USING THE CORRBIN PACKAGE FOR NONPARAMETRIC ANALYSIS OF CORRELATED BINARY DATA

ANIKO SZABO

The `CorrBin` package focuses on non-parametric methods for exchangeable correlated binary data with varying cluster sizes. Exchangeability implies that the order of responses within a cluster does not matter, only the total number of responses; the package uses the clustersize/ number of responses combination as input. Currently only one categorical cluster-level predictor, treatment group, is allowed. Many of the functions are geared toward testing trend, so treatment groups are usually treated as ordered categories.

```
> library(CorrBin)
> library(lattice)
```

1. Data input

All the analysis functions in the package work on `CBData` objects, so we start by setting up the data in the format needed for analysis. The Shell toxicology data set is available in the CBData format in the package, however we will load it from a text file using the `read.CBData` function to show more typical usage. The "ShellTox.txt" file contains four space-delimited columns. The first column contains an integer $1 - 4$ giving the treatment group, the second column gives the size of the cluster, the third the number of responses in the cluster, and the last gives the number of times the given combination occured in the data.

```
> sh <- read.CBData("ShellTox.txt", with.freq = TRUE)
> levels(sh$Trt) <- c("Control", "Low", "Medium", "High")
> str(sh)
```

```
Classes 'CBData' and 'data.frame':        67 obs. of  4 variables:
 $ Trt        : Factor w/ 4 levels "Control","Low",..: 2 3 1 2 3 4 2 4 1 2 ...
 $ ClusterSize: num  1 3 4 4 4 4 5 5 6 6 ...
 $ NResp      : num  0 0 0 0 0 0 0 0 0 0 ...
 $ Freq       : int  1 1 1 1 1 1 1 1 2 3 ...
```

Alternatively, if the data is already in a data frame, the `CBData` function can be used to define the roles of the variables. Both `read.CBData` and `CBData` can accomodate cluster-level data that has not been summarized to have frequencies of each clustersize/number of responses combination.

2. Marginal compatibility

A basic assumption of all of the following analyses is that of *marginal compatibility* (MC), which states that the size of the cluster has no effect on either the marginal probability of response, or the correlation (any order) within the cluster. Of course, this is only relevant if the clusters of different sizes are present in the data. We can test for marginal compatibility:

```
> mc.test.chisq(sh)
```

```
$overall.chi
[1] 4.017923

$overall.p
[1] 0.4035857

$individual
$individual$chi.sq
cbdata$Trt
   Control        Low     Medium       High
0.46055742 2.04650267 0.04703645 1.46382641
```

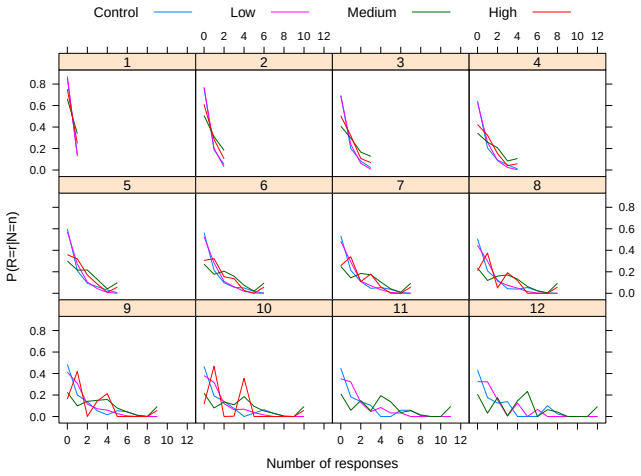

FIGURE 1. Density function of number of responses under MC by cluster-size estimated separately for each treatment group

```
$individual$p
cbdata$Trt
  Control       Low    Medium      High
0.4973636 0.1525563 0.8283027 0.2263223
```

Neither the overall p-value of 0.4, or the individual treatment group p-values show evidence of deviation from marginal compatibility.

Now we can obtain non-parametric estimates of the distribution of the number of responses in the cluster under MC:

```
> sh.mc <- mc.est(sh)
```

Even though the `mc.est` functions gives estimates for all cluster-sizes, due to the marginal compatibility assumption the estimates $\pi_{r,M}$ for the largest cluster-size M determine all the other estimates:

$$P(r \text{ responses} \mid \text{cluster size } n) = \pi_{r,n} = \sum_{t=0}^{M} h(r,t,n)\pi_{t,M}, \tag{1}$$

where $h(r,t,n) = \binom{t}{r}\binom{M-t}{n-r}/\binom{M}{n}$ is the hypergeometric density function. Figure 1 shows the estimates.

```
> print(xyplot(Prob ~ NResp | factor(ClusterSize), groups = Trt, data = sh.mc,
+     subset = ClusterSize > 0 & ClusterSize < 13, type = "l", as.table = TRUE,
+     auto.key = list(columns = 4, lines = TRUE, points = FALSE), xlab = "Number of responses",
+     ylab = "P(R=r|N=n)"))
```

The density functions in Figure 1 are difficult to compare (there is no obvious shift); distribution functions often provide a cleaner comparison, so they are plotted in Figure 2. These plots show that curves for the "Control" and "Low" groups tend to be above the "Medium" and "High" group, suggesting the presence of a dose related trend.

```
> panel.cumsum <- function(x, y, ...) {
+     x.ord <- order(x)
+     panel.xyplot(x[x.ord], cumsum(y[x.ord]), ...)
+ }
> print(xyplot(Prob ~ NResp | factor(ClusterSize), groups = Trt, data = sh.mc,
+     subset = ClusterSize > 0 & ClusterSize < 13, type = "s", panel = panel.superpose,
+     panel.groups = panel.cumsum, as.table = T, auto.key = list(columns = 4,
+         lines = T, points = F), xlab = "Number of responses", ylab = "Cumulative Probability R(R>=r|N=n)
+     ylim = c(0, 1.1)))
```

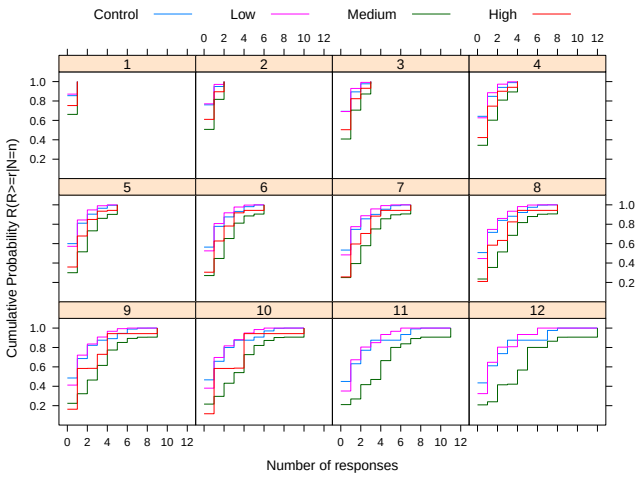

FIGURE 2. Distribution functions of number of responses under MC by cluster-size estimated separately for each treatment group

3. TESTING FOR TREND

Several methods have been proposed for testing for trend with correlated binary data; this package implements 3 types of tests: the Rao-Scott (RS) test, 3 versions of a GEE-based test[1], and a stochastic ordering (SO) based test. RS and GEE test for linear trend in the marginal probability of response, while SO tests for ordering of the distribution functions (as in Figure 2) of the number of responses..

The common interface for accessing the trend tests is the `trend.test` function. It allows to pick the test, whether a permutation-test based exact or an asymptotic (only for RS and GEE) p-value should be used, and set algorithm options for the SO test. In this vignette we use $R = 50$ permutations to limit running time, but in actual work larger values should be used.

```
> set.seed(4461)
> (so.res <- trend.test(sh, test = "SO", R = 50, control = soControl(eps = 0.1,
+     max.directions = 40)))
$statistic
[1] 18.81289
attr(,"ll0")
[1] -133.1832
attr(,"ll1")
[1] -123.7767

$p.val
[1] 0.02

attr(,"boot")

DATA PERMUTATION


Call:
boot(data = dat2, statistic = boot.LRT.fun, R = R, sim = "permutation")


Bootstrap Statistics :
    original   bias    std. error
t1* 18.81289 -9.47545    4.318644
```

[1]The GEE approach is implemented, even though it is not quite a non-parametric test

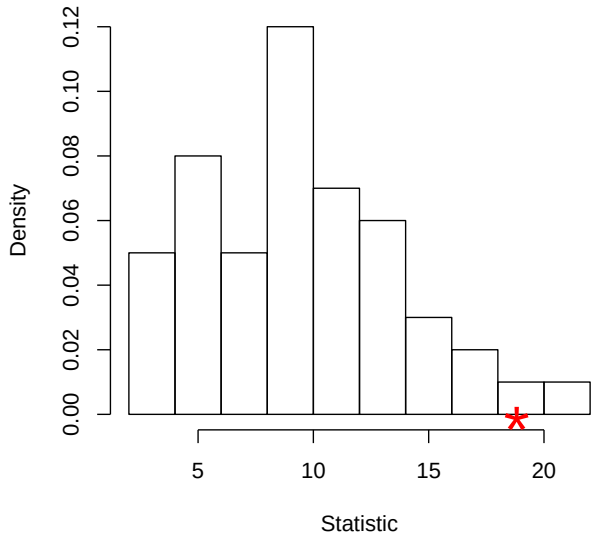

FIGURE 3. Null distribution of the SO test statistic with the observed value marked with a red star.

The `eps` parameter sets the limit on the absolute error of the log-likelihood (and thus, the test statistic), while `max.directions` is a tuning parameter for the default ISDM algorithm: it affects only the running time, and, in general, larger values lead to fewer iterations that however take longer. The details of the permutation test are saved in the output object, so the null distribution of the test statistic can be extracted for, say, a plot as in Figure 3.

```
> hist(attr(so.res, "boot")$t[, 1], freq = FALSE, xlab = "Statistic", ylab = "Density",
+     main = "")
> points(so.res$statistic, 0, pch = "*", col = "red", cex = 3)
```

The other tests also show the presence of a statistically significant trend:

```
> trend.test(sh, test = "RS")
$statistic
[1] 2.364614

$p.val
[1] 0.009024435
> trend.test(sh, test = "GEE")
$statistic
[1] 2.532378

$p.val
[1] 0.00566459
```

The stochastic ordering approach provides not only a test for trend, but also the estimated distribution of the number of responses under the alternative hypothesis of stochastic order. These can be obtained by the `SO.mc.est` function, with the value of the log-likelihood, and convergence information. The estimates are then plotted (see Figure 4).

```
> sh.SO.est <- SO.mc.est(sh, control = soControl(eps = 0.1, max.directions = 40))
> str(sh.SO.est)
'data.frame':        416 obs. of  4 variables:
 $ NResp      : num  0 1 0 1 2 0 1 2 3 0 ...
 $ ClusterSize: num  1 1 2 2 2 3 3 3 3 4 ...
 $ Trt        : Factor w/ 4 levels "Control","Low",..: 1 1 1 1 1 1 1 1 1 1 ...
```

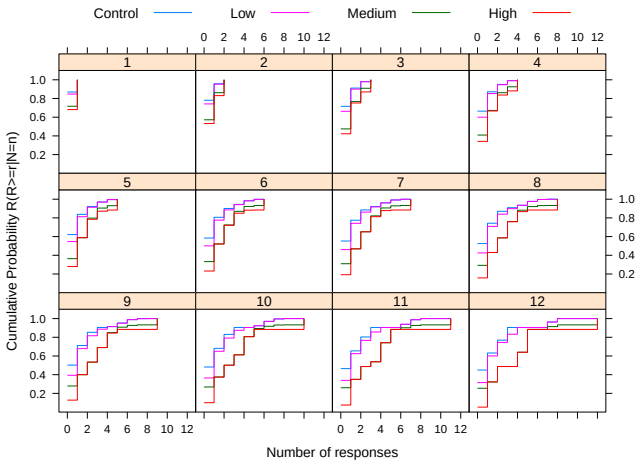

FIGURE 4. Stochastically ordered distribution functions of the number of responses under MC by cluster-size.

```
 $ Prob       : num  0.8695 0.1305 0.7814 0.1763 0.0424 ...
 - attr(*, "loglik")= num -124
 - attr(*, "converge")= Named num  0.0794 18
  ..- attr(*, "names")= chr  "rel.error" "n.iter"
```

```
> print(xyplot(Prob ~ NResp | factor(ClusterSize), groups = Trt, data = sh.SO.est,
+     subset = ClusterSize < 13, type = "s", panel = panel.superpose, panel.groups = panel.cumsum,
+     as.table = T, auto.key = list(columns = 4, lines = T, points = F), xlab = "Number of responses",
+     ylab = "Cumulative Probability R(R>=r|N=n)", ylim = c(0, 1.1), main = ""))
```

4. Risk assessment

The No-Statistical-Significance-Of-Trend dose – the largest dose at which no trend in the rate of response has been observed – is often used to determine a safe dosage level for a potentially toxic compound. The `NOSTASOT` function computes this dose by a step-down approach of testing all doses, all but the last, etc. All three tests (stochastic order, Rao-Scott, and GEE) are available.

```
> NOSTASOT(sh, test = "RS")
```

```
$NOSTASOT
[1] "Low"

$p
        Low      Medium        High
0.496667059 0.001410127 0.009024435
```

For the Shell toxicology data, the "Low" dose of the drug shows no trend, but all higher doses do, so that's the NOSTASOT dose.