New Features in the **car** and **effects** Packages

John Fox [1]   Brad Price [2]   Sanford Weisberg [3]

[1]McMaster University

[2]West Virginia University

[3]University of Minnesota

useR! 2018, Brisbane, Australia

Introduction

- The **car** and **effects** packages are associated with Fox and Weisberg, *An R Companion to Applied Regression* (Fox and Weisberg, 2018a).
- Based on CRAN download statistics (e.g., at `https://www.rdocumentation.org/trends`), these packages are in very wide use.
- In conjunction with the third edition of the book, to be published in the Fall, we prepared new major versions of these two packages.
- We also moved the data sets in the packages to a new data-only package, **carData**.
- The current CRAN versions of the packages, described in this talk, are **car** 3.0-1 and **effects** 4.0-2.

Focus

- The **car** package focuses on tools, many of them graphical, that are useful for applied regression analysis (broadly construed to include linear, generalized linear, mixed-effects, and some other statistical models).
- The package includes tools for *importing*, *preparing*, *examining*, and *transforming* data prior to specification of a regression model, and tools that are useful for *testing*, *summarizing*, *comparing*, and *assessing* regression models that have been fit to data.
- The **effects** packages focuses on graphical methods for *interpreting* fitted regression models.

Highlights of New Features and Improvements in the **car** Package

- Improved facilities for bootstrapping regression models, in the new function `Boot()`.
- Several new functions that provide more flexible summaries of statistical models (and in certain cases other objects) than corresponding standard R functions, for example optionally accommodating non-standard coefficient covariance matrices:
  - `S()` (replacing `summary()`)
  - `brief()` (replacing `print()`)
  - `Confint()` (replacing `confint()`)
  - `Predict()` (replacing `predict()`)
  - `Tapply()` (a formula-oriented interface to `tapply()`)
- Deletion diagnostics for clusters and individual cases in mixed-effects models, implemented as methods for the `influence()` function.
- A new `poTest()` function for testing for proportional odds in `"polr"` models.

Highlights of New Features and Improvements in the **car** Package

- Improved handling of variable transformations, including introduction of the `bcnPower()` family of transformations, an extension of the Box-Cox family to variables with zero and negative values, introduced by Hawkins and Weisberg (2017).
- Reorganization of arguments to graphics functions in the **car** package to make the use of these functions simpler and more consistent.

Highlights of New Features and Improvements in the **effects** Package

- A new conceptualization of effects plots, termed *predictor effects* (Fox and Weisberg, 2018b), which focus in turn on each predictor in a regression model, showing how that predictor, perhaps conditioning on values of other predictors with which it interacts, influences the response.
  - Other predictors are held to typical values or, in the case of factors, to a typical (but user-modifiable) distribution of values over the levels of the factor.
  - Predictor effects are computed by the new functions `predictorEffect()` and `predictorEffects()`.
  - Predictor effects have improved invariance properties and are more easily interpreted than the "term effects" produced by the `allEffects()` function previously available in the package.
- Partial residuals may be added to effect plots for linear and generalized linear models (Fox and Weisberg, 2018b), permitting the data analyst, under well understood circumstances, to determine whether the functional form of the model is correct.
  - The implementation is very general and includes interaction terms of arbitrary degree and complexity.
  - The most recent version of the package extends partial residuals to plots against factors.

Highlights of New Features and Improvements in the **effects** Package

- Reorganization of the arguments to the functions in the **effects** package that compute and graph effects, to make their use simpler and more consistent, and to work more effectively with the **lattice** package, used to build and display effect graphs.
- There are also significant improvements under the hood that are invisible to users:
  - Most notably, there is an improved `default` method for the work-horse `Effect()` generic function, which is more likely to work with other classes of statistical models, and which, when it doesn't work directly, makes it simpler to write new `Effect()` methods.
  - In addition to the `default` method, we supply `Effect()` methods for models produced by `lm()` and `glm()`; `betareg()` (in the **betareg** package); `clm()`, `clm2()`, and `clmm()` (in **ordinal**); `gls()` and `lme()` (in **nlme**); `lmer()` and `glmer()` (in **lme4**); `multinom()` (in **nnet**); `polr()` (in **MASS**); `rlmer()` (in **robustlmm**); and `svyglm()` (**survey**).
  - Improved handling of rank-deficient models and estimability checks for effects in LMs and GLMs.

Examples

- To illustrate some of the new features of the **car** and **effects** packages we develop two examples drawing on data sets in the **carData** package:
  1. Fitting a linear model by OLS to data on bank transactions in the `Transact` data set.
  2. Fitting a generalized linear mixed-effects model to data on police stops of individuals in Minneapolis, combining data from the individual-level `MplsStops` and neighbourhood-level `MplsDemo` data sets.

Bank Transactions

- The `Transact` data frame contains data from Cunningham and Heathcote (1989) on the numbers of two types of transactions, `t1` and `t2`, and the total labour `time` in minutes in 261 branch offices of a large Australian bank:

```
> library(car)
> brief(Transact)
```

```
261 x 3 data.frame (256 rows omitted)
     t1   t2  time
    [i]  [i]   [i]
1     0 1166  2396
2     0 1656  2348
3     0  899  2403
. . .
260 370 2644  7930
261 825 4429 13610
```


Bank Transactions

- An OLS regression estimates the average time devoted to a transaction of each type, with the intercept estimating average branch non-transaction time:

```
> S(mod.trans <- lm(time ~ t1 + t2, Transact), brief=TRUE)
```

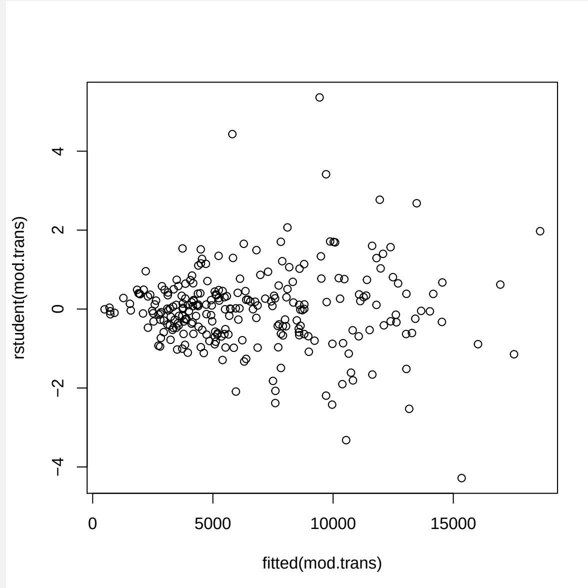
```
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 144.36944  170.54410   0.847    0.398
t1            5.46206    0.43327  12.607   <2e-16
t2            2.03455    0.09434  21.567   <2e-16

Residual standard deviation: 1143 on 258 degrees of freedom
Multiple R-squared: 0.9091
F-statistic:  1289 on 2 and 258 DF,  p-value: < 2.2e-16
    AIC     BIC
4421.08 4435.34
```

Bank Transactions

- There is, however, very strong evidence of non-constant error variance; for example:

```
> plot(fitted(mod.trans), rstudent(mod.trans))
> ncvTest(mod.trans)
```

```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 61.65942, Df = 1,  p = 4.0831e-15
```



Bank Transactions

- One approach is to stick with the OLS estimator but use a corrected estimate of the coefficient-covariance matrix.
- Another approach is to fit a model, such as a gamma GLM, where the variance increases with the mean (as discussed for the `Transact` data in Cunningham and Heathcote, 1989, and Fox and Weisberg, 2018a, Sec. 6.10.2).
- Functions in the **car** package make it easy to use alternative coefficient-covariance matrix estimators,
  - either a sandwich estimator (see Fox, 2016, Sec. 12.2.3, or Weisberg, 2014, Sec. 7.2.1), e.g., as provided by the `hccm()` function,
  - or a coefficient-covariance matrix based on the bootstrap, using the `Boot()` function, which provides a convenient front-end to `boot::boot()` (see Fox, 2016, Ch. 21, or Weisberg, 2014, Sec. 7.7).

Bank Transactions

```
> brief(mod.trans, vcov.=hccm)

           (Intercept)    t1    t2
Estimate           144 5.462 2.035
Std. Error         203 0.729 0.163

 Residual SD = 1143 on 258 df, R-squared = 0.909

> Anova(mod.trans, vcov.=hccm)

Analysis of Deviance Table (Type II tests)

Response: time
           Df       F    Pr(>F)
t1          1  56.167 1.065e-12
t2          1 154.935 < 2.2e-16
Residuals 258
```


Bank Transactions

```
> (seed <- sample(.Machine$integer.max, 1))

[1] 1135084836

> set.seed(seed) # for reproducibility
> boot.mod.trans <- Boot(mod.trans, ncores=4)
> brief(mod.trans, vcov.=vcov(boot.mod.trans))

           (Intercept)    t1   t2
Estimate           144 5.462 2.03
Std. Error         189 0.683 0.15

 Residual SD = 1143 on 258 df, R-squared = 0.909
```

Bank Transactions

```
> Anova(mod.trans, vcov.=vcov(boot.mod.trans))

Coefficient covariances computed by vcov(boot.mod.trans)
Analysis of Deviance Table (Type II tests)

Response: time
           Df       F    Pr(>F)
t1          1  63.919 4.364e-14
t2          1 184.007 < 2.2e-16
Residuals 258

> Confint(boot.mod.trans)

Bootstrap bca confidence intervals

              Estimate       2.5 %     97.5 %
(Intercept) 144.369443 -262.217918 476.521316
t1            5.462057    3.879273   6.603760
t2            2.034549    1.785787   2.404706
```


Bank Transactions

- The same approaches work uniformly in the **car** package.
- For example, we can use the delta method for the ratio of the time required for the two types of transactions, with or without correction for non-constant variance:

```
> deltaMethod(mod.trans, "t2/t1")

       Estimate         SE     2.5 %    97.5 %
t2/t1 0.3724877 0.04425834 0.2857429 0.4592324

> deltaMethod(mod.trans, "t2/t1", vcov.=hccm)

       Estimate         SE     2.5 %    97.5 %
t2/t1 0.3724877 0.07747019 0.2206489 0.5243265
```

Minneapolis Police Stops

- The data in `MplsStops` are for nearly all stops made by the Minneapolis Police Department for the year 2017; there are 51,920 stops in the data set.
- The data in `MplsDemo` contain demographic data on 84 Minneapolis neighbourhoods from the 2015 American Community Survey.
- We built a combined data set (`Mpls`), merging the neighborhood-level with the stops-level data, and restricting the data to non-traffic stops, for which racial and gender information about the person stopped was collected. We focus on whether the person stopped was searched.
- After removing stops in three neighbourhoods in which there is no housing and filtering out missing data, we're left with 9612 stops:

Minneapolis Police Stops

```
> summary(Mpls)

         neighborhood                  race          gender          black
 Downtown West   :1407   White          :3457   Female:2087   Min.   :0.0040
 Whittier        : 554   Black          :5007   Male  :7525   1st Qu.:0.1350
 East Phillips   : 466   Native American:1148                 Median :0.2110
 Lyndale         : 447                                        Mean   :0.2334
 Hawthorne       : 377                                        3rd Qu.:0.2970
 Midtown Phillips: 327                                        Max.   :0.6560
 (Other)         :6034

 personSearch
 NO :7070
 YES:2542
```

- The variables are largely self-explanatory; `black` is the proportion of residents in the neighbourhood who are black.

Minneapolis Police Stops

- We proceed to use glmer() in the **lme4** package to fit a binomial GLMM to the data, with personSearch as the response variable:

```
> library("lme4")
> mod.mpls <- glmer(personSearch ~ race*gender + black
+                   + (1 | neighborhood), data=Mpls, family=binomial)
> Anova(mod.mpls)
```

```
Analysis of Deviance Table (Type II Wald chisquare tests)

Response: personSearch
               Chisq Df Pr(>Chisq)
race        109.6020  2  < 2.2e-16
gender       57.7897  1  2.917e-14
black        14.9714  1  0.0001092
race:gender   9.2222  2  0.0099411
```


Minneapolis Police Stops

```
> S(mod.mpls)

Generalized linear mixed model fit by ML
Call: glmer(formula = personSearch ~ race * gender + black + (1 |
            neighborhood), data = Mpls, family = binomial)

Estimates of Fixed Effects:
                               Estimate Std. Error z value Pr(>|z|)
(Intercept)                     -2.0413     0.1283 -15.916  < 2e-16
raceBlack                        0.1516     0.1321   1.148 0.251125
raceNative American              0.5987     0.1477   4.055 5.02e-05
genderMale                       0.2308     0.1111   2.078 0.037742
black                            1.3581     0.3510   3.869 0.000109
raceBlack:genderMale             0.4369     0.1446   3.022 0.002509
raceNative American:genderMale   0.2126     0.1746   1.218 0.223301
```

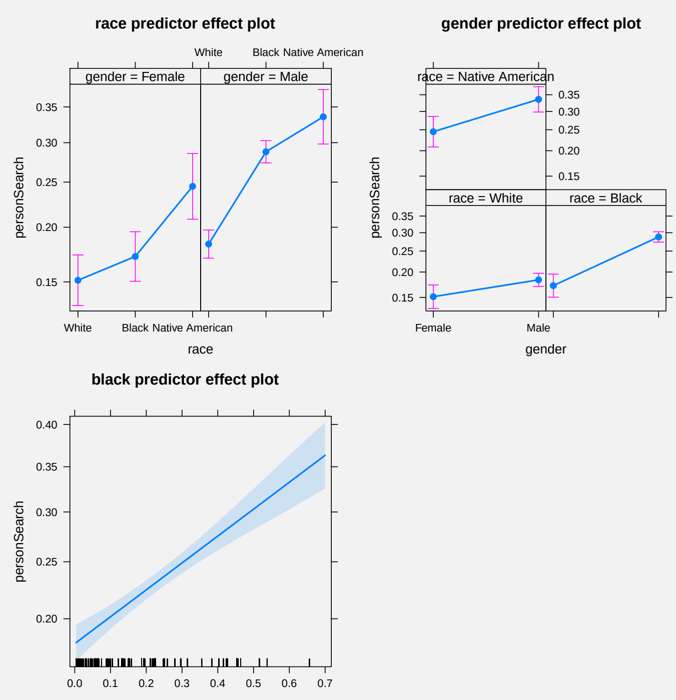
Minneapolis Police Stops

```
Exponentiated Fixed Effects and Confidence Bounds:
                              Estimate     2.5 %    97.5 %
(Intercept)                  0.1298553 0.1009919 0.1669677
raceBlack                    1.1636372 0.8982785 1.5073849
raceNative American          1.8198078 1.3624941 2.4306164
genderMale                   1.2596694 1.0131615 1.5661541
black                        3.8888989 1.9545833 7.7374723
raceBlack:genderMale         1.5478534 1.1659725 2.0548083
raceNative American:genderMale 1.2368592 0.8784905 1.7414196

Estimates of Random Effects (Covariance Components):
 Groups       Name        Std.Dev.
 neighborhood (Intercept) 0.3485

Number of obs: 9612, groups:  neighborhood, 84

  logLik       df      AIC      BIC
-5305.92        8 10627.83 10685.20
```


Minneapolis Police Stops

- Partly because of the `race`-by-`gender` interaction, and partly because the model is on the log-odds scale (though here the exponentiated coefficients help), it's hard to interpret the fixed effects from the coefficients.
- Predictor effect plots can help. Here are the default fixed-effect plots:

```
> library("effects")
> plot(predictorEffects(mod.mpls))
```

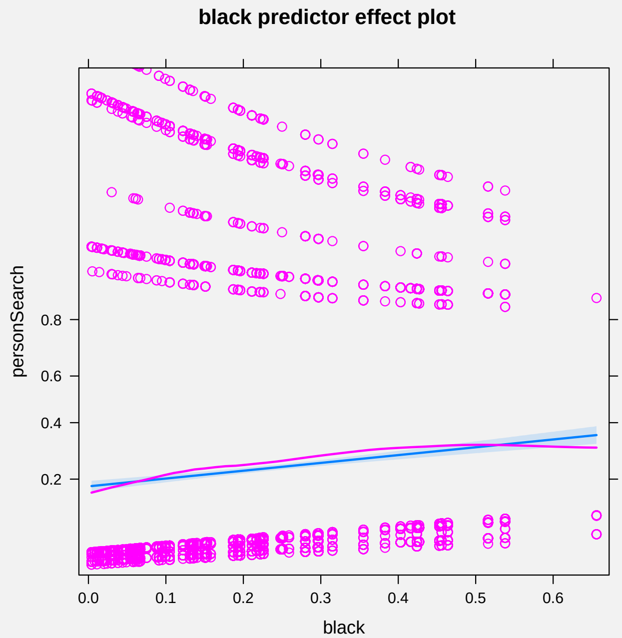


Minneapolis Police Stops

- Adding partial residuals to the effect plots helps us to judge lack of fit; for example, for the neighbourhood-level predictor `black`, which enters the model additively:

```
> plot(predictorEffect("black",
+     mod.mpls, residuals=TRUE))
```

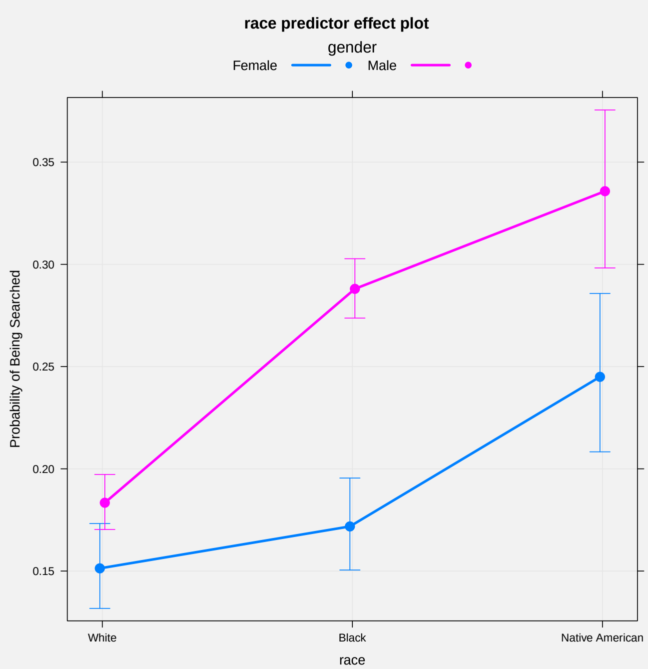
- The blue line represents the fitted model, the magenta line a loess smooth.
- There seems to be some unmodeled nonlinearity, but it is slight.



Minneapolis Police Stops

- Effect plots can be customized in a variety of ways. For example:

```
plot(predictorEffect("race", mod.mpls),
  lines=list(multiline=TRUE, lwd=3),
  symbols=list(cex=1.5, pch=16),
  confint=list(style="bars"),
  axes=list(y=list(
    lab="Probability of Being Searched",
    type="response")),
  lattice=list(key.args=list(
    cex=1, cex.title=1.2)),
  grid=TRUE)
```



Minneapolis Police Stops

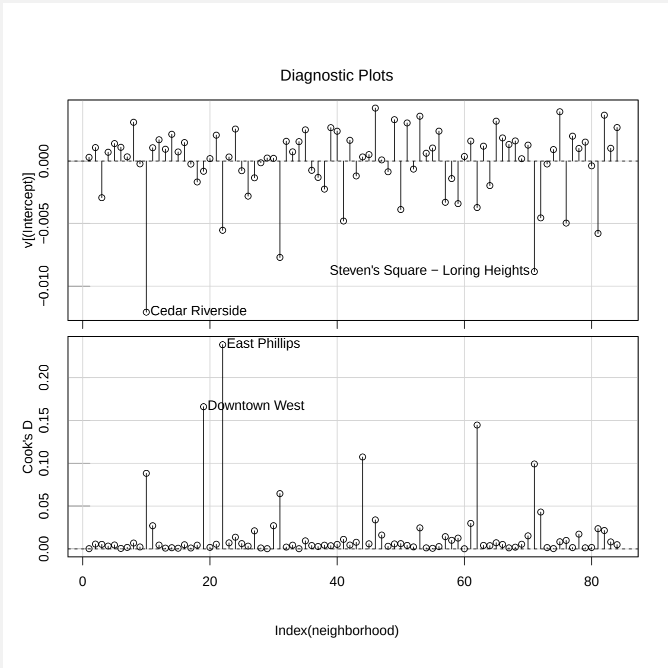
- The **car** package includes methods for the generic `influence()` function to compute case-deletion diagnostics for linear and generalized linear models fit by functions in the **nlme** and **lme4** package.
- The `influence()` methods allow us to remove individual-level cases as well as clusters from hierarchical (or longitudinal) data. The latter are likely to be of more concern.
- Because these methods work by refitting the models with a case or cluster removed, they are computationally intensive, even after efficiencies gained by selecting starting values from the initial model and limiting iterations.
- Revised methods that take advantage of parallel computations are in the works.

Minneapolis Police Stops

- Here, we apply `influence()` to the binomial GLMM that we fit to the Minneapolis poice-stops data:

```
> infl.mod.mpls <- influence(mod.mpls,
+   groups="neighborhood")
> infIndexPlot(infl.mod.mpls,
+   vars=c("cookd", "var.cov.comps"))
```

- For brevity, we plot only the Cook's Ds for the neighbourhoods (which summarize influence on the fixed-effects coefficients) and the single variance component for the intercept, omitting the dfbeta values for fixed-effect coefficients, which would be included by default.



Minneapolis Police Stops

- None of the neighbourhoods has much of an impact on the variance of the intercepts, but a couple may substantially affect the fixed-effects estimates.
- We try removing East Phillips (a neighbourhood in the center of the city with a large public-housing complex and many Native-American residents), which has the largest Cook's D (abbreviating the output):

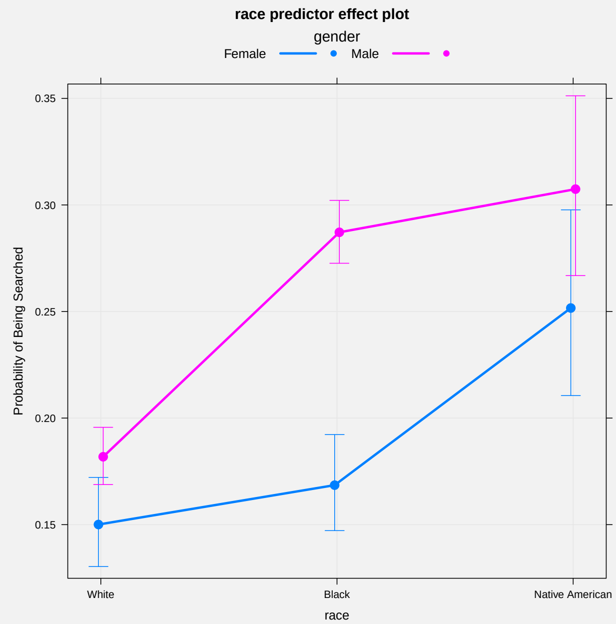
Minneapolis Police Stops

```
> mod.mpls.1 <- update(mod.mpls, subset = neighborhood != "East Phillips")
> compareCoefs(mod.mpls, mod.mpls.1)

. . .
                               Model 1 Model 2
(Intercept)                     -2.041  -2.038
SE                               0.128   0.129

. . .

black                            1.358   1.315
SE                               0.351   0.346

raceBlack:genderMale             0.437   0.457
SE                               0.145   0.147

raceNative American:genderMale  0.2126  0.0475
SE                              0.1746  0.1913
```

Minneapolis Police Stops

- The only coefficient that's substantially affected is "raceNative American:genderMale", which alters the shape of the race $\times$ gender interaction:

```
> plot(predictorEffect("race", mod.mpls.1),
+   lines=list(multiline=TRUE, lwd=3),
+   symbols=list(cex=1.5, pch=16),
+   confint=list(style="bars"),
+   axes=list(y=list(
+     lab="Probability of Being Searched",
+     type="response")),
+   lattice=list(key.args=list(
+     cex=1, cex.title=1.2)),
+   grid=TRUE)
```



References

Cunningham, R. and Heathcote, C. (1989). Estimating a non-gaussian regressino model with multicolinearity. *Australian Journal of Statistics*, 31:12–17.

Fox, J. (2016). *Applied Regression Analysis and Generalized Linear Models*. Sage Publications, Thousand Oaks, CA, third edition.

Fox, J. and Weisberg, S. (2018a). *An R Companion to Applied Regression (in press)*. Sage Publications, Thousand Oaks CA, third edition.

Fox, J. and Weisberg, S. (2018b). Visualizing fit and lack of fit in complex regression models with predictor effect plots and partial residuals. *Journal of Statistical Software (in press)*.

Hawkins, D. M. and Weisberg, S. (2017). Combining the Box-Cox power and generalised log transformations to accommodate negative responses in linear and mixed-effects linear models. *South African Statistics Journal*, 51:pp. 317–328.

Weisberg, S. (2014). *Applied Linear Regression*. John Wiley & Sons, Hoboken, NJ, fourth edition.