R Workshop

Session 4: Mixed Effects Models

Bill Venables, CSIRO, Australia

UseR! 2012

Nashville

11 June, 2012

Contents

1 An introductory example: petroleum extraction

The petrol data of N. L. Prater.

- *No* crude oil sample identification label. (Factor.)
- *SG* specific gravity, degrees API. (Constant within sample.)
- *VP* vapour pressure in pounds per square inch. (Constant within sample.)
- *V10* volatility of crude; ASTM 10% point. (Constant within sample.)
- *EP* desired volatility of gasoline. (The end point. Varies within sample.)
- *Y* yield as a percentage of crude.

For a description in **R**:

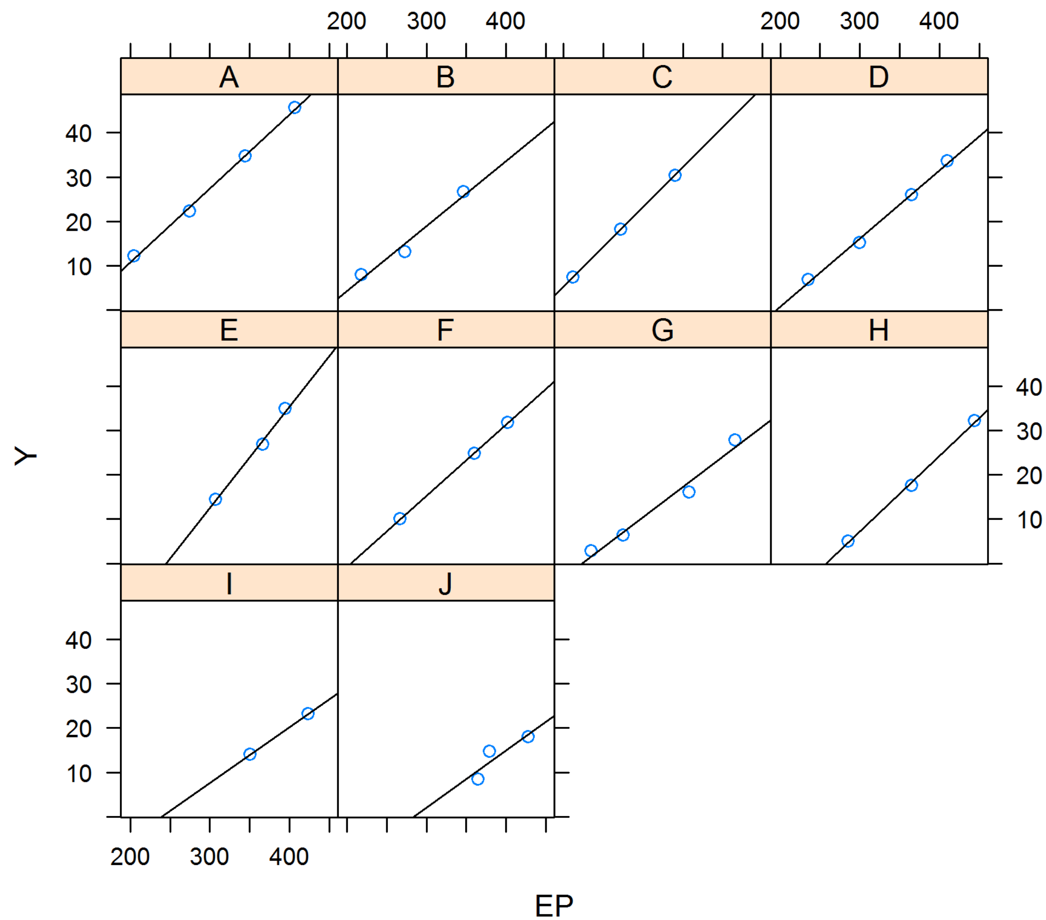
```
> library(MASS)
> ?petrol
> head(petrol)
```

```
  No   SG  VP V10  EP    Y
1  A 50.8 8.6 190 205 12.2
2  A 50.8 8.6 190 275 22.3
3  A 50.8 8.6 190 345 34.7
4  A 50.8 8.6 190 407 45.7
5  B 40.8 3.5 210 218  8.0
6  B 40.8 3.5 210 273 13.1
```

For a more complete description of the data and an alternative (somewhat fussy) analysis see the `betareg` package, (Cribari-Neto and Zeileis, 2010). `?GasolineYield`.

An initial look at the data:

```
> require(lattice)
> ## lattice.options(default.theme = standard.theme(color=TRUE))
> print(xyplot(Y ~ EP | No, petrol, as.table = TRUE, aspect=1,
               panel = function(x, y, ...) {
                 panel.xyplot(x, y, ...)
                 panel.lmline(x, y)
               }))
> petrol <- within(petrol, EPc <- EP - mean(EP))  ### for convenience
> Store(petrol)
```


A
B
C
D
E
F
G
H
I
J
Y
EP
200
300
400
10
20
30
40

1.1 Fixed or random?

A pure fixed effects model treats the crude oil samples as independent with the residual error as the only source of randomenss.

A random effects model treats them as possibly dependent, in that they may share the value of a latent random variable, addition to the residual error.

The obvious candidate predictor to be regarded as injecting an additional source of randomenss is the crude oil sample indicator, `No`.

Fixed effects only.

```
> options(show.signif.stars = FALSE)
> m3 <- lm(Y ~ 0 + No/EP, petrol)               ## 10 ints + 10 slopes
> m2 <- lm(Y ~ 0 + No+EP, petrol)               ## 10 ints + 1 slope
> m1 <- lm(Y ~ 1 + SG+VP+V10+EP, petrol)        ## (1 int + 3 coeffs) + 1 slope
> anova(m1, m2, m3)

Analysis of Variance Table
Model 1: Y ~ 1 + SG + VP + V10 + EP
Model 2: Y ~ 0 + No + EP
Model 3: Y ~ 0 + No/EP
  Res.Df     RSS Df Sum of Sq      F  Pr(>F)
1     27 134.804
2     21  74.132  6    60.672 4.0009 0.01965
3     12  30.329  9    43.803 1.9257 0.14390
```

Parallel regressions, but differences between samples cannot quite be explained by regression on the other variables.

Random effects alternatives:

```
> suppressPackageStartupMessages(library(lme4))  ## alt. nlme
> Rm1 <- lmer(Y ~ 1 + SG+VP+V10 + EPc + (1|No),     data = petrol)
> Rm2 <- lmer(Y ~ 1 + SG+VP+V10 + EPc + (1+EPc|No), data = petrol)
> anova(Rm1, Rm2)

Data: petrol
Models:
Rm1: Y ~ 1 + SG + VP + V10 + EPc + (1 | No)
Rm2: Y ~ 1 + SG + VP + V10 + EPc + (1 + EPc | No)
    Df    AIC    BIC  logLik Chisq Chi Df Pr(>Chisq)
Rm1  7 150.10 160.36 -68.049
Rm2  9 154.18 167.38 -68.092     0      2          1
```

Emphatically different slopes are not needed!

The problem is that the variance estimates are REML rather than maximum likelihood.

```
> Rm1_ML <- update(Rm1, REML = FALSE)
> Rm2_ML <- update(Rm2, REML = FALSE)
> anova(Rm1_ML, Rm2_ML)

Data: petrol
Models:
Rm1_ML: Y ~ 1 + SG + VP + V10 + EPc + (1 | No)
Rm2_ML: Y ~ 1 + SG + VP + V10 + EPc + (1 + EPc | No)
       Df    AIC    BIC  logLik  Chisq Chi Df Pr(>Chisq)
Rm1_ML  7 149.38 159.64 -67.692
Rm2_ML  9 153.34 166.54 -67.672 0.0385      2     0.9809
```

Still pretty emphatically not needed.

Inspecting the random effects fit:

```
> print(summary(Rm1), correlation = FALSE)

Linear mixed model fit by REML
Formula: Y ~ 1 + SG + VP + V10 + EPc + (1 | No)
   Data: petrol
   AIC   BIC logLik deviance REMLdev
 166.4 176.6 -76.19    136.1   152.4
Random effects:
 Groups   Name        Variance Std.Dev.
 No       (Intercept) 2.0873   1.4447
 Residual             3.5052   1.8722
Number of obs: 32, groups: No, 10
Fixed effects:
             Estimate Std. Error t value
(Intercept) 46.062547  14.691579   3.135
SG           0.219398   0.146929   1.493
VP           0.545864   0.520499   1.049
V10         -0.154242   0.039960  -3.860
EPc          0.157177   0.005588  28.128
```

```
> print(summary(Rm2), correlation = FALSE)
Linear mixed model fit by REML
Formula: Y ~ 1 + SG + VP + V10 + EPc + (1 + EPc | No)
   Data: petrol
   AIC   BIC logLik deviance REMLdev
 170.4 183.6 -76.19    136.2   152.4
Random effects:
 Groups   Name        Variance   Std.Dev.  Corr
 No       (Intercept) 2.1389e+00 1.4624819
          EPc         1.2918e-05 0.0035942 0.057
 Residual             3.4329e+00 1.8527988
Number of obs: 32, groups: No, 10
Fixed effects:
             Estimate Std. Error t value
(Intercept) 45.934617  14.778375   3.108
SG           0.219234   0.147711   1.484
VP           0.552160   0.523528   1.055
V10         -0.153799   0.040236  -3.822
EPc          0.157258   0.005688  27.647
```

Use *fixef* for fixed effect estimates and *ranef* for BLUPs:

```
> cbind(Rm1 = ranef(Rm1)$No, Rm2 = ranef(Rm2)$No)

  (Intercept) Rm2.(Intercept)       Rm2.EPc
A -0.05943595     -0.04214312  0.0007608813
B -0.21857463     -0.21898402 -0.0002975789
C  1.92034361      1.96463075  0.0002788768
D -1.92767172     -1.95660916 -0.0004374530
E -0.21650155     -0.22746503  0.0010007798
F  0.56933023      0.57479310  0.0002256160
G  0.06701389      0.05027548 -0.0014955586
H  0.19194964      0.18789649  0.0006531943
I -0.40278261     -0.40719252 -0.0004969090
J  0.07632910      0.07479803 -0.0001918488
```

Variances and correlations

```
> VarCorr(Rm2)
$No
            (Intercept)          EPc
(Intercept) 2.138853253 3.012290e-04
EPc         0.000301229 1.291827e-05
attr(,"stddev")
(Intercept)         EPc
1.462481881 0.003594199
attr(,"correlation")
            (Intercept)        EPc
(Intercept)  1.00000000 0.05730653
EPc          0.05730653 1.00000000
attr(,"sc")
[1] 1.852799
```

2 An extended example: going fishing

The `Headrope` data set gives catch and effort data from a prawn fishery.

- The fishery has 7 *`Stock`* regions *`Tig1`*, ..., *`Tig7`*, West to East.
- The data is for 20 seasons (*`YearF`*) 1987, ..., 2006. (*`Y2K`* = year - 2000.)
- There are 236 *`Vessel`*s, which visit one or more stock regions within a season, each for one or more *`Days`*.
- The *response* for which a model is required is the total *`Catch`* in kg, by a vessel within a stock region for a season.
- Additionally the vessels have *`Hull`* size, engine *`Power`* and the *`Head`*rope length they were using recorded. (These are constant within season, but may change between seasons.)

```
> data(Headrope)
> dim(Headrope)

[1] 8594   13

> head(Headrope, 2)

     YearF Y2K Stock Vessel Days Head Hull Power Catch Banana Tiger
0001  1987 -13  Tig1   V008   20   20  133   350  4355   2509   975
0002  1987 -13  Tig1   V012   13   20  134   336  4746   3612   252
     Endeavour King
0001       871    0
0002       882    0

> Headrope <- within(Headrope, YearF <- factor(YearF)) ## needed
> Store(Headrope)
```

The purpose of the study was to gain some insight on the marginal effect of headrope length on the catch.

A multiplicative (log-linear) model was suggested, with additive random effects for a) vessel and b) stock regions over seasons.

Two random effects models: the first is the simpler

```
> HRmodel1 <- lmer(log(Catch) ~ 0 + log(Days) + Y2K + log(Head) +
                   log(Power) + log(Hull) + Stock +
                   (1|Vessel) + (1|YearF/Stock), Headrope)
> HRmodel1_ML <- update(HRmodel1, REML = FALSE)
> Store(HRmodel1)
```

The second has a more elaborate random effect structure:

```
> HRmodel2 <- lmer(log(Catch) ~ 0 + log(Days) + Y2K + log(Head) +
                   log(Power) + log(Hull) + Stock +
                   (1|Vessel) + (0+Stock|YearF),
                   data = Headrope)
> HRmodel2_ML <- update(HRmodel2, REML = FALSE)
> Store(HRmodel2, HRmodel2_ML)
```

The more elaborate model seems justified by AIC, but not BIC!

```
> anova(HRmodel1_ML, HRmodel2_ML)

Data: Headrope
Models:
HRmodel1_ML: log(Catch) ~ 0 + log(Days) + Y2K + log(Head) + log(Power) + log(Hul
HRmodel1_ML:     Stock + (1 | Vessel) + (1 | YearF/Stock)
HRmodel2_ML: log(Catch) ~ 0 + log(Days) + Y2K + log(Head) + log(Power) + log(Hul
HRmodel2_ML:     Stock + (1 | Vessel) + (0 + Stock | YearF)
            Df   AIC   BIC  logLik  Chisq Chi Df Pr(>Chisq)
HRmodel1_ML 16 11676 11788 -5821.8
HRmodel2_ML 42 11613 11910 -5764.5 114.48     26  4.501e-13
```

The fixed effects estimates are very similar:

```
> cbind(m1 = fixef(HRmodel1), m2 = fixef(HRmodel2))
                    m1         m2
log(Days)   1.16175483 1.16092768
Y2K         0.02742618 0.03477335
log(Head)   0.30270318 0.30165523
log(Power)  0.11566443 0.11413855
log(Hull)   0.20684716 0.20812540
StockTig1   2.84888023 2.88901846
StockTig2   2.39961474 2.43791784
StockTig3   2.14663298 2.18116835
StockTig4   2.32495242 2.35987247
StockTig5   2.41038414 2.44535334
StockTig6   2.55092062 2.56127067
StockTig7   2.19454067 2.17294398
```

Some notes:

- The coefficient on `log(Days)` is slightly larger than 1, (but significantly). A coeficient of 1 would imply that, *mutatis mutandis*, catch is proportional to "effort" (measured in boat days).

- The coefficient of `Y2K` suggests an average fishing power increase in the order of 2.5%-3.5% per year. This looks about right, but it is confounded with change in the stock abundance. Essentially the job of disentangling this confounding is what stock assessment is all about (and why it is so hard).

For reference we include a copy of the summary of the more elaborate model below.

```
> print(summary(HRmodel2), correlation = FALSE)
Linear mixed model fit by REML
Formula: log(Catch) ~ 0 + log(Days) + Y2K + log(Head) + log(Power) + log(Hull) +
   Data: Headrope
   AIC   BIC logLik deviance REMLdev
 11677 11973  -5797    11529   11593
Random effects:
 Groups   Name        Variance Std.Dev. Corr
 Vessel   (Intercept) 0.010275 0.10137
 YearF    StockTig1   0.161463 0.40182
          StockTig2   0.113516 0.33692   0.296
          StockTig3   0.022821 0.15106   0.110  0.377
          StockTig4   0.019230 0.13867   0.201  0.764  0.547
          StockTig5   0.020210 0.14216   0.161  0.624  0.616  0.927
          StockTig6   0.159386 0.39923  -0.071  0.146  0.357  0.221  0.534
          StockTig7   0.178439 0.42242   0.026  0.345  0.256  0.059  0.302
 Residual             0.211245 0.45961
  0.798

Number of obs: 8594, groups: Vessel, 236; YearF, 20
```

```
Fixed effects:
           Estimate Std. Error t value
log(Days)  1.160928   0.004801  241.79
Y2K        0.034773   0.004265    8.15
log(Head)  0.301655   0.049989    6.03
log(Power) 0.114139   0.037573    3.04
log(Hull)  0.208125   0.031700    6.57
StockTig1  2.889018   0.195866   14.75
StockTig2  2.437918   0.188496   12.93
StockTig3  2.181168   0.175106   12.46
StockTig4  2.359872   0.174620   13.51
StockTig5  2.445353   0.175128   13.96
StockTig6  2.561271   0.193476   13.24
StockTig7  2.172944   0.196107   11.08
```

2.1 A brief look at generalized linear/additive mixed models

Software for GLMMs is still somewhat developmental.

- *glmmPQL* in `MASS` is based on `nlme`, but handles general cases.
- *glmer* from the `lme4` package handles some GLMMs but is restricted in the families it can take. (In particular, *quasipoisson* is NOT included.)

The software for GAMMs also uses a linear ME engine.

- *gamm* from the `mgcv` package uses `nlme` engine,
- *gamm4* from he `gamm4` package (Wood, 2011), uses `lme4` engine, (and so has the same limitations).

Both *gamm* and *gamm4* return a composite object with an `lme` and a `gam` component. Manipulation is tricky.

To illuatrate, we construct a GLMM and a GAMM for the Tiger Prawn species split example. The model structure is slightly simplified relative to the working model.

We use two helper functions, `Hyear` and `twoWay` which will be defined at the end.

First, the GIMM:

```
> library(splines)
> library(MASS)
> TModelGLMM <- glmmPQL(Psem/Total ~ ns(Coast, 6) + ns(Sea, 5) +
                        twoWay(DayOfYear, Sea) + ns(Depth, k=5) +
                        Hyear(DayOfYear, 2) + ns(Mud, k=5),
                        random = ~1|Survey,
                        family = quasibinomial, data= Tigers,
                        niter = 40, weights = Total)
> Store(TModelGLMM)
```

Note that the random component is defined separately from the main formula, in `nlme` style.

For a GAM with smoothed terms:

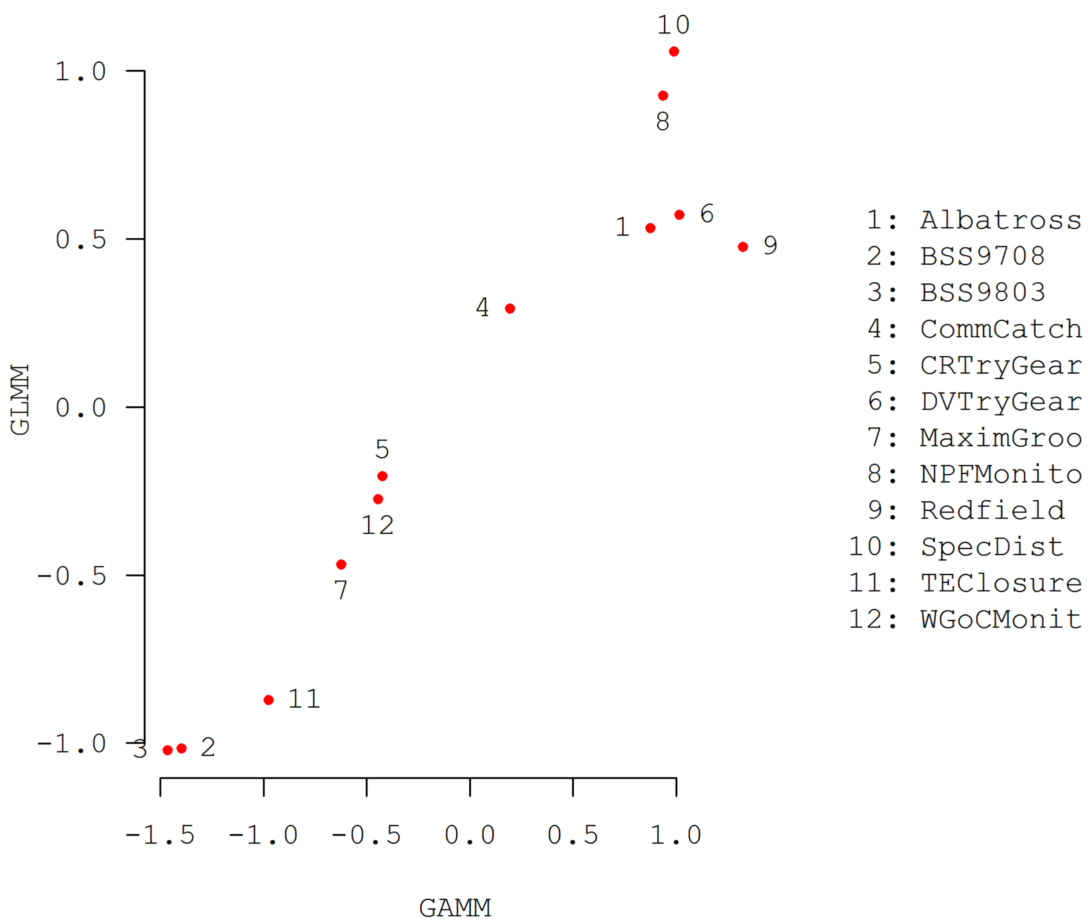
```
> library(mgcv)
> TModelGAMM <- gamm(formula = Psem/Total ~ s(Coast, k=5) + s(Sea,k=5) +
                     twoWay(DayOfYear, Sea) +
                     s(DayOfYear, k=5, bs="cc") + s(Depth,k=5) +
                     s(Mud, k=5),
                     random = list(Survey = ~1),
                     family = quasibinomial, data = Tigers,
                     niterPQL = 40,
                     weights = Total)
> Store(TModelGAMM)
```

The random effects from these different models are quite similar. We illustrate below. (We also use the *thigmophobe* function from the plotrix package, (Lemon, 2006), to minimise clashes in annotation of the poings

```
> re1 <- ranef(TModelGLMM)
> re2 <- ranef(TModelGAMM$lme)$Survey  ## obscure
> re12 <- cbind(re1, re2)
> names(re12) <- c("GLMM", "GAMM")
> re12

                   GLMM       GAMM
Albatross     0.5318898  0.8748097
BSS9708      -1.0162057 -1.3977773
BSS9803      -1.0212852 -1.4673097
CommCatch     0.2927734  0.1937862
CRTryGear    -0.2056465 -0.4244961
DVTryGear     0.5717618  1.0143913
MaximGroote -0.4686942 -0.6236263
NPFMonitor   0.9254647  0.9366853
Redfield      0.4769554  1.3248063
SpecDist      1.0579519  0.9879133
TEClosure    -0.8708619 -0.9751100
WGoCMonitor -0.2741036 -0.4440728
```

```
> layout(rbind(1:2), widths = c(3.5,1))
> library(plotrix)
> with(re12, {
    nos <- seq_along(GLMM)
    plot(GLMM ~ GAMM, pch = 20,col="red", bty="n")
    pos <- thigmophobe(GAMM, GLMM)
    text(GLMM ~ GAMM, labels = nos, xpd = NA, pos=pos)
    par(mar = c(0,0,0,0), xpd = NA)
    frame()
    legend("center", paste(format(nos), rownames(re12),
                               sep = ": "), bty="n")
  })
```


10
1.0
8
1
6
9
0.5
4
GLMM
0.0
5
12
-0.5
7
11
-1.0
3
2
-1.5
-1.0
-0.5
0.0
0.5
1.0
GAMM
1: Albatross
2: BSS9708
3: BSS9803
4: CommCatch
5: CRTryGear
6: DVTryGear
7: MaximGroo
8: NPFMonito
9: Redfield
10: SpecDist
11: TEClosure
12: WGoCMonit

2.2 Appendix: Two helper functions:

These are needed to define harmonic terms and interactions.

```
> Harm <- function (theta, k = 4) {
    X <- matrix(0, length(theta), 2 * k)
    nam <- as.vector(outer(c("c", "s"), 1:k, paste, sep = ""))
    dimnames(X) <- list(names(theta), nam)
    m <- 0
    for (j in 1:k) {
      X[, (m <- m + 1)] <- cos(j * theta)
      X[, (m <- m + 1)] <- sin(j * theta)
    }
    X
  }
> Hyear <- function(x, k = 4)
      Harm(2*base::pi*x/365.25, k)
> twoWay <- local({

    `%star%` <- function(X, Y) {
```

```
    X <- as.matrix(X)
    Y <- as.matrix(Y)
    stopifnot(is.numeric(X), is.numeric(Y),
              nrow(X) == nrow(Y))
    XY <- matrix(NA, nrow(X), ncol(X)*ncol(Y))
    k <- 0
    for(i in 1:ncol(X))
        for(j in 1:ncol(Y)) {
          k <- k+1
          XY[, k] <- X[, i] * Y[, j]
        }
    XY
  }

  function(day, sea, k = c(3,2))
      Hyear(day, k[1]) %star% ns(sea, k[2])
})
```

3 Technical highlights

- Slide ...

References

Cribari-Neto, F. and A. Zeileis (2010). Beta regression in **R**. *Journal of Statistical Software 34*(2), 1–24.

Lemon, J. (2006). Plotrix: a package in the red light district of R. *R-News 6*(4), 8–12.

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

Wood, S. (2011). *gamm4: Generalized additive mixed models using* `mgcv` *and* `lme4`. CRAN. R package version 0.1-5.

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: lattice 0.20-6, lme4 0.999375-42, MASS 7.3-18, Matrix 1.0-6, plotrix 3.4-1, SOAR 0.99-10
- Loaded via a namespace (and not attached): grid 2.15.0, mgcv 1.7-17, nlme 3.1-104, stats4 2.15.0, tools 2.15.0