Package 'HeterFunctionalData'

August 20, 2020

Type Package

Title Test of No Main and/or Interaction Effects in Functional Data

Version 0.1.0

Author Haiyan Wang, Mickael Akritas, James Higgins, Dale Blasi

Maintainer Haiyan Wang <hwang@ksu.edu>

Description Distribution free heteroscedastic tests for functional data.
The following tests are included in this package: test of no main treatment or contrast effect and no simple treatment effect given in
Wang, Higgins, and Blasi (2010) <doi:10.1016/j.spl.2009.11.016>,
no main time effect, and no interaction effect based on
original observations given in Wang and Akritas (2010a)
<doi:10.1080/10485250903171621>
and tests based on ranks given in Wang and Akritas (2010b)
<doi:10.1016/j.jmva.2010.03.012>.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Depends R (>= 3.1.0)

Imports stats

NeedsCompilation no

Repository CRAN

Date/Publication 2020-08-20 09:00:03 UTC

R topics documented:

`CF` *Classical F test for mixed effects model*

Description

#' This function conducts the test of main effect of treatment, interaction effect of treatment and time, main effect of time, and simple effect of treament based on mixed effects models

Usage

```
CF(data, a, b, mn)
```

Arguments

`data`	The data in long format (see the example in function dataformat_wide_to_long( )).
`a`	The number of treatments.
`b`	The number of time points or repeated measurements per subject.
`mn`	The vector of sample sizes in treatments.

Value

a list of p-values from mixed effect model for the test of no main treatment effect (palpha), test of no main time effect (pbeta), test of no interaction effect between treatment and time (pgamma), and test of no simple effect of treatment (pphi).

cov_squared_jackknife *Jackknife estimate of the squared covariance between two time points*

Description

This function provides the Jackknife estimate of the squared covariance using i.i.d. obervations from one variable given in vector x and i.i.d. observations from a second variable given in vector y.

Usage

```
cov_squared_jackknife(x, y)
```

Arguments

x	a vector of i.i.d. observations
y	a second vector of i.i.d. observations

Value

Jackknife estimate of the squared covariance

dataformat_wide_to_long *Convert data from wide format to long format*

Description

The function dataformat_wide_to_long() takes data in wide format (see the example) and convert to long format so that it will be ready for function Heter.test().

Usage

```
dataformat_wide_to_long(data)
```

Arguments

data	The data stored in wide format. The first column is the index for subject (named as sub). The second column is index for treatment (named as trt). The remaining columns are named time1, time2, etc. See the example.

Value

The data in long format.

Examples

```
# Example of data in wide format
# sub trt      time1          time2         time3           time4         time5
 # 1   1  2.4644642  1.7233498 -1.1374695 -0.5242729 -2.379145
 # 2   1  2.5746848  1.0181738 -0.8325308 -2.4873067 -3.463602
 # 3   1  2.5813995 -0.7528324 -3.1457645 -3.3135573 -4.364621
 # 4   1  0.8232141  0.2394987 -2.2073150 -3.3583005 -6.073399
 # 5   1  0.8274860  0.8323298 -2.1028060 -2.6015848 -3.291307
 # 1   2 -2.2217084  0.6779049  3.6310542  3.2052691  4.310316
 # 2   2 -3.3954705 -0.7827040  3.1364749  3.7184895  5.118996
 #
 # Data stored in long format
 # x_{ijk}, k=1, ..., n_i are the kth observation from the ith subject at time j.
 # 1 1 1 x111
 # 1 1 2 x112
 # 1 2 1 x121
 # 1 2 2 x122
 # 1 2 3 x123

 #  The following example generate a data set that contains data from
 # 3 treatments, with 3 subjects in treatment 1, 3 subjects in treatment 2,
 # and 4 subjects in treatment 3. Each subject contains m=50
 # repeated observations from Poisson distribution. For the 1st treatment,
 # the mean vector of the repeated observations from the same subject is
 # equal to mu1 plus a random effect vector generated by NorRanGen( ).
 # The m is the number of repeated measurements per subject.
 f1<-function(m, mu1, raneff) {
     currentmu=mu1+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
 f2<-function(m, mu2, raneff) {
     currentmu=mu2+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
 f3<- function(m, mu3, raneff){
     currentmu=mu3+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}

 # The a is the number of treatments. The mn stores the number of subjects in treatments.
 a=3;  mn=c(3, 3, 4); mu1=3; mu2=3; mu3=3; m=50
 raneff=NorRanGen(m) # generate random effects with AR(1) structure.

 # Generate data and store in wide format.
datawide=numeric()
now=0
for (i in 1:a){
   fi=function(x1, x2) f1(m,x1, x2)*(i==1)+f2(m,x1, x2)*(i==2)+f3(m, x1, x2)*(i==3)
   mu=mu1*(i==1)+mu2*(i==2)+mu3*(i==3)
   for (k in 1:mn[i]){
          now=now+1
          datawide<-rbind(datawide, c(k, i, fi(mu, raneff) )    )
```

```
            colnames(datawide)=c("sub", "trt", paste("time", seq(m), sep=""))
            #this is a typical way to store data in practice
       }
    } #end of j

    dat=dataformat_wide_to_long(datawide)
```

---

eu *A function to calaculate cell residues.*

Description

This function calcualtes the residual $x_ijk-barx_ij.$ for the kth replication from the ith group at jth time point.

Usage

```
eu(x, coln, Rij, Rik)
```

Arguments

x	the data matrix in long format, whose first column gives the index i, 2nd columns given the index j, 3rd column gives the index k, and the last column gives the observation x_ijk. For given i and j, $x_ijk, k=1, ..., n_ij$ are assumed to be i.i.d. from the same distribution.
coln	takes value 4 or 5. If coln=4, the calculation uses original data; if coln=5, the calculation uses rank. The default value for coln is 5.
Rij	is the mean of all observations in ith treatment at jth measurement calculated in other functions
Rik	is the mean for the kth subject in ith treatment calculated in other functions

Value

The residual to be used in Heter.test()

---

fun.sigijj12 *Unbiased estimate of $sigma_ijj1^2$*

Description

This function calculatess an unbiased estimate of $sigma_ijj1^2$ using the u-statisitic of vectors $x=(x_1, x_2, ..., x_ni)$ and $y=(y_1, y_2, ..., y_ni)$, where X_j and Y_j are correlated, but X_j and Y_j1 are independent if $j ne j1$. Note: $sum_k1 ne k2 ne k3 ne k4 (x_k1-x_k2))(y_k1-y_k2)) (x_k3-x_k4)) (y_k3-y_k4))$ is an unbiased est. of $4*ni*(ni-1)*(ni-2)*(ni-3) [E(X_ijk-mu_ij u_ij1k)]^2$.

Usage

```
fun.sigijj12(x, y)
```

Arguments

x	a vector
y	a vector

Value

A list containing two variables sigmaijj12 and ssijj1 in it. The sigmaijj12 variable gives an unbiased estimate of $sigma_ijj1^2$. The ssijj1 variable gives an unbiased estimate of $sigma_ijj sigma_ij_1j_1$.

Heter.test *Heteroscedastic test for functional data*

Description

This function conducts the test of main effect of treatment, interaction effect of treatment and time, main effect of time, and simple effect of treament based on original observations (see Wang and Akritas 2010a) or ranks (see Wang and Akritas 2010b).

Usage

```
Heter.test(
  data,
  a,
  b,
  mn,
  h = 0.45,
  method = "rank",
  Ca = cbind(as.vector(rep(1, a - 1)), -diag(a - 1))
)
```

Arguments

data	The data in long format (see the example in function dataformat_wide_to_long( )).
a	The number of treatments.
b	The number of time points or repeated measurements per subject.
mn	The vector of sample sizes in treatments.

h	The h value used in the estimators in Proposition 3.3 of Wang and Akritas (2010a) and Theorem 3.2 of Wang, Higgins, and Blasi (2010). The value of h should be in (0, 0.5) for the test of no main treatment effect or contract effect of treatments. For other tests, h should be in (0, 1). Recommend to use the default value h=0.45 as given in the function. Note: If multiple values are provided to h as a vector, then the calculation will be carried out for each h value, which results in multiple p-values in the returned result for palpha, pbeta, pgamma, pphi.
method	Specifying method='rank' to use rank test. For all other values, the test based on original data will be used.
Ca	Contrast matrix for the contrast effect of the treatments.

Value

a matrix of p-values for the test of no main treatment effect (palpha), test of no main time effect (pbeta), test of no interaction effect between treatment and time (pgamma), and test of no simple effect of treatment (pphi). For each h value, the p-values are in the same column.

References

Haiyan Wang and Michael Akritas (2010a). Inference from heteroscedastic functional data, Journal of Nonparametric Statistics. 22:2, 149-168. DOI: 10.1080/10485250903171621

Haiyan Wang and Michael Akritas (2010b). Rank test for heteroscedastic functional data. Journal of Multivariate Analysis. 101: 1791-1805. https://doi.org/10.1016/j.jmva.2010.03.012

Haiyan Wang, James Higgins, and Dale Blasi (2010). Distribution-Free Tests For No Effect Of Treatment In Heteroscedastic Functional Data Under Both Weak And Long Range Dependence. Statistics and Probability Letters. 80: 390-402. Doi:10.1016/j.spl.2009.11.016

Examples

```
# Generate a data set that contains data from 3 treatments,
# with 3 subjects in treatment 1, 3 subjects in treatment 2,
# and 4 subjects in treatment 3. Each subject contains m=50
# repeated observations from Poisson distribution. For the 1st treatment,
# the mean vector of the repeated observations from the same subject is
# equal to mu1 plus a random effect vector generated by NorRanGen( ).
# The m is the number of repeated measurements per subject.
 f1<-function(m, mu1, raneff) {
     currentmu=mu1+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
 f2<-function(m, mu2, raneff) {
     currentmu=mu2+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
f3<- function(m, mu3, raneff){
     currentmu=mu3+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
# The a is the number of treatments. The mn stores the number of subjects in treatments.
```

```
a=3;  mn=c(3, 3, 4); mu1=3; mu2=3; mu3=3; m=50
# Generate the time effects via random effects with AR(1) structure.
raneff=NorRanGen(m)
# Generate data and store in wide format.
datawide=numeric()
now=0
for (i in 1:a){
   fi=function(x1, x2) f1(m,x1, x2)*(i==1)+f2(m,x1, x2)*(i==2)+f3(m, x1, x2)*(i==3)
   mu=mu1*(i==1)+mu2*(i==2)+mu3*(i==3)
   for (k in 1:mn[i]){
       now=now+1
       datawide<-rbind(datawide, c(k, i, fi(mu, raneff) )    )
       colnames(datawide)=c("sub", "trt", paste("time", seq(m), sep=""))
       #this is a typical way to store data in practice
   }
} #end of j

# Note:There are different time effects since values in raneff vector are different
dat=dataformat_wide_to_long(datawide)   #dat is in long format
# Define the h value used in Proposition 3.3 of Wang and Akritas (2010a)
h=c(0.45, 0.49)
  #Note: The resulting palpha, pbeta, pgamma, pphi each contains
  #      two p-values, one corresponds to each h value
  #      (see Proposition 3.3 of Wang and Akritas (2010a))
# test based on original data.
(org= Heter.test(dat, a, m, mn, h, method='original') )
#test based on ranks
(rankt= Heter.test(dat, a, m, mn, h, method='rank') )
```

NorRanGen *Generate a vector of random effects with specific correlation structure and given variance*

Description

Generate a vector of random effects that follow AR(1) correlation structure with rho=exp(-1/m) and variances sigmaj being given or andomly generated from uniform distribution on (1.2, 1.4).

Usage

```
NorRanGen(m, sigmaj = stats::runif(m, 1.2, 1.4))
```

Arguments

m	the length of the vector of the random effects
sigmaj	standard deviations for the random effect vector. Default is a vector from U(1.2, 1.4).

Value

the random effect vector of length m

Examples

```
   m=50;  raneff=NorRanGen(m)
# Note: If X ~ N(0, I), then tran X ~ N(0, A) with
# A being the cov matrix of AR(1), which contains the standard deviations sigma and the
# correlation coeff rho=exp(-1/m).
# i.e. corr= (1  rho  rho^2 ... rho^(m-1)
#             rho 1   rho   ... rho^(m-2)
#                 ...................
#             rho^(m-1) rho^(m-2) ... rho )
#
# To see the correlation values, run the following example
# j1=seq(25); cv=numeric()
# for (j in 1:25){
#  lag=abs(j1-j)/25; cv=rbind(cv, exp(-lag))
#}
# row.names(cv)=j1;  colnames(cv)=j1;  cv[1,]
```

---

sigma4 *Unbiased estimate of squared variance $sigma^4$ based on U-statistic of an i.i.d. sample*

Description

This function takes a random sample and produces an unbiased estimate of the squared variance based on the U-statistic $sum_k1 ne k2 ne k3 ne k4 (x_k1-x_k2))^2 (x_k3-x_k4))^2$.

Usage

```
sigma4(x)
```

Arguments

x	is a vector of i.i.d. observations.

Value

unbiased estimate of squared variance $sigma^4$, where $sigma^2$ is the variance.

Examples

```
x=stats::rnorm(10)
sigma4(x)
```

sigma4bootstrap *Bootstrap estimate of $sigma^4$ using an i.i.d. sample*

Description

This function takes a random sample and approximates an unbiased estimate of the squared variance based on Bootstrap estimate.

Usage

```
sigma4bootstrap(x)
```

Arguments

x	is a vector of i.i.d. observations.

Value

Bootstrap estimate of $ sigma^4$, where $sigma^2$ is the variance.

Examples

```
x=stats::rnorm(10)
sigma4bootstrap(x)
```

sigma4jackknife *Jackknife estimate of $sigma^4$ using an i.i.d. sample*

Description

This function takes a random sample and approximates an unbiased estimate of the squared variance based on Jackknife estimate.

Usage

```
sigma4jackknife(x)
```

Arguments

x	is a vector of i.i.d. observations.

Value

Jackknife estimate of $sigma^4$, where $sigma^2$ is the variance.

Examples

```
x=stats::rnorm(10)
sigma4jackknife(x)
```

taufun *An intermediate function to calculate the partial sums*

Description

This function is internally used to calculate the partial sums which will then be used in calculating the asymptotic variances of the test statistics in Heter.test()

Usage

```
taufun(u, ranks, d1, d2, d3, a, b, mn, mcon, coln)
```

Arguments

u	a vector
ranks	a vector corresponds to either the fourth column of data or rank of the observations.
d1	a vector corresponds to the first column of data in the long format.
d2	a vector corresponds to the second column of data in the long format.
d3	a vector corresponds to the third column of data in the long format.
a	The number of treatments.
b	The number of time points or repeated measurements per subject.
mn	The vector of sample sizes in treatments.
mcon	a vector corresponds to b^h in Theorem 3.2 of Wand and Akritas (2010a)
coln	takes value 4 or 5. Value 4 is for the tests based on original data and value 5 is for the tests based on ranks.

Value

Asymptotic variances to be used in Heter.test().

tcontrast *Test of no contrast effect of the treatments*

Description

This function conducts the test of no contract effect of treatments based on Theorem 3.2 of Wang, Higgins, and Blasi (2010).

Usage

```
tcontrast(
  data,
  a,
  b,
  mn,
  h = 0.45,
  method = "rank",
  Ca = cbind(as.vector(rep(1, a - 1)), -diag(a - 1))
)
```

Arguments

data	The data in long format (see the example in function dataformat_wide_to_long( )).
a	The number of treatments.
b	The number of time points or repeated measurements per subject.
mn	The vector of sample sizes in treatments.
h	The h value used in the estimators in Theorem 3.2 of Wang, Higgins, and Blasi (2010). The value of h should be in (0, 0.5). Recommend to use the default value h=0.45 as given in the function. Note: If multiple values are provided to h as a vector, then the calculation will be carried out for each h value, which results in multiple p-values in the returned result.
method	Specifying method='rank' to use rank test. For all other values, the test based on original data will be used.
Ca	Contrast matrix for the contrast effect of the treatments. The default contrast corresponds to the main treatment effect.

Value

a matrix that contains the test statistics and pvalues for each h value.

References

Haiyan Wang and Michael Akritas (2010a). Inference from heteroscedastic functional data, Journal of Nonparametric Statistics. 22:2, 149-168. DOI: 10.1080/10485250903171621

Haiyan Wang and Michael Akritas (2010b). Rank test for heteroscedastic functional data. Journal of Multivariate Analysis. 101: 1791-1805. https://doi.org/10.1016/j.jmva.2010.03.012

Haiyan Wang, James Higgins, and Dale Blasi (2010). Distribution-Free Tests For No Effect Of Treatment In Heteroscedastic Functional Data Under Both Weak And Long Range Dependence. Statistics and Probability Letters. 80: 390-402. Doi:10.1016/j.spl.2009.11.016

Examples

```
# Generate a data set that contains data from 3 treatments,
# with 3 subjects in treatment 1, 3 subjects in treatment 2,
```

```
# and 4 subjects in treatment 3. Each subject contains m=50
# repeated observations from Poisson distribution. For the 1st treatment,
# the mean vector of the repeated observations from the same subject is
# equal to mu1 plus a random effect vector generated by NorRanGen( ).
# The m is the number of repeated measurements per subject.
 f1<-function(m, mu1, raneff) {
     currentmu=mu1+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
 f2<-function(m, mu2, raneff) {
     currentmu=mu2+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
 f3<- function(m, mu3, raneff){
     currentmu=mu3+raneff;
     currentmu[abs(currentmu)<1e-2]=1e-2;
     rpois(m, abs(currentmu))}
# The a is the number of treatments. The mn stores the number of subjects in treatments.
a=3;  mn=c(3, 3, 4); mu1=3; mu2=3; mu3=2; m=50
# Note treatment 3 has mean mu3=2, which is different from the mean of
#  the other two treatments.
# Generate the time effects via random effects with AR(1) structure.
raneff=NorRanGen(m)
# Generate data and store in wide format.
datawide=numeric()
now=0
for (i in 1:a){
   fi=function(x1, x2) f1(m,x1, x2)*(i==1)+f2(m,x1, x2)*(i==2)+f3(m, x1, x2)*(i==3)
   mu=mu1*(i==1)+mu2*(i==2)+mu3*(i==3)
   for (k in 1:mn[i]){
       now=now+1
       datawide<-rbind(datawide, c(k, i, fi(mu, raneff) )    )
       colnames(datawide)=c("sub", "trt", paste("time", seq(m), sep=""))
       #this is a typical way to store data in practice
   }
} #end of j

# Note:There are different time effects since values in raneff vector are different
dat=dataformat_wide_to_long(datawide)   #dat is in long format
  #Note: For each h value below, the test statistic and p-value are calculated.
  #      (see Theorem 3.2 of Wang, Higgins, and Blasi (2010))

tcontrast(dat, a, m, mn, h=c(0.45, 0.49), method='original')
```

Index