

# Package ‘divergence’

July 14, 2025

**Title** Divergence: Functionality for assessing omics data by divergence with respect to a baseline

**Version** 1.25.0

**Author** Wikum Dinalankara <wdd4001@med.cornell.edu>, Luigi Marchionni <marchion@jhu.edu>, Qian Ke <qke1@jhu.edu>

**Maintainer**

Wikum Dinalankara <wdd4001@med.cornell.edu>, Luigi Marchionni <marchion@jhu.edu>

**Description** This package provides functionality for performing divergence analysis as presented in Dinalankara et al, ``Digitizing omics profiles by divergence from a baseline'', PANS 2018. This allows the user to simplify high dimensional omics data into a binary or ternary format which encapsulates how the data is divergent from a specified baseline group with the same univariate or multivariate features.

**Depends** R (>= 3.6), SummarizedExperiment

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.1.1

**biocViews** Software, StatisticalMethod

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr

**git\_url** <https://git.bioconductor.org/packages/divergence>

**git\_branch** devel

**git\_last\_commit** d8c2152

**git\_last\_commit\_date** 2025-04-15

**Repository** Bioconductor 3.22

**Date/Publication** 2025-07-14

## Contents

breastTCGA_ER . . . . .	2
breastTCGA_Group . . . . .	2
breastTCGA_Mat . . . . .	3
computeChiSquaredTest . . . . .	3
computeMultivariateBinaryMatrix . . . . .	4

computeMultivariateDigitization . . . . .	5
computeMultivariateSupport . . . . .	6
computeQuantileMatrix . . . . .	7
computeUnivariateDigitization . . . . .	8
computeUnivariateSupport . . . . .	9
computeUnivariateTernaryMatrix . . . . .	10
findMultivariateGammaWithSupport . . . . .	11
findUnivariateGammaWithSupport . . . . .	12
msigdb_Hallmarks . . . . .	13

<b>Index</b>	<b>14</b>
--------------	-----------

---

breastTCGA_ER	<i>ER positive or negative status of breast tumor samples</i>
---------------	---

---

### Description

A factor indicating whether 887 breast samples in breastTCGA\_Mat are ER positive or ER negative. The matched normals have empty values.

### Usage

```
breastTCGA_ER
```

### Format

A Factor of length 887 of levels Negative and Positive (with 111 missing values for the normals).

### Source

<https://cancergenome.nih.gov/>

---

breastTCGA_Group	<i>Normal or Tumor status of breast samples</i>
------------------	---

---

### Description

A factor indicating whether 887 breast samples in breastTCGA\_Mat are tumor or matched normal.

### Usage

```
breastTCGA_Group
```

### Format

A Factor of length 887 of levels NORMAL and TUMOR.

### Source

<https://cancergenome.nih.gov/>

---

breastTCGA_Mat	<i>Gene expression for 260 genes in 887 breast samples</i>
----------------	--

---

**Description**

A data matrix containing a subset of the TCGA breast cancer dataset, with the gene level expression estimates in log2 transcripts per million for 887 breast samples.

**Usage**

```
breastTCGA_Mat
```

**Format**

A data matrix with 260 rows and 887 columns.

**Source**

<https://cancergenome.nih.gov/>

---

computeChiSquaredTest	<i>Compute chi-squared test</i>
-----------------------	---------------------------------

---

**Description**

Given a binary or ternary data matrix with class associations of samples, computes chi-squared tests for each feature between given groups

**Usage**

```
computeChiSquaredTest(Mat, Groups, classes)
```

**Arguments**

Mat	Matrix of digitized binary or ternary data with each column corresponding to a sample and each row corresponding to a feature
Groups	Factor indicating class association of samples
classes	Vector of class labels; the test will be applied between the classes given.

**Value**

A data frame with columns 'statistic' and 'pval'.

### Examples

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
dataMat = breastTCGA_Mat[, breastTCGA_Group != "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
seMat = SummarizedExperiment(assays=list(data=dataMat))
div = computeUnivariateDigitization(
  seMat = seMat,
  seMat.base = seMat.base,
  parallel = TRUE
)
assays(seMat)$div = div$Mat.div
sel = which(colnames(seMat) %in% colnames(dataMat))
div.chi = computeChiSquaredTest(Mat=assays(seMat)$div,
                                Groups=breastTCGA_ER[sel],
                                classes=c("Positive", "Negative"))
```

---

computeMultivariateBinaryMatrix

*Compute the binary matrix with digitized divergence coding*

---

### Description

Function for obtaining the binary form for a matrix for multivariate divergence of data given a baseline range

### Usage

```
computeMultivariateBinaryMatrix(seMat, Baseline)
```

### Arguments

seMat	SummarizedExperiment with assay to be digitized, in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
Baseline	A Baseline object; this corresponds to the output of findMultivariateGammaW-ithSupport() or computeMultivariateSupport()

### Value

A matrix with the same columns as Mat, with rows being the multivariate features, containing the binary form data.

### Examples

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
assays(seMat.base)$quantile = computeQuantileMatrix(seMat.base)
baseline = computeMultivariateSupport(seMat=seMat.base, FeatureSets=msigdb_Hallmarks)
dataMat = breastTCGA_Mat[, breastTCGA_Group != "NORMAL"]
seMat = SummarizedExperiment(assays=list(data=dataMat))
assays(seMat)$quantile = computeQuantileMatrix(seMat)
```

```
Mat.div = computeMultivariateBinaryMatrix(seMat=seMat, Baseline=baseline)
```

---

```
computeMultivariateDigitization
```

*Perform binary digitization*

---

## Description

Function for obtaining the digitized form, along with other relevant statistics and measures given a data matrix and a baseline matrix with multivariate features of interest

## Usage

```
computeMultivariateDigitization(seMat, seMat.base, FeatureSets,
  computeQuantiles = TRUE, gamma = c(1:9/100, 1:9/10), beta = 0.95,
  alpha = 0.01, distance = "euclidean", verbose = TRUE,
  findGamma = TRUE, Groups = NULL, classes = NULL)
```

## Arguments

seMat	SummarizedExperiment with assay to be digitized, in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
seMat.base	SummarizedExperiment with baseline assay in [0, 1], with each column corresponding to a sample and each row corresponding to a feature
FeatureSets	The multivariate features in list or matrix form. In list form, each list element should be a vector of individual features; in matrix form, it should be a binary matrix with rownames being individual features and column names being the names of the feature sets.
computeQuantiles	Apply quantile transformation to both data and baseline matrices (TRUE or FALSE; defaults to TRUE).
gamma	Range of gamma values to search through. By default gamma = 0.01, 0.02, ... 0.09, 0.1, 0.2, ..., 0.9.
beta	Parameter for eliminating outliers ( $0 < \beta \leq 1$ ). By default beta=0.95.
alpha	Expected proportion of divergent features per sample to be estimated. The optimal gamma providing this level of divergence in the baseline data will be searched for.
distance	Type of distance to be calculated between points. Any type of distance that can be passed on to the dist function can be used (default 'euclidean').
verbose	Logical indicating whether to print status related messages during computation (defaults to TRUE).
findGamma	Logical indicating whether to search for optimal gamma values through the given gamma values (defaults to TRUE). If FALSE, the first value given in gamma will be used.
Groups	Factor indicating class association of samples
classes	Vector of class labels

**Value**

A list with elements: Mat.div: divergence coding of data matrix in binary form, of same dimensions at seMat baseMat.div: divergence coding of base matrix in binary form, of same column names at seMat.base, rows being multivariate features. div: data frame with the number of divergent features in each sample features.div: data frame with the divergent probability of each feature; divergence probability for each phenotype is included as well if 'Groups' and 'classes' inputs were provided. Baseline: a list containing a "Ranges" data frame with the baseline interval for each feature, and a "Support" binary matrix of the same dimensions as Mat indicating whether each sample was a support or a feature or not (1=support, 0=not in the support), gamma: selected gamma value alpha: the expected number of divergent features per sample computed over the baseline data matrix

**Examples**

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
dataMat = breastTCGA_Mat[, breastTCGA_Group != "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
seMat = SummarizedExperiment(assays=list(data=dataMat))
div = computeMultivariateDigitization(
  seMat = seMat,
  seMat.base = seMat.base,
  FeatureSets = msigdb_Hallmarks
)
```

---

```
computeMultivariateSupport
```

*Estimate the baseline support*

---

**Description**

Function for computing the baseline support for multivariate features given gamma and beta parameters.

**Usage**

```
computeMultivariateSupport(seMat, FeatureSets, gamma = 0.1,
  beta = 0.95, distance = "euclidean", verbose = TRUE)
```

**Arguments**

seMat	SummarizedExperiment with an assay in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
FeatureSets	The multivariate features in list or matrix form. In list form, each list element should be a vector of individual features; in matrix form, it should be a binary matrix with rownames being individual features and column names being the names of the feature sets.
gamma	Parameter for selecting radius around each support point ( $0 < \text{gamma} < 1$ ). By default $\text{gamma} = 0.1$ .
beta	Parameter for eliminating outliers ( $0 < \text{beta} \leq 1$ ). By default $\text{beta} = 0.95$ .

distance	Type of distance to be calculated between points. Any type of distance that can be passed on to the dist function can be used (default 'euclidean').
verbose	Logical indicating whether to print status related messages during computation (defaults to TRUE).

### Value

A list with elements: Support: a matrix indicating which samples were included in the support. Baseline\_list: a list where each element is the baseline of a multivariate feature. featureMat: the multivariate features in matrix form. alpha: the expected number of divergent multivariate features per sample. gamma: the gamma parameter used for baseline computation. distance: the type of distance used for baseliem computation.

### Examples

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
assays(seMat.base)$quantile = computeQuantileMatrix(seMat.base)
baseline = computeMultivariateSupport(seMat=seMat.base, FeatureSets=msigdb_Hallmarks)
```

---

computeQuantileMatrix *Compute quantile transformations*

---

### Description

Function for computing the quantile transformation for one or more samples supplied as columns of a matrix.

### Usage

```
computeQuantileMatrix(seMat)
```

### Arguments

seMat	A data matrix in SummarizedExperiment form, with each column corresponding to a sample and each row corresponding to a feature.
-------	---

### Value

A matrix of the same dimensions with the quantile data.

### Examples

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
assays(seMat.base)$quantile = computeQuantileMatrix(seMat.base)
```

---

computeUnivariateDigitization

*Perform ternary digitization*


---

## Description

Function for obtaining the digitized form, along with other relevant statistics and measures given a data matrix and a baseline matrix

## Usage

```
computeUnivariateDigitization(seMat, seMat.base, computeQuantiles = TRUE,
  gamma = c(1:9/100, 1:9/10), beta = 0.95, alpha = 0.01,
  parallel = TRUE, verbose = TRUE, findGamma = TRUE, Groups = NULL,
  classes = NULL)
```

## Arguments

seMat	SummarizedExperiment with assay to be digitized, in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
seMat.base	SummarizedExperiment with baseline assay in [0, 1], with each column corresponding to a sample and each row corresponding to a feature
computeQuantiles	Logical; apply quantile transformation to both data and baseline matrices (TRUE or FALSE; defaults to TRUE).
gamma	Range of gamma values to search through. By default gamma = 0.01, 0.02, ... 0.09, 0.1, 0.2, ..., 0.9.
beta	Parameter for eliminating outliers ( $0 < \beta \leq 1$ ). By default beta=0.95.
alpha	Expected proportion of divergent features per sample to be estimated. The optimal gamma providing this level of divergence in the baseline data will be searched for.
parallel	Logical indicating whether to compute features parallelly with mclapply on Unix based systems (defaults to TRUE, switched to FALSE if parallel package is not available).
verbose	Logical indicating whether to print status related messages during computation (defaults to TRUE).
findGamma	Logical indicating whether to search for optimal gamma values through the given gamma values (defaults to TRUE). If FALSE, the first value given in gamma will be used.
Groups	Factor indicating class association of samples (optional).
classes	Vector of class labels (optional).

## Value

A list with elements: Mat.div: divergence coding of data matrix in ternary (-1, 0, 1) form, of same dimensions at seMat baseMat.div: divergence coding of base matrix in ternary (-1, 0, 1) form, of same dimensions at seMat.base div: data frame with the number of divergent features in each

sample, including upper and lower divergence features. `div`: data frame with the divergent probability of each feature; divergence probability for each phenotype is included as well if 'Groups' and 'classes' inputs were provided. `Baseline`: a list containing a "Ranges" data frame with the baseline interval for each feature, and a "Support" binary matrix of the same dimensions as `Mat` indicating whether each sample was a support or a feature or not (1=support, 0=not in the support), `gamma`: selected gamma value, `alpha`: the expected number of divergent features per sample computed over the baseline data matrix, `optimal`: logical indicating whether the selected gamma value provided the necessary alpha requirement, `alpha_space`: a data frame with alpha values for each gamma searched

## Examples

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
dataMat = breastTCGA_Mat[, breastTCGA_Group != "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
seMat = SummarizedExperiment(assays=list(data=dataMat))
div = computeUnivariateDigitization(
  seMat = seMat,
  seMat.base = seMat.base,
  parallel = TRUE
)
assays(seMat)$div = div$Mat.div
```

---

computeUnivariateSupport

*Estimate the baseline support*

---

## Description

Function for computing the baseline support for univariate features given gamma and beta parameters.

## Usage

```
computeUnivariateSupport(seMat, gamma = 0.1, beta = 0.95,
  parallel = TRUE, verbose = TRUE)
```

## Arguments

<code>seMat</code>	SummarizedExperiment with an assay in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
<code>gamma</code>	Parameter for selecting radius around each support point ( $0 < \text{gamma} < 1$ ). By default <code>gamma = 0.1</code> .
<code>beta</code>	Parameter for eliminating outliers ( $0 < \text{beta} \leq 1$ ). By default <code>beta=0.95</code> .
<code>parallel</code>	Logical indicating whether to compute features parallelly with <code>mclapply</code> on Unix based systems (defaults to <code>TRUE</code> , switched to <code>FALSE</code> if <code>parallel</code> package is not available).
<code>verbose</code>	Logical indicating whether to print status related messages during computation (defaults to <code>TRUE</code> ).

**Value**

A list with elements "Ranges": data frame with the baseline interval for each feature, "Support": binary matrix of the same dimensions as Mat indicating whether each sample was a support for a feature or not (1=support, 0=not in the support), "gamma": gamma value, and "alpha": the expected number of divergent features per sample estimated over the samples.

**Examples**

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
assays(seMat.base)$quantile = computeQuantileMatrix(seMat.base)
baseline = computeUnivariateSupport(seMat=seMat.base)
```

---

computeUnivariateTernaryMatrix

*Compute the ternary matrix with digitized divergence coding*

---

**Description**

Function for obtaining the ternary form for a matrix of data given a baseline range

**Usage**

```
computeUnivariateTernaryMatrix(seMat, Baseline)
```

**Arguments**

seMat	SummarizedExperiment with an assay in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
Baseline	A list with a data frame element "Ranges" containing the baseline range of each features; this corresponds to the output of findUnivariateGammaWithSupport() or computeUnivariateSupport()

**Value**

A matrix containing the ternary form data.

**Examples**

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
assays(seMat.base)$quantile = computeQuantileMatrix(seMat.base)
baseline = computeUnivariateSupport(seMat=seMat.base)
dataMat = breastTCGA_Mat[, breastTCGA_Group != "NORMAL"]
seMat = SummarizedExperiment(assays=list(data=dataMat))
assays(seMat)$quantile = computeQuantileMatrix(seMat)
assays(seMat)$div = computeUnivariateTernaryMatrix(seMat=seMat, Baseline=baseline)
```

---

findMultivariateGammaWithSupport

*Find optimal gamma and corresponding support for list of feature sets*


---

## Description

Function for searching through a range of gamma values for finding the smallest gamma and support that provides expected proportion of divergent features per sample less than or equal to alpha.

## Usage

```
findMultivariateGammaWithSupport(seMat, FeatureSets, gamma = 1:9/10,
  beta = 0.95, alpha = 0.01, distance = "euclidean",
  verbose = TRUE)
```

## Arguments

seMat	SummarizedExperiment with an assay in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
FeatureSets	The multivariate features in list or matrix form. In list form, each list element should be a vector of individual features; in matrix form, it should be a binary matrix with rownames being individual features and column names being the names of the feature sets.
gamma	Range of gamma values to search through. By default gamma = {0.01, 0.02, ..., 0.09, 0.1, 0.2, ..., 0.9}.
beta	Parameter for eliminating outliers ( $0 < \beta \leq 1$ ). By default beta=0.95.
alpha	Expected proportion of divergent features per sample to be estimated over the samples in Mat. By default alpha = 0.01; i.e. search for the smallest gamma that provides 1% or less number of divergent features per sample.
distance	Type of distance to be calculated between points. Any type of distance that can be passed on to the dist function can be used (default 'euclidean').
verbose	Logical indicating whether to print status related messages during computation (defaults to TRUE).

## Value

A list with elements: Support: a matrix indicating which samples were included in the support. Baseline: a list where each element is the baseline of a multivariate feature. featureMat: the multivariate features in matrix form. alpha: the expected number of divergent multivariate features per sample. gamma: the gamma parameter selected. distance: the type of distance used for baselien computation. optimal: TRUE or FALSE indicating whether the alpha criteria was met alpha\_space: the alpha values corresponding to the gamma values searched through

## Examples

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
assays(seMat.base)$quantile = computeQuantileMatrix(seMat.base)
baseline = findMultivariateGammaWithSupport(seMat=seMat.base, FeatureSets=msigdb_Hallmarks)
```

---

findUnivariateGammaWithSupport

*Search for optimal gamma and associated support*


---

## Description

Function for searching through a range of gamma values for finding the smallest gamma that provides expected proportion of divergent features per sample less than or equal to alpha.

## Usage

```
findUnivariateGammaWithSupport(seMat, gamma = c(1:9/100, 1:9/10),
  beta = 0.95, alpha = 0.01, parallel = TRUE, verbose = TRUE)
```

## Arguments

seMat	SummarizedExperiment with an assay in [0, 1], with each column corresponding to a sample and each row corresponding to a feature; usually in quantile form.
gamma	Range of gamma values to search through. By default gamma = {0.01, 0.02, ... 0.09, 0.1, 0.2, ..., 0.9}.
beta	Parameter for eliminating outliers ( $0 < \beta \leq 1$ ). By default beta=0.95.
alpha	Expected proportion of divergent features per sample to be estimated over the samples in Mat. By default alpha = 0.01; i.e. search for the smallest gamma that provides 1% or less number of divergent features per sample.
parallel	Logical indicating whether to compute features parallelly with mclapply on Unix based systems (defaults to TRUE, switched to FALSE if parallel package is not available).
verbose	Logical indicating whether to print status related messages during computation (defaults to TRUE).

## Value

A list with elements "Ranges": data frame with the baseline interval for each feature, "Support": binary matrix of the same dimensions as Mat indicating whether each sample was a support for a feature or not (1=support, 0=not in the support), "gamma": gamma value, and "alpha": the expected number of divergent features per sample estimated over the samples, "optimal": logical indicating whether the selected gamma value provided the necessary alpha requirement, and "alpha\_space": a data frame with alpha values for each gamma searched.

## Examples

```
baseMat = breastTCGA_Mat[, breastTCGA_Group == "NORMAL"]
seMat.base = SummarizedExperiment(assays=list(data=baseMat))
assays(seMat.base)$quantile = computeQuantileMatrix(seMat.base)
baseline = findUnivariateGammaWithSupport(seMat=seMat.base)
```

---

`msigdb_Hallmarks`*Cancer Hallmark gene sets from the MSigDB collection*

---

**Description**

A subset of the cancer hallmarks functional gene sets from the MSigDB collection.

**Usage**

`msigdb_Hallmarks`

**Format**

A list of length 10, with the hallmark gene set name, each a character vector of gene symbols.

**Source**

<https://software.broadinstitute.org/gsea/msigdb/>

# Index

- \* **baseline**,
    - computeMultivariateSupport, [6](#)
    - computeUnivariateSupport, [9](#)
  - \* **binary**
    - computeMultivariateBinaryMatrix, [4](#)
  - \* **chi-squared**
    - computeChiSquaredTest, [3](#)
  - \* **datasets**
    - breastTCGA\_ER, [2](#)
    - breastTCGA\_Group, [2](#)
    - breastTCGA\_Mat, [3](#)
    - msigdb\_Hallmarks, [13](#)
  - \* **digitization**
    - computeMultivariateBinaryMatrix, [4](#)
    - computeUnivariateTernaryMatrix, [10](#)
  - \* **digitize**
    - computeMultivariateDigitization, [5](#)
    - computeUnivariateDigitization, [8](#)
  - \* **gamma**
    - findMultivariateGammaWithSupport, [11](#)
    - findUnivariateGammaWithSupport, [12](#)
  - \* **support**
    - computeMultivariateSupport, [6](#)
    - computeUnivariateSupport, [9](#)
  - \* **ternary**
    - computeMultivariateDigitization, [5](#)
    - computeUnivariateDigitization, [8](#)
    - computeUnivariateTernaryMatrix, [10](#)
- breastTCGA\_ER, [2](#)  
breastTCGA\_Group, [2](#)  
breastTCGA\_Mat, [3](#)
- computeChiSquaredTest, [3](#)  
computeMultivariateBinaryMatrix, [4](#)  
computeMultivariateDigitization, [5](#)  
computeMultivariateSupport, [6](#)  
computeQuantileMatrix, [7](#)  
computeUnivariateDigitization, [8](#)  
computeUnivariateSupport, [9](#)  
computeUnivariateTernaryMatrix, [10](#)
- findMultivariateGammaWithSupport, [11](#)  
findUnivariateGammaWithSupport, [12](#)  
msigdb\_Hallmarks, [13](#)