Ecotoxicology is not normal.

A comparison of statistical approaches for analysis of count and proportion data in ecotoxicology.

Eduard Szöcs, Ralf B. Schäfer

March 23, 2015

Supplement 2 - Worked R examples

1 Count data example

1.1 Introduction

In this example we will analyse data from (Brock et al., 2015). 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.

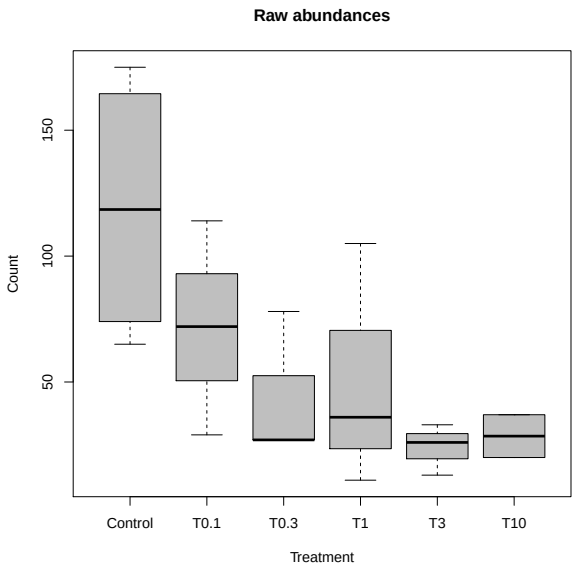
First, we load the data, bring it to the long format and remove NA values.

```
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')
require(reshape2)
dfm <- melt(df, value.name = 'abu', variable.name = 'treatment')
dfm <- dfm[!is.na(dfm['abu']), ]
head(dfm)

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

This results in a table with two columns - one indicating the treatment and one with the measured abundances.

Let's have a first look at the data:

```
boxplot(abu ~ treatment, data = dfm, xlab = 'Treatment',
        ylab = 'Count', col = 'grey75', main = 'Raw abundances')
```



We clearly see a treatment related response. Moreover, we may note that variances are increasing with increasing abundances.

1.2 Assuming a normal distribution of transformed abundances

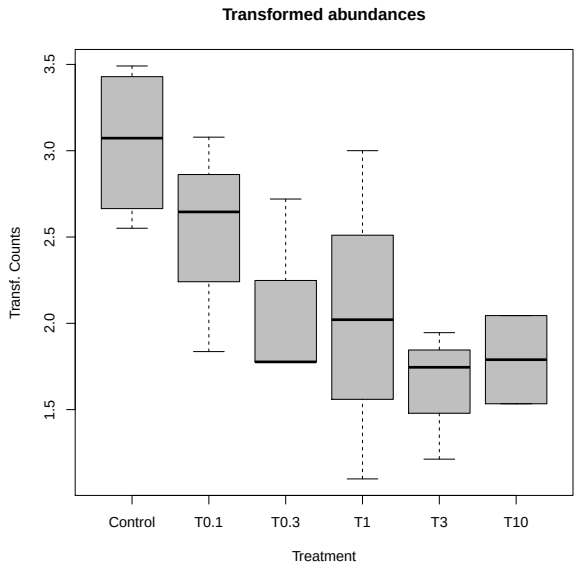
1.2.1 Data transformation

Next we transform the data using a $\ln(Ax + 1)$ transformation. A is chosen so that the term Ax equals two for the lowest non-zero abundance. We add these transformed abundances as extra column to our table.

```
A <- 2 / min(dfm$abu[dfm$abu != 0])
A

## [1] 0.1818182

dfm$abu_t <- log(A * dfm$abu + 1)
head(dfm)

##   treatment abu    abu_t
## 1   Control 175 3.490983
## 2   Control  65 2.550865
## 3   Control 154 3.367296
## 4   Control  83 2.778254
## 5      T0.1  29 1.836211
## 6      T0.1 114 3.078568
```

It looks like the transformation does a good job in equalizing the variances:

```
boxplot(abu_t ~ treatment, data = dfm,
        xlab = 'Treatment', ylab = 'Transf. Counts',
        col = 'grey75', main = 'Transformed abundances')
```



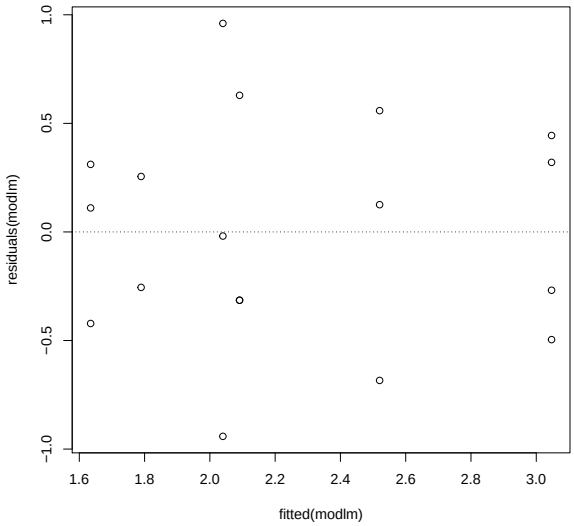
1.2.2 Model fitting

The model from eqn. 2 can be easily fitted using the `lm()` function:

```
modlm <- lm(abu_t ~ treatment, data = dfm)
```

The residuals vs. fitted values diagnostic plot show no problematic pattern, though it might be difficult to decide with such a small sample size

```
plot(residuals(modlm) ~ fitted(modlm))
abline(h = 0, lty = 'dotted')
```



The `summary()` gives the estimated parameters with standard errors and Wald t tests:

```
summary(modlm)

##
## Call:
## lm(formula = abu_t ~ treatment, data = dfm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.94133 -0.31454  0.04576  0.31813  0.96033
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     3.0468     0.2970  10.260 2.71e-07 ***
## treatmentT0.1  -0.5267     0.4536  -1.161  0.26814
## treatmentT0.3  -0.9558     0.4536  -2.107  0.05682 .
## treatmentT1    -1.0069     0.4536  -2.220  0.04646 *
## treatmentT3    -1.4121     0.4536  -3.113  0.00897 **
## treatmentT10   -1.2575     0.5144  -2.445  0.03089 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.5939 on 12 degrees of freedom
## Multiple R-squared:  0.5167,Adjusted R-squared:  0.3154
## F-statistic: 2.566 on 5 and 12 DF,  p-value: 0.08406
```

1.2.3 Inference on general treatment effect

Or, if you want to have the ANOVA table with an F-test:

```
summary.aov(modlm)

##             Df Sum Sq Mean Sq F value Pr(>F)
## treatment    5  4.526  0.9052   2.566 0.0841 .
## Residuals   12  4.233  0.3528
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From this output we might infer that we cannot detect any treatment effect ($F = 2.566$, $p = 0.084$).

1.2.4 Inference on LOEC

Let's move on to the LOEC determination. This can be easily done using the multcomp package (Hothorn et al., 2008):

Here we perform a one-sided (`alternative = 'less'`) using Dunnett contrasts of treatment (`mcp(treatment='Dunnett')`). Moreover, we adjust for multiple testing using Holm's method (`test = adjusted('holm')`):

```
require(multcomp)
summary(glht(modlm, linfct = mcp(treatment = 'Dunnett'),  alternative = 'less'),
        test = adjusted('holm'))

##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Dunnett Contrasts
##
##
## Fit: lm(formula = abu_t ~ treatment, data = dfm)
##
## Linear Hypotheses:
##                       Estimate Std. Error t value Pr(<t)
## T0.1 - Control >= 0  -0.5267     0.4536  -1.161 0.1341
## T0.3 - Control >= 0  -0.9558     0.4536  -2.107 0.0697 .
## T1 - Control >= 0    -1.0069     0.4536  -2.220 0.0697 .
## T3 - Control >= 0    -1.4121     0.4536  -3.113 0.0224 *
## T10 - Control >= 0   -1.2575     0.5144  -2.445 0.0618 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

This indicates that only treatment 3 shows a statistically significant from control and is the determined LOEC. The column `'Estimate'` gives the estimated difference in means between treatments and control and `'Std. Error'` the standard errors of these estimated parameters.

To make Williams type comparisons simply change `'Dunnett'` to `'Williams'`:

```
summary(glht(modlm, linfct = mcp(treatment = 'Williams'),
             alternative = 'less'),
        test = adjusted('holm'))

##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Williams Contrasts
##
##
## Fit: lm(formula = abu_t ~ treatment, data = dfm)
##
## Linear Hypotheses:
##          Estimate Std. Error t value Pr(<t)
## C 1 >= 0  -1.2575     0.5144  -2.445 0.0154 *
## C 2 >= 0  -1.3503     0.3984  -3.389 0.0134 *
## C 3 >= 0  -1.2215     0.3637  -3.359 0.0134 *
## C 4 >= 0  -1.1491     0.3468  -3.313 0.0134 *
## C 5 >= 0  -1.0157     0.3367  -3.016 0.0134 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

Note, this multiple contrats test is different from the original Williams test (Williams, 1972), because of different variance estimators (Bretz, 1999).

1.3 Assuming a Poisson distribution of abundances

1.3.1 Model fitting

We are dealing with count data, so a Poisson GLM might be a good choice. GLMs can be fitted using the `glm()` function and here we fit the model from eqn. 3:

```
modpois <- glm(abu ~ treatment, data = dfm, family = poisson(link = 'log'))
```

Here `family = poisson(link = 'log')` specifies that we want to fit a poisson model using a log link between response and predictors.

The `summary` gives the estimated parameters, standard errors and Wald Z tests:

```
(sum_pois <- summary(modpois))

##
## Call:
## glm(formula = abu ~ treatment, family = poisson(link = "log"),
##     data = dfm)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -6.7625  -2.7621  -0.8219   2.7172   6.6602
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    4.78122    0.04579 104.423  < 2e-16 ***
## treatmentT0.1 -0.50920    0.08214  -6.199 5.69e-10 ***
## treatmentT0.3 -0.99703    0.09835 -10.138  < 2e-16 ***
## treatmentT1   -0.85595    0.09314  -9.190  < 2e-16 ***
## treatmentT3   -1.60317    0.12643 -12.680  < 2e-16 ***
## treatmentT10  -1.43132    0.14014 -10.213  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 604.79  on 17  degrees of freedom
## Residual deviance: 273.77  on 12  degrees of freedom
## AIC: 387.63
##
## Number of Fisher Scoring iterations: 5
```

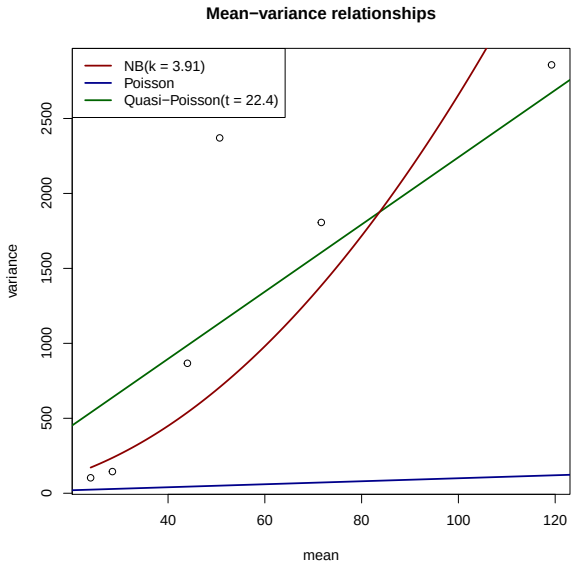
But is a poisson distribution appropriate here? A property of the poisson distribution is that its variance is equal to the mean. A simple diagnostic would be to plot group variances vs. group means:

```
require(plyr)
# mean and variance per treatment
musd <- ddply(dfm, .(treatment), summarise,
              mu = mean(abu),
              var = var(abu))
musd
```

```
##   treatment        mu      var
## 1   Control 119.25000 2857.583
## 2      T0.1  71.66667 1806.333
## 3      T0.3  44.00000  867.000
## 4        T1  50.66667 2370.333
## 5        T3  24.00000  103.000
## 6       T10  28.50000  144.500
```

```
# plot mean vs var
plot(var ~ mu, data = musd,
     xlab = 'mean', ylab = 'variance', main = 'Mean-variance relationships')
# poisson
abline(a = 0, b = 1, col = 'darkblue', lwd = 2)
# quasi-Poisson
abline(a = 0, b = 22.41, col = 'darkgreen', lwd = 2)
# negative binomial
curve(x + (x^2 / 3.91), from = 24, to = 119.25, add = TRUE,
      col = 'darkred', lwd = 2)
```

```
legend('topleft',
       legend = c('NB(k = 3.91)', 'Poisson', 'Quasi-Poisson(t = 22.4)'),
       col = c('darkred', 'darkblue', 'darkgreen'),
       lty = c(1,1, 1),
       lwd = c(2,2, 2))
```



I also added the assumed mean-variance relationships of the Poisson, quasi-Poisson and negative binomial models (see below). We clearly see that the variance increases much more than would be expected under the poisson distribution (the data is overdispersed). Moreover, we could check overdispersion from the `summary`: If the ratio of residual deviance to degrees of freedom is >1 the data is overdispersed.

```
sum_pois$deviance / sum_pois$df.residual

## [1] 22.81412
```

1.4 Apply quasi-Poisson to deal with overdispersion

The plot above suggests that the variance may increasing stronger than the mean and a quasi-Poisson or negative binomial model might be more appropriate for this data.

1.4.1 Model fitting

Fitting a quasi-Poisson model (eqn. 4) is straight forward:

```
modqpois <- glm(abu ~ treatment, data = dfm, family = 'quasipoisson')
```

The summary gives the estimated parameters:

```
summary(modqpois)
```

```
##
## Call:
## glm(formula = abu ~ treatment, family = "quasipoisson", data = dfm)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.7625  -2.7621  -0.8219   2.7172   6.6602
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)     4.7812     0.2168  22.058 4.43e-11 ***
## treatmentT0.1  -0.5092     0.3889  -1.309   0.2149
## treatmentT0.3  -0.9970     0.4656  -2.142   0.0534 .
## treatmentT1    -0.8560     0.4409  -1.941   0.0761 .
## treatmentT3    -1.6032     0.5985  -2.679   0.0201 *
## treatmentT10   -1.4313     0.6634  -2.157   0.0519 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 22.41055)
##
##     Null deviance: 604.79  on 17  degrees of freedom
## Residual deviance: 273.77  on 12  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

, with the dispersion parameter $\Theta = 22.41055$. Note, that the parameter estimates are the same as from the Poisson model, only the standard errors are scaled/wider.

1.4.2 Inference on general treatment effect

An F-test can be performed using `drop1()`:

```
drop1(modqpois, test = 'F')
```

```
## Single term deletions
##
## Model:
## abu ~ treatment
##           Df Deviance F value  Pr(>F)
## <none>         273.77
## treatment  5   604.79  2.9019 0.06059 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we would reject that there is treatment effect (at alpha = 0.05).

1.4.3 Inference on LOEC

The LOEC can be determined with `multcomp`:

```
summary(glht(modqpois, linfct = mcp(treatment = 'Dunnett'),
             alternative = 'less'),
        test = adjusted('holm'))

##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Dunnett Contrasts
##
##
## Fit: glm(formula = abu ~ treatment, family = "quasipoisson", data = dfm)
##
## Linear Hypotheses:
##                      Estimate Std. Error z value Pr(<z)
## T0.1 - Control >= 0  -0.5092     0.3889  -1.309 0.0952 .
## T0.3 - Control >= 0  -0.9970     0.4656  -2.142 0.0619 .
## T1 - Control >= 0    -0.8560     0.4409  -1.941 0.0619 .
## T3 - Control >= 0    -1.6032     0.5985  -2.679 0.0185 *
## T10 - Control >= 0   -1.4313     0.6634  -2.157 0.0619 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

, which determines 3 mg/L as LOEC.

1.5 Assuming a negative binomial distribution of abundances

1.5.1 Model fitting

To fit a negative binomial GLM (eqn. 5) we could use `glm.nb()` from the MASS package (Venables and Ripley, 2002):

```
require(MASS)
modnb <- glm.nb(abu ~ treatment, data = dfm)
```

The estimated parameters:

```
summary(modnb)

##
## Call:
## glm.nb(formula = abu ~ treatment, data = dfm, init.theta = 3.905898474,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2554  -0.8488  -0.3020   0.5954   1.5899
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)     4.7812     0.2571  18.596  < 2e-16 ***
```

```
## treatmentT0.1  -0.5092     0.3951  -1.289  0.19746
## treatmentT0.3  -0.9970     0.3988  -2.500  0.01241 *
## treatmentT1    -0.8560     0.3975  -2.153  0.03130 *
## treatmentT3    -1.6032     0.4066  -3.943 8.05e-05 ***
## treatmentT10   -1.4313     0.4601  -3.111  0.00186 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(3.9059) family taken to be 1)
##
##     Null deviance: 39.057  on 17  degrees of freedom
## Residual deviance: 18.611  on 12  degrees of freedom
## AIC: 181.24
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  3.91
##           Std. Err.:  1.37
##
##  2 x log-likelihood:  -167.238
```

, with $\kappa = 3.905898$ (glm.nb uses a slightly other parametrisation).

1.5.2 Inference on general treatment effect (LR-test)

For an LR-Test we need to first fit a reduced model:

```
modnb.null <- glm.nb(abu ~ 1, data = dfm)
```

, so that the dispersion parameter κ is re-estimated for the reduced model. Then we can compare these two models with a LR-Test:

```
anova(modnb, modnb.null, test = 'Chisq')

## Likelihood ratio tests of Negative Binomial Models
##
## Response: abu
##       Model    theta Resid. df    2 x log-lik.   Test    df LR stat.
## 1         1 1.861577        17       -181.2281
## 2 treatment 3.905898        12       -167.2383 1 vs 2     5 13.98985
##    Pr(Chi)
## 1
## 2 0.015674
```

, which suggests a treatment related effect on abundances.

1.5.3 Inference on general treatment effect (parametric bootstrap)

To test the LR statistic using paramtric bootstrap, we use two custom functions:

The first function `myPBrefdist` generates a boostrap sample and return the LR statistic for this sample:

```r
#' PB of LR statistic
#' @param m1 Full model
#' @param m0 reduced model
#' @param  data data used in the models
#' @return LR of boostrap
# generate reference distribution
myPBrefdist <- function(m1, m0, data){
  # simulate from null
  x0 <- simulate(m0)
  # refit with new data
  newdata0 <- data
  newdata0[ , as.character(formula(m0)[[2]])] <- x0
  m1r <-  try(update(m1, .~., data = newdata0), silent = TRUE)
  m0r <- try(update(m0, .~., data = newdata0), silent = TRUE)
  # check convergence (otherwise return NA for LR)
  if(inherits(m0r, "try-error") | inherits(m1r, "try-error")){
    LR <- 'convergence error'
  } else {
    if(!is.null(m0r[['th.warn']]) | !is.null(m1r[['th.warn']])){
      LR <- 'convergence error'
    } else {
      LR <- -2 * (logLik(m0r) - logLik(m1r))
    }
  }
  return(LR)
}
```

The second one (`myPBmodcomp`) repeats `myPBrefdist` many time and returns a p-value:

```r
#' generate LR distribution and return p value
#' @param m1 Full model
#' @param m0 reduced model
#' @param data data used in m1 and m0
#' @param npb number of bootstrap samples
#' @return p-value of boostrapped LR values
myPBmodcomp <- function(m1, m0, data, npb){
  ## calculate reference distribution
  LR <- replicate(npb, myPBrefdist(m1 = m1, m0 = m0, data = data),
                  simplify = TRUE)
  LR <- as.numeric(LR)
  nconv_LR <- sum(!is.na(LR))
  ## original stats
  LRo <- c(-2 * (logLik(m0) - logLik(m1)))
  ## p-value from parametric bootstrap
  p.pb <- mean(c(LR, LRo) >= LRo, na.rm = TRUE)
  return(list(nconv_LR = nconv_LR, p.pb = p.pb))
}
```

Sounds complicated, but we can easily apply this to the negativ binomial model using:

```r
set.seed(1234)
myPBmodcomp(modnb, modnb.null, data = dfm, npb = 500)

## $nconv_LR
## [1] 499
##
## $p.pb
## [1] 0.042
```

Here, we specify to generate 500 bootstrap samples (`npb = 500`). Of these 500 samples, 499 converged (`nconv_LR`) (one did not and throws some errors) and gives a p-value of 0.042.

1.5.4 Inference on LOEC

This is similar to the other parametric models:

```r
summary(glht(modnb, linfct = mcp(treatment = 'Dunnett'),  alternative = 'less'),
        test = adjusted('holm'))

##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Dunnett Contrasts
##
##
## Fit: glm.nb(formula = abu ~ treatment, data = dfm, init.theta = 3.905898474,
##     link = log)
##
## Linear Hypotheses:
##                      Estimate Std. Error z value   Pr(<z)
## T0.1 - Control >= 0  -0.5092     0.3951  -1.289 0.098731 .
## T0.3 - Control >= 0  -0.9970     0.3988  -2.500 0.018615 *
## T1 - Control >= 0    -0.8560     0.3975  -2.153 0.031300 *
## T3 - Control >= 0    -1.6032     0.4066  -3.943 0.000201 ***
## T10 - Control >= 0   -1.4313     0.4601  -3.111 0.003727 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

which suggests a LOEC at the 0.3 mg/l treatment.

1.6 Non-parametric methods

1.6.1 Kruskal-Wallis Test

We can use the Kruskal-Wallies test to check if there is a difference between treatments:

```r
kruskal.test(abu ~ treatment, data = dfm)

##
##  Kruskal-Wallis rank sum test
##
```

```
## data:  abu by treatment
## Kruskal-Wallis chi-squared = 8.219, df = 5, p-value = 0.1446
```

1.6.2 Pairwise Wilcoxon test

To determine the LOEC we could use a Pairwise Wilcoxon test. The built-in `pairwise.wilcox.test()` compares by default all levels (Tukey-contrasts). We are only interested in a subset of these comparisons (Dunnett-contrast).

Therefore, we use a custom function, which is a wrapper around `wilcox.exact()` from the exactRankTests package:

```
#' pairwise wilcox.test with dunnett contrasrs
#' @param y numeric; vector of data values
#' @param g factor; grouping vector
#' @param dunnett logical; if TRUE dunnett contrast, otherwise Tukey-contrasts
#' @param padj character; method for p-adjustment, see ?p.adjust.
#' @param ... other arguments passed to wilcox.exact {exactRankTests}
pairwise_wilcox <- function(y, g, dunnett = TRUE, padj = 'holm', ...){
  if(!require(exactRankTests)){
    stop('Install exactRankTests package')
  }
  tc <- t(combn(nlevels(g), 2))
  # take only dunnett comparisons
  if(dunnett){
    tc <- tc[tc[ , 1] == 1, ]
  }
  pval <- numeric(nrow(tc))
  # use wilcox.exact (for tied data)
  for(i in seq_len(nrow(tc))){
    pval[i] <- wilcox.exact(y[as.numeric(g) == tc[i, 2]],
                            y[as.numeric(g) == tc[i, 1]], exact = TRUE,
                            ...)$p.value
  }

  # adjust p-values
  pval <- p.adjust(pval, padj)
  names(pval) = paste(levels(g)[tc[,1]], levels(g)[tc[,2]], sep = ' vs. ')
  return(pval)
}
```

Here, we use one-sided Dunnett contrasts and adjust p-values using Holm's method:

```
pairwise_wilcox(y = dfm$abu, g = dfm$treatment,
                dunnett = TRUE, p.adj = 'holm', alternative = 'less')

## Control vs. T0.1 Control vs. T0.3   Control vs. T1   Control vs. T3
##        0.2285714        0.2285714        0.2285714        0.1428571
##  Control vs. T10
##        0.2285714
```

This indicates no treatment effect at no level of concentration.

2 Binomial data example

2.1 Introduction

Here we will show how to analyse binomial data (*x out of n*). 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 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)
```

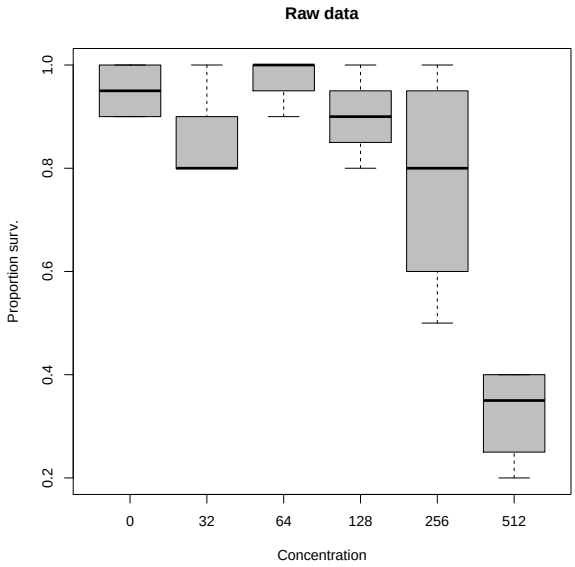
So after data cleaning the data looks like

```
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.',
        main = 'Raw data', col = 'grey75')
```



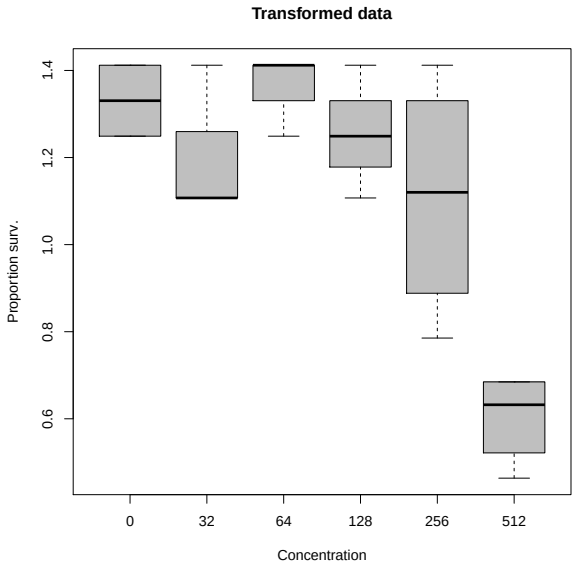
This plot indicates a strong effect at the highest concentration.

2.2 Assuming a normal distribution of transformed proportions

First, we arcsine transform (eqn. 6) 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.',
        main = 'Transformed data', col = 'grey75')
```



Then, like in the count data example we fit the model using `lm()`:

```
modlm <- lm(y_asin ~ conc, data = dfm)
```

The summary gives the estimated parameters:

```
summary(modlm)
```

```
##
## Call:
## lm(formula = y_asin ~ conc, data = dfm)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32401 -0.08149 -0.00527  0.08150  0.30261
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.33053    0.07693  17.295 1.16e-12 ***
## conc32      -0.14717    0.10880  -1.353   0.1929
## conc64       0.04074    0.10880   0.374   0.7124
## conc128     -0.07622    0.10880  -0.701   0.4925
## conc256     -0.22113    0.10880  -2.032   0.0571 .
## conc512     -0.72735    0.10880  -6.685 2.86e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1539 on 18 degrees of freedom
## Multiple R-squared:  0.7871,Adjusted R-squared:  0.7279
## F-statistic: 13.31 on 5 and 18 DF,  p-value: 1.612e-05
```

The F-test suggests a treatment related effect:

```
drop1(modlm, test = 'F')

## Single term deletions
##
## Model:
## y_asin ~ conc
##        Df Sum of Sq     RSS     AIC F value    Pr(>F)
## <none>              0.42613 -84.746
## conc    5    1.5753 2.00142 -57.621  13.308 1.612e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And the LOEC is at the highest concentration:

```
summary(glht(modlm, linfct = mcp(conc = 'Dunnett'), alternative = 'less'),
        test = adjusted('holm'))

##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Dunnett Contrasts
##
##
## Fit: lm(formula = y_asin ~ conc, data = dfm)
##
## Linear Hypotheses:
##               Estimate Std. Error t value   Pr(<t)
## 32 - 0 >= 0  -0.14717    0.10880  -1.353    0.289
## 64 - 0 >= 0   0.04074    0.10880   0.374    0.644
## 128 - 0 >= 0 -0.07622    0.10880  -0.701    0.493
## 256 - 0 >= 0 -0.22113    0.10880  -2.032    0.114
## 512 - 0 >= 0 -0.72735    0.10880  -6.685 7.14e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

2.3 Assuming a binomial distribution

The binomial model with a logit link (eqn. 7) between predictors and response can be fitted using the `glm()` function:

```
modglm <- glm(y ~ conc , data = dfm, family = binomial(link = 'logit'),
              weights = rep(10, nrow(dfm)))
```

Here the weights arguments, indicates how many fish where exposed in each treatment (N=10, eqn .7).

The summary gives the estimated parameters:

```
summary(modglm)

##
## Call:
## glm(formula = y ~ conc, family = binomial(link = "logit"), data = dfm,
##     weights = rep(10, nrow(dfm)))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8980  -0.5723   0.0000   0.7869   2.2578
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.9444     0.7255   4.059 4.94e-05 ***
## conc32       -1.2098     0.8499  -1.423   0.1546
## conc64        0.7191     1.2458   0.577   0.5638
## conc128      -0.7472     0.8967  -0.833   0.4047
## conc256      -1.7077     0.8183  -2.087   0.0369 *
## conc512      -3.6753     0.8002  -4.593 4.37e-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: 72.862
##
## Number of Fisher Scoring iterations: 5
```

To perform a LR-test we can used the `drop1()` function:

```
drop1(modglm, test = 'Chisq')

## Single term deletions
##
## Model:
## y ~ conc
##        Df Deviance     AIC    LRT  Pr(>Chi)
## <none>      23.889  72.862
## conc    5   88.672 127.645 64.783 1.243e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Also with the binomial model the LOEC is at the highest concentration:

```
summary(glht(modglm, linfct = mcp(conc = 'Dunnett'), alternative = 'less'),
        test = adjusted('holm'))

##
##   Simultaneous Tests for General Linear Hypotheses
```

```
##
## Multiple Comparisons of Means: Dunnett Contrasts
##
##
## Fit: glm(formula = y ~ conc, family = binomial(link = "logit"), data = dfm,
##     weights = rep(10, nrow(dfm)))
##
## Linear Hypotheses:
##              Estimate Std. Error z value   Pr(<z)
## 32 - 0 >= 0   -1.2098     0.8499  -1.423   0.2319
## 64 - 0 >= 0    0.7191     1.2458   0.577   0.7181
## 128 - 0 >= 0  -0.7472     0.8967  -0.833   0.4047
## 256 - 0 >= 0  -1.7077     0.8183  -2.087   0.0738 .
## 512 - 0 >= 0  -3.6753     0.8002  -4.593 1.09e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

References

Bretz, F. (1999). *Powerful modifications of Williams' test on trend*. PhD thesis.

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. (2015). The minimum detectable difference (MDD) and the interpretation of treatment-related effects of pesticides in experimental ecosystems. *Environmental Science and Pollution Research*, 22(2):1160–1174.

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

Hothorn, T., Bretz, F., and Westfall, P. (2008). Simultaneous Inference in General Parametric Models. *Biometrical Journal*, 50(3):346–363.

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.

Williams, D. A. (1972). The comparison of several dose levels with a zero dose control. *Biometrics*, pages 519–531.