# Exact p-values for pairwise comparison of Friedman rank sums, with application to comparing classifiers
# Eisinga, Heskes, Pelzer & Te Grotenhuis
# BMC Bioinformatics, January 2 2017
# The main function 'pexactfrsd.2parts' given below computes the exact p-values for
# MCE-euclid-FC vs PLS-AREA-time, presented in the main text in Table 6, using the
data by Zagar et al. [70]. The main function calls function 'part1' to compute
probabilities and function 'part2' to compute p-values, and calculates the exact
p-value for the incomplete data using the method described in the manuscript.

pexactfrsd.2parts <- function(d,k1,n1,k2,n2) {
  result  <- 0
  i1 <- n1*(k1-1)
  for (i in -i1:+i1){
    result <- result + part1(i,k1,n1) * part2(d-i,k2,n2)
  }
  return(as.numeric(result))
}

part1 <- function(d,k,n) {
  result <- 0
  sum2   <- 0
  for (h in 0:n){
    sum1 <- choose(n,h) / (k^h * (1-k)^n)
    sum2 <- 0
    for (i in 0:h){
      for (j in 0:h){
        if (k*(i-j)-d-h >= 0) {
          sum2 <- sum2 + (-1)^(i-j) * choose(h,i) * choose(h,j) *
          choose(k*(i-j)-d+h-1,k*(i-j)-d-h)
        }
      }
    }
    result <- result + sum1 * sum2
  }
  return(as.numeric(result))
}

part2 <- function(d,k,n) {
  result <- 0
  sum2   <- 0
  for (h in 0:n){
    sum1 <- choose(n,h) / (k^h * (1-k)^n)
    sum2 <- 0
    for (i in 0:h){
      for (j in 0:h){
        if (k*(i-j)-d-h >= 0) {
          sum2 <- sum2 + (-1)^(i-j) * choose(h,i) * choose(h,j) *
          choose(k*(i-j)-d+h,k*(i-j)-d-h)
        }
      }
    }
    result <- result + sum1 * sum2
  }
  return(as.numeric(2*result))
}

# Data Zagar et al. [70]
# ----------------------
k=12;n=9
k1=12;n1=9
k2=10;n2=1
# P-values excluding GDS2688
# --------------------------
d=37
# unadjusted p-values
unadj <- part2(d=d, k=k, n=n)

# Bonferroni-adjusted p-values for N x 1 comparison
BonfadjNx1 <- part2(d=d, k=k, n=n) * (k-1)

# Bonferroni-adjusted p-values for N x N comparison
BonfadjNxN <- part2(d=d, k=k, n=n) * choose(k, 2)

# P-values including GDS2688
# --------------------------
d=46

c21 <- pexactfrsd.2parts(d=d, k1=k1, n1=n1, k2=k2, n2=n2)

c22 <- pexactfrsd.2parts(d=d, k1=k1, n1=n1, k2=k2, n2=n2) * (k-1)

c23 <- pexactfrsd.2parts(d=d, k1=k1, n1=n1, k2=k2, n2=n2) * choose(k, 2)

# Print out of results presented in Table 6
# ------------------------------------------
rbind(cbind(unadj, BonfadjNx1, BonfadjNxN), cbind(c21, c22, c23))

# ----------------------------------- End of file ---------------------------------- #
# Exact p-values for pairwise comparison of Friedman rank sums, with application to comparing classifiers
# Eisinga, Heskes, Pelzer & Te Grotenhuis
# BMC Bioinformatics, January 2 2017

pexactfrsd

Description
The function pexactfrsd computes the exact p-value of the absolute value of rank sum difference d of a pair of Friedman rank sums, in an analysis of k treatments and n blocks. It also offers the possibility to compute the mass point probability, the number of compositions, and the cumulative number of compositions of the absolute value of rank sum difference d, by specifying an optional argument.

Usage
pexactfrsd(d,k,n,option)

Arguments
- d absolute value of rank sum difference.
- k number of treatments.
- n number of blocks.
- option character string indicating the desired statistic: "pvalue", "probability", "no of compositions", or "cumulative no of compositions". If the character string is not provided in the function call, the function returns the exact p-value (default).

Values
The potential range of rank sum difference d is 0 to n(k-1) inclusive. The number of treatments k should be at least 2, and the number of blocks n at least 1. Depending on the option specified, the function computes the following:
- "pvalue" returns P(D>=d;k,n)
- "probability" returns P(D=d;k,n)
- "no of compositions" returns W(D=d;k,n)
- "cumulative no of compositions" returns W(D>=d;k,n).

Details
The function pexactfrsd is an implementation of the algorithm provided in Eisinga et al. [1]. The function requires the R package Rmpfr [2] to be installed. In the script below the maximal precision, in bits, is set at (precBits =) 2048, which is sufficient even for rather large values of n (100, say).

Note
It is important to note that the results pertain to the absolute value of d. The rank sum difference distribution with positive and negative d values is symmetric around 0. Hence the probability mass to the left of d=0 may be folded over, producing a discrete distribution of non-negative d, ranging from 0 to n(k-1). The probability P(D=d;k,n) of all d except d=0 in the distribution of non-negative d is doubled relative to the probability in the symmetric distribution, so that they sum to unity. The same doubling goes for the (cumulative) number of compositions. Consequently, the number of compositions of d=n(k-1), for example, equals 2 and not 1, as is the case for the symmetric discrete distribution with support d=[-n(k-1),n(k-1)].

Examples
Example 1 following the R function with argument values d=k=n=100 returns a p-value of 0.8085251. The result is obtained in about 0.5 secs, using an Intel Core i7-3520M CPU @ 2.9Ghz. Example 2 calculates for k=n=10 the exact p-value of d values ranging from 1 to n(k-1), and subsequently plots and prints out the results. Example 3 generates the statistics presented in the example application in Additional file 4 of Eisinga et al. [1]. Example 4 computes the p-value and the
# mid p-value for k=n=5 in Table 4 of Eisinga et al. [1].

# References

# [1] Eisinga, Rob, Tom Heskes, Ben Pelzer, Manfred Te Grotenhuis (2017). Exact
# p-values for pairwise comparison of Friedman rank sums, with application
# to comparing classifiers, BMC Bioinformatics
# Reliable, Version 0.6-0, December 4 2015,
# https://cran.r-project.org/web/packages/Rmpfr/index.html.

```r
library(Rmpfr)
pexactfrsd <- function(d,k,n,option) {
  if (any(n < 1))                      stop("n out-of-bounds: min = 1")
  if (any(k < 2))                      stop("k out-of-bounds: min = 2")
  if (any(d < 0) || any(d > n*(k-1))) stop("d out-of-bounds: min,max = 0,n(k-1)")
  if (missing(option)) {option = "pvalue"}
  result <- 0
  for (h in 0:n) {
    sum1 <- chooseZ(n,h)/mpfr((pow.bigz(k,h) * pow.bigz(1-k,n)), precBits = 2048)
    sum2 <- 0
    for (s in 0:h) {
      if (any(k*s-d-h >= 0)) {
        if (option == "pvalue" || option == "cumulative no of compositions"){
          sum2 <- sum2 + (-1)^s * chooseZ(2*h,h+s) * chooseZ(k*s-d+h,k*s-d-h)}
        if (option == "probability" || option == "no of compositions"){
          sum2 <- sum2 + (-1)^s * chooseZ(2*h,h+s) * chooseZ(k*s-d-h-1,k*s-d-h)}
      }
    }
    result <- result + sum1 * sum2
  }
  if (any(d == 0) & option== "pvalue") return(1)
  if (any(d != 0) & option== "pvalue") return(as.numeric(2*result))
  if (any(d == 0) & option== "probability") return(as.numeric(result))
  if (any(d != 0) & option== "probability") return(as.numeric(2*result))
  if (any(d == 0) & option== "no of compositions" || option== "cumulative no of compositions")
    return(round(2*result*mpfr(pow.bigz(k*(k-1),n), precBits = 2048)))
  if (any(d == 0) & option== "no of compositions")
    return(round(result*mpfr(pow.bigz(k*(k-1),n), precBits = 2048)))
  if (any(d == 0) & option== "cumulative no of compositions")
    return(round(mpfr(pow.bigz(k*(k-1),n), precBits = 2048)))
}
```

```r
# Example 1
pexactfrsd(d=100,k=100,n=100)
nrep=1; system.time(replicate(nrep,pexactfrsd(d=100,k=100,n=100))) / nrep

# Example 2
k=10;n=10;d=c(1:(n*(k-1)))
pvalue <- pexactfrsd(d=d,k=k,n=n)
plot(d,pvalue,type="s",col="blue"); pvalue

# Example 3
k=3;n=2;d=c(1:(n*(k-1)))
pexactfrsd(d,k,n, "no of compositions")
pexactfrsd(d,k,n, "cumulative no of compositions")
pexactfrsd(d,k,n, "probability")
```
pexactfrsd(d,k,n, "pvalue")
#
#
# Example 4
# ---------
#
# k=5; n=5; d=11
#
pvalue <- pexactfrsd(d, k, n)
midpvalue <- (1/2) * pexactfrsd(d=(d-1), k, n,"probability") + pexactfrsd(d, k, n)
pvalue
midpvalue
#
# ----------------------------------- End of file ----------------------------------