Use the GLM, Luke!

How the use of proper statistical models can increase statistical power in ecotoxicological experiments.

Eduard Szöcs

January 15, 2015

1 Supplement 2 - Examples

1.1 Binomial data

1.1.1 Introduction

Here we will show how to analyse binomial data. Data is provided in Newman (2012) (example 5.1, page 223) and EPA (2002). Ten fathead minnow (*Pimephales promelas* larvals were exposed to sodium pentachlorophenol (NaPCP) and proportions of the total number alive at the end of the exposure reported.

First we load the data:

```
df <- read.table(header = TRUE, text = 'conc A B C D
0 1 1 0.9 0.9
32 0.8 0.8 1 0.8
64 0.9 1 1 1
128 0.9 0.9 0.8 1
256 0.7 0.9 1 0.5
512 0.4 0.3 0.4 0.2')
df
```

```
##   conc   A   B   C   D
## 1    0 1.0 1.0 0.9 0.9
## 2   32 0.8 0.8 1.0 0.8
## 3   64 0.9 1.0 1.0 1.0
## 4  128 0.9 0.9 0.8 1.0
## 5  256 0.7 0.9 1.0 0.5
## 6  512 0.4 0.3 0.4 0.2
```

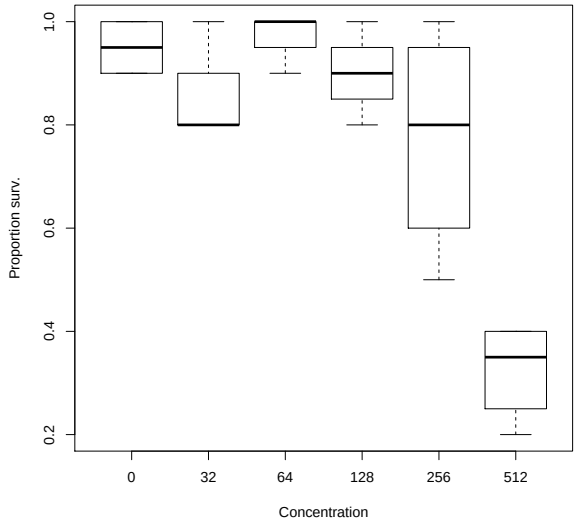
The we do some house-keeping, reformat the data to the long format and convert concentration to a factor:

```
require(reshape2)
# wide to long
dfm <- melt(df, id.vars = 'conc', value.name = 'y', variable.name = 'tank')
# conc as factor
dfm$conc <- factor(dfm$conc)
head(dfm)
```

```
##   conc tank   y
## 1    0    A 1.0
## 2   32    A 0.8
## 3   64    A 0.9
## 4  128    A 0.9
## 5  256    A 0.7
## 6  512    A 0.4
```

Let's have a first look at the data:

```
boxplot(y ~ conc, data = dfm,
        xlab = 'Concentration', ylab = 'Proportion surv.')
```

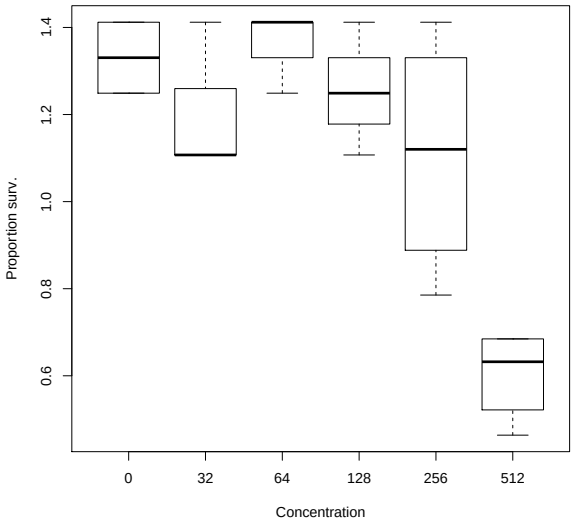


1.1.2 Transforming data

Next we arcsin transform the proportions:

```
dfm$y_asin <- ifelse(dfm$y == 1, asin(1) - asin(sqrt(1/40)),
                                          ifelse(dfm$y == 0, asin(sqrt(1/40)),
                                                 asin(sqrt(dfm$y))
                                                 )
                                          )
```

```
boxplot(y_asin ~ conc, data = dfm,
        xlab = 'Concentration', ylab = 'Proportion surv.')
```



1.1.3 Analysing data

We will use the bbmle-package to fit the models, as we used this also for the simulations. However, also standard R functions can be use (see below). Let's start with the model assuming a normal distribution of transformed proportions.

```
require(bbmle)
mod_normal <- mle2(y_asin ~ dnorm(mean = mu, sd = s),
                   parameters = list(mu ~ conc, s ~ 1),
                   data = dfm,
                   start = list(mu = 0.5, s = 1)
                   )
summary(mod_normal)

## Maximum likelihood estimation
##
## Call:
## mle2(minuslogl = y_asin ~ dnorm(mean = mu, sd = s), start = list(mu = 0.5,
##     s = 1), data = dfm, parameters = list(mu ~ conc, s ~ 1))
##
## Coefficients:
##                Estimate Std. Error z value   Pr(z)
## mu.(Intercept)   1.3305     0.0666   19.97 < 2e-16 ***
## mu.conc32       -0.1472     0.0942   -1.56   0.118
## mu.conc64        0.0407     0.0942    0.43   0.665
## mu.conc128      -0.0762     0.0942   -0.81   0.419
## mu.conc256      -0.2211     0.0942   -2.35   0.019 *
## mu.conc512      -0.7274     0.0942   -7.72 1.2e-14 ***
## s                0.1333     0.0192    6.93 4.3e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## -2 log L: -28.64
```

This may gives some warnings as we did not put boundaries on the parameters (s cannot be 0 or lower), however we can ignore them. The summary gives us the estimated parameters, their standard error and accompanying p-values. Note that this model is parametrised a contrast to the control group so we can directly use the summary-output to determine the LOEC. Note that these p-values are not adjusted for multiple testing (you can use p.adjust() for this).

The perform a Likelihood-Ratio-Test we specify a null model and compare both.

```
mod_normal.null <- update(mod_normal,
                      parameters = list(mu ~ 1, sd ~ 1)
                      )
anova(mod_normal, mod_normal.null)

## Likelihood Ratio Tests
## Model 1: mod_normal, y_asin~dnorm(mean=mu,sd=s): mu~conc, s~1
## Model 2: mod_normal.null, y_asin~dnorm(mean=mu,sd=s): mu~1, sd~1
##   Tot Df Deviance Chisq Df Pr(>Chisq)
## 1      7   -28.64
## 2      2     8.49  37.1  5    5.7e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

With base R we could do:

```
mod_normal <- lm(y_asin ~ conc, data = dfm)
drop1(mod_normal, test = 'Chisq')

## Single term deletions
##
## Model:
## y_asin ~ conc
##        Df Sum of Sq   RSS   AIC Pr(>Chi)
## <none>              0.426 -84.7
## conc    5      1.57 2.001 -57.6  5.7e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

or using the lmtest package (Zeileis and Hothorn, 2002):

```
require(lmtest)
lrtest(mod_normal)

## Likelihood ratio test
##
## Model 1: y_asin ~ conc
## Model 2: y_asin ~ 1
##   #Df LogLik Df Chisq Pr(>Chisq)
## 1   7  14.32
## 2   2  -4.24 -5  37.1    5.7e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Next we assume that the number of surviving larvals are drawn from a binomial distribution. First we backtransform the proportions reported to counts of surviving and dead larvals:

```
dfm$surv <- dfm$y * 10
dfm$dead <- 10 - dfm$surv
```

And then we specify the logistic model:

```
mod_bin <- mle2(surv ~ dbinom(size = 10, prob = plogis(lp)),
                parameters = list(lp ~ conc),
                data = dfm,
                start = list(lp = 0)
                )
summary(mod_bin)

## Maximum likelihood estimation
##
## Call:
## mle2(minuslogl = surv ~ dbinom(size = 10, prob = plogis(lp)),
##     start = list(lp = 0), data = dfm, parameters = list(lp ~
##         conc))
##
## Coefficients:
##                Estimate Std. Error z value   Pr(z)
## lp.(Intercept)    2.944      0.725    4.06 4.9e-05 ***
## lp.conc32        -1.210      0.850   -1.42   0.155
## lp.conc64         0.719      1.246    0.58   0.564
## lp.conc128       -0.747      0.897   -0.83   0.405
## lp.conc256       -1.708      0.818   -2.09   0.037 *
## lp.conc512       -3.675      0.800   -4.59 4.4e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## -2 log L: 60.86
```

And perform the LRT

```
mod_bin.null <- update(mod_bin,
                       parameters = list(lp ~ 1))
anova(mod_bin, mod_bin.null)

## Likelihood Ratio Tests
## Model 1: mod_bin, surv~dbinom(size=10,prob=plogis(lp)): lp~conc
## Model 2: mod_bin.null, surv~dbinom(size=10,prob=plogis(lp)): lp~1
##   Tot Df Deviance Chisq Df Pr(>Chisq)
## 1      6     60.9
## 2      1    125.6  64.8  5    1.2e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

With base R we could use

```r
mod_bin_b <- glm(cbind(surv, dead) ~ conc, data = dfm, family = binomial(link = 'logit'))
summary(mod_bin_b)
```

```
##
## Call:
## glm(formula = cbind(surv, dead) ~ conc, family = binomial(link = "logit"),
##     data = dfm)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.898  -0.572   0.000   0.787   2.258
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.944      0.725    4.06  4.9e-05 ***
## conc32        -1.210      0.850   -1.42    0.155
## conc64         0.719      1.246    0.58    0.564
## conc128       -0.747      0.897   -0.83    0.405
## conc256       -1.708      0.818   -2.09    0.037 *
## conc512       -3.675      0.800   -4.59  4.4e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 88.672  on 23  degrees of freedom
## Residual deviance: 23.889  on 18  degrees of freedom
## AIC: 72.86
##
## Number of Fisher Scoring iterations: 5
```

```r
drop1(mod_bin_b, test = 'Chisq')
```

```
## Single term deletions
##
## Model:
## cbind(surv, dead) ~ conc
##        Df Deviance   AIC  LRT Pr(>Chi)
## <none>        23.9  72.9
## conc    5     88.7 127.6 64.8  1.2e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

1.2 Count data

1.2.1 Introduction

In this example we will analyse data from (Brock et al., 2014). The data are count of mayfly larvae in Macroinvertebrate Artificial Substrate Samplers in 18 mesocosms at one sampling day. There are 5 Treatments and one control group.

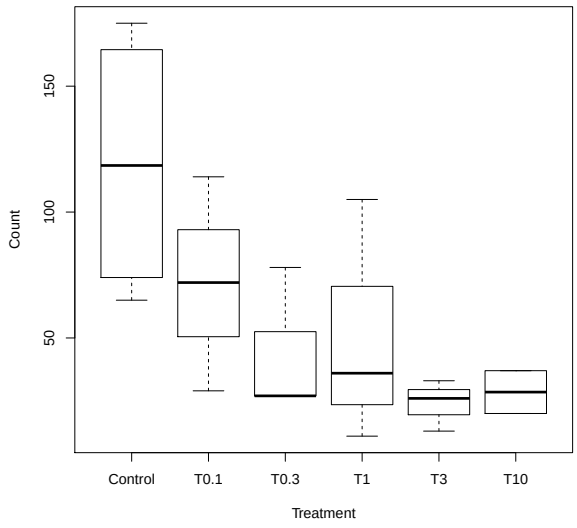
First we load the data and bring it to the long format and remove NA values.

```
df <- read.table(header = TRUE, text = 'Control  T0.1 T0.3 T1 T3 T10
175 29 27 36 26 20
65 114 78 11 13 37
154 72 27 105 33 NA
83 NA NA NA NA NA
')
dfm <- melt(df, value.name = 'n', variable.name = 'treatment')
dfm <- dfm[!is.na(dfm['n']), ]
head(dfm)

##   treatment   n
## 1   Control 175
## 2   Control  65
## 3   Control 154
## 4   Control  83
## 5      T0.1  29
## 6      T0.1 114
```

Next we have a look at the data:

```
boxplot(n ~ treatment, data = dfm, xlab = 'Treatment', ylab = 'Count')
```

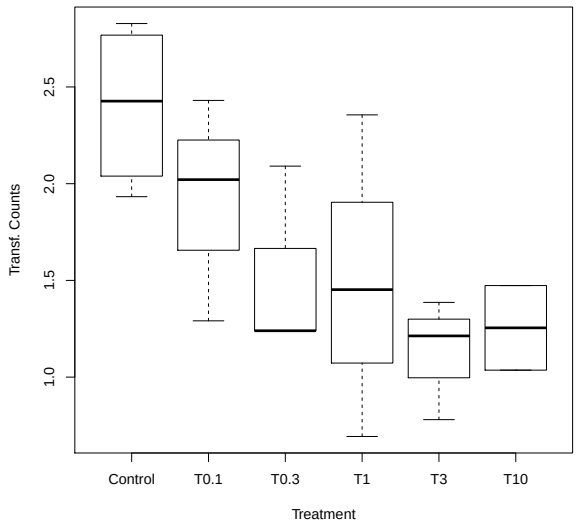


1.2.2 Transforming data

Next we transform the data using a $\ln(Ax + 1)$ transformation:

```
A <- 1 / min(dfm$n[dfm$n != 0])
dfm$nt <- log(A * dfm$n + 1)
```

```r
boxplot(nt ~ treatment, data = dfm,
        xlab = 'Treatment', ylab = 'Transf. Counts')
```



1.3 Analsying data

Again we start with the model assuming a normal distribution of transformed counts:

```r
mod_normal <- mle2(nt ~ dnorm(mean = mu, sd = s),
                   parameters = list(mu ~ treatment, s ~ 1),
                   data = dfm,
                   start = list(mu = mean(dfm$nt), s = sd(dfm$nt)))
summary(mod_normal)

## Maximum likelihood estimation
##
## Call:
## mle2(minuslogl = nt ~ dnorm(mean = mu, sd = s), start = list(mu = mean(dfm$nt),
##     s = sd(dfm$nt)), data = dfm, parameters = list(mu ~ treatment,
##     s ~ 1))
##
## Coefficients:
##                   Estimate Std. Error z value   Pr(z)
## mu.(Intercept)     2.4035     0.2169   11.08 < 2e-16 ***
## mu.treatmentT0.1  -0.4894     0.3313   -1.48 0.13956
## mu.treatmentT0.3  -0.8801     0.3313   -2.66 0.00789 **
## mu.treatmentT1    -0.9032     0.3313   -2.73 0.00640 **
## mu.treatmentT3    -1.2771     0.3313   -3.86 0.00012 ***
## mu.treatmentT10   -1.1488     0.3756   -3.06 0.00222 **
## s                  0.4337     0.0723    6.00   2e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## -2 log L: 21.01
```

```
mod_normal.null <- update(mod_normal,
                          parameters = list(mu ~ 1, s ~ 1))
anova(mod_normal, mod_normal.null)

## Likelihood Ratio Tests
## Model 1: mod_normal, nt~dnorm(mean=mu,sd=s): mu~treatment, s~1
## Model 2: mod_normal.null, nt~dnorm(mean=mu,sd=s): mu~1, s~1
##   Tot Df Deviance Chisq Df Pr(>Chisq)
## 1      7     21.0
## 2      2     34.3  13.3  5      0.021 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

With base R you could do:

```
mod_normal_b <- lm(nt ~ treatment, data = dfm)
summary(mod_normal_b)
drop1(mod_normal_b, test = 'Chisq')
```

Next we analyse the raw counts assuming a poisson distribution with a log link:

```
mod_pois <- mle2(n ~ dpois(lambda = exp(logmu)),
                 parameters = list(logmu ~ treatment),
                 data = dfm,
                 start = list(logmu = mean(dfm$nt)))
summary(mod_pois)

## Maximum likelihood estimation
##
## Call:
## mle2(minuslogl = n ~ dpois(lambda = exp(logmu)), start = list(logmu = mean(dfm$nt)),
##     data = dfm, parameters = list(logmu ~ treatment))
##
## Coefficients:
##                     Estimate Std. Error z value   Pr(z)
## logmu.(Intercept)     4.7812     0.0458  104.42 < 2e-16 ***
## logmu.treatmentT0.1  -0.5093     0.0821   -6.20 5.7e-10 ***
## logmu.treatmentT0.3  -0.9970     0.0983  -10.14 < 2e-16 ***
## logmu.treatmentT1    -0.8560     0.0931   -9.19 < 2e-16 ***
## logmu.treatmentT3    -1.6032     0.1264  -12.68 < 2e-16 ***
## logmu.treatmentT10   -1.4313     0.1401  -10.21 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## -2 log L: 375.6
```

Or with base R

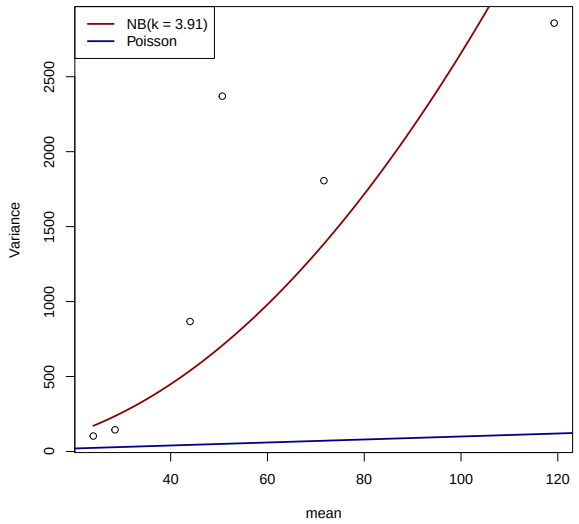
```
mod_pois <- glm(n ~ treatment, data = dfm, family = poisson(link = 'log'))
summary(mod_pois)
```

But is a poisson distribution appropriate here? A property of the poisson distribution is that its variance is equal to the mean. A simple diagnostic would be to plot group variances vs group means, other more formal statistics are also available.

```
require(plyr)
musd <- ddply(dfm, .(treatment), summarise,
              mu = mean(n),
              var = var(n))
musd
```

```
##   treatment     mu    var
## 1   Control 119.25 2857.6
## 2      T0.1  71.67 1806.3
## 3      T0.3  44.00  867.0
## 4        T1  50.67 2370.3
## 5        T3  24.00  103.0
## 6       T10  28.50  144.5
```

```
plot(var ~ mu, data = musd, xlab = 'mean', ylab = 'Variance')
abline(a = 0, b = 1, col = 'darkblue', lwd = 2)
curve(x + (x^2 / 3.91), from = 24, to = 119.25, add = TRUE, col = 'darkred', lwd = 2)
legend('topleft', c('NB(k = 3.91)', 'Poisson'),
       col = c('darkred', 'darkblue'),
       lty = c(1,1),
       lwd = c(2,2))
```



We clearly see that the variance increases much more than would be expected under the poisson distribution (the data is overdispersed). One possibility to deal with overdispersion is using a negative binomial distribution:

```
mod_negbin <- mle2(n ~ dnbinom(mu = exp(logmu), size = k),
                   parameters = list(logmu ~ treatment, k ~ 1),
                   data = dfm,
                   start = list(logmu = log(mean(dfm$n)), k = 1))
```

```
summary(mod_negbin)

## Maximum likelihood estimation
##
## Call:
## mle2(minuslogl = n ~ dnbinom(mu = exp(logmu), size = k), start = list(logmu = log(mean
##     k = 1), data = dfm, parameters = list(logmu ~ treatment,
##     k ~ 1))
##
## Coefficients:
##                      Estimate Std. Error z value   Pr(z)
## logmu.(Intercept)       4.781      0.257   18.60 < 2e-16 ***
## logmu.treatmentT0.1    -0.509      0.395   -1.29  0.1974
## logmu.treatmentT0.3    -0.997      0.399   -2.50  0.0124 *
## logmu.treatmentT1      -0.856      0.398   -2.15  0.0313 *
## logmu.treatmentT3      -1.603      0.407   -3.94 8.1e-05 ***
## logmu.treatmentT10     -1.431      0.460   -3.11  0.0019 **
## k                       3.906      1.366    2.86  0.0042 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## -2 log L: 167.2

mod_negbin.null <- update(mod_negbin,
                          parameters = list(logmu ~ 1))
anova(mod_negbin, mod_negbin.null)

## Likelihood Ratio Tests
## Model 1: mod_negbin, n~dnbinom(mu=exp(logmu),size=k): logmu~treatment, k~1
## Model 2: mod_negbin.null, n~dnbinom(mu=exp(logmu),size=k): logmu~1
##   Tot Df Deviance Chisq Df Pr(>Chisq)
## 1      7      167
## 2      2      181    14  5      0.016 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Or using the MASS package (Venables and Ripley, 2002):

```
require(MASS)
mod_negbin_m <- glm.nb(n ~ treatment, data = dfm)
summary(mod_negbin_m)

##
## Call:
## glm.nb(formula = n ~ treatment, data = dfm, init.theta = 3.905898474,
##     link = log)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.255  -0.849  -0.302   0.595   1.590
##
```

```
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)      4.781      0.257   18.60  < 2e-16 ***
## treatmentT0.1   -0.509      0.395   -1.29   0.1975
## treatmentT0.3   -0.997      0.399   -2.50   0.0124 *
## treatmentT1     -0.856      0.398   -2.15   0.0313 *
## treatmentT3     -1.603      0.407   -3.94  8.1e-05 ***
## treatmentT10    -1.431      0.460   -3.11   0.0019 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(3.906) family taken to be 1)
##
##     Null deviance: 39.057  on 17  degrees of freedom
## Residual deviance: 18.611  on 12  degrees of freedom
## AIC: 181.2
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  3.91
##           Std. Err.:  1.37
##
##  2 x log-likelihood:  -167.24

lrtest(mod_negbin_m)

## Likelihood ratio test
##
## Model 1: n ~ treatment
## Model 2: n ~ 1
##   #Df LogLik Df Chisq Pr(>Chisq)
## 1   7  -83.6
## 2   2  -90.6 -5    14      0.016 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note, that we cannot use drop1() with glm.nb()!

References

Brock, T. C. M., Hammers-Wirtz, M., Hommen, U., Preuss, T. G., Ratte, H.-T., Roessink, I., Strauss, T., and Van den Brink, P. J. (2014). The minimum detectable difference (MDD) and the interpretation of treatment-related effects of pesticides in experimental ecosystems. *Environmental Science and Pollution Research*.

EPA (2002). *Methods for Measuring the Acute Toxicity of Effluents and Receiving Waters to Freshwater and Marine Organisms*.

Newman, M. C. (2012). *Quantitative ecotoxicology*. Taylor & Francis, Boca Raton, FL.

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

Zeileis, A. and Hothorn, T. (2002). Diagnostic checking in regression relationships. *R News*, 2(3):7–10.