TEST FOR TREND WITH A MULTINOMIAL OUTCOME

ANIKO SZABO

1. Introduction

- Consider a study in which a multinomial outcome with K possible unordered values is measured in subjects belonging to one of G ordered groups. The size of each group, $n_{i\cdot}$, is defined by the study design, and will be treated as fixed. Let $\mathbf{p}_i = (p_{i1}, \ldots, p_{iK})^{\mathrm{T}}$ denote the probabilities of the multinomial outcomes in the ith group. The hypothesis of interest is to evaluate the homogeneity of these probabilities across the groups with a targeted alternative of a trend in at least one of the categories. Formally, we consider testing $H_0 = \bigcap_{j=1}^{K} H_{0j}$ versus $H_1 = \bigcup_{j=1}^{K} H_{1j}$, where

$$\begin{aligned} &H_{0j} : p_{1j} = \cdots = p_{Gj} \\ &H_{1j} : p_{1j} \leq \cdots \leq p_{Gj} \text{ or } p_{1j} \geq \cdots \geq p_{Gj} \text{ with at least one inequality} \end{aligned} \tag{1}$$

The test is based on the following result:

Theorem 1. *Let $\mathcal{J} \subset \{1, \ldots, K\}$, then under $H_{0\mathcal{J}} = \bigcap_{j\in\mathcal{J}} H_{0j}$ as $N \to \infty$*

$$W_{\mathcal{J}} = \sum_{j\in\mathcal{J}} (1 - p_{\cdot j}) T_j^2 + \big(\sum_{j\in\mathcal{J}} p_{\cdot j}\big) T_{\mathcal{J}}^2 \xrightarrow{d} \chi^2_d, \tag{2}$$

where $d = \min(|\mathcal{J}|, K-1)$, $T_{\mathcal{J}} = \big[\sum_{i=1}^{G} \sum_{j\in\mathcal{J}} n_{ij}(c_i - \bar{c})\big] \big/ \sqrt{p_{\cdot\mathcal{J}}(1-p_{\cdot\mathcal{J}})s^2}$ denotes the Cochran-Armitage trend test statistic for testing for marginal trend in $p_{i\mathcal{J}} = \sum_{j\in\mathcal{J}} p_{ij}$, $i = 1, \ldots, G$.

2. Implementing the overall test

The main `multiCA.test` function is a generic, with methods for a matrix and formula input.

Date: May 2, 2016.

"../R/aaa-generics.R" 2a≡

```
#'Multinomial Cochran-Armitage trend test
#'
#'The \code{multiCA.test} performs a multinomial generalization of the
#' Cochran-Armitage trend test.
#'
#'
#'@export
#'@param x a two-dimensional matrix or a formula
#'@param \dots other arguments
#'@return a list with two components
#' \item{overall}{an object of class "htest" with the results of the overall test}
#' \item{individual}{a vector with adjusted p-values for individual outcomes}
#'@author Aniko Szabo
#'@references Szabo, A. (2016) Test for trend with a multinomial outcome.
#'@keywords nonparametric
#'@examples
#'
#'data(stroke)
#'## using formula interface
#'multiCA.test(Type ~ Year, weights=Freq, data=stroke)
#'
#'## using matrix interface and testing only the first 3 outcomes
#'strk.mat <- xtabs(Freq ~ Type + Year, data=stroke)
#'multiCA.test(strk.mat, outcomes=1:3)
#'
#'@name multiCA.test

multiCA.test <- function(x,...) UseMethod("multiCA.test")

◇
```

The actual calculation of the test statitistic, overall and unadjusted individual p-values is encapsulated in an internal function that operates on a matrix. No error control is provided here.

"../R/multiCA.R" 2b≡

```
#'@keywords internal

.multiCA.test <- function(x, scores, outcomes){
  K <- nrow(x)
  full <- length(outcomes) == K  #full test

  nidot <- apply(x, 2, sum)
  n <- sum(nidot)

  cbar <- sum(nidot * scores)/n

  s2 <- sum(nidot * (scores - cbar)^2)
  pdot <- prop.table(rowSums(x))[outcomes]
  nonz <- (pdot > 0)

  if (!any(nonz)) return(1)

  X <- x[outcomes, ,drop=FALSE] %*% (scores - cbar)

  #individual tests
  CAT <- X[nonz]^2 / (pdot[nonz] * (1-pdot[nonz])) / s2
  CAT.p.value <- pchisq(CAT, df=1, lower.tail=FALSE)

  #overall test
  if (full || sum(pdot) >= 1){
    Tt <- ( sum(X[nonz]^2 / pdot[nonz])) / s2
  } else {
    Tt <- (sum(X)^2 / (1-sum(pdot)) + sum(X[nonz]^2 / pdot[nonz])) / s2
  }

  df <- length(outcomes) - full
  p.value <- pchisq(Tt, df=df, lower.tail=FALSE)

  res <- list(statistic = Tt, parameter = df, p.value = p.value,
              indiv.statistics = CAT, indiv.p.value = CAT.p.value)
  return(res)
}
◇
```

File defined by 2b, 3, 4a, ?.
Defines: `.multiCA.test` 3, 5b.

The default method uses a two-dimensional contingency matrix with the outcomes as rows and ordered groups as columns.

"../R/multiCA.R" 3≡

```
#'@rdname multiCA.test
#'@method multiCA.test default
#'@param scores numeric vector of the same length as the number of ordered groups. Defaults to linearly i
#'@param outcomes integer or character vector defining the set of outcomes (by row index or row name) ove
#'@param p.adjust.method character string defining the correction method for individual outcome p-values.
#'@export

multiCA.test.default <- function(x, scores=1:ncol(x), outcomes=1:nrow(x),
  p.adjust.method=c("none","closed.set","holm-schaffer"),...){
  if (!is.matrix(x)) {
    cat(str(x))
    stop("x should be a two-dimensional matrix")
}
  if (length(scores) != ncol(x)) stop("The length of the score vector should equal the number of columns

  testres <- .multiCA.test(x=x, scores=scores, outcomes=outcomes)

  Tt <- c(W = testres$statistic)
  df <- c(df = testres$parameter)

  p.value <- testres$p.value
  null.value <- 0
  names(null.value) <- sprintf("slope for outcomes %s", deparse(substitute(outcomes)))

  res <- list(statistic = Tt, parameter = df, p.value = p.value,
              method="Multinomial Cochran-Armitage trend test",
              alternative="two.sided",
              null.value=null.value,
              data.name = deparse(substitute(x)))
  class(res) <- "htest"

  ⟨ Calculate adjusted p-values 4b ⟩
```

```
    return(list(overall = res, individual = indiv.res))
  }
  ◇
```

File defined by 2b, 3, 4a, ?.
Defines: `multiCA.test.default` Never used.
Uses: `.multiCA.test` 2b.

The formula interface converts data into the appropriate contingency matrix for use with the default method. The code is based on `t.test.formula`.

`"../R/multiCA.R"` 4a≡

```
#'@rdname multiCA.test
#'@method multiCA.test formula
#'@param formula a formula of the form \code{outcome ~ group} where \code{outcome} is a factor representi
#'@param data  an optional matrix or data frame containing the variables in the formula \code{formula}. B
#'@param subset  an optional vector specifying a subset of observations to be used.
#'@param na.action       a function which indicates what should happen when the data contain NAs. Default
#'@param weights an integer-valued variable representing the number of times each \code{outcome} - \code{
#'@export

multiCA.test.formula <- function(formula, data, subset, na.action,  weights, ...){
    if (missing(formula) || (length(formula) != 3L) || (length(attr(terms(formula[-2L]),
        "term.labels")) != 1L))
        stop("'formula' missing or incorrect")
    m <- match.call(expand.dots = FALSE)
    if (is.matrix(eval(m$data, parent.frame())))
        m$data <- as.data.frame(data)
    m[[1L]] <- quote(stats::model.frame)
    m$... <- NULL
    mf <- eval(m, parent.frame())
    responsevar <- attr(attr(mf, "terms"), "response")
    response <- mf[[responsevar]]
    weightvar <- which(names(mf)=="(weights)")
    w <- if(length(weightvar) > 0)  mf[[weightvar]] else rep(1L, nrow(mf))
    g <- factor(mf[,-c(responsevar, weightvar)])

    tab <- xtabs(w ~ response + g)
    multiCA.test(tab, ...)
}
◇
```

File defined by 2b, 3, 4a, ?.
Defines: `multiCA.test.formula` Never used.

3. Multiple testing adjusted inference for individual outcomes

⟨ *Calculate adjusted p-values* 4b ⟩ ≡

```
if (missing(p.adjust.method)){
  if (length(outcomes)<=3) p.adjust.method <- "closed.set"
  else p.adjust.method <- "holm-schaffer"
} else {
  p.adjust.method <- match.arg(p.adjust.method)
}

full.set <- (length(outcomes) == nrow(x))
if (p.adjust.method=="none") {
  indiv.res <- testres$indiv.p.value
} else if (p.adjust.method=="closed.set") {
  ⟨ Closed set adjustment 5b ⟩
} else if (p.adjust.method=="holm-schaffer") {
  ⟨ Holm-Schaffer adjustment 5a ⟩
}
◇
```

Fragment referenced in 3.

3.1. **Holm-Schaffer approach.**

Schaffer's modification of Holm's adjustment involves multiplying the ordered p-values by t_s, the maximum number of possibly true hypotheses, given that at least $s-1$ hypotheses are false. In our case the logical restriction means that if there is at least one false null hypothesis, then no more than $K-2$ null hypotheses could be true. So

$$p^{HS}_{(j)} = \max_{s\leq j}(\min(t_s p_{(s)}, 1))$$

$$\text{where}t_s = \begin{cases} K-s+1, & s \neq 2 \\ K-2, & s = 2 \end{cases}$$

⟨ *Holm-Schaffer adjustment* 5a ⟩ ≡

```
    s <- seq_along(testres$indiv.p.value)
    if (full.set) s[2] <- 3
    o <- order(testres$indiv.p.value)
    ro <- order(o)
    indiv.res <- pmin(1, cummax((length(outcomes) - s + 1L) * testres$indiv.p.value[o]))[ro]
◇
```

Fragment referenced in 4b.

3.2. **Closed set adjustment.**

In a closed testing procedure an elementary hypothesis H_{0j} is rejected if and only if all composite hypotheses $H_{0\mathcal{J}}$, where $j \in \mathcal{J}$ are rejected. The process can be rewritten using adjusted p-values for $H_{0j}, j = 1, \dots K$:

$$p^*_j = \max_{\mathcal{J}:j\in\mathcal{J}} p(\mathcal{J}), \tag{3}$$

where $p(\mathcal{J}) = P(W_j \geq \chi^2_{|\mathcal{J}|})$ is the unadjusted p-value for testing $H_{0\mathcal{J}}$. From the logical constraints sets $\mathcal{J}$ of cardinality $K-1$ do not need to be considered.

⟨ *Closed set adjustment* 5b ⟩ ≡

```
  mytest <- function(hypotheses){
    .multiCA.test(x, scores, hypotheses)$indiv.p.values
  }
  indiv.res <- .p.adjust.closed(mytest, outcomes, remove=full.set)
◇
```

Fragment referenced in 4b.
Uses: `.multiCA.test` 2b.

The actual adjustment calculation is based on code from `cherry::closed`, removing the $K-1$ element sets if the full set of hypotheses is being tested.

`"multiCA.R"` 5c≡

```
#' @importFom bitops bitAnd
.bit2boolean <- function (x, N)
{
  base <- 2^(1:N - 1)
  bitAnd(x, base) != 0
}

#’ @param test function that performs the local test. The function should accept a subvector of the hypot
#’ @param  hypotheses identifiers of the collection of elementary hypotheses.
#’ @param remove logical indicator of whether hypotheses of length N-1 should be removed
#’ @param  ...  additional parameters to the ’test’ function
#’ @return  numeric vector of adjusted p-values for each hypothesis
.p.adjust.closed <- function (test, hypotheses, remove=FALSE, ...)
{
  N <- length(hypotheses)
  Nmax <- log2(.Machine$integer.max + 1)
  if (N > Nmax)
    stop("no more than ", Nmax, " hypotheses supported in full closed testing.\n Use a shortcut-based tes
  closure <- 1:(2^N - 1)
  base <- 2^(1:N - 1)
  offspring <- function(x) {
    res <- bitAnd(x, closure)
    res[res != 0]
  }
  lengths <- rowSums(sapply(base, function(bs) bitAnd(closure, bs) != 0))

  idx <- sort.list(lengths, decreasing = TRUE)
  closure <- closure[idx]
  lengths <- lengths[idx]
  if (remove)  closure <- closure[lengths != (N-1)]

  adjusted <- numeric(2^N - 1)
  for (i in closure) {
    if (adjusted[i] < 1) {
      localtest <- test(hypotheses[.bit2boolean(i,N)], ...)
      if (localtest > adjusted[i]) {
        offs <- offspring(i)
        adjusted[offs] <- pmax(adjusted[offs], localtest)
      }
    }
  }

  out <- adjusted[base]
  names(out) <- hypotheses
  return(out)
}
```
◇

4. Power and sample size calculation

The calculation is based on the following result: Let $\nu_i = n_{i\cdot}/N$ denote the proportion of subjects in group i.

Theorem 2. *Under H_a, the asymptotic distribution of W is approximately $\chi^2_{K-1}(\lambda)$ with non-centrality parameter*

$$\lambda = N s_\nu^2 \sum_{j=1}^{K} \frac{\beta_j^2}{p_{\cdot j}}, \tag{4}$$

where $s_\nu^2 = \sum_{i=1}^G \nu_i (c_i - \bar{c})^2 = s^2/N$ and $\beta_j = \left[\sum_{i=1}^G \nu_i (p_{ij} - p_{\cdot j})(c_i - \bar{c})\right]/s_\nu^2$ is the slope of p_{ij}, $i = 1, \ldots, G$ regressed on c_i with weights ν_i.

A non-centrality parameter calculation function can be useful by itself. It calculates the non-centrality parameter for a chi-square distribution that achieves the target power at a given significance level.

"../R/multiCA.R" ?≡

```
#' Non-centrality parameter for chi-square distribution
#'
#' Calculates the non-centrality parameter for a chi-square distribution that achieves the target power a
#'@param df an integer giving the degrees of freedom of the chi-square variable
#'@param alpha a numeric value giving the significance level of the test
#'@param beta a numeric value giving the desired type II error (1-\code{beta} is the power)
#'@examples
#' cnonct(6, 0.05, 0.2)
#'@export

cnonct <- function(df, alpha, beta){
  crit.value <- qchisq(alpha, df=df, lower.tail=FALSE)

  f <- function(ncp){pchisq(crit.value, df=df, ncp=pmax(0,ncp)) - beta}

  res <- uniroot(f, interval=c(0, 100), extendInt="downX")
  res$root
}
◇
```

File defined by 2b, 3, 4a, ?.
Defines: `cnonct` Never used.

5. Files

"../R/aaa-generics.R" Defined by 2a.
"../R/multiCA.R" Defined by 2b, 3, 4a, ?.
"multiCA.R" Defined by 5c.

6. Macros

⟨Calculate adjusted p-values 4b⟩ Referenced in 3.
⟨Closed set adjustment 5b⟩ Referenced in 4b.
⟨Holm-Schaffer adjustment 5a⟩ Referenced in 4b.

7. Identifiers

`.multiCA.test`: 2b, 3, 5b.
`cnonct`: ?.
`multiCA.test.default`: 3.
`multiCA.test.formula`: 4a.