Package 'sbivar'

November 20, 2026

Type Package

Title Test for Spatial Bivariate Association Across Omics Types

Version 0.99.49

Description Sbivar implements a suite of tests for Spatial BIVARiate association across omics modalities, with possibly disjoint coordinate sets. Implemented tests are generalized additive models (GAMs), modified t-test, bivariate Moran's I and Gaussian processes (GPs). Both single images and replicated experiments can be analysed.

License GPL-2

Encoding UTF-8

RdMacros Rdpack

biocViews Transcriptomics, Spatial, Proteomics, Metabolomics

Depends R (>= 4.6.0)

Imports abind, BiocParallel, concaveman, ggplot2, gstat(>= 2.0.3), lme4, lmerTest, Matrix, methods, mgcv, MultiAssayExperiment, nlme, Rcpp, Rdpack, reformulas, sf, smoppix(>= 1.1.8), SpatialExperiment, SpatialPack, spatstat.geom, stats, SummarizedExperiment

Suggests anndataR, htmltools, knitr, RANN, rhdf5, rmarkdown, openxlsx, testthat

LinkingTo Rcpp, RcppArmadillo

VignetteBuilder knitr

BugReports https://github.com/sthawinke/sbivar/issues

URL https://github.com/sthawinke/sbivar

Config/roxygen2/version 8.1.1

git_url https://git.bioconductor.org/packages/sbivar

git_branch devel

git_last_commit fbd3f9a

git_last_commit_date 2026-11-13

Repository Bioconductor 3.24

Date/Publication 2026-11-19

Author Stijn Hawinkel [cre, aut] (ORCID:
<https://orcid.org/0000-0002-4501-5180>),
Wangjun Hu [aut]

Maintainer Stijn Hawinkel <stijn.hawinkel@psb.ugent.be>

Contents

addFeatureColumn . . . 3
arrayDeriv . . . 4
arrayMatProd . . . 4
arrayProd2tr . . . 5
arrayProdTr . . . 5
assayT . . . 6
bdiagn . . . 6
buildAltSigmas . . . 7
buildDerivArray . . . 7
buildNewGrid . . . 8
buildSigmaGp . . . 9
buildWeightMat . . . 9
CCT . . . 10
checkControlVals . . . 10
checkInputSingle . . . 11
computeSigXws . . . 11
correlationsMulti . . . 12
evalVariogram . . . 12
evalVariogramCpp . . . 13
exploreWeights . . . 13
extractResultsMulti . . . 14
findDoubleUnderScore . . . 16
fitGAM . . . 16
fitGP . . . 17
fitLinModel . . . 17
fitManyGAMs . . . 18
fitManyGPs . . . 19
GAMsMulti . . . 19
GAMsSingle . . . 20
GaussKernel . . . 21
getApproxVar . . . 21
getDiscreteVars . . . 22
getFeaturesList . . . 22
getLengthScale . . . 23
getSize . . . 23
getSpatialCoords . . . 24
getX . . . 24
GPsSingle . . . 25
makeNames . . . 26
makeOffset . . . 26
makePval . . . 27
matheronVariograms . . . 27
ModTtestSingle . . . 28
moranControl . . . 28

MoransIMulti . . . 30
MoransISingle . . . 31
moveTwoCoords . . . 32
normMat . . . 33
plotCoords . . . 33
plotGAMs . . . 34
plotGAMsTopPair . . . 36
plotPairSingle . . . 36
plotSpotSums . . . 38
plotTopPair . . . 39
printIteration . . . 41
printProgress . . . 41
replaceLhs . . . 42
sbivar . . . 42
sbivarMulti . . . 44
sbivarResults-class . . . 45
sbivarSingle . . . 46
scaleHelpFun . . . 47
scaleMinusOne . . . 48
scaleZeroOne . . . 48
scoreTestInternals_cpp . . . 49
selfName . . . 49
show,sbivarResults-method . . . 50
splitSpatialExperiment . . . 50
strwrapPaste . . . 51
sund . . . 51
testGAM . . . 52
testGP . . . 52
topPairs . . . 53
vcovPredGam . . . 54
Vicari . . . 54
writeSbivarToXlsx . . . 55

Index **57**

`addFeatureColumn` *Add columns with feature names to a matrix by splitting rownames, and remove rownames*

Description

Add columns with feature names to a matrix by splitting rownames, and remove rownames

Usage

```
addFeatureColumn(x)
```

Arguments

x	A results matrix

Value

The matrix extended with two columns of feature names in front

---

`arrayDeriv` *Construct the nxnxg/2 array of derivatives for a nxn matrix to the g/2 covariance matrix parameters.*

Description

Construct the nxnxg/2 array of derivatives for a nxn matrix to the g/2 covariance matrix parameters.

Usage

```
arrayDeriv(fittedGP, distMat, what)
```

Arguments

`fittedGP`	The fitted Gaussian process (vector of 4 parameters)
`distMat`	The distance matrix
`what`	For which parameter is the derivative required?

Value

The matrix of derivatives

`arrayMatProd` *Multiply an array with a matrix*

Description

Array resulting from all matrix products of slices of array A of size mxmxp, with matrix M of size mxm. This is a faster version of `vapply(seq_len(p), FUN.VALUE = mat, function(i){mat %*% arr[,,i]})`, although it may consume more memory.

Usage

```
arrayMatProd(A, M)
```

Arguments

`A`	An mxmxp array
`M`	An mxm matrix

Details

The speedup comes from a single call to %*%, very efficient in BLAS.

Value

An mxmxp array

arrayProd2tr — *Find traces of all inner products of matrices composing arrays A and B yielding a pxp matrix*

Description

A faster version of `vapply(seq_len(p), FUN.VALUE = double(p), function(i){ vapply(seq_len(p), FUN.VALUE = double(1), function(j){ tr(arr[,,i] %*% arr2[,,j]) }) })`

Usage

```
arrayProd2tr(A, B)
```

Arguments

A, B	mxmxp arrays

Value

pxp matrix of traces

arrayProdTr — *Find traces of all inner products of matrices composing arrays A and B, after transposing the second yielding a pxp matrix*

Description

Returns the matrix resulting from the traces of all matrix products of slices of array A mxmxp with those of array B of the same dimensions. It is a faster version of `vapply(seq_len(p), FUN.VALUE = double(p), function(i){ vapply(seq_len(p), FUN.VALUE = double(1), function(j){ tr(crossprod(arr2[, arr[,,j])) }) })`

Usage

```
arrayProdTr(A, B)
```

Arguments

A, B	mxmxp arrays

Details

The speedup comes from a single call to crossprod, very efficient in BLAS.

Value

A pxp matrix of traces

assayT *Extract an assay, and transpose*

Description

Extract an assay, and transpose

Usage

```
assayT(x, assayName)
```

Arguments

x	The SummarizedExperiment object
assayName	The name of the assay

Value

The required assay, transposed

bdiagn *A wrapper for Matrix::bdiag maintaining names*

Description

A wrapper for Matrix::bdiag maintaining names

Usage

```
bdiagn(A, B)
```

Arguments

A, B	Matrix to be used in bdiag

Value

Same as bdiag but with dimnames

buildAltSigmas — *Construct cross-blocks of alternative covariance matrices at different length scales*

Description

These matrices are used to test for bivariate association at different length scales. Each alternative covariance matrix has the block structure $\Sigma_{alt,l} = \begin{bmatrix} I_n & C_l \\ C_l^T & I_m \end{bmatrix}$ where C_l is the Gaussian cross-covariance kernel evaluated at length scale l. Only the $n \times m$ cross-blocks are returned; the identity diagonal is implicit.

Usage

```
buildAltSigmas(distMat, numLscAlts, Quants, idN, idM)
```

Arguments

distMat	The complete distance matrix of Cx and Ey stacked. Only needed when sx, sy, derivX, or derivY are not supplied.
numLscAlts	Number of length scales (and thus number of cross-blocks to be built)
Quants	Most extreme quantiles of the distance distribution to be used as length scales.
idN, idM	indices for x and y in the distance matrix

Value

An $n \times m \times L$ array of cross-blocks C_l

buildDerivArray — *Build a 3-slice array of covariance-parameter derivatives*

Description

Build a 3-slice array of covariance-parameter derivatives

Usage

```
buildDerivArray(fittedGP, distMat, suffix)
```

Arguments

fittedGP	The fitted Gaussian process (vector of 4 parameters)
distMat	Distance matrix for the relevant modality
suffix	Character suffix to append to parameter names ("X" or "Y")

Value

An $n \times n \times 3$ array with 3rd-dim names `c("sigma<suffix>", "range<suffix>", "nugget<suffix>")`

`buildNewGrid` *Build a new grid covering the convex hull around two coordinate matrices*

Description

Given two coordinate matrices, concave hulls are found around them. The intersection between these two hulls is found, and in that area an evenly spaced, discrete grid is constructed.

Usage

```
buildNewGrid(Cx, Ey, n_points_grid)
```

Arguments

`Cx`, `Ey`	The coordinate matrices
`n_points_grid`	An integer, the number of points desired for the new grid

Details

The new grid will contain approximately the number of new points requested, depending on the size of the concave hull

Value

A data frame of two columns with all points of the grid, with column names x and y.

Note

This function is mainly used to create a grid on which two fitted GAMs can be evaluated to calculate correlations.

See Also

concaveman

Examples

```
Cx <- matrix(runif(40, 0, 1), 20, 2)
Ey <- matrix(runif(30, 0, 1), 15, 2)
buildNewGrid(Cx, Ey, n_points_grid = 50)
```

`buildSigmaGp` *Build the SAC matrix for a Gaussian process*

Description

Build the SAC matrix for a Gaussian process

Usage

```
buildSigmaGp(pars, distMat)
```

Arguments

`pars`	A vector of parameters
`distMat`	The complete distance matrix of Cx and Ey stacked. Only needed when `sx`, `sy`, `derivX`, or `derivY` are not supplied.

Value

The covariance matrix

See Also

corGaus, corMatrix

`buildWeightMat` *Build a weight matrix for bivariate Moran's I*

Description

Build a weight matrix to be used in the calculation of the bivariate Moran's I statistic, normalized to sum to one.

Usage

```
buildWeightMat(
  Cx,
  Ey,
  wo,
  eta,
  numNN,
  distMat = spatstat.geom::crossdist(Cx[, 1], Cx[, 2], Ey[, 1], Ey[, 2])
)
```

Arguments

`Cx, Ey`	Two coordinate matrices
`wo`	The weighting option, either "nn" or "Gauss"
`eta`	parameter that controls the decay of the weights with distance, see details
`numNN`	An integer, the number of neighbours
`distMat`	The distance matrix

Details

For wo = "Gauss", the weight decays as exp(-d^2/eta) with d the distance between observations. For wo = "nn" the numNN nearest neighbours are given equal weight, all others are zero.

Value

A weight matrix

CCT *An analytical p-value combination method using the Cauchy distribution.*

Description

The CCT function takes in a numeric vector of p-values, and returns the aggregated p-value using Cauchy combination rule. The code was taken from the xihaoli/STAAR github repo (Liu and Xie 2020), and adapted.

Usage

```
CCT(pvals)
```

Arguments

pvals	a numeric vector of p-values, where each of the elements is between 0 to 1, to be combined.

Value

The aggregated p-value

References

Liu Y, Xie J (2020). "Cauchy combination test: A powerful test with analytic p-value calculation under arbitrary dependency structures." *J. Am. Stat. Assoc.*, **115**(529), 393 - 402. doi:10.1080/01621459.2018.1554485.

---

checkControlVals *Check validity of control list*

Description

Check validity of control list

Usage

```
checkControlVals(method, controlvals, correlation, gpParams)
```

Arguments

method	A character string, indicating which method to apply
controlvals	The list of control values
correlation	Correlation structure, passed onto fitGP or fitGAM
gpParams	Pre-estimated GP parameters

Value

Finishes silently if no problems found

---

checkInputSingle *Check input of matrices and lists*

Description

Checks dimensions of matrices and lists, to be used for all exported functions

Usage

```
checkInputSingle(X, Y, Cx, Ey)
```

Arguments

X, Y	Matrices of omics measurements
Cx, Ey	Coordinate matrices of dimension two, belonging to X and Y respectively

Value

Throws error when a fault is found with the input, otherwise finishes silently

---

computeSigXws *Batch-compute lower triangles and traces of W_iˆTΣ_XW_i*

Description

For each weight-matrix slice W_i of the $n \times m \times numWs$ array W, constructs the spatial covariance matrix Σ_X from pre-evaluated lower-triangle covariance values and computes $W_i^T \Sigma_X W_i$ ($m \times m$), returning its lower triangle and trace.

Usage

```
computeSigXws(vgVals, W)
```

Arguments

vgVals	Covariance values for the strictly lower triangle of Σ_X, in column-major order (length $n(n-1)/2$). The diagonal is set to 1 (unit variance after standardisation).
W	$n \times m \times numWs$ array of weight matrices

Details

The lower-triangle iteration order (column-major) matches R's `which(lower.tri(diag(n)))` so that `vgVals` can be passed directly from `evalVariogramCpp(distX, ...)`, where `distX` is `as.vector(stats::dist(C`

Value

A list with

sigXws $mm2 \times numWs$ matrix, where $mm2 = m(m-1)/2$; column i holds the lower-triangle entries of $W_i^T \Sigma_X W_i$ in column-major order.

traces Length-numWs vector of $tr(W_i^T \Sigma_X W_i)$.

---

correlationsMulti *Find all cross-correlations for a list of matrices*

Description

Find all raw cross-correlations between lists of observations matrices from different modalities.

Usage

```
correlationsMulti(Xl, Yl, featuresX, featuresY, verbose)
```

Arguments

`Xl`, `Yl`	Lists of matrices of omics measurements
`featuresX`, `featuresY`	Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.
`verbose`	Should progress be printed?

Value

A list of named correlation vectors

---

evalVariogram *Evaluate a variogram on a set of distances*

Description

Evaluate a variogram on a set of distances

Usage

```
evalVariogram(vg, distVec)
```

Arguments

vg	The variogram model resulting from a call to fit.variogram)
distVec	A vector of pairwise distances

Value

A vector of covariances

---

evalVariogramCpp *Evaluate Exp or Lin variogram model on a vector of distances*

Description

Faster C++ replacement for the R evalVariogram wrapper. Implements the Exponential model (psill * exp(-d / range)) and the Linear model (psill * (1 - d / range) for d < range, else 0).

Usage

```
evalVariogramCpp(distVec, psill, range_, modelExp)
```

Arguments

distVec	Numeric vector of pairwise distances
psill	Partial sill of the structured spatial component
range_	Range parameter
modelExp	Logical: TRUE for Exponential, FALSE for Linear

Value

Numeric vector of covariance values, same length as distVec

---

exploreWeights *Visualize different weighting functions*

Description

The weighting functions for calculating bivariate Moran's I resulting from different choices of decay parameters eta are visualized in a lineplot.

Usage

```
exploreWeights(
  etas,
  dists = seq(0, 0.2, length.out = 1000),
  palette = "paired",
  legend.position = "topright"
)
```

Arguments

etas	A vector of positive decay parameters
dists	A set of distances, smaller than the square root of 2 since all coordinates are scaled to the unit square.
palette	The colour palette
legend.position	Legend position, passed onto legend as x

Value

Plots the weighting functions

Examples

```
exploreWeights(10^c(-5, -4, -3))
```

---

extractResultsMulti *Extract results from linear models*

Description

extractResultsMulti() adds the results as matrix to the object, including adjusted p-values, and sorted by p-value.

Given result estimated from replicated images using sbivarMulti, fit linear models to determine significance. To maintain interpretability of the intercept, continuous fixed effect variables are centered, and sum coding is used for the categorical ones.

Usage

```
extractResultsMulti(result, designDf, method = "BH")

## S4 method for signature 'sbivarResults'
extractResultsMulti(result, designDf, method = "BH")

fitLinModels(
  result,
  designDf,
  Formula,
  verbose = TRUE,
  inverseWeigh = FALSE,
  scaleByMax = TRUE,
 Control = lmerControl(check.conv.grad = .makeCC("ignore", tol = 0.002, relTol = NULL),
   check.conv.singular = .makeCC(action = "ignore", tol = 1e-04), check.conv.hess =
     .makeCC(action = "ignore", tol = 1e-06))
)

## S4 method for signature 'sbivarResults'
fitLinModels(
  result,
  designDf,
```

```
  Formula,
  verbose = TRUE,
  inverseWeigh = FALSE,
  scaleByMax = TRUE,
 Control = lmerControl(check.conv.grad = .makeCC("ignore", tol = 0.002, relTol = NULL),
   check.conv.singular = .makeCC(action = "ignore", tol = 1e-04), check.conv.hess =
     .makeCC(action = "ignore", tol = 1e-06))
)
```

Arguments

result	A result with the result of bivariate spatial association, from a call to the sbivar function with multiple images
designDf	A design dataframe
method	Multiplicity correction method passed onto p.adjust
Formula	A formula for the linear model to be fitted, can contain random effects.
verbose	Should a message with number of linear models and cores be printed?
inverseWeigh	A boolean, should estimates be inverse weighed by variance for GAMs and Moran's I?
scaleByMax	A boolean, should Moran's I be scaled by maximum values before plugging into the linear model?
Control	A control list for lmerTest::lmer

Details

The left hand side of "Formula" can be provided or not, but it will be overridden by "out" for downstream analysis

Value

The modified results object

For fitLinModels(), a list of linear models

See Also

lmer, lm, sbivarMulti, p.adjust

Examples

```
# Multi-image analysis on Vicari data, using GAMs
data(Vicari)
# Subset to 5 images and 500 spots limit computing time
VicariMultiTest <- lapply(Vicari, function(x) lapply(x[1:5], function(y) y[1:500, ]))
VicariRes <- sbivar(VicariMultiTest$TranscriptOutcomes,
    VicariMultiTest$MetaboliteOutcomes,
    VicariMultiTest$TranscriptCoords, VicariMultiTest$MetaboliteCoords,
    method = "GAMs", control = list(
        "families" =
            list("X" = mgcv::nb(), "Y" = Gamma(lin = "log"))
    )
)
mouse <- substr(names(Vicari$TranscriptOutcomes)[1:5], 1, 10)
```

```
designDf <- data.frame("mouse" = mouse) # The design matrix
multiGAMLmms <- fitLinModels(VicariRes, designDf, Formula = ~ (1 | mouse))
# Extract the results
resGAMsMulti <- extractResultsMulti(multiGAMLmms, designDf = designDf)
topPairs(resGAMsMulti)
```

`findDoubleUnderScore` *Is there any double underscore in the character vector?*

Description

Is there any double underscore in the character vector?

Usage

```
findDoubleUnderScore(charVec)
```

Arguments

`charVec`	The character vector

Value

A boolean

`fitGAM` *Fit a GAM model to a single variable*

Description

Fit a generalized additive model (GAM) captures spatial dependence through a bivariate spatial spline.

Usage

```
fitGAM(df, outcome, family = gaussian(), Gamm, correlation)
```

Arguments

`df`	The dataframe containing outcome and coordinates
`outcome`	A character vector indicating the outcome variable
`family`	A character string indicating the family, see family.
`Gamm`	A boolean, should a gamm with Gaussian covariance kernel be fitted? See gamm
`correlation`	Correlation structure, passed onto fitGP or fitGAM

Details

If a GAMM fit is attempted and fails, a GAM is tried instead. If a gamma fit is attempted and fails, which frequently happens for sparse data, a negative binomial fit is attempted instead.

Value

A fitted GAM or GAMM model, or try-error when the fit fails

See Also

gam, s

`fitGP` *Fit a Gaussian process (GP) to a single outcome vector*

Description

A Gaussian process (GP) models the spatial autocovariance using a kernel function that describes the covariance as a decreasing function of distance between observations.

Usage

```
fitGP(x, coord, GPmethod, correlation, optControl)
```

Arguments

`x`	outcome vector
`coord`	Coordinate matrix
`GPmethod`	The method by which to fit the Gaussian processes, passed onto gls as "method".
`correlation`	A corStruct object, see corStruct. At this point, corGaus is hard-coded,
`optControl`	List of control values for gls, see glsControl.

Value

A vector of length 4 with components range, nugget, sigma and mean

Note

Fitting GPs can be computation and memory intensive!

`fitLinModel` *Fit a linear model for an individual feature pair*

Description

Fit a linear model for an individual feature pair

Usage

```
fitLinModel(ff, y, Control, Terms, modMat, MM, Assign, weights = NULL)
```

Arguments

ff	The prepared frame
y	outcome vector
Control	A control list for lmerTest::lmer
modMat	Design matrix of the fixed effects model
MM	a Boolean, should a mixed model fit be attempted?
Assign, Terms	Added to fitted fixed effects model
weights	weights vector

Details

The code is based on fitSingleLmmModel, but may diverge

Value

A fitted model

A fitted lmer or lm model

---

fitManyGAMs *Fit GAMs to all columns of a dataframe, as a wrapper for fitGAM*

Description

Fit GAMs to all columns of a dataframe, as a wrapper for fitGAM

Usage

```
fitManyGAMs(
  mat,
  coord,
  family = gaussian(),
  modality,
  features,
  Gamm,
  correlation,
  pseudoCount = 1e-08,
  ...
)
```

Arguments

mat	The matrix of outcomes
coord	The coordinate matrix
family	A character string indicating the family, see family.
modality	Character vector indicating which modality is being fit. For debugging purposes mainly
features	Features to be fit, others are only used to estimate the offset

Gamm	A boolean, should a gamm with Gaussian covariance kernel be fitted? See gamm
correlation	Correlation structure, passed onto fitGP or fitGAM
pseudoCount	Pseudocount added to avoid zeroes for gamma distribution
...	Passed onto fitGAM

Value

A list of GAM models

`fitManyGPs` *A wrapper to fit GPs on all columns of a matrix*

Description

A wrapper to fit GPs on all columns of a matrix

Usage

```
fitManyGPs(mat, coord, features, ...)
```

Arguments

mat	Matrix of observations
coord	Matrix of coordinates
features	Features to be fit, others are only used to estimate the offset
...	passed onto fitGP

Value

Matrix of fitted GP components

`GAMsMulti` *Fit GAMs and find correlations and standard error for data lists*

Description

Wraps GAMsSingle for lists

Usage

```
GAMsMulti(
  Xl,
  Yl,
  Cxl,
  Eyl,
  control,
  verbose,
  featuresX,
  featuresY,
  findVariances = FALSE
)
```

Arguments

Xl, Yl	Lists of matrices of omics measurements
Cxl, Eyl	Lists of corresponding coordinate matrices of dimension two
control	Control arguments passed onto MoransISingle, fitGP or fitGAM, replacing defaults by MoransISingle, fitGP or fitGAM, respectively
verbose	Should info on type of analysis be printed?
featuresX, featuresY	Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.
findVariances	A boolean, should variances be found for bivariate Moran's I and GAMs

Value

A list named like Xl, containing all results

GAMsSingle *Fit univariate GAMs and test bivariate combinations*

Description

Fit univariate GAMs with 2D cubic smoothing splines for both modalities, and test for correlation between all bivariate combinations.

Usage

```
GAMsSingle(
  X,
  Y,
  Cx,
  Ey,
  families,
  n_points_grid,
  verbose,
  featuresX,
  featuresY,
  Gamm,
  correlation,
  findVariances = TRUE
)
```

Arguments

X, Y	Matrices of omics measurements
Cx, Ey	Coordinate matrices of dimension two, belonging to X and Y respectively
families	A vector of length 2 giving the distributional families for the outcome values. See details of sbivarSingle.
n_points_grid	The number of points in the new grid for the GAMs to be evaluated on.

verbose	Should info on type of analysis be printed?
featuresX, featuresY	Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.
Gamm	A boolean, should a gamm with Gaussian covariance kernel be fitted? See gamm
correlation	Correlation structure, passed onto fitGP or fitGAM
findVariances	Should variances be calculated? For internal use only

Value

A named list of results

`GaussKernel` *Construct the SAC part of the covariance matrix using the Gaussian covariance kernel,*

Description

Construct the SAC part of the covariance matrix using the Gaussian covariance kernel,

Usage

```
GaussKernel(distMat, range)
```

Arguments

distMat	The complete distance matrix of Cx and Ey stacked. Only needed when sx, sy, derivX, or derivY are not supplied.
range	Range (or length-scale) parameter of the Gaussian covariance kernel

Value

The SAC covariance matrix

`getApproxVar` *Approximate variance of the spatial covariance numerator*

Description

Computes $v^T \mathrm{Cov}(\hat{f})\, v$ using the factored form $(B^T v)^T C (B^T v)$, where B is the basis matrix and $C = \mathrm{Var}(\beta)$. This avoids materialising the $N_{\text{grid}} \times N_{\text{grid}}$ prediction covariance matrix.

Usage

```
getApproxVar(predInfo, cen, x, link)
```

Arguments

predInfo	List returned by vcovPredGam (must contain basis and coef_cov)
cen	Centered predictions of the *other* modality
x	Raw predictions (needed for non-identity links)
link	Link function name

Value

Scalar approximate variance

---

getDiscreteVars *Get all categrocial variables from a dataframe*

Description

Get all categrocial variables from a dataframe

Usage

```
getDiscreteVars(df)
```

Arguments

df	The data frane

Value

A character vector of variable names

---

getFeaturesList *Return feature names from a list*

Description

Return feature names from a list

Usage

```
getFeaturesList(Xl)
```

Arguments

Xl	A list of feature matrices

Value

A vector of feature names

getLengthScale *Derive a default length scale to start GP fitting*

Description

Derive a default length scale to start GP fitting

Usage

```
getLengthScale(Cx)
```

Arguments

Cx	The coordinate matrix

Value

90

getSize *Construct the size variable*

Description

Construct the size variable

Usage

```
getSize(X, Y, normX, normY, size, scaleBySampleSums)
```

Arguments

X, Y	Matrices of omics measurements
normX, normY	Character strings, indicating what normalization is required for X and Y matrices, respectively, before plotting, see details.
size	The maximum output size
scaleBySampleSums	boolean, should the point size be scaled by sample sums?

Value

The final point size

getSpatialCoords *Extract coordinate matrix*

Description

Extract coordinate matrix

Usage

```
getSpatialCoords(X, Cx)
```

Arguments

X	The matrix or SpatialExperiment object
Cx	The coordinate matrix

Value

A coordinate matrix

getX *Extract data matrix*

Description

Extract data matrix

Usage

```
getX(X, assay)
```

Arguments

X	The matrix or SpatialExperiment object
assay	The name of the assay

Value

A matrix

GPsSingle *Fit Gaussian processes (GPs) if needed, and perform score tests*

Description

Fit all univariate GPs on both modalities, and perform all bivariate tests across them

Usage

```
GPsSingle(
  X,
  Y,
  Cx,
  Ey,
  gpParams,
  numLscAlts,
  Quants,
  GPmethod,
  correlation,
  optControl,
  verbose,
  featuresX,
  featuresY
)
```

Arguments

X, Y	Matrices of omics measurements
Cx, Ey	Coordinate matrices of dimension two, belonging to X and Y respectively
gpParams	Parameters of the Gaussian processes, see details
numLscAlts	Number of length scales to be tested for bivariate association
Quants	Most extreme quantiles of the distance distribution to take as length scales
GPmethod	The method by which to fit the Gaussian processes, passed onto gls as "method".
correlation	Correlation structure, passed onto fitGP or fitGAM
optControl	List of control values for gls, see glsControl.
verbose	Should info on type of analysis be printed?
featuresX, featuresY	Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.

Details

gpParams must be a list of length 2 with names 'X' and 'Y', consisting of matrices with rownames "mean", "nugget", "range" and "sigma", and column names as in X and Y. This argument allows to pass parameters of the Gaussian processes estimated with other software (e.g. with GPU acceleration) to perform the score test.

Value

A named list of results

`makeNames` *Make unique names*

Description

Make unique names

Usage

```
makeNames(featX, featY)
```

Arguments

`featX`, `featY`	vectors of feature names

Value

A vector of names

`makeOffset` *Make a list of offsets*

Description

Make a list of offsets

Usage

```
makeOffset(X, family)
```

Arguments

`X`	The data matrix
`family`	The distribution family

Value

A list of length two with offsets

makePval — *Convert z-value to p-value*

Description

Convert z-value to p-value

Usage

```
makePval(z)
```

Arguments

z	The z-value to be converted

Value

The p-value

matheronVariograms — *Estimate variograms using Matheron's binning estimator for many features at once, and evaluate*

Description

Estimate variograms using Matheron's binning estimator for many features at once, and evaluate

Usage

```
matheronVariograms(X, Cx, width, cutoff, variogramModels)
```

Arguments

X	Outcome matrix
Cx	Coordinate matrix
cutoff, width	Cutoff and width of the variogram estimation, passed onto vgm
variogramModels	A character vector, indicating the variogram model passed onto vgm. Currently, only "Exp" and "Lin" are implemented for computational reasons.

Details

The best fitting variogram model, measured by the squared error, will be used.

Value

A list of evaluated variograms

ModTtestSingle *Perform modified t-tests for all pairs*

Description

The modified t-test is applied to all pairs, which tests for the significance of the Pearson correlation while accounting for spatial autocorrelation (Clifford et al. 1989).

Usage

```
ModTtestSingle(X, Y, Cx, verbose)
```

Arguments

X, Y	Matrices of omics measurements
Cx	The shared coordinate matrix
verbose	Should info on type of analysis be printed?

Value

A dataframe of results sorted by p-value, also containing effective sample size (ESS) and correlation estimate.

References

Clifford P, Richardson S, Hemon D (1989). "Assessing the Significance of the Correlation between Two Spatial Processes." *Biometrics*, **45**(1), 123 - 134. ISSN 0006341X, 15410420. doi:10.2307/2532039.

See Also

modified.ttest

moranControl *Create default argument lists*

Description

Define default argument lists for the MoransISingle(), GAMsSingle() and GPsingle() functions

Usage

```
moranControl(
  etas = c(5e-06, 2e-04, 0.02),
  findMaxW = FALSE,
  returnSEsMoransI = TRUE,
  variogramModels = c("Exp", "Lin"),
  width = cutoff/15,
  cutoff = sqrt(2)/3,
```

```
  wo = c("Gauss", "nn"),
  numNNs = c(4, 8, 24)
)

GAMcontrol(
  Gamm = FALSE,
  Cx,
  n_points_grid = 600,
  families = list(X = gaussian(), Y = gaussian())
)

GPcontrol(
  GPmethod = c("REML", "ML"),
  Cx,
  Quants = c(0.005, 0.5),
  numLscAlts = 5,
 optControl = lmeControl(opt = "optim", maxIter = 500, msMaxIter = 500, niterEM = 1000,
    msMaxEval = 1000)
)
```

Arguments

findMaxW	Is the maximum bivariate Moran's I needed?
returnSEsMoransI	A boolean, are standard errors of Moran's I to be returned?
variogramModels	A character vector, indicating the variogram model passed onto vgm. Currently, only "Exp" and "Lin" are implemented for computational reasons.
cutoff, width	Cutoff and width of the variogram estimation, passed onto vgm
wo	type of weight parameter, passed onto buildWeightMat
numNNs, etas	Vectors of weight matrix parameters, whose elements are passed onto buildWeightMat
Gamm	A boolean, should a gamm with Gaussian covariance kernel be fitted? See gamm
Cx	Coordinate matrix of the first modality
n_points_grid	The number of points in the new grid for the GAMs to be evaluated on.
families	A vector of length 2 giving the distributional families for the outcome values. See details of sbivarSingle.
GPmethod	The method by which to fit the Gaussian processes, passed onto gls as "method".
Quants	Most extreme quantiles of the distance distribution to take as length scales
numLscAlts	Number of length scales to be tested for bivariate association
optControl	List of control values for gls, see glsControl.

Value

A list of control elements

See Also

MoransISingle, fitGAM, fitGP

MoransIMulti *Find all Moran's I statistics for a list of matrices*

Description

Find all Moran's I values for lists of observations matrices from different modalities, by calling the MoransISingle function.

Usage

```
MoransIMulti(
  Xl,
  Yl,
  Cxl,
  Eyl,
  verbose,
  featuresX,
  featuresY,
  findVariances = FALSE,
  control,
  ...
)
```

Arguments

Xl, Yl	Lists of matrices of omics measurements
Cxl, Eyl	Lists of corresponding coordinate matrices of dimension two
verbose	Should info on type of analysis be printed?
featuresX, featuresY	Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.
findVariances	A boolean, should variances be found for bivariate Moran's I and GAMs
control	Control arguments passed onto MoransISingle, fitGP or fitGAM, replacing defaults by MoransISingle, fitGP or fitGAM, respectively
...	passed onto MoransISingle

Value

A list of Moran's I estimates, standard errors and maximum values

See Also

MoransISingle

`MoransISingle` *Calculate bivariate Moran's I between two modality matrix, with variance and p-value*

Description

The variance calculation requires estimation of the spatial autocorrelation structure of every feature separately, using Matheron's variogram estimator (Matheron 1963).

Usage

```
MoransISingle(
  X,
  Y,
  Cx,
  Ey,
  wo,
  etas,
  numNNs,
  cutoff,
  width,
  verbose,
  findMaxW,
  variogramModels,
  returnSEsMoransI,
  featuresX,
  featuresY,
  findVariances = TRUE,
  ...
)
```

Arguments

`X`, `Y`	Matrices of omics measurements
`Cx`, `Ey`	Coordinate matrices of dimension two, belonging to X and Y respectively
`wo`	type of weight parameter, passed onto buildWeightMat
`numNNs`, `etas`	Vectors of weight matrix parameters, whose elements are passed onto buildWeightMat
`cutoff`, `width`	Cutoff and width of the variogram estimation, passed onto vgm
`verbose`	Should info on type of analysis be printed?
`findMaxW`	Is the maximum bivariate Moran's I needed?
`variogramModels`	A character vector, indicating the variogram model passed onto vgm. Currently, only "Exp" and "Lin" are implemented for computational reasons.
`returnSEsMoransI`	A boolean, are standard errors of Moran's I to be returned?
`featuresX`, `featuresY`	Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.

`findVariances`	Should variances be calculated? For internal use only
`...`	passed onto variogram

Details

By default, a number of range parameters and corresponding weight matrices are screened for spatial association, and their p-value combined using the Cauchy combination rule by (Liu and Xie 2020). The maximum value of the bivariate Moran's I statistics are returned conditionally, as it is computation intensive and not always needed.

Value

A dataframe of results sorted by p-value, also containing the estimated Moran's I statistic and its variance. In addition, the maximum value of the Moran's I statistic, and the parameters of the weight matrix

Note

No multithreading is implemented for the variance calculation, as the matrix calculations involved may use inherent multithreading with OpenBLAS. Computations may require much RAM for building weight matrices.

References

Liu Y, Xie J (2020). "Cauchy combination test: A powerful test with analytic p-value calculation under arbitrary dependency structures." *J. Am. Stat. Assoc.*, **115**(529), 393 - 402. doi:10.1080/01621459.2018.1554485.

Matheron G (1963). "Principles of geostatistics." *Economic Geology*, **58**(8), 1246 - 1266. doi:10.2113/gsecongeo.58.8.1246.

`moveTwoCoords` *Move two sets of coordinates to 0-1. without shifting them with respect to each other*

Description

Move two sets of coordinates to 0-1. without shifting them with respect to each other

Usage

```
moveTwoCoords(Cx, Ey)
```

Arguments

`Cx, Ey`	Coordinate matrices

Value

A list with the shifted and scaled coordinates

normMat — *Normalize a data matrix, and ensure correct column names*

Description

Normalize to relative expression, and potentially add pseudocount and log-normalize.

Usage

```
normMat(x, norm, pseudoCount = 1e-08, warnDrop = TRUE)
```

Arguments

x	The matrix
norm	A character string, either "none", "log" or "rel"
pseudoCount	A pseudocount added prior to log-normalization to avoid taking the log of zero
warnDrop	Is a warning to be emitted when spots are dropped?

Details

norm = "none" is pass-through, norm = "rel" divides by sample sums, "log" adds a pseudocount, divides by sample sums and log-normalizes.

Value

A normalized matrix

Examples

```
mat <- matrix(rpois(2000, lambda = 3), 40, 50)
nMat <- normMat(mat, norm = "rel")
```

plotCoords — *Plot the coordinates of two omics modalities*

Description

Plot the coordinates of two modalities onto the same coordinate framework, in two different colours. This is a useful check of the alignment and overlap.

Usage

```
plotCoords(Cx, Ey, pchX = 1, pchY = 3, cex = 0.8, ...)

plotCoordsMulti(Cxl, Eyl, ...)
```

Arguments

Cx, Ey	Coordinate matrices of dimension two, belonging to X and Y respectively
pchX, pchY	Point shapes for x and y
cex	Expansion factor
...	passed onto plot()
Cxl, Eyl	Lists of corresponding coordinate matrices of dimension two

Details

plotCoordsMulti() is a wrapper for plotCoords for lists of coordinates, and requires the user to set par(mar =) appropriately, such that all plots are shown.

Value

Plots to the plotting device

Examples

```
data(Vicari)
plotCoords(Vicari$TranscriptCoords[[1]], Vicari$MetaboliteCoords[[1]])
# For multiple coordinates
par(mfrow = c(2, 3))
foo <- lapply(names(Vicari$TranscriptCoords), function(nam) {
   plotCoords(Vicari$TranscriptCoords[[nam]], Vicari$MetaboliteCoords[[nam]], main = nam)
})
par(mfrow = c(1, 1))
```

---

plotGAMs *Plot the fitted splines, and the correlation between them*

Description

Spawns a three-panel plot with the splines fitted for the two variables, plus a visualization of regions with positive and negative correlations between those splines. The splines are refitted using fitGAM, so no GAM-objects can be provided. This function takes entire outcome matrices X and Y as argument, to be able to account for offsets in refitting the GAMs. plotGAMsTopResults() plots the feature with the smallest p-value in the 'results' object.

Usage

```
plotGAMs(
  X,
  Y,
  Cx,
  Ey,
  features,
  scaleFun = scaleMinusOne,
  families = list(X = gaussian(), Y = gaussian()),
  addTitle = TRUE,
  normX = c("none", "rel", "log"),
```

```
  normY = c("none", "rel", "log"),
  pseudoCount = 1e-08,
  n_points_grid = 600,
  Gamm = FALSE,
  correlation = corGaus(form = ~x + y, nugget = TRUE, value = c(1, 0.25)),
  ...
)
```

Arguments

X, Y	Matrices of omics measurements, or lists thereof
Cx, Ey	Corresponding coordinate matrices of dimension two, or lists thereof
features	The features to plot
scaleFun	The scaling function to be applied before plotting
families	A vector of length 2 giving the distributional families for the outcome values. See details of sbivarSingle.
addTitle	A boolean, should a title be plotted
normX, normY, pseudoCount	Normalization parameters, passed onto normMat
n_points_grid	The number of points in the new grid for the GAMs to be evaluated on.
Gamm	A boolean, should a gamm with Gaussian covariance kernel be fitted? See gamm
correlation	Correlation structure, passed onto fitGP or fitGAM
...	passed onto fitGAM

Value

A ggplot object

Note

Both spline surfaces are scaled to the [-1,1] range, the same as the correlation has naturally, for legibility.

Examples

```
# Single image
example(sbivar, "sbivar")
plotGAMs(X, Y, Cx, Ey, features = c("X1", "Y2"))
plotGAMsTopPair(resGAMs, X, Y, Cx = Cx, Ey = Ey)
# Multi image, arbitrary pair
data(Vicari)
plotGAMs(Vicari$TranscriptOutcomes, Vicari$MetaboliteOutcomes,
    Vicari$TranscriptCoords, Vicari$MetaboliteCoords,
    features = c("Pcp4", "Dopamine")
)
```

plotGAMsTopPair *Plot GAMs of top features*

Description

Fit GAMS for the top feature pair chosen, and plot the resulting spline surfaces

Usage

```
plotGAMsTopPair(x, X, Y, Cx, Ey, topRank = 1, parameter = "Intercept", ...)

## S4 method for signature 'sbivarResultsGAMs'
plotGAMsTopPair(x, X, Y, Cx, Ey, topRank = 1, parameter = "Intercept", ...)
```

Arguments

x	a sbivarResultsGAMs object
X, Y	Matrices of omics measurements
Cx, Ey	Coordinate matrices of dimension two, belonging to X and Y respectively
topRank	An integer, the feature pair with the rank-th smallest p-value is plotted
parameter	The linear model parameter used to find the feature with the strongest effect. The default is the intercept, i.e. the overall effect.
...	passed onto lower level functions

Value

a ggplot2 object

Examples

```
example(sbivar, "sbivar")
plotGAMsTopPair(resGAMs, X, Y, Cx, Ey)
```

plotPairSingle *Plot a specified feature pair for the data provided*

Description

Plot a specified feature pair for the data provided

Usage

```
plotPairSingle(
  X,
  Y,
  Cx,
  Ey,
  features,
  normX = c("none", "rel", "log"),
  normY = c("none", "rel", "log"),
  assayX,
  assayY,
  scaleBySampleSums = FALSE,
  size = 0.02,
  ...
)

plotPairMulti(
  Xl,
  Yl,
  Cxl,
  Eyl,
  features,
  normX = c("none", "rel", "log"),
  scaleBySampleSums = FALSE,
  normY = c("none", "rel", "log"),
  size = 0.02,
  assayX,
  assayY,
  theme = theme_bw()
)

plotPairSingleVectors(
  x,
  y,
  Cx,
  Ey,
  size,
  modalityNames = c("Modality X", "Modality Y"),
  theme = theme_bw(),
  ...
)
```

Arguments

X	the input object, containing measurements of the first modality, see methods('sbivar')
Y	Matrix or SpatialExperiment object of second modality
Cx, Ey	Coordinate matrices of dimension two, belonging to X and Y respectively
features	Feature vector of length 2 to be plotted
normX, normY	Character strings, indicating what normalization is required for X and Y matrices, respectively, before plotting, see details.
assayX, assayY	Assay names to be used in the analysis, see assay

scaleBySampleSums	boolean, should the point size be scaled by sample sums?
size	Point size
...	additional constructor and analysis arguments
Xl, Yl	Lists of matrices of omics measurements
Cxl, Eyl	Lists of corresponding coordinate matrices of dimension two
theme	the ggplot2 theme
x, y	Outcome vectors
modalityNames	Names to be given to the modalities, appearing in the strip text of the columns. For plotTopPair() and plotPairSingle(), the feature names are used.

Value

a ggplot2 object

Examples

```
example(sbivar, "sbivar")
plotPairSingle(X, Y, Cx, Ey, features = c("X1", "Y1"))
```

---

plotSpotSums *Plot the spot-wise sums of the two modalities across the slices*

Description

This is an important exploratory plot to visualize spot-wise sums (library sizes, total ion counts etc.) spatially

Usage

```
plotSpotSums(
  Xl,
  Yl,
  Cxl,
  Eyl,
  viewNames = c("Transcriptomics", "Metabolomics"),
  pointSize = 0.01,
  stripTextSize = 8.5
)
```

Arguments

Xl, Yl	Lists of matrices of omics measurements
Cxl, Eyl	Lists of corresponding coordinate matrices of dimension two
viewNames	The names of the views, showed in the column names
pointSize	The point size for geom_point()
stripTextSize	The size of the text in the strips (grey panels)

Value

A ggplot2 object

Examples

```
data(Vicari)
plotSpotSums(
    Vicari$TranscriptOutcomes, Vicari$MetaboliteOutcomes,
    Vicari$TranscriptCoords, Vicari$MetaboliteCoords
)
```

`plotTopPair` *Plot a the top feature pair according to a sbivar analysis*

Description

Plot a chosen feature pair, or the highest ranking feature pair, for a single image or multiple images.

Usage

```
plotTopPair(
  x,
  ...,
  topRank = 1,
  parameter = "Intercept",
  scaleBySampleSums = FALSE,
  normX = x@normX,
  normY = x@normY
)

## S4 method for signature 'sbivarResults'
plotTopPair(
  x,
  ...,
  topRank = 1,
  parameter = "Intercept",
  scaleBySampleSums = FALSE,
  normX = x@normX,
  normY = x@normY
)
```

Arguments

`x`	The sbivarResults object
`...`	passed onto lower level functions
`topRank`	An integer, the feature pair with the rank-th smallest p-value is plotted
`parameter`	The linear model parameter used to find the feature with the strongest effect. The default is the intercept, i.e. the overall effect.

`scaleBySampleSums`
: A boolean, should the size of the spots be scaled by their sample sum, e.g. library size or total ion count? Recommended to reflect differences in certainty depending on sample sums.

`normX`, `normY`
: Character strings, indicating what normalization is required for X and Y matrices, respectively, before plotting, see details.

Details

For sequence count data, such as transcriptomics, normalization may be indicated to achieve clear plots (normX = "rel" or "log", see normMat). The normalization used for plotting is not necessarily the same as the one used for the analysis.

Value

A ggplot object

See Also

extractResultsMulti, fitLinModels

Examples

```
### Single image
# Single image analysis on synthetic data
n <- 8e1
m <- 9e1
p <- 3
k <- 2
X <- matrix(rnorm(n * p), n, p,
    dimnames =
        list(paste0("sampleX", seq_len(n)), paste0("X", seq_len(p)))
)
Y <- matrix(rnorm(m * k), m, k,
    dimnames =
        list(paste0("sampleY", seq_len(m)), paste0("Y", seq_len(k)))
)
Cx <- matrix(runif(n * 2), n, 2, dimnames = list(rownames(X), c("x", "y")))
Ey <- matrix(runif(m * 2), m, 2, dimnames = list(rownames(Y), c("x", "y")))
resMoransI <- sbivar(X, Y, Cx, Ey, method = "Moran's I")
# Plot the feature pair with the most significant signal
plotTopPair(resMoransI, X, Y, Cx, Ey)
# Plot an arbitrary feature pair
plotPairSingle(X, Y, Cx, Ey, features = c("X1", "Y1"))
### Multi image
data(Vicari)
# Plot an arbitrary feature pair
plotPairMulti(Vicari$TranscriptOutcomes, Vicari$MetaboliteOutcomes,
    Vicari$TranscriptCoords, Vicari$MetaboliteCoords,
    normX = "rel", normY = "rel", features = c("Gnas", "Tocopherol")
)
```

printIteration *Print a message for the current iteration*

Description

Print a message for the current iteration

Usage

```
printIteration(current, all)
```

Arguments

current	current state of the iterator
all	vector of all iterators

Value

prints message to output

printProgress *Print feature progress*

Description

Print feature progress

Usage

```
printProgress(feat, allFeats, verbose)
```

Arguments

feat	The current feature
allFeats	Vector of all features
verbose	Boolean, should output be printed?

Value

Prints progress message to output

replaceLhs *Replace the left hand side of a formula by a fixed string*

Description

Replace the left hand side of a formula by a fixed string

Usage

```
replaceLhs(x, repl = "out")
```

Arguments

x	a formula
repl	the replacement string

Value

A formula

sbivar *Spatial bivariate association analysis*

Description

Perform a bivariate spatial association analysis, either on a single or multiple images. Depending on the input, the workhorse functions sbivarSingle (single-image) or sbivarMulti (multi-image) are called.

Usage

```
sbivar(X, Y, ...)

## S4 method for signature 'matrix,matrix'
sbivar(X, Y, Cx, Ey, ...)

## S4 method for signature 'list,list'
sbivar(X, Y, Cx, Ey, assayX = NULL, assayY = NULL, ...)

## S4 method for signature 'SpatialExperiment,SpatialExperiment'
sbivar(
  X,
  Y,
  assayX,
  assayY,
  sample_id_x = "sample_id",
  sample_id_y = sample_id_x,
  ...
)
```

```
## S4 method for signature 'MultiAssayExperiment,ANY'
sbivar(X, experimentX, experimentY, assayX, assayY, ...)
```

Arguments

X	the input object, containing measurements of the first modality, see methods('sbivar')
Y	Matrix or SpatialExperiment object of second modality
...	additional constructor and analysis arguments
Cx, Ey	Coordinate matrices of dimension two, belonging to X and Y respectively
assayX, assayY	Assay names to be used in the analysis, see assay
sample_id_x, sample_id_y	If provided, these are used to discriminate between different images included in the same SpatialExperiment objects X and Y. By default, they are assumed to be the same for both X and Y.
experimentX, experimentY	Names of the experiments in X and Y to be used in the analysis

Value

A list containing the analysis result, along with parameters used in the analysis

Note

Feature names are converted using the make.names() function

See Also

sbivarSingle, sbivarMulti

Examples

```
# Single image analysis on synthetic data
n <- 8e1
m <- 9e1
p <- 3
k <- 2
X <- matrix(rnorm(n * p), n, p,
    dimnames =
        list(paste0("sampleX", seq_len(n)), paste0("X", seq_len(p)))
)
Y <- matrix(rnorm(m * k), m, k,
    dimnames =
        list(paste0("sampleY", seq_len(m)), paste0("Y", seq_len(k)))
)
Cx <- matrix(runif(n * 2), n, 2, dimnames = list(rownames(X), c("x", "y")))
Ey <- matrix(runif(m * 2), m, 2, dimnames = list(rownames(Y), c("x", "y")))
resMoransI <- sbivar(X, Y, Cx, Ey, method = "Moran's I")
resGAMs <- sbivar(X, Y, Cx, Ey, method = "GAMs")
Y2 <- matrix(rnorm(n * k), n, k,
    dimnames =
        list(paste0("sampleX", seq_len(n)), paste0("Y", seq_len(k)))
)
resModtTestJoint <- sbivar(X, Y2, Cx, method = "Modified t-test")
```

```
# Single image analysis on synthetic data, converted to SpatialExperiment
library(SpatialExperiment)
seX <- SpatialExperiment(
    assays = list("transcripts" = t(X)),
    spatialCoords = Cx
)
seY <- SpatialExperiment(
    assays = list("metabolites" = t(Y)),
    spatialCoords = Ey
)
resModtGPs <- sbivar(seX, seY,
    assayX = "transcripts", assayY = "metabolites",
    method = "GPs"
)
```

`sbivarMulti` *Estimate measures of bivariate spatial association for multiple images*

Description

This function calculates measures of spatial association for every image. The resulting estimates can then be analysed further using the fitLinModels function.

Usage

```
sbivarMulti(
  Xl,
  Yl,
  Cxl,
  Eyl,
  method = c("Moran's I", "GAMs", "Correlation"),
  control,
  featuresX = getFeaturesList(Xl),
  featuresY = getFeaturesList(Yl),
  normX = c("none", "rel", "log"),
  normY = c("none", "rel", "log"),
  pseudoCount = 1e-08,
  verbose = TRUE,
  findVariances = FALSE,
  findMaxW = TRUE
)
```

Arguments

`Xl`, `Yl`	Lists of matrices of omics measurements
`Cxl`, `Eyl`	Lists of corresponding coordinate matrices of dimension two
`method`	A character string, indicating which method to apply
`control`	Control arguments passed onto MoransISingle, fitGP or fitGAM, replacing defaults by MoransISingle, fitGP or fitGAM, respectively

`featuresX`, `featuresY`
: Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.

`normX`, `normY`, `pseudoCount`
: Normalization parameters, passed onto normMat

`verbose`
: Should info on type of analysis be printed?

`findVariances`
: A boolean, should variances be found for bivariate Moran's I and GAMs

`findMaxW`
: A boolean, should maximum values be found for bivariate Moran's I

Value

A sbivarResults-class object with multi = TRUE

Note

All methods use multithreading on the cluster provided using the BiocParallel package

See Also

fitLinModels, MoransIMulti, correlationsMulti, GAMsMulti

`sbivarResults-class` *sbivarResults class*

Description

An S4 class representing the result of a sbivar call, testing for bivariate spatial association in one or more images

Slots

`method` The statistical test used

`normX` Normalization applied to the first modality

`normY` Normalization applied to the second modality

`multi` A boolean, were multiple images tested?

`assayX` Name of the assay of the first modality

`assayY` Name of the assay of the second modality

`models` Optional, the linear model fits if multi=TRUE

`result` The result of the statistical test: a dataframe

`estimates` Raw estimates of association measures, for input to linear model fitting

`sbivarSingle` *Test for bivariate spatial association in a single image*

Description

The tests can be applied to either joint or disjoint coordinate sets.

Usage

```
sbivarSingle(
  X,
  Y,
  Cx,
  Ey,
  method = c("Moran's I", "GAMs", "Modified t-test", "GPs"),
  normX = c("none", "rel", "log"),
  normY = c("none", "rel", "log"),
  pseudoCount = 1e-08,
  correlation = corGaus(form = ~x + y, nugget = TRUE, value = c(getLengthScale(Cx),
    0.25)),
  control,
  gpParams,
  featuresX = colnames(X),
  featuresY = colnames(Y),
  verbose = TRUE
)
```

Arguments

`X`, `Y`	Matrices of omics measurements
`Cx`, `Ey`	Coordinate matrices of dimension two, belonging to X and Y respectively
`method`	A character string, indicating which method to apply
`normX`, `normY`, `pseudoCount`	Normalization parameters, passed onto normMat
`correlation`	Correlation structure, passed onto fitGP or fitGAM
`control`	Control arguments passed onto MoransISingle, fitGP or fitGAM, replacing defaults by MoransISingle, fitGP or fitGAM, respectively
`gpParams`	Pre-estimated GP parameters
`featuresX`, `featuresY`	Features to be tested. Defaults to all features, but specifying them allows to test a limited feature set, while using the whole matrix to calculate library sizes as offset or for normalization.
`verbose`	Should info on type of analysis be printed?

Details

If only Cx is supplied and X and Y have the same number of rows, a joint analysis is performed If Cx and Ey are provided, and X and Y have the same number of rows, equality of Cx and Ey is checked. If true, a joint analysis is run, with a warning.

X and Y need to have rownames for matching to the coordinates, and column names for identifying the features. Cx and Ey must have rownames matching those in X and Y, and have two columns. For GAMs, usually no normalization is needed, as the non-gaussianity is taken care of by the outcome distribution, offset and link functions. Currently, identity, inverse and log-link are implemented.

Value

A sbivarResults-class object with multi = FALSE

Note

All methods use multithreading on the cluster provided using the BiocParallel package.

See Also

MoransISingle, ModTtestSingle, GAMsSingle, GPsSingle

`scaleHelpFun` *Wrapper to normalize, select feature and scale*

Description

Wrapper to normalize, select feature and scale

Usage

```
scaleHelpFun(X, feat)
```

Arguments

`X`	data matrix
`feat`	the feature name

Details

Returns vector of NA if feature not found, leading to grey in the plots

Value

A vector of values

scaleMinusOne *Scale to [-1,1] range*

Description

Scale to [-1,1] range

Usage

```
scaleMinusOne(y, na.rm = TRUE)
```

Arguments

y	The vector to be scaled
na.rm	passed onto min and range

Value

The scaled vector

scaleZeroOne *Scale to [0,1] range*

Description

Scale to [0,1] range

Usage

```
scaleZeroOne(y, na.rm = TRUE)
```

Arguments

y	The vector to be scaled
na.rm	passed onto min and range

Value

The scaled vector

scoreTestInternals_cpp

Compute Itautau, Itautheta and score statistics for the GP score test

Description

For each length-scale slice l, exploits the block structure of $\Sigma_{alt,l} = I + \begin{bmatrix} 0 & C_l \\ C_l^T & 0 \end{bmatrix}$ to compute $P\Sigma_{alt,l}$ block-wise without materialising the full $(n+m)^2 \times L$ intermediate array.

Usage

```
scoreTestInternals_cpp(P, crossBlocks, derivX, derivY, vecPos)
```

Arguments

`P`	$(n+m) \times (n+m)$ projection matrix
`crossBlocks`	$n \times m \times L$ array of cross-blocks C_l (the off-diagonal blocks of each $\Sigma_{alt,l}$)
`derivX`	$n \times n \times 3$ covariance-parameter derivative arrays for X
`derivY`	$m \times m \times 3$ covariance-parameter derivative arrays for Y
`vecPos`	$(n+m)$ score vector ($\Omega^{-1}(z-\mu)$ for the positive direction)

Value

A list with

Itautau Length-L vector of $0.5 \left\| P\Sigma_{alt,l} \right\|_F^2$

Itautheta $L \times 6$ matrix of trace cross-products with derivative matrices

UPos Length-L vector of score statistics for the positive direction

UNeg Length-L vector of score statistics for the negative direction

selfName

Name a character vector after itself

Description

Name a character vector after itself

Usage

```
selfName(x)
```

Arguments

`x`	The vector to be named

Value

The named vector

Examples

```
selfName(LETTERS[1:5])
```

`show,sbivarResults-method`

Show basic properties of a sbivarResults object

Description

Show basic properties of a sbivarResults object

Usage

```
## S4 method for signature 'sbivarResults'
show(object)
```

Arguments

`object`	The sbivarResults object

Value

Prints a description to console

`splitSpatialExperiment`

Split a SpatialExperiment object into images

Description

Split Spatial Experiment into a list of SpatialExperiment objects, based on a variable present in the colData slot

Usage

```
splitSpatialExperiment(spe, sample_id)
```

Arguments

`spe`	The SpatialExperiment object
`sample_id`	A character vector

Value

A list of SpatialExperiment objects

strwrapPaste *Collapse a set of character vectors, and wrap the string*

Description

Collapse a set of character vectors, and wrap the string

Usage

```
strwrapPaste(...)
```

Arguments

...	A number of character vectors

Value

Formats the character vector

sund *Split a string*

Description

Split a string

Usage

```
sund(string, split = "__")
```

Arguments

string	The string
split	string to split by

Value

A character vector of length 2

testGAM — *Test for correlation between the predictions of two GAM models*

Description

The variance of the correlation between the spline surfaces is found by propagating the uncertainties on the spline parameters through uncertainty on the spline predictions to uncertainty on the correlation estimate.

Usage

```
testGAM(modelx, modely, predx, predy, findVariances)
```

Arguments

modelx, modely	Two fitted GAMs
predx, predy	Predictions and covariance matrices of fitted GAMs in common grid
findVariances	Should variances be calculated? For internal use only

Value

A vector with the correlation estimate, its standard error and the p-value

testGP — *Perform a score test on the bivariate spatial association in a Gaussian process.*

Description

This function tests for the variance of a random effect, causing the covariance, to be zero. It is a score test as developed by (Zhang and Lin 2003), with the test statistic having a scaled chi-square distribution.

Usage

```
testGP(x, y, crossBlocks, solXonly, solYonly, sx, sy, derivX, derivY, distMat)
```

Arguments

x, y	outcome vectors
crossBlocks	An $n \times m \times L$ array of cross-blocks C_l from buildAltSigmas
solXonly, solYonly	Parameters of the Gaussian processes of x and y
sx, sy	Inverses of the covariance matrices of x and y respectively. Computed from distMat and solXonly/solYonly when missing.
derivX, derivY	$n \times n \times 3$ / $m \times m \times 3$ arrays of covariance-parameter derivative matrices. Computed when missing.
distMat	The complete distance matrix of Cx and Ey stacked. Only needed when sx, sy, derivX, or derivY are not supplied.

Details

Two tests are performed, one for positive and one for negative association, and two times the smallest p-value to achieve a two-sided test. The sign indicates which direction was most significant.

Value

A vector of length 2: a p-value and an indicator of the sign: +1 for positive association, -1 for negative

References

Zhang D, Lin X (2003). "Hypothesis testing in semiparametric additive mixed models." *Biostatistics*, **4**(1), 57 - 74.

---

topPairs *Return the top results found by sbivar*

Description

Return the top results found by sbivar

Usage

```
topPairs(x, ...)

## S4 method for signature 'sbivarResults'
topPairs(x, parameter = "Intercept", digits = 4L, ...)
```

Arguments

x	The sbivarResults object
...	passed onto head
parameter	For multi-image analysis, the parameter for which top results should be shown
digits	The number of significant digits to be rounded to

Value

A dataframe of top results

Examples

```
example(sbivar, "sbivar")
topPairs(resGAMs)
```

`vcovPredGam` *Return predictions of a GAM, along with the factored coefficient covariance*

Description

Spline predictions are $B\beta$, where B is the basis (design) matrix and β are the spline coefficients. Rather than materialising the full $N_{\text{grid}} \times N_{\text{grid}}$ prediction covariance $B\,\text{Var}(\beta)\,B^T$, only B ($N_{\text{grid}} \times q$) and $\text{Var}(\beta)$ ($q \times q$) are returned. The quadratic form $v^T B\,\text{Var}(\beta) B^T v = (B^T v)^T \text{Var}(\beta)(B^T v)$ is then computed cheaply in q dimensions by getApproxVar.

Usage

```
vcovPredGam(model, newdata, findVariances = TRUE)
```

Arguments

`model`	The fitted GAM
`newdata`	The grid on which predictions are made
`findVariances`	Should variances be calculated? For internal use only

Value

A list with components

`pred`	A vector of predictions on the response scale
`basis`	$N_{\text{grid}} \times q$ basis matrix for the smooth of interest
`coef_cov`	$q \times q$ unconditional covariance matrix of the smooth coefficients

See Also

vcov.gam, predict.gam

`Vicari` *Spatial transcriptomics and metabolomics data of mouse brain*

Description

Spatial transcriptomics and metabolomics data measured on the same tissue sections of mouse brains on a regular grid by Vicari et al. 2024. Only a subset of the data, consisting of the 5 most abundant transcripts and metabolites for 6 samples, are included in the package for computational and memory reasons. The images were pre-aligned manually with the help of MAGPIE (Williams et al. 2026). The data consist of two lists of outcome variables and their coordinates.

Usage

```
data(Vicari)
```

Format

Four lists of data matrices:

TranscriptCoords,MetaboliteCoords Coordinate lists

TranscriptOutcomes,MetaboliteOutcomes Outcome matrices

Source

doi:10.1038/s4158702301937y

References

Vicari M, Mirzazadeh R, Nilsson A, Shariatgorji R, Bjärterot P, Larsson L, Lee H, Nilsson M, Foyer J, Ekvall M, Czarnewski P, Zhang X, Svenningsson P, Käll L, Andrén PE, Lundeberg J (2024). "Spatial multimodal analysis of transcriptomes and metabolomes in tissues." *Nat. Biotechnol.*, **42**(7), 1046 - 1050. doi:10.1038/s4158702301937y.

Williams EC, Franzén L, Lindvall MO, Hamm G, Oag S, Majumder MM, Denholm J, Hamidinekoo A, Morlanes JE, Vicari M, Lundeberg J, Setyo L, Godfrey TM, Eberlin LS, Zakirov A, Hornberg JJ, Stamou M, Ståhl PL, Ollerstam A, Tan JY, Mohorianu I (2026). "Spatially resolved integrative analysis of transcriptomic and metabolomic changes in tissue injury studies." *Nat. Commun.*, **17**(1), 205. doi:10.1038/s4146702568003w.

`writeSbivarToXlsx` *Write* sbivar *results to an excel worksheet*

Description

The results of single- or multi-image analysis are written to an excel spreadsheet with separate tabs per parameter tested, sorted by increasing p-value.

Usage

```
writeSbivarToXlsx(x, file, overwrite = FALSE, digits = 3, sigLevel = 0.05)

## S4 method for signature 'sbivarResults'
writeSbivarToXlsx(x, file, overwrite = FALSE, digits = 3, sigLevel = 0.05)
```

Arguments

`x`	The analysis results, from a call to sbivar (single-image) or extractResultsMulti (multi-image)
`file`	The file to write the results to
`overwrite`	A boolean, should the file be overwritten if it exists already?
`digits`	An integer, the number of significant digits to retain for the effect size, raw and adjusted p-values
`sigLevel`	The significance level threshold to use for the adjusted p-values, only features exceeding the threshold are written to the file. Set this parameter to 1 to write all features

Details

If no feature exceeds the significance threshold for a certain parameter, an empty tab is created. For each fixed effect, a single tab is written. The "baseline" tabs indicate the overall patterns, the other tabs are named after the fixed effects and indicate departure from this baseline for this fixed effect.

Value

Returns invisible with a message when writing operation successful, otherwise throws a warning.

See Also

createWorkbook

Examples

```
example(sbivar, "sbivar")
# The significance level is set to 1 here for illustration,
# meaning that all feature pairs will be written to the spreadsheet.
# Single result
writeSbivarToXlsx(resGAMs,
    file = tmpFile <- tempfile(fileext = ".xlsx"),
    sigLevel = 1
)
file.exists(tmpFile)
```

Index

∗ **datasets**
Vicari, 54
∗ **internal**
computeSigXws, 11
evalVariogramCpp, 13
scoreTestInternals_cpp, 49

addFeatureColumn, 3
arrayDeriv, 4
arrayMatProd, 4
arrayProd2tr, 5
arrayProdTr, 5
assay, *37*, *43*
assayT, 6

bdiag, *6*
bdiagn, 6
buildAltSigmas, 7, *52*
buildDerivArray, 7
buildNewGrid, 8
buildSigmaGp, 9
buildWeightMat, 9, *29*, *31*

CCT, 10
checkControlVals, 10
checkInputSingle, 11
computeSigXws, 11
concaveman, *8*
corGaus, *9*, *17*
corMatrix, *9*
correlationsMulti, 12, *45*
corStruct, *17*
createWorkbook, *56*

evalVariogram, 12
evalVariogramCpp, 13
exploreWeights, 13
extractResultsMulti, 14, *40*, *55*
extractResultsMulti,sbivarResults-method *(*extractResultsMulti*)*, 14

family, *16*, *18*
findDoubleUnderScore, 16
fit.variogram, *13*
fitGAM, *11*, *16*, 16, *19–21*, *25*, *29*, *30*, *34*, *35*, *44*, *46*
fitGP, *11*, *16*, 17, *19–21*, *25*, *29*, *30*, *35*, *44*, *46*
fitLinModel, 17
fitLinModels, *40*, *44*, *45*
fitLinModels *(*extractResultsMulti*)*, 14
fitLinModels,sbivarResults-method *(*extractResultsMulti*)*, 14
fitManyGAMs, 18
fitManyGPs, 19
fitSingleLmmModel, *18*

gam, *17*
GAMcontrol *(*moranControl*)*, 28
gamm, *16*, *19*, *21*, *29*, *35*
GAMsMulti, 19, *45*
GAMsSingle, *19*, 20, *47*
GaussKernel, 21
getApproxVar, 21, *54*
getDiscreteVars, 22
getFeaturesList, 22
getLengthScale, 23
getSize, 23
getSpatialCoords, 24
getX, 24
gls, *17*, *25*, *29*
glsControl, *17*, *25*, *29*
GPcontrol *(*moranControl*)*, 28
GPsSingle, 25, *47*

head, *53*

legend, *14*
lm, *15*
lmer, *15*

makeNames, 26
makeOffset, 26
makePval, 27
matheronVariograms, 27
min, *48*
modified.ttest, *28*
ModTtestSingle, 28, *47*
moranControl, 28

MoransIMulti, 30, *45*
MoransISingle, *20*, *29*, *30*, 31, *44*, *46*, *47*
moveTwoCoords, 32

normMat, 33, *35*, *40*, *45*, *46*

p.adjust, *15*
plotCoords, 33
plotCoordsMulti *(*plotCoords*)*, 33
plotGAMs, 34
plotGAMsTopPair, 36
plotGAMsTopPair,sbivarResultsGAMs-method *(*plotGAMsTopPair*)*, 36
plotPairMulti *(*plotPairSingle*)*, 36
plotPairSingle, 36
plotPairSingleVectors *(*plotPairSingle*)*, 36
plotSpotSums, 38
plotTopPair, 39
plotTopPair,sbivarResults-method *(*plotTopPair*)*, 39
predict.gam, *54*
printIteration, 41
printProgress, 41

range, *48*
replaceLhs, 42

s, *17*
sbivar, *15*, 42, *55*
sbivar,list,list-method *(*sbivar*)*, 42
sbivar,matrix,matrix-method *(*sbivar*)*, 42
sbivar,MultiAssayExperiment,ANY-method *(*sbivar*)*, 42
sbivar,SpatialExperiment,SpatialExperiment-method *(*sbivar*)*, 42
sbivarMulti, *14*, *15*, *42*, *43*, 44
sbivarResults-class, *45*, 45, *47*
sbivarSingle, *20*, *29*, *35*, *42*, *43*, 46
scaleHelpFun, 47
scaleMinusOne, 48
scaleZeroOne, 48
scoreTestInternals_cpp, 49
selfName, 49
show,sbivarResults-method, 50
splitSpatialExperiment, 50
strwrapPaste, 51
sund, 51

testGAM, 52
testGP, 52
topPairs, 53
topPairs,sbivarResults-method *(*topPairs*)*, 53

variogram, *32*
vcov.gam, *54*
vcovPredGam, *22*, 54
vgm, *27*, *29*, *31*
Vicari, 54

writeSbivarToXlsx, 55
writeSbivarToXlsx,sbivarResults-method *(*writeSbivarToXlsx*)*, 55