CORRELATED BINARY DATA

ANIKO SZABO

Abstract. We define a class for describing data from toxicology experiments and implement fitting of a variety of existing models and trend tests.

"../R/CorrBin-package.R" 1≡

```
#'Nonparametrics for Correlated Binary and Multinomial Data
#'
#'This package implements nonparametric methods for analyzing exchangeable
#'binary and multinomial data with variable cluster sizes with emphasis on trend testing. The
#'input should specify the treatment group, cluster-size, and the number of
#'responses (i.e. the number of cluster elements with the outcome of interest)
#'for each cluster.
#'
#'\itemize{ \item The \code{\link{CBData}/\link{CMData}} and \code{\link{read.CBData}/\link{read.CMData}}
#'functions create a 'CBData' or 'CMData' object used by the analysis functions.
#'\item \code{\link{ran.CBData}} and \code{\link{ran.CMData}} can be used to generate random
#' binary or multinomial data using a variety of distributions.
#'\item \code{\link{mc.test.chisq}} tests the assumption of marginal compatibility
#'underlying all the methods, while \code{\link{mc.est}} estimates the
#'distribution of the number of responses under marginal compatibility.
#'\item Finally, \code{\link{trend.test}} performs three different tests for trend
#'along the treatment groups for binomial data. }
#'
#'@name CorrBin-package
#'@aliases CorrBin-package CorrBin
#'@docType package
#'@author Aniko Szabo
#'
#'Maintainer: Aniko Szabo <aszabo@@mcw.edu>
#'@references Szabo A, George EO. (2009) On the Use of Stochastic Ordering to
#'Test for Trend with Clustered Binary Data. \emph{Biometrika}
#'
#'Stefanescu, C. & Turnbull, B. W. (2003) Likelihood inference for exchangeable
#'binary data with varying cluster sizes. \emph{Biometrics}, 59, 18-24
#'
#'Pang, Z. & Kuk, A. (2007) Test of marginal compatibility and smoothing
#'methods for exchangeable binary data with unequal cluster sizes.
#'\emph{Biometrics}, 63, 218-227
#'@keywords package nonparametric
NULL
```

◇

Uses: ran.CBData 9b.

Date: August 28, 2024.

1. Defining CBData – a class for Clustered Binary Data

We start with defining an S3 class describing data from toxicology experiments. The class is a data frame with the following columns:

Trt: a factor defining (treatment) groups
ClusterSize: an integer-valued variable defining the cluster size
NResp: an integer-valued variable defining the number of responses (1s)
Freq: an integer-valued variable defining frequency for each Trt/ClusterSize/NResp combination

`CBData` converts a data frame to a CBData object. `x` is the input data frame; `trt`, `clustersize`, `nresp` and `freq` could be strings or column indices defining the appropriate variable in `x` (`freq` can also be NULL, in which case it is assumed that each combination has frequency 1).

"../R/CBData.R" 2a≡

```
#'Create a 'CBdata' object from a data frame.
#'
#'The \code{CBData} function creates an object of class \dfn{CBData} that is
#'used in further analyses. It identifies the variables that define treatment
#'group, clustersize and the number of responses.
#'
#'@export
#'@importFrom stats aggregate
#'@param x a data frame with one row representing a cluster or potentially a
#'set of clusters of the same size and number of responses
#'@param trt the name of the variable that defines treatment group
#'@param clustersize the name of the variable that defines cluster size
#'@param nresp the name of the variable that defines the number of responses in
#'the cluster
#'@param freq the name of the variable that defines the number of clusters
#'represented by the data row. If \code{NULL}, then each row is assumed to
#'correspond to one cluster.
#'@return A data frame with each row representing all the clusters with the
#'same trt/size/number of responses, and standardized variable names:
#'@return \item{Trt}{factor, the treatment group}
#'@return \item{ClusterSize}{numeric, the cluster size}
#'@return \item{NResp}{numeric, the number of responses}
#'@return \item{Freq}{numeric, number of clusters with the same values}
#'@author Aniko Szabo
#'@seealso \code{\link{read.CBData}} for creating a \code{CBData} object
#'directly from a file.
#'@keywords classes manip
#'@examples
#'
#'data(shelltox)
#'sh <- CBData(shelltox, trt="Trt", clustersize="ClusterSize", nresp="NResp")
#'str(sh)
#'
◇
```

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.

"../R/CBData.R" 2b≡

```
CBData <- function(x, trt, clustersize, nresp, freq=NULL){
  if (!is.data.frame(x)) stop("x has to be a data frame")
  nms <- names(x)
  process.var <- function(var){
    if (is.character(var)){
       if (var %in% nms) res <- x[[var]]
```

```
      else stop(paste("Variable '", var, "' not found"))
    }
    else {
      if (is.numeric(var)){
        if (var %in% seq(along=nms)) res <- x[[var]]
        else stop(paste("Column", var, " not found"))
      }
      else stop(paste("Invalid variable specification:",var))
    }
  }
  trtvar <- factor(process.var(trt))
  csvar <- process.var(clustersize)
  nrespvar <- process.var(nresp)
  if (is.null(freq)) freqvar <- rep(1, nrow(x))
  else freqvar <- process.var(freq)

  d <- data.frame(Trt=trtvar, ClusterSize=csvar, NResp=nrespvar, Freq=freqvar)
  d <- aggregate(d$Freq, list(Trt=d$Trt, ClusterSize=d$ClusterSize, NResp=d$NResp),sum)
  names(d)[4] <- "Freq"
  d$ClusterSize <- as.numeric(as.character(d$ClusterSize))
  d$NResp <- as.numeric(as.character(d$NResp))
  class(d) <- c("CBData", "data.frame")
  d}
```

◇

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Defines: `CBdata.data.frame` Never used.

The `read.CBData` function reads in clustered binary data from a tab-delimited text file. The first column should give the treatment group, the second the size of the cluster, the third the number of responses in the cluster. Optionally, a fourth column could give the number of times the given combination occurs in the data.

"../R/CBData.R" 3a≡

```
#'Read data from external file into a CBData object
#'
#'A convenience function to read data from specially structured file directly
#'into a \code{CBData} object.
#'
#'@export
#'@importFrom utils read.table
#'@param file name of file with data. The first column should contain the
#'treatment group, the second the size of the cluster, the third the number of
#'responses in the cluster. Optionally, a fourth column could give the number
#'of times the given combination occurs in the data.
#'@param with.freq logical indicator of whether a frequency variable is present
#'in the file
#'@param ... additional arguments passed to \code{\link[utils]{read.table}}
#'@return a \code{CBData} object
#'@author Aniko Szabo
#'@seealso \code{\link{CBData}}
#'@keywords IO file
#'
```

◇

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.

"../R/CBData.R" 3b≡

```
read.CBData <- function(file, with.freq=TRUE, ...){
  d <- read.table(file, col.names=c("Trt","ClusterSize","NResp", if (with.freq) "Freq"), ...)
  if (!with.freq) d$Freq <- 1
  d <- aggregate(d$Freq, list(Trt=d$Trt, ClusterSize=d$ClusterSize, NResp=d$NResp),sum)
  names(d)[4] <- "Freq"
  d$ClusterSize <- as.numeric(as.character(d$ClusterSize))
  d$NResp <- as.numeric(as.character(d$NResp))
  d <- CBData(d, "Trt", "ClusterSize", "NResp", "Freq")
  d}
```

◇

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Defines: `read.CBdata` Never used.

The `[.CMData` function defines subsetting of `CMData` objects. If the subsetting is only affecting the rows, then the appropriate attributes are preserved, and the unused levels of `Trt` are dropped. Otherwise the returned object does not have a `CMData` class anymore.

"../R/CBData.R" 4≡

```
#'Extract from a CBData or CMData object
#'
#'The extracting syntax works as for \code{\link{[.data.frame}}, and in general the returned object is no
#'However if the columns are not modified, then the result is still a \code{CBData} or \code{CMData} obje
#' and the unused levels of treatment groups dropped.
#'
#'@param x \code{CMData} object.
#'@param i numeric, row index of extracted values
#'@param j numeric, column index of extracted values
#'@param drop logical. If TRUE the result is coerced to the lowest possible dimension.
#'The default is the same as for \code{\link{[.data.frame}}: to drop if only one column is left, but not
#'@return a \code{CBData} or \code{CMData} object
#'@author Aniko Szabo
#'@seealso \code{CBData}, \code{\link{CMData}}
#'@keywords manip
#'@name Extract
#'
NULL
```

◇

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.

"../R/CBData.R" 5a≡

```
#'@rdname Extract
#'@export
#'@examples
#'
#'data(shelltox)
#'str(shelltox[1:5,])
#'str(shelltox[1:5, 2:4])

"[.CBData" <- function(x, i, j, drop){
  res <- NextMethod("[")
  if (NCOL(res) == ncol(x)){
    res <- "[.data.frame"(x, i, )
    if (is.factor(res$Trt)) res$Trt <- droplevels(res$Trt)
    res
  }
  else {
    class(res) <- setdiff(class(res), "CBData")
  }
  res
}
```

◇

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Defines: `[.CBData` Never used.

`unwrap.CBData` is a utility function that reformats a CBData object so that each row is one observation (instead of one cluster). A new 'ID' variable is added to indicate clusters. It is first defined as a generic function to allow generalization.

"../R/aaa-generics.R" 5b≡

```
#'Unwrap a clustered object
#'
#'\code{unwrap} is a utility function that reformats a CBData or CMData object so
#'that each row is one observation (instead of one or more clusters). A new
#'‘ID’ variable is added to indicate clusters. This form can be useful for
#'setting up the data for a different package.
#'
#'@aliases unwrap unwrap.CBData
#'@export
#'@param object a \code{\link{CBData}} object
#'@param \dots other potential arguments; not currently used
#'@return For \code{unwrap.CBData}: a data frame with one row for each cluster element (having a binary
#'outcome) with the following standardized column names
#'@return \item{Trt}{factor, the treatment group}
#'@return \item{ClusterSize}{numeric, the cluster size}
#'@return \item{ID}{factor, each level representing a different cluster}
#'@return \item{Resp}{numeric with 0/1 values, giving the response of the cluster
#'element}
#'@author Aniko Szabo
#'@keywords manip
#'@examples
#'
#'data(shelltox)
#'ush <- unwrap(shelltox)
#'head(ush)
#'
```

```
unwrap <- function(object,...) UseMethod("unwrap")
```
◇
Uses: unwrap.CBData 6a.

"../R/CBData.R" 6a≡

```
#'@rdname unwrap
#'@method unwrap CBData
#'@export

unwrap.CBData <- function(object,...){
  freqs <- rep(1:nrow(object), object$Freq)
  cb1 <- object[freqs,]
  cb1$Freq <- NULL
  cb1$ID <- factor(1:nrow(cb1))
  pos.idx <- rep(1:nrow(cb1), cb1$NResp)
  cb.pos <- cb1[pos.idx,]
  cb.pos$Resp <- 1
  cb.pos$NResp <- NULL
  neg.idx <- rep(1:nrow(cb1), cb1$ClusterSize-cb1$NResp)
  cb.neg <- cb1[neg.idx,]
  cb.neg$Resp <- 0
  cb.neg$NResp <- NULL
  res <- rbind(cb.pos, cb.neg)
  res[order(res$ID),]
  }
```
◇
File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Defines: unwrap.CBData 5b, 8.

2. Rao-Scott adjusted Cochran-Armitage test

The RS-adjusted CA test for trend is based on design-effect adjustment.

"../R/CBData.R" 6b≡

```
#'Rao-Scott trend test
#'
#'\code{RS.trend.test} implements the Rao-Scott adjusted Cochran-Armitage test
#'for linear increasing trend with correlated data.
#'
#'The test is based on calculating a \dfn{design effect} for each cluster by
#'dividing the observed variability by the one expected under independence. The
#'number of responses and the cluster size are then divided by the design
#'effect, and a Cochran-Armitage type test statistic is computed based on these
#'adjusted values.
#'
#'The implementation aims for testing for \emph{increasing} trend, and a
#'one-sided p-value is reported. The test statistic is asymptotically normally
#'distributed, and a two-sided p-value can be easily computed if needed.
#'
#'@export
#'@param cbdata a \code{\link{CBData}} object
#'@return A list with components
#'@return \item{statistic}{numeric, the value of the test statistic}
#'@return \item{p.val}{numeric, asymptotic one-sided p-value of the test}
```

```
#'@author Aniko Szabo, aszabo@@mcw.edu
#'@seealso \code{\link{SO.trend.test}}, \code{\link{GEE.trend.test}} for
#'alternative tests; \code{\link{CBData}} for constructing a CBData object.
#'@references Rao, J. N. K. & Scott, A. J. A (1992) Simple Method for the
#'Analysis of Clustered Data \emph{Biometrics}, 48, 577-586.
#'@keywords htest nonparametric
#'@examples
#'
#'data(shelltox)
#'RS.trend.test(shelltox)
#'
```

◇

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Uses: `GEE.trend.test` 8, `RS.trend.test` 7a.

"../R/CBData.R" 7a≡

```
RS.trend.test <- function(cbdata){
        dat2 <- cbdata[rep(1:nrow(cbdata), cbdata$Freq),]  #each row is one sample
        dat2$Trt <- factor(dat2$Trt)  #remove unused levels
  x.i <- pmax(tapply(dat2$NResp, dat2$Trt, sum), 0.5)  #"continuity" adjustment to avoid RS=NaN
  n.i <- tapply(dat2$ClusterSize, dat2$Trt, sum)
  m.i <- table(dat2$Trt)
  p.i.hat <- x.i/n.i
  r.ij <- dat2$NResp - dat2$ClusterSize*p.i.hat[dat2$Trt]
  v.i <- m.i/(m.i-1)/n.i^2*tapply(r.ij^2, dat2$Trt, sum)
  d.i <- n.i * v.i / (p.i.hat*(1-p.i.hat))   #design effect
  x.i.new <- x.i/d.i
  n.i.new <- n.i/d.i
  p.hat <- sum(x.i.new)/sum(n.i.new)

  scores <- (1:nlevels(dat2$Trt))-1
  mean.score <- sum(scores*n.i.new)/sum(n.i.new)
  var.scores <- sum(n.i.new*(scores-mean.score)^2)
  RS <- (sum(x.i.new*scores) - p.hat*sum(n.i.new*scores)) /
        sqrt(p.hat*(1-p.hat)*var.scores)
  p.val <- pnorm(RS, lower.tail=FALSE)
  list(statistic=RS, p.val=p.val)
  }
```

◇

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Defines: `RS.trend.test` 6b, 7b.

3. GEE based test

"../R/CBData.R" 7b≡

```
#'GEE-based trend test
#'
#'\code{GEE.trend.test} implements a GEE based test for linear increasing trend
#'for correlated binary data.
#'
#'The actual work is performed by the \code{\link[geepack]{geese}} function of
#'the \code{geepack} library, which is required for this feature to work.
#'This function only provides a convenient wrapper
#'to obtain the results in the same format as \code{\link{RS.trend.test}} and
#'\code{\link{SO.trend.test}}.
```

```
#'
#'The implementation aims for testing for \emph{increasing} trend, and a
#'one-sided p-value is reported. The test statistic is asymptotically normally
#'distributed, and a two-sided p-value can be easily computed if needed.
#'
#'@export
#'@importFrom stats binomial pnorm
#'@param cbdata a \code{\link{CBData}} object
#'@param scale.method character string specifying the assumption about the
#'change in the overdispersion (scale) parameter across the treatment groups:
#'"fixed" - constant scale parameter (default); "trend" - linear trend for the
#'log of the scale parameter; "all" - separate scale parameter for each group.
#'@return A list with components
#'@return \item{statistic}{numeric, the value of the test statistic}
#'@return \item{p.val}{numeric, asymptotic one-sided p-value of the test}
#'@author Aniko Szabo, aszabo@@mcw.edu
#'@seealso \code{\link{RS.trend.test}}, \code{\link{SO.trend.test}} for
#'alternative tests; \code{\link{CBData}} for constructing a CBData object.
#'@keywords htest models
#'@examples
#'
#'data(shelltox)
#'if (require(geepack)){
#'  GEE.trend.test(shelltox, "trend")
#'}
◇
```

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Uses: `GEE.trend.test` 8, `RS.trend.test` 7a.

"../R/CBData.R" 8≡

```
GEE.trend.test <- function(cbdata, scale.method=c("fixed", "trend", "all")){
  if (!requireNamespace("geepack", quietly = TRUE)) {
        stop("Please install geepack: install.packages('geepack')")
        }
  ucb <- unwrap.CBData(cbdata)
  scale.method <- match.arg(scale.method)
  if (scale.method=="fixed") {
    geemod <- geepack::geese(Resp~unclass(Trt), id=ucb$ID, scale.fix=FALSE, data=ucb,
                    family=binomial, corstr="exch") }
  else if (scale.method=="trend"){
    geemod <- geepack::geese(Resp~unclass(Trt), sformula=~unclass(Trt), id=ucb$ID,  data=ucb,
                   family=binomial, sca.link="log", corstr="exch")}
  else if (scale.method=="all"){
    geemod <- geepack::geese(Resp~unclass(Trt), id=ucb$ID,  sformula=~Trt, data=ucb,
                    family=binomial, sca.link="log", corstr="exch") }
  geesum <- summary(geemod)
  testres <- geesum$mean[2,"estimate"]/geesum$mean[2,"san.se"]
  p <- pnorm(testres, lower.tail=FALSE)
  list(statistic=testres, p.val=p)
 }
◇
```

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Defines: `GEE.trend.test` 6b, 7b.
Uses: `unwrap.CBData` 6a.

4. Generating random data

`ran.CBData` generates a random CBData object from a given two-parameter distribution. `sample.sizes` is a dataset with variables Trt, ClusterSize and Freq giving the number of clusters to be generated for each Trt/ClusterSize combination. `p.gen.fun` and `rho.gen.fun` are functions that generate the parameter values for each treatment group ($g = 1$ corresponds to the lowest group, $g = 2$ to the second, etc). `pdf.fun` is a function(p, rho, n) generating the pdf of the number of responses given the two parameters `p` and `rho`, and the cluster size `n`.

"../R/CBData.R" 9a≡

```
#'Generate random correlated binary data
#'
#'\code{ran.mc.CBData} generates a random \code{\link{CBData}} object from a
#'given two-parameter distribution.
#'
#'\dfn{p.gen.fun} and \dfn{rho.gen.fun} are functions that generate the
#'parameter values for each treatment group; \dfn{pdf.fun} is a function
#'generating the pdf of the number of responses given the two parameters
#'\dfn{p} and \dfn{rho}, and the cluster size \dfn{n}.
#'
#'\code{p.gen.fun} and \code{rho.gen.fun} expect the parameter value of 1 to
#'represent the first group, 2 - the second group, etc.
#'
#'@export
#'@importFrom stats rmultinom
#'@param sample.sizes a dataset with variables Trt, ClusterSize and Freq giving
#'the number of clusters to be generated for each Trt/ClusterSize combination.
#'@param p.gen.fun a function of one parameter that generates the value of the
#'first parameter of \code{pdf.fun} (\emph{p}) given the group number.
#'@param rho.gen.fun a function of one parameter that generates the value of
#'the second parameter of \code{pdf.fun} (\emph{rho}) given the group number.
#'@param pdf.fun a function of three parameters (\emph{p, rho, n}) giving the
#'PDF of the number of responses in a cluster given the two parameters
#'(\emph{p, rho}), and the cluster size (\emph{n}). Functions implementing two
#'common distributions: the beta-binomial (\code{\link{betabin.pdf}}) and
#'q-power (\code{\link{qpower.pdf}}) are provided in the package.
#'@return a CBData object with randomly generated number of responses with
#'sample sizes specified in the call.
#'@author Aniko Szabo, aszabo@@mcw.edu
#'@seealso \code{\link{betabin.pdf}} and \code{\link{qpower.pdf}}
#'@keywords distribution
#'@examples
#'
#' set.seed(3486)
#' ss <- expand.grid(Trt=0:3, ClusterSize=5, Freq=4)
#' #Trt is converted to a factor
#' rd <- ran.CBData(ss, p.gen.fun=function(g)0.2+0.1*g)
#' rd
#'
◇
```

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Uses: `ran.CBData` 9b.

"../R/CBData.R" 9b≡

```
ran.CBData <- function(sample.sizes, p.gen.fun=function(g)0.3,
```

```
                          rho.gen.fun=function(g)0.2, pdf.fun=qpower.pdf){
  ran.gen <- function(d){
  # d is subset(sample.sizes, Trt==trt, ClusterSize==cs)
    cs <- d$ClusterSize[1]
    trt <- unclass(d$Trt)[1]
    n <- d$Freq[1]
    p <- p.gen.fun(trt)
    rho <- rho.gen.fun(trt)
    probs <- pdf.fun(p, rho, cs)
    tmp <- rmultinom(n=1, size=n, prob=probs)[,1]
    cbind(Freq=tmp, NResp=0:cs, ClusterSize=d$ClusterSize, Trt=d$Trt)}

  sst <- if (is.factor(sample.sizes$Trt)) sample.sizes$Trt else factor(sample.sizes$Trt)
  a <- by(sample.sizes, list(Trt=sst, ClusterSize=sample.sizes$ClusterSize), ran.gen)
  a <- data.frame(do.call(rbind, a))
  a$Trt <- factor(a$Trt, labels=levels(sst))
  a <- a[a$Freq>0, ]
  class(a) <-  c("CBData", "data.frame")
  a
 }
◇
```

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Defines: `ran.CBData` 1, 9a, 11a.

4.1. **Parametric pdf generating functions.** `betabin.pdf` and `qpower.pdf` provide two classic distributions – beta-binomial and q-power – for generating correlated binary data. Either can be used in `ran.CBData`.

"../R/CBData.R" 11a≡

```
#'Parametric distributions for correlated binary data
#'
#'\code{qpower.pdf} and \code{betabin.pdf} calculate the probability
#'distribution function for the number of responses in a cluster of the q-power
#'and beta-binomial distributions, respectively.
#'
#'The pdf of the q-power distribution is \deqn{P(X=x) =
#'{{n}\choose{x}}\sum_{k=0}^x (-1)^k{{x}\choose{k}}q^{(n-x+k)^\gamma},}{P(X=x)
#'= C(n,x)\sum_{k=0}^x (-1)^kC(x,k)q^((n-x+k)^g),} \eqn{x=0,\ldots,n}, where
#'\eqn{q=1-p}, and the intra-cluster correlation \deqn{\rho =
#'\frac{q^{2^\gamma}-q^2}{q(1-q)}.}{rho = (q^(2^g)-q^2)/(q(1-q)).}
#'
#'The pdf of the beta-binomial distribution is \deqn{P(X=x) = {{n}\choose{x}}
#'\frac{B(\alpha+x, n+\beta-x)}{B(\alpha,\beta)},}{P(X=x) = C(n,x)
#'B(a+x,n+b-x)/B(a,b),} \eqn{x=0,\ldots,n}, where \eqn{\alpha=
#'p\frac{1-\rho}{\rho}}{a=p(1-rho)/rho}, and \eqn{\alpha=
#'(1-p)\frac{1-\rho}{\rho}}{b=(1-p)(1-rho)/rho}.
#'
#'@export
#'@name pdf
#'@aliases qpower.pdf betabin.pdf
#'@param p numeric, the probability of success.
#'@param rho numeric between 0 and 1 inclusive, the within-cluster correlation.
#'@param n integer, cluster size.
#'@return a numeric vector of length \eqn{n+1} giving the value of \eqn{P(X=x)}
#'for \eqn{x=0,\ldots,n}.
#'@author Aniko Szabo, aszabo@@mcw.edu
#'@seealso \code{\link{ran.CBData}} for generating an entire dataset using
#'these functions
#'@references Kuk, A. A (2004) Litter-based approach to risk assessment in
#'developmental toxicity studies via a power family of completely monotone
#'functions \emph{Applied Statistics}, 52, 51-61.
#'
#'Williams, D. A. (1975) The Analysis of Binary Responses from Toxicological
#'Experiments Involving Reproduction and Teratogenicity \emph{Biometrics}, 31,
#'949-952.
#'@keywords distribution
#'@examples
#'
#'#the distributions have quite different shapes
#'#with q-power assigning more weight to the "all affected" event than other distributions
#'plot(0:10, betabin.pdf(0.3, 0.4, 10), type="o", ylim=c(0,0.34),
#'   ylab="Density", xlab="Number of responses out of 10")
#'lines(0:10, qpower.pdf(0.3, 0.4, 10), type="o", col="red")
#'
◇
```

File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.
Uses: ran.CBData 9b.

"../R/CBData.R" 11b≡

```
betabin.pdf <- function(p, rho, n){
  a <- p*(1/rho-1)
  b <- (1-p)*(1/rho-1)
  idx <- 0:n
  res <- choose(n, idx)*beta(a+idx, b+n-idx)/beta(a,b)
  res
```

```
  }
```
◇
File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.

"../R/CBData.R" 12≡

```
#'@rdname pdf
#'@export
 qpower.pdf <- function(p, rho, n){
   .q <- 1-p
   gamm <- log2(log(.q^2+rho*.q*(1-.q))/log(.q))
   res <- numeric(n+1)
   for (y in 0:n){
     idx <- 0:y
     res[y+1] <- choose(n,y) * sum( (-1)^idx * choose(y,idx) * .q^((n-y+idx)^gamm))
   }
   res <- pmax(pmin(res,1),0)  #to account for numerical imprecision
   res
 }
```
◇
File defined by 2ab, 3ab, 4, 5a, 6ab, 7ab, 8, 9ab, 11ab, 12.