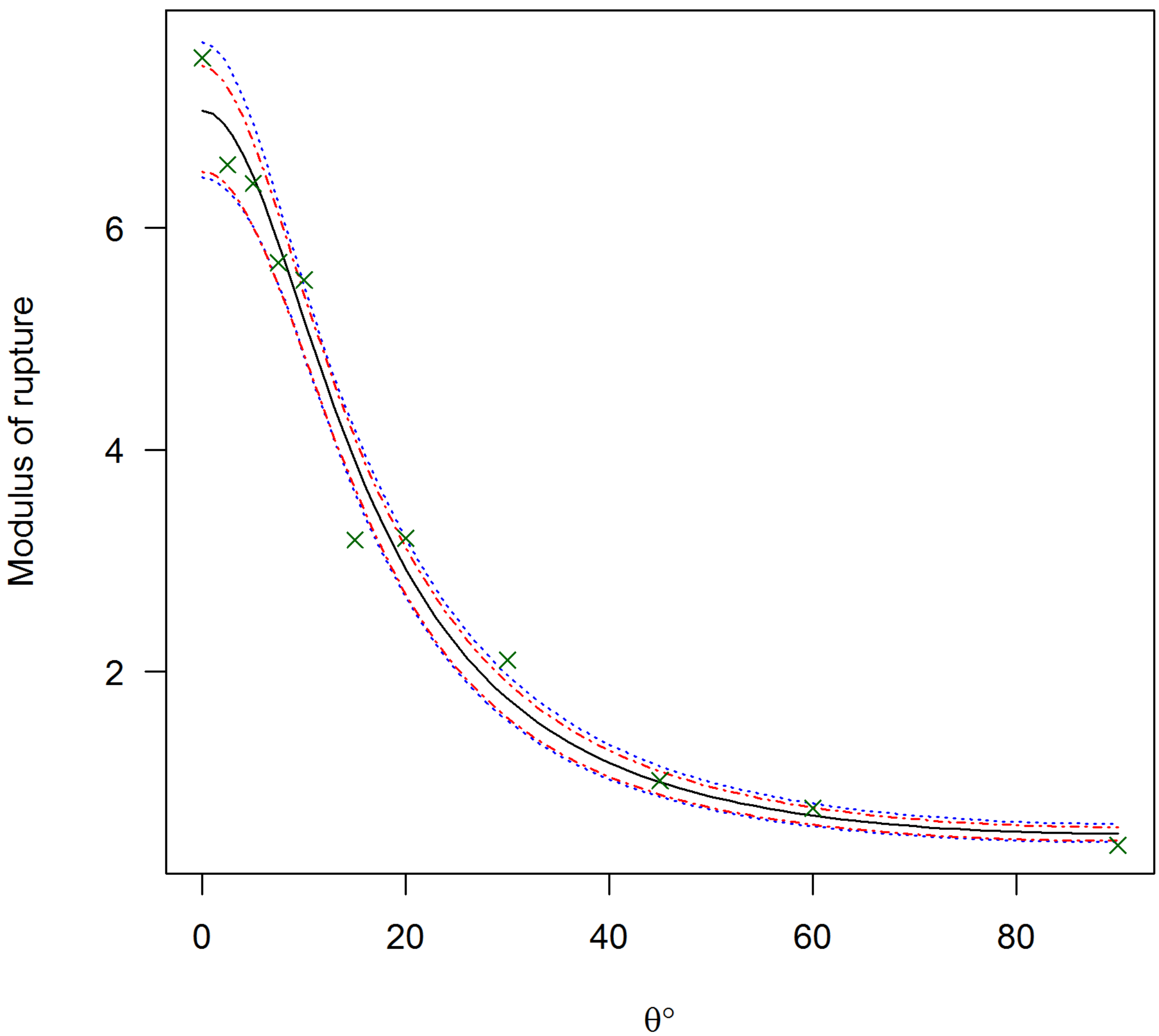

Figure 1: Estimates of the modulus of rupture, with bootstrap confidence intervals