# Package ‘BIDistances’

May 11, 2025

**Type** Package

**Title** Bioinformatic Distances

**Version** 0.1.3

**Date** 2025-05-06

**Maintainer** Quirin Stier <Quirin_Stier@gmx.de>

**Description** A selection of distances measures for bioinformatics data. Other important distance measures for bioinformatics data are selected from the R package 'parallelDist'. A special distance measure for the Gene Ontology is available.

**Depends** R (>= 3.5.0)

**Imports** Rcpp (>= 1.0.8), RcppParallel, parallelDist, parallel, DataVisualizations, diptest, e1071, vegan, methods, pracma, ggplot2

**Suggests** knitr, rmarkdown, remotes, sphet, transport, ineq

**LinkingTo** Rcpp, RcppArmadillo, RcppParallel

**NeedsCompilation** yes

**SystemRequirements** GNU make, pandoc (>=1.12.3, needed for vignettes)

**License** GPL-3

**LazyLoad** yes

**LazyData** TRUE

**Encoding** UTF-8

**VignetteBuilder** knitr

**Author** Quirin Stier [aut, rev, ctb, cre] (ORCID: <https://orcid.org/0000-0002-7896-4737>),
Michael Thrun [aut] (ORCID: <https://orcid.org/0000-0001-9542-5543>),
Luca Brinkmann [ctb]

**Repository** CRAN

**Date/Publication** 2025-05-10 23:00:05 UTC

# Contents

---

`CosinusDistance` *Cosine Distance*

---

### Description

Calculates the cosine distance

### Usage

```
CosinusDistance(Data)
```

### Arguments

| | |
|---|---|
| `Data` | [1:n,1:d] matrix with n cases, d variables |

### Details

https://en.wikipedia.org/wiki/Cosine_similarity

### Value

| | |
|---|---|
| `Distance` | [1:n,1:n] symmetric matrix, containing the distanes of the cases (rows) for the given data |

## Note

The cosine distance is calculated by calculating the cosine similarity $d(i,j) = \max s - s(i,j)$, where $s$ is the cosine similarity and the $d$ the cosine distance.

## Author(s)

Michael Thrun

## Examples

```
data(Hepta)
distMatrix = CosinusDistance(Hepta$Data)
```

---

# Dist2All *Distances to all data points*

---

## Description

Calculates all distances from a given vector to the rows of a matrix.

## Usage

```
Dist2All(X, Data, SelectFeatures, method = "euclidean",p=2,knn=1)
```

## Arguments

| | |
|---|---|
| `X` | A vector containing the data point to be compared to data. |
| `Data` | A matrix containing the data points to be compared with x. |
| `SelectFeatures` | A vector of the same length as x and the rows of data, containing TRUE for all columns of the data to be compared and any other value for columns to be discarded. |
| `method` | (Optional) String marking, which distance measure is to be used. Euclidean by default. |
| `p` | (Optional) Scalar, The pp-th root of the sum of the pp-th powers of the differences of the components. Default is 2 |
| `knn` | (Optional) Scalar, gives the number of the indices of the k nearest neighbors returned. Default is 1 |

## Value

List with

| | |
|---|---|
| `distToAll` | A vector containing the distances from x to all rows of data. |
| `KNN` | Numeric vector, containing the indices of the k nearest neighbors (rows) to the given points |

### Note

This function is very inefficient for large Data.

### Author(s)

Michael Thrun

### Examples

```
data(Hepta)
Dist2All(Hepta$Data[1,],Hepta$Data)
```

---

## DistanceDistributions *Distance Distribution*

---

### Description

Calculates the distribution of the distances between the data points

### Usage

```
DistanceDistributions(Data, DistanceMethods=c('bhjattacharyya', 'bray',
                                                      'canberra', 'chord',
                                                    'divergence', 'euclidean',
                                                      'minkowski', 'geodesic',
                                                      'hellinger', 'kullback',
                                                      'manhattan', 'maximum',
                                                      'soergel', 'wave',
                                                      'whittaker'),
                                  CosineNonParallel = TRUE, CorrelationDist = TRUE,
                                   Mahalanobis = FALSE, Podani = FALSE,
                                   PlotIt = FALSE, PlotSampleSize = 5e3)
```

### Arguments

| | |
|---|---|
| `Data` | [1:n, 1:m] A matrix, containing data as rows. |
| `DistanceMethods` | Character vector stating all distance methods such as 'euclidean'. |
| `CosineNonParallel` | Boolean stating if cosine should be computed in parallel. |
| `CorrelationDist` | Boolean stating if CorrelationDist should be computed. |
| `Mahalanobis` | Boolean stating if Mahalanobis should be computed. |
| `Podani` | Boolean stating if Podani should be computed. |
| `PlotIt` | Boolean: TRUE => create plot. FALSE => no plot. |
| `PlotSampleSize` | Integer stating the number of samples for plotting. |

### Value

List with elements

`DistanceMatrix` [1:n, 1:n] numeric matrix containing the distance matrix

`DistanceChoice` [1:n, 1:n] numeric matrix containing the distance matrix

`OrderedDistances`
[1:n, 1:n] numeric matrix containing the distance matrix

`ggobject` ggplot object

### Author(s)

Michael Thrun

### Examples

```
iris=datasets::iris
if(requireNamespace("DataVisualizations",quietly=TRUE)){
library(DataVisualizations)
DistanceDistributions(as.matrix(iris[,1:4]), c("euclidean"), PlotIt = FALSE)
}
```

---

## `DistanceMatrix` *Pairwise distance between pairs of objects*

---

### Description

computes the distance between objects in the data matrix, X, using the method specified by method

### Usage

```
DistanceMatrix(X,method='euclidean',dim=2,outputisvector=FALSE)
```

### Arguments

`X` data matrix [1:n,1:d], n cases d variables

`method` Optional, method specified by distance string: ’binary’,’canberra’,’cityblock’,’euclidean,
’sqEuclidean’, ’maximum’,’cosine’,’chebychev’,’jaccard,’kendallM’,’kendallD’
’mahalanobis’,’minkowski’,’manhattan’,’braycur’,’cosine’,’wasserstein’,’pearsonD’,’spearmanD’,’pearso

`dim` Optional: if method="minkowski", or wasserstein, choose scalar. For minkowski the ppth root of the sum of the ppth powers of the differences of the components. For wasserstein the order, default should be then 1

`outputisvector` Optional: should the output be converted to a vector

## Details

If possible uses implementation parallelized by the parallelDist package. Otherwise R implementations besides Euclidean for which a GPU implementation is provided.

'binary' (aka asymmetric binary): The vectors are regarded as binary bits, so non-zero elements are 'on' and zero elements are 'off'. The distance is the proportion of bits in which only one is on amongst those in which at least one is on.

'cityblock'==manhattan

'maximum': Maximum distance between two components of x and y (supremum norm)

'cosine' calculates a similarity matrix sim between all column vectors of a matrix x. This matrix might be a document-term matrix, so columns would be expected to be documents and rows to be terms. the distances is than defined with D=max(sim)-sim

'jaccard' Jaccard index is computed as 2B/(1+B), where B is Bray-Curtis dissimilarity: the number of items which occur in both elements divided by the total number of items in the elements (Sneath, 1957). This measure is often also called: binary, asymmetric binary, etc.

'mahalanobis' the squared generalized Mahalanobis distance between all pairs of rows in a data frame with respect to a covariance matrix. The element of the i-th row and j-th column of the distance matrix is defined as $D_{ij}^2 = (\boldsymbol{x}_i - \boldsymbol{x}_j)'\boldsymbol{S}^{-1}(\boldsymbol{x}_i - \boldsymbol{x}_j)$

'minkowski':The p norm, the pth root of the sum of the pth powers of the differences of the components.

'manhattan': Absolute distance between the two vectors (1 norm aka L_1).

'chebychev'=max(abs(x-y)),

'canberra'=sum abs(x-y)/sum(abs(x)-abs(y)), Terms with zero numerator and denominator are omitted from the sum and treated as if the values were missing. This is intended for non-negative values (e.g., counts): taking the absolute value of the denominator is a 1998 R modification to avoid negative distances.

'braycur'=sum abs(x -y)/abs(x+y)

'pearsonM' metric, see [Legendre, 1986] or [Bock,1974, pp.77-79] sqrt((1 - r)+1)/2) with r beeing the Pearson's correlation coefficient.

'spearmanM' metric, see [Legendre, 1986] or [Bock,1974, pp.77-79] sqrt((1 - r)+1)/2) with r beeing Spearman's correlation coefficient.

'kendallM' metric, see [Legendre, 1986] or [Bock,1974, pp.77-79] sqrt((1 - r)+1)/2) with tau beeing Kendalls's correlation coefficient.

'pearsonD' dissimilarity 1 - r with r beeing the Pearson's correlation coefficient.

'spearmanD' dissimilarity 1 - r with r beeing Spearman's correlation coefficient.

'kendallD' dissimilarity 1 - r with tau beeing Kendalls's correlation coefficient.

'cosine' s. wiki for similarity conversion: max(S)-S(i,j)

## Value

| | |
|---|---|
| `Dmatrix` | [1:n,1:n] Distance Marix: Pairwise distance between pairs of objects |

## Author(s)

Michael Thrun

## References

Sneath, P. H. A. (1957) Some thoughts on bacterial classification. Journal of General Microbiology 17, pages 184-200.

Leydesdorff, L. (2005) Similarity Measures, Author Cocitation Analysis,and Information Theory. In: JASIST 56(7), pp.769-772.

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) The New S Language. Wadsworth & Brooks/Cole.

Mardia, K. V., Kent, J. T. and Bibby, J. M. (1979) Multivariate Analysis. Academic Press.

Borg, I. and Groenen, P. (1997) Modern Multidimensional Scaling. Theory and Applications. Springer.

Mahalanobis, P. C. (1936) On the generalized distance in statistics. Proceedings of The National Institute of Sciences of India, 12:49-55.

## Examples

```
  data(Hepta)
Dmatrix = DistanceMatrix(Hepta$Data,method='euclidean')
```

---

# `fastPdist` *fastPdist*

---

## Description

calculates pairwise euclidean distances

## Usage

```
fastPdist(X)
```

## Arguments

| | |
|---|---|
| `X` | [1:n,1:m] data to calculate distances to |

## Value

dist[1:n,1:n] distances

## Author(s)

Michael Thrun

## Examples

```
fastPdist(as.matrix(iris[,1:4]))
```

## fastPdistC *fastPdist*

### Description

calculates pairwise euclidean distances

### Usage

```
fastPdistC(Ar,Br)
```

### Arguments

| | |
|---|---|
| `Ar` | [1:n,1:m] data to calculate distances to |
| `Br` | [1:n,1:m] data to calculate distances to |

### Value

dist[1:n,1:n] distances

### Author(s)

Felix Riede

### References

https://blog.felixriedel.com/2013/05/pairwise-distances-in-r/

## FractionalDistance *Calculates fractional distances*

### Description

Calculates distance matrix, through $\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p}$

### Usage

```
FractionalDistance(Data, p)
```

### Arguments

| | |
|---|---|
| `Data` | [1:n,1:d] Matrix, with n cases, d variables |
| `p` | Scalar, value for p |

## Details

Values of p < 1 can be used, which can be useful for high-dimensional data, see references.

## Value

`DistanceMatrix` [1:n,1:n] symmetric Matrix, containing the distances between the cases (rows) of the input matrix

## Author(s)

Michael Thrun

## References

Aggrawal, C. C., Hinneburg, A., Keim, D. (2001), On the Suprising Behavior of Distance Metrics in High Dimensional Space.

## Examples

```
data(Hepta)
distMatrix = FractionalDistance(Hepta$Data, p = 1/2)
```

---

# `GiniDist` *GiniDist*

---

## Description

Calculates pairwise gini distances

## Usage

```
GiniDist(Data)
```

## Arguments

| | |
|---|---|
| `Data` | [1:n,1:d] data to calculate distances to |

## Value

dist[1:n,1:n] distances

## Author(s)

Michael Thrun

## Examples

```
GiniDist(as.matrix(iris[,1:4]))
```

## Hearingloss_N109 *Hearingloss data*

### Description

Hearingloss data, with Gene2GoTerm matrix.

### Usage

```
data('Hearingloss_N109')
```

### Details

FeatureMarix_Gene2Term contains the dataset, NCBI are the row names for the genes and GoTerm_Header contains the column names for the GoTerms. Size of data matrix is 109 with dimension 829.

### Source

NCBI OtoGenome Test for Hearing Loss, accessed 24 June 2022.

### References

GeneTestingRegistry (2018). OtoGenome Test for Hearing Loss Retrieved 2017

### Examples

```
data(Hearingloss_N109)
str(Hearingloss_N109)
```

## Hepta *Hepta introduced in [Ultsch, 2003]*

### Description

Clearly defined clusters, different variances. Detailed description of dataset and its clustering challenge is provided in [Thrun/Ultsch, 2020].

### Usage

```
data('Hepta')
```

### Details

Size 212, Dimensions 3, stored in `Hepta$Data`

Classes 7, stored in `Hepta$Cls`

### References

[Ultsch, 2003] Ultsch, A.: Maps for the visualization of high-dimensional data spaces, Proc. Workshop on Self organizing Maps (WSOM), pp. 225-230, Kyushu, Japan, 2003.

[Thrun/Ultsch, 2020] Thrun, M. C., & Ultsch, A.: Clustering Benchmark Datasets Exploiting the Fundamental Clustering Problems, Data in Brief, Vol. 30(C), pp. 105501, doi:10.1016/j.dib.2020.105501, 2020.

### Examples

```
data(Hepta)
str(Hepta)
```

## `jaccard` *Computes dissimilarity indices Jaccard*

### Description

The function computes dissimilarity indices Jaccard, which index is computed as 2B/(1+B), where B is Bray-Curtis dissimilarity

### Usage

```
jaccard(X)
```

### Arguments

| | |
|---|---|
| `X` | Distance Matrix |

### Value

Kosinusdistanz der beiden Vektoren x,y

### Author(s)

MT

### Examples

```
jaccard(as.matrix(iris[,1:4]))
```

## Mahalanobis *Pairwise Squared Generalized Mahalanobis Distances*

### Description

Function to calculate the squared generalized Mahalanobis distance between all pairs of rows in a data frame with respect to a covariance matrix. The element of the *i*-th row and *j*-th column of the distance matrix is defined as

$$D_{ij}^2 = (\boldsymbol{x}_i - \boldsymbol{x}_j)'\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_i - \boldsymbol{x}_j)$$

### Usage

```
Mahalanobis(X, cov, inverted = FALSE)
```

### Arguments

| | |
|---|---|
| `X` | a matrix of data (*n x d*) n cases, d variables |
| `cov` | a variance-covariance matrix (*p x p*). |
| `inverted` | logical. If `FALSE` (default), `cov` is supposed to be a variance-covariance matrix. |

### Value

Distances[1:n,1:n]

### Note

copy of function in biotools package, because this packages doesnt work under mac os

### Author(s)

Anderson Rodrigo da Silva <anderson.agro@hotmail.com>

### References

Mahalanobis, P. C. (1936) On the generalized distance in statistics. *Proceedings of The National Institute of Sciences of India*, 12:49-55.

### See Also

`dist`

### Examples

```
# Manly (2004, p.65-66)
x1 <- c(131.37, 132.37, 134.47, 135.50, 136.17)
x2 <- c(133.60, 132.70, 133.80, 132.30, 130.33)
x3 <- c(99.17, 99.07, 96.03, 94.53, 93.50)
x4 <- c(50.53, 50.23, 50.57, 51.97, 51.37)
x <- cbind(x1, x2, x3, x4)
Cov <- matrix(c(21.112,0.038,0.078,2.01, 0.038,23.486,5.2,2.844,
0.078,5.2,24.18,1.134, 2.01,2.844,1.134,10.154), 4, 4)
Mahalanobis(x, Cov)

# End (not run)
```

## msmd *msmd*

### Description

msmd

### Usage

```
msmd(Values1, Values2, ParameterC)
```

### Arguments

`Values1` [1:N1] Numeric vector with values of the first time series.

`Values2` [1:N1] Numeric vector with values of the second time series.

`ParameterC` Numeric vector with time stamps of the first time series.

### Value

List with elements

`Value` Distance measure

### Author(s)

Quirin Stier

### Examples

```
msmd(1:10, 1:10)
```

## `nearest` *Nearest*

### Description

returns the index of the nearest neighbour of a given data point.

### Usage

```
nearest(Data, i, defined)
```

### Arguments

| | |
|---|---|
| `Data` | A matrix holding n data points as row vectors. |
| `i` | the index of the data point, who's nearest neighbour is sought. |
| `defined` | A row vector with 1 for all columns of data that are used for the computation. If missing, all columns are used. |

### Value

| | |
|---|---|
| `nNInd` | The index of the nearest neighbour of data[i, ] |

### Author(s)

Michael Thrun, Raphael Paebst

### Examples

```
nearest(Data = as.matrix(iris[,1:4]), i = 1)
```

## `SharedNeighborDistance` *Shared Neighbor Distance*

### Description

Calculates the Shared Neighbor Distance

### Usage

```
SharedNeighborDistance(Data, k = 5, NThreads = NULL, ComputationInR = FALSE)
```

## Arguments

Data — [1:n,1:d] matrix with n cases, d variables

k — Integer defining the number of nearest neighbors

NThreads — Number of threads in parallel computation.

ComputationInR — Boolean (Default ComputationInR = FALSE). If FALSE, do computation in Rcpp, else in R (very slow).

## Value

Distance — [1:n,1:n] symmetric matrix, containing the distanes of the cases (rows) for the given data

## Author(s)

Quirin Stier

## References

https://github.com/albert-espin/snn-clustering/blob/master/SNN/snn.py

## Examples

```
data(Hepta)
distMatrix = SharedNeighborDistance(Hepta$Data, NThreads = 1, ComputationInR=TRUE)
```

# Tfidf_dist — *Term frequency-inverse document frequency distance*

## Description

Computes the term frequency inverse document frequency (tfidf) distance for a FeatureMatrix_Gene2GoTerm. In case of genes with annotated GOterms from gene ontology genes can be interpreted as documents and GOterms as terms.

## Usage

```
Tfidf_dist(FeatureMatrix_Gene2GoTerm, tf_fun = mean)
```

## Arguments

FeatureMatrix_Gene2GoTerm — [1:n,1:d] Matrix, with n genes and d GO-Terms.

tf_fun — Function, defining the numerator value in the normalized Term-frequency. The default is the mean of the not 0 values.

### Details

For the FeatureMatrix_Gene2GoTerm it is:
FeatureMatrix_Gene2GoTerm[i,j] > 0 iff GOterm j is relevant for gene i. The FeatureMatrix_Gene2GoTerm[i,j] > 1 if the specific gene is annotated by in a specific GO-Term with more than one evidence code FeatureMatrix_Gene2GoTerm[i,j] is the frequency of term js occurance in document i.

### Value

List with

`dist` Numeric vector containing the tdfidf distances between the documents = absolute difference of TfidfWeights

`TfidfWeights` [1:n] Numeric vector containing the term frequence inverse document frequency weights used for the distance, given as the Term frequency*Inverse document frequency

### Author(s)

Michael Thrun

### References

Stier, Q. and Thrun, M., C.: Deriving homogeneous subsets from gene sets by exploiting the Gene Ontology, Informatica, in review, 2023

### Examples

```
data(Hearingloss_N109)
V = Tfidf_dist(Hearingloss_N109$FeatureMatrix_Gene2Term)
dist = V$dist
TfidfWeights = V$TfidfWeights
```

---

## `ToroidDist2All` *Calculate toroid Euclidean Distances*

---

### Description

Calculate toroid Euclidean Distances

### Arguments

`positionxy` One datapoint

`AllPositions(1:AnzData:2)`
All Other dataPoints

`Lines, Columns` Size of planar grid

## Value

Dist2All(1:AnzData,1:AnzData); distance(s) between XY and AllPositions

## Author(s)

MT

## Examples

```
positionxy = c(1,1)
AllPositions = rbind(c(2,3), c(5,2))
Lines = 40
Columns = 80
ToroidDist2All(positionxy, AllPositions, Lines, Columns)
```

---

# `TransformSimilarity2MetricDistance` *TransformSimilarity2MetricDistance*

---

## Description

TransformSimilarity2MetricDistance

## Usage

```
TransformSimilarity2MetricDistance(Similarity)
```

## Arguments

| | |
|---|---|
| `Similarity` | Similarity |

## Value

Similarity

## Author(s)

Michael Thrun

## Examples

```
Data_S = fastPdist(as.matrix(iris[,1:4]))
Data_S = Data_S-min(Data_S)
Data_S = Data_S/max(Data_S)
diag(Data_S) = 1
TransformSimilarity2MetricDistance(Data_S)
```

## `twed` *twed*

### Description

twed

### Usage

```
twed(Values1, Values2, Time1, Time2, Nu = 1, Lambda = 1, Degree = 2)
```

### Arguments

| | |
|---|---|
| `Values1` | [1:N1] Numeric vector with values of the first time series. |
| `Values2` | [1:N1] Numeric vector with values of the second time series. |
| `Time1` | [1:N1] Numeric vector with time stamps of the first time series. |
| `Time2` | [1:N1] Numeric vector with time stamps of the second time series. |
| `Nu` | Optional, Numeric: Elasticity parameter - nu >=0 needed for distance measure. |
| `Lambda` | Optional, Numeric: Penalty for deletion operation. |
| `Degree` | Optional, Integer: Degree of the p norm for local cost. |

### Value

List with elements

| | |
|---|---|
| `TWED` | TWED distance between time series Values1 (Time1) and Values2 (Time2) |
| `DPMatrix` | [1:n, 1:m] Numeric matrix |

### Author(s)

Quirin Stier

### Examples

```
twed(1:10, 1:10, 1:10, 1:10)
```

## VariablePrecision *VariablePrecision*

### Description

Computes the variable precision

### Usage

```
VariablePrecision(Variable)
```

### Arguments

| | |
|---|---|
| `Variable` | Numeric vector [1:n] or matrix [1:n, 1:d] |

### Value

MinAbsDiff, MinAbsNZDiff, MinExpo

### Author(s)

Michael Thrun

### Examples

```
data(Hepta)
distMat = VariablePrecision(as.matrix(iris[, 1]))

distMat = VariablePrecision(as.matrix(iris[, 1:4]))
```

## WassersteinDist *Wasserstein Distance*

### Description

Computes the Wasserstein distance for a data matrix

### Usage

```
WassersteinDist(Data, p = 1, InverseWeighting = FALSE)
```

### Arguments

| | |
|---|---|
| `Data` | data matrix of n cases and d feautures |
| `p` | scalar higher than one, the power to which the Euclidean distance between points is taken in order to compute transportation costs. |
| `InverseWeighting` | weighting per row can be either 1 (FALSE) or 1/n (TRUE) |

## Details

Wasserstein distance, also known as Earth Mover's Distance (EMD) is the distance between two probability distributions over a region D. The Wasserstein distance of order p is defined as the p-th root of the total cost incurred when transporting measure a to measure b in an optimal way, where the cost of transporting a unit of mass from x to y is given as the p-th power of the Euclidean distance.

It is claimed to be useful for distributions that do not align well with traditional measures like Euclidean distance.

## Value

matrix of distances, symmetric

## Author(s)

Michae Thrun

## References

...

## See Also

`wasserstein1d`

## Examples

```
data(Hepta)
distMat=WassersteinDist(Hepta$Data)
```

# Index