Package: BET (via r-universe)

October 4, 2026

Type Package

Title Binary Expansion Testing

Version 0.6.0

Depends R (>= 3.6.0)

Description Nonparametric detection of nonuniformity and dependence with Binary Expansion Testing (BET). See Kai Zhang (2019) BET on Independence, Journal of the American Statistical Association, 114:528, 1620-1637, <DOI:10.1080/01621459.2018.1537921>, Kai Zhang, Wan Zhang, Zhigen Zhao, Wen Zhou. (2023). BEAUTY Powered BEAST, <doi:10.48550/arXiv.2103.00674> and Wan Zhang, Zhigen Zhao, Michael Baiocchi, Yao Li, Kai Zhang. (2023) SorBET: A Fast and Powerful Algorithm to Test Dependence of Variables, Technical report.

License GPL

Encoding UTF-8

LazyData true

Imports Rcpp (>= 0.12.3)

LinkingTo Rcpp

RoxygenNote 7.1.1

NeedsCompilation yes

Author Wan Zhang [aut, cre], Zhigen Zhao [aut], Michael Baiocchi [aut], Kai Zhang [aut]

Maintainer Wan Zhang <wanz63@live.unc.edu>

Repository https://cran.r-universe.dev

Date/Publication 2026-10-04 21:39:13 UTC

RemoteUrl https://github.com/cran/BET

RemoteRef HEAD

RemoteSha ae258fcaa39f8ebaebed8998516c64ac5bbe3da5

Contents

BEAST 2
BEAST.asymptotic 4
BEAST.asymptotic.calibrate 5
bet.plot 6
BET_package 7
cell.counts 8
get.signs 8
MaxBET 9
MaxBETs 11
star 12
symm 13

Index **14**

`BEAST` *Binary Expansion Adaptive Symmetry Test*

Description

`BEAST` (Binary Expansion Adaptive Symmetry Test) is used for nonparametric detection of nonuniformity or dependence.

Usage

```
BEAST(
  X,
  dep,
  subsample.percent = 1/2,
  B = 100,
  unif.margin = FALSE,
  lambda = NULL,
  index = list(c(1:ncol(X))),
  method = "p",
  num = NULL
)
```

Arguments

`X`	a matrix to be tested.
`dep`	depth of the binary expansion for the `BEAST`.
`subsample.percent`	Fraction of observations in each subsample. The effective size is the positive integer truncation of n * subsample.percent.
`B`	Number of subsamples.

`unif.margin` logicals. If `TRUE` the marginal distribution is known to be Uniform[0,1]. Default is `FALSE`, and empirical cdf transformation will be applied to each marginal distribution.

`lambda` tuning parameter for soft-thresholding, default to be $\sqrt{\log(2^{p \cdot dep})/(8n)}$.

`index` a list of indices. If provided, test the independence among two or more groups of variables. For example, `index = list(c(1,2), c(3))` refers to test the independence between (X_1, X_2) and X_3. Default to be `list(c(1:p))` to test if the data follow the multivariate uniform distribution over $[0,1]^p$, where `p = ncol(X)`.

`method` If `"p"`, then compute null distribution with permutations. If `"s"`, then compute null distribution with simulations. If `"stat"`, only return interaction and BEAST Statistic. The `method = "s"` option is only available for testing uniformity and bivariate independence.

`num` number of permutations if method == "p" (default to be 100), or simulations if method == "s" (supply an explicit positive value).

Value

`Interaction` the most frequent interaction among all subsamples.

`BEAST.Statistic` BEAST statistic.

`Null.Distribution` simulated null distribution.

`p.value` simulated p-value.

Examples

```
## Elapsed times 7.32 secs
## Measured in R 4.0.2, 32 bit, on a processor 3.3 GHz 6-Core Intel Core i5 under MacOS, 2024/9/6
## Not run:
  x1 = runif(128)
  x2 = runif(128)
  y = sin(4*pi*(x1 + x2)) + 0.8*rnorm(128)
  ##test independence between (x1, x2) and y
  BEAST(cbind(x1, x2, y), 3, index = list(c(1,2), c(3)))
  ##test mutual independence among x1, x2 and y
  BEAST(cbind(x1, x2, y), 3, index = list(1, 2, 3))

  ##test bivariate uniformity
  x1 = rbeta(128, 2, 4)
  x2 = rbeta(128, 2, 4)
  BEAST(cbind(x1, x2), 3)
  ##test multivariate uniformity
  x1 = rbeta(128, 2, 4)
  x2 = rbeta(128, 2, 4)
  x3 = rbeta(128, 2, 4)
  BEAST(cbind(x1, x2, x3), 3)

## End(Not run)
```

`BEAST.asymptotic` *Bivariate Asymptotic BEAST Test*

Description

Tests independence between two continuous variables using a stored calibration based on the joint Gaussian limit of full-sample and internally reranked subsample interactions.

Usage

```
BEAST.asymptotic(X, calibration, alpha = 0.05, seed = NULL)
## S3 method for class 'BEASTAsymptoticResult'
print(x, ...)
```

Arguments

`X`	Numeric matrix or all-numeric data frame with exactly two columns, finite observations, no missing values, and nonconstant margins.
`calibration`	An object from `BEAST.asymptotic.calibrate`. Its sample size must equal `nrow(X)`. Tuning parameters are taken from this object.
`alpha`	Significance level strictly between zero and one.
`seed`	Optional integer seed in $[0, 2147483646]$ for the observed statistic's subsampling. An explicit seed preserves the caller's RNG state; `NULL` uses the current RNG stream.
`x`	A test result.
`...`	Reserved for print-method compatibility.

Details

The function calls `BEAST(method="stat")` with the tuning parameters stored in `calibration` and compares $\sqrt{n-1}B_\lambda$ with its `G0` draws.

The p-value is `mean(calibration$G0 >= Scaled.Statistic)`. The critical value is `sort(calibration$G0)[ceiling((1-` and rejection uses `Scaled.Statistic > critical.value`. At atoms, this strict decision can differ from comparing the empirical p-value with `alpha`.

The dataset must have two columns and the calibration's sample size. Marginal ties produce a warning; permutation calibration is recommended for tied data. A nonpositive critical value also requires permutation calibration and produces an error. A nonfinite BEAST statistic produces an error.

The same calibration can be reused across datasets with matching sample size and tuning parameters, and at different significance levels. See `BEAST.asymptotic.calibrate` for its construction.

Value

An object of class `BEASTAsymptoticResult`, with `BEAST.Statistic`, `Scaled.Statistic`, `p.value`, `critical.value`, logical `reject`, `alpha`, `Interaction` from the BEAST statistic, calibration configuration/method summary, tie indicator and optional seed.

See Also

BEAST.asymptotic.calibrate, BEAST

Examples

```
# Small counts for illustration; use larger H and G for inference.
i <- seq_len(32)
X1 <- cbind(i/33, i/33 + 0.08*sin(i))
X2 <- cbind(i/33, -i/33 + 0.08*cos(i))
cal <- BEAST.asymptotic.calibrate(32, dep=2, subsample.percent=1/2,
  B=2, lambda=0, H=12, G=64, seed=1)
BEAST.asymptotic(X1, cal, seed=11)
BEAST.asymptotic(X2, cal, seed=12)
```

BEAST.asymptotic.calibrate

Reusable Bivariate Asymptotic BEAST Calibration

Description

Constructs a reusable joint Gaussian reference for BEAST testing independence between two continuous variables.

Usage

```
BEAST.asymptotic.calibrate(n, dep = 3, subsample.percent = 3/16,
  B = 128, lambda = NULL, H = 1000, G = 10000, seed = NULL,
  ncores = 1)
## S3 method for class 'BEASTAsymptoticCalibration'
print(x, ...)
```

Arguments

n	Integer sample size, at least two.
dep	Binary expansion depth, an integer from 1 through 5.
subsample.percent	Subsample fraction in $(0, 1]$. The effective size is the positive integer truncation $r = \lfloor n \, \texttt{subsample.percent} \rfloor$.
B	Positive integer number of subsamples, denoted by m in the paper.
lambda	Finite nonnegative soft threshold. NULL uses the bivariate BEAST default $\sqrt{\log(2^{2\,\texttt{dep}})/(8n)}$.
H	Number of independent null replications for covariance estimation; integer at least two.
G	Number of joint Gaussian draws; integer at least two.
seed	Optional integer seed in $[0, 2147483646]$. See reproducibility details.
ncores	Must equal 1 for serial calibration.
x	A calibration object.
...	Reserved for print-method compatibility.

Details

Calibration uses all $q = (2^{\texttt{dep}} - 1)^2$ cross interactions. For each of `H` independent bivariate normal null samples, it computes the full-sample interaction means and their average over `B` size-r subsamples drawn without replacement. Marginal ranks are recomputed within each subsample. The covariance of this joint vector, scaled by $\sqrt{n-1}$, is estimated and negative eigenvalues are set to zero.

Joint Gaussian draws are soft-thresholded at $\sqrt{n-1}\lambda$. The BEAST limit map divides the inner product of the thresholded full and subsample vectors by the norm of the full vector, returning zero when that norm is zero. The resulting `G0` draws are stored for testing at different significance levels.

Both `n` and the effective r must be divisible by $2^{\texttt{dep}}$, with $r \geq 2$. A calibration can be saved with `saveRDS()` and reused for bivariate datasets with the same sample size and tuning parameters. For multivariate blocks, the covariance also depends on within-block dependence, so this reusable calibration is bivariate only.

An explicit `seed` makes calibration reproducible and preserves the caller's RNG state. With `seed=NULL`, a master seed is drawn from the current RNG stream and recorded in the object. Calibration is serial.

Value

An object of class `BEASTAsymptoticCalibration`, containing `n`, `dep`, `subsample.percent`, effective `r`, `B`, actual `lambda`, `q`, `H`, `G`, the covariance `Gamma`, pre-clipping `Gamma.raw`, `G0`, nonzero fraction, minimum raw eigenvalue, count of clipped eigenvalues, basis/configuration, RNG metadata, package version, method identifier and bivariate continuous-margin scope.

See Also

`BEAST.asymptotic`, `BEAST`

Examples

```
# Small counts for illustration; use larger H and G for inference.
cal <- BEAST.asymptotic.calibrate(32, dep = 2, subsample.percent = 1/2,
  B = 2, H = 8, G = 32, seed = 1)
cal
length(cal$G0)
```

`bet.plot` *Plotting Binary Expansion Testing (2-dimensions)*

Description

`bet.plot` shows the cross interaction of the strongest asymmetry, which the BET returns with the rejection of independence null. This function only works for the test on two variables, that is, `X` can only have two columns. There are $2^{2dep}-1$ nontrivial binary variables in the σ-field and $(2^dep-1)^2$ of them are cross interactions, whose positive regions are in plotted in white and whose negative regions are plotted in blue. `bet.plot` shows the cross interaction where the difference of number of observations in the positive and negative region is largest.

Usage

```
## S3 method for class 'plot'
bet(X, dep, unif.margin = FALSE, cex=0.5, index = list(c(1:ncol(X))), ...)
```

Arguments

X	a matrix with two columns.
dep	depth of BET.
unif.margin	logicals. If TRUE the marginal distribution is known to be Uniform[0,1]. Default is FALSE, and empirical cdf transformation will be applied to each marginal distribution.
cex	number indicating the amount by which plotting text and symbols should be scaled relative to the default.
index	a list of indices. If provided, test the independence among two or more groups of variables. For example, index = list(c(1,2)) refers to test data uniformity, index = list(1, 2) refers to test independence between X_1 and X_2. Default to be list(c(1:p)), where p = ncol(X), then test data uniformity.
...	graphical parameters to plot

Examples

```
v <- runif(128, -pi, pi)
X1 <- cos(v) + 2.5 * rnorm(128, 0, 1/20)
X2 <- sin(v) + 2.5 * rnorm(128, 0, 1/20)
bet.plot(cbind(X1, X2), 3, index = list(1,2))
```

BET_package *Binary Expansion Testing*

Description

The BET package provides functions for nonparametric detection of nonuniformity and dependence with Binary Expansion Testing (BET).

BET functions

MaxBET symm get.signs cell.counts bet.plot MaxBETs BEAST

Reference(s)

Kai Zhang (2019) BET on Independence, Journal of the American Statistical Association, 114:528, 1620-1637, doi:10.1080/01621459.2018.1537921, Kai Zhang, Zhigen Zhao, and Wen Zhou (2021). BEAUTY Powered BEAST, <arXiv:2103.00674> and Wan Zhang, Zhigen Zhao, Michael Baiocchi, Yao Li, Kai Zhang. SorBET: A Fast and Powerful Algorithm to Test Dependence of Variables. Techinical report, 2023.

cell.counts *Counts the amount of points in each cell after binary expansion.*

Description

cell.counts returns the amount of data points in each cell getting from binary expansion.

Usage

```
cell.counts(X, dep, unif.margin = FALSE)
```

Arguments

X	a matrix to be tested.
dep	depth of the marginal binary expansions.
unif.margin	logicals. If TRUE the data has been uniformed based on empirical cumulative distribution function. Default to be FALSE and the function uniforms the data.

Value

The result is a dataframe with 2 rows and $2^(p * dep)$ columns, where p is the number of columns of X. The first column is the binary index, the second column is the amount of data points.

Examples

```
v <- runif(128, -pi, pi)
X1 <- cos(v) + 2.5 * rnorm(128, 0, 1/20)
X2 <- sin(v) + 2.5 * rnorm(128, 0, 1/20)
cell.counts(cbind(X1, X2), 3)
```

get.signs *Signs of Colors of all Points for all Interactions*

Description

get.signs returns all the signs of colors for each point under all interactions up to depth d in marginal binary expansions for the tests BET and BETs.

Usage

```
get.signs(X, dep, unif.margin = FALSE)
```

Arguments

X	a matrix to be tested.
dep	depth of the marginal binary expansions.
unif.margin	logicals. If TRUE the data has been uniformed based on empirical cumulative distribution function. Default to be FALSE and the function uniforms the data.

Value

The result is a dataframe with n rows and $2^(p * dep)$ columns, where p is the number of columns of X and n is the number of rows of X. The values of 1 or -1 stand for the sign of color, while the marginal interactions return 0.

Examples

```
v <- runif(128, -pi, pi)
X1 <- cos(v) + 2.5 * rnorm(128, 0, 1/20)
X2 <- sin(v) + 2.5 * rnorm(128, 0, 1/20)
get.signs(cbind(X1, X2), 3)
```

MaxBET *Binary Expansion Testing at a Certain Depth*

Description

MaxBET stands for Binary Expansion Testing. It is used for nonparametric detection of nonuniformity or dependence. It can be used to test whether a column vector is [0, 1]-uniformly distributed. It can also be used to detect dependence between columns of a matrix X, if X has more than one column.

Usage

```
MaxBET(
  X,
  dep,
  unif.margin = FALSE,
  asymptotic = TRUE,
  plot = FALSE,
  index = list(c(1:ncol(X)))
)
```

Arguments

X	a matrix to be tested. When X has only one column, BET will test whether X is [0, 1]-uniformly distributed (an error will be given if data is out of range [0, 1]). When X has two or more columns, BET tests the independence among those column vectors.

dep	depth of the binary expansion for the BET.
unif.margin	logicals. If TRUE the marginal distribution is known to be Uniform[0,1]. Default is FALSE, and empirical cdf transformation will be applied to each marginal distribution.
asymptotic	logicals. If TRUE the p-value is computed by asymptotic distribution. Default to be TRUE. Ignored if X has three or more columns.
plot	logicals. If TRUE, make the plot of cross interaction of the strongest asymmetry. Default to be FALSE. This option only works for X with two columns.
index	a list of indices. If provided, test the independence among two or more groups of variables. For example, index = list(c(1,2), c(3))) refers to test the independence between (X_1, X_2) and X_3. Default to be list(c(1:p)), where p = ncol(X), then test data uniformity.

Details

MaxBET tests the independence or uniformity by considering the maximal magnitude of the symmetry statistics in the $sigma$-field generated from marginal binary expansions at the depth d.

Value

Interaction	a dataframe with p columns, where p is the number of columns of X. It displays the interactions where the extreme symmetry statistics happens. For each column in X, we use a binary index to indicate binary variables involved in the extreme symmetry statistic.
Extreme.Asymmetry	the extreme asymmetry statistics.
p.value.bonf	p-value of the test with Bonferroni adjustment.
z.statistic	normal approximation of the test statistic.

Examples

```
##test mutual independence
v <- runif(128, -pi, pi)
X1 <- cos(v) + 2.5 * rnorm(128, 0, 1/20)
X2 <- sin(v) + 2.5 * rnorm(128, 0, 1/20)
MaxBET(cbind(X1, X2), 3, asymptotic = FALSE, index = list(1,2))

##test independence between (x1, x2) and y
x1 = runif(128)
x2 = runif(128)
y = sin(4*pi*(x1 + x2)) + 0.4*rnorm(128)
MaxBET(cbind(x1, x2, y), 3, index = list(c(1,2), c(3)))

##test uniformity
x1 = rbeta(128, 2, 4)
x2 = rbeta(128, 2, 4)
x3 = rbeta(128, 2, 4)
MaxBET(cbind(x1, x2, x3), 3)
```

MaxBETs *Binary Expansion Testing up to a Certain Depth*

Description

MaxBETs is used for nonparametric dependence detection. Extended from BET, for a chosen maximal depth d.max, MaxBETs does a sequential test up to d.max and avoids overlapping symmetry statistics in different depths, for all $2 \leq d \leq d.max$. The adjustment is done by multiplying the number of interactions which are in the σ-field generated by marginal binary expansions at depth d but not in that at depth $d-1$.

Usage

```
MaxBETs(
  X,
  d.max = 4,
  unif.margin = FALSE,
  asymptotic = TRUE,
  plot = FALSE,
  index = list(c(1:ncol(X)))
)
```

Arguments

X	a matrix to be tested. When X has only one column, BETs will test whether X is [0, 1]-uniformly distributed (an error will be given if data is out of range [0, 1]). When X has two or more columns, BETs tests the independence among those column vectors.
d.max	the maximal depth of the binary expansion for BETs.
unif.margin	logicals. If TRUE the marginal distribution is known to be Uniform[0,1]. Default is FALSE, and empirical cdf transformation will be applied to each marginal distribution.
asymptotic	logicals. If TRUE the p-value is computed by asymptotic distribution. Default to be TRUE. Ignored if X has three or more columns.
plot	logicals. If TRUE, make the plot of cross interaction of the strongest asymmetry. Default to be FALSE. This option only works for X with two columns.
index	a list of indices. If provided, test the independence among two or more groups of variables, for example, (X_1, X_2) and X_3.

Value

bet.s.pvalue.bonf	the overall p-value on the test.
bet.s.index	the interaction that the p-value is minimal.
bet.s.zstatistic	normal approximation of the test statistic.

Examples

```
##test mutual independence
v <- runif(128, -pi, pi)
X1 <- cos(v) + 2.5 * rnorm(128, 0, 1/20)
X2 <- sin(v) + 2.5 * rnorm(128, 0, 1/20)
MaxBETs(cbind(X1, X2), 3, asymptotic = FALSE, index = list(1,2))

##test independence between (x1, x2) and y
x1 = runif(128)
x2 = runif(128)
y = sin(4*pi*(x1 + x2)) + 0.4*rnorm(128)
MaxBETs(cbind(x1, x2, y), 3, index = list(c(1,2), c(3)))

##test uniformity
x1 = rbeta(128, 2, 4)
x2 = rbeta(128, 2, 4)
x3 = rbeta(128, 2, 4)
MaxBETs(cbind(x1, x2, x3), 3)
```

---

star *Coordinates of Brightest Stars in the Night Sky*

Description

This data set collects the galactic coordinates of the 256 brightest stars in the night sky (Perryman et al. 1997). We consider the longitude (x) and sine latitude (y) here.

Usage

```
data(star)
```

Format

An object of class `data.frame` with 256 rows and 2 columns.

Examples

```
data(star)
MaxBETs(cbind(star$x.raw, star$y.raw), asymptotic = FALSE, plot = TRUE, index = list(1,2))
```

symm *Symmetry Statistics for all Interactions*

Description

symm returns all the symmetry statistics up to depth d in marginal binary expansions for the tests BET and BETs.

Usage

```
symm(
  X,
  dep,
  unif.margin = FALSE,
  print.sample.size = TRUE
)
```

Arguments

X	a matrix to be tested.
dep	depth of the marginal binary expansions.
unif.margin	logicals. If TRUE the data has been uniformed based on empirical cumulative distribution function. Default to be FALSE and the function uniforms the data.
print.sample.size	logicals. If TRUE print the sample size. Default to be TRUE.

Value

The result is a dataframe with $(p + 2)$ columns, where p is the number of columns of X. The first column gives the binary index for all variables, the next p columns displays all the interactions of respective variables, the last column of Statistics gives the respective symmetry statistic.

Examples

```
v <- runif(128, -pi, pi)
X1 <- cos(v) + 2.5 * rnorm(128, 0, 1/20)
X2 <- sin(v) + 2.5 * rnorm(128, 0, 1/20)
symm(cbind(X1, X2), 3)
```

Index

∗ **datasets**
star, 12

BEAST, 2, *5–7*
BEAST.asymptotic, 4, *6*
BEAST.asymptotic.calibrate, *4*, 5, *5*
BET-package *(*BET_package*)*, 7
bet.plot, 6, *7*
BET_package, 7

cell.counts, *7*, 8

get.signs, *7*, 8

MaxBET, *7*, 9
MaxBETs, *7*, 11

print.BEASTAsymptoticCalibration
*(*BEAST.asymptotic.calibrate*)*, 5
print.BEASTAsymptoticResult
*(*BEAST.asymptotic*)*, 4

star, 12
symm, *7*, 13