Package 'CompositionalNAimp'

March 31, 2026

Type Package

Title Missing Value Imputation with Compositional Data

Version 1.0

Date 2026-03-26

Author Michail Tsagris [aut, cre]

Maintainer Michail Tsagris <mtsagris@uoc.gr>

Depends R (>= 4.0)

Imports Compositional, graphics, Rfast, Rnanoflann

Suggests Rfast2

Description Functions to perform missing value imputation with compositional data using the Jensen-Shannon divergence based k--NN and a--k--
NN algorithms. The functions are based on the following paper:
Tsagris M., Alenazi A. and Stewart C. (2026). ``A Jensen--Shannon divergence based k--
NN algorithm for missing value imputation in compositional data''.
Journal of Applied Statistics (Accepted for publication).

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2026-03-31 09:50:09 UTC

Contents

CompositionalNAimp-package
                        Missing Value Imputation with Compositional Data.

Description

Functions to perform missing value imputation with compositional data using the Jensen-Shannon divergence k–NN algorithm.

Details

Package:	CompositionalNAimp	Type:	Package
Version:	1.0		
Date:	2026-03-26		
License:	GPL-2		

Maintainers

Michail Tsagris <mtsagris@uoc.gr>.

Author(s)

Michail Tsagris <mtsagris@uoc.gr>.

References

Tsagris M., Alenazi A. and Stewart C. (2026). A Jensen–Shannon divergence based k–NN algorithm for missing value imputation in compositional data. Journal of Applied Statistics (Accepted for publication).

Cross-validation for the Jensen-Shannon based a-k-NN algorithm
                Cross-validation for the Jensen-Shannon based α–k–NN algorithm

---

Description

Cross-validation for the Jensen-Shannon based α–k–NN algorithm.

Usage

```
alfaknnimp.tune(x, k = 2:10, a = seq(-1, 1, by = 0.1), type = "kl",
R = 50, graph = FALSE)
```

Arguments

x	A matrix with the compositional data, zeros are allowed.
k	A vector with the number of nearest neighbours to explore.
a	A vector with the values of α to consider.
type	The type of distance to use in order to see the difference between the true and the imputed data. If there are no zeros, The Kullback-Leibler divergence ("kl", default value) may be used, otherwise the (squared) Jensen-Shannon divergence ("js") may be used.
R	The number of repetitions to perform.
graph	Do you want to produce a heatmap with the performance of each combination of α and k?

Details

The algorithm performs a repeated cross-validation for the Jensen-Shanon divergence based α–k–NN algorithm. It selects the optimal pair of α and k.

Value

A list including:

perf	A matrix with the average performance (Kullback-Leibler or Jensen-Shannon divergence) for each value of α and k.
performance	The value of the optimal performance.
best_a	The optimal value of α.
best_k	The optimal value of k.
runtime	The runtime of the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Tsagris M., Alenazi A. and Stewart C. (2026). A Jensen–Shannon divergence based k–NN algorithm for missing value imputation in compositional data. Journal of Applied Statistics (Accepted for publication).

See Also

alfa.knnimp, cv.knnimp

Examples

```
x <- matrix( rgamma(100 * 7, runif(7, 4,6), 1), ncol = 7, byrow = TRUE )
x <- w <- x / Rfast::rowsums(x)  ## Dirichlet simulated values
dm <- dim(x)     ;    n <- dm[1]    ;    p <- dm[2]
xam <- matrix(nrow = 10, ncol =  3)
nu <- sample(1:n, 10)
for ( vim in 1:10)  xam[vim, ] <- sample(1:p, 3)
for ( vim in 1:10)  x[ nu[vim], xam[vim, ] ] <- NA
mod <- alfaknnimp.tune( x, k = 2:4, a = c(0.5, 0.75, 1), R = 20 )
```

Cross-validation for the Jensen-Shannon based k–NN algorithm
                        Cross-validation for the Jensen-Shannon based k–NN algorithm

---

Description

Cross-validation for the Jensen-Shannon based k–NN algorithm.

Usage

```
cv.knnimp(x, k = 2:10, type = "Ait", R = 200)
```

Arguments

x	A matrix with the compositional data, zeros are allowed.
k	A vector with the number of nearest neighbours to explore.
type	The type of distance to use in order to see the difference between the true and the imputed data. If there are no zeros, Aitchison's distance ("Ait", default value) may be used, otherwise the (squared) Jensen-Shannon divergence ("js") may be used.
R	The number of repetitions to perform.

Details

The algorithm performs a repeated cross-validation for the Jensen-Shanon divergence based k–NN algorithm. It selects the optimal value of k, the number of nearest neighbours.

Value

A list including:

perf	A matrix with the performance (Aitchison or Jensen-Shannon divergence) for each value of k, at each repetition.
performance	The averagge performance (Aitchison or Jensen-Shannon divergence) for each value of k.
best_k	The optimal value of k.
runtime	The runtime of the cross-validation procedure.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Tsagris M., Alenazi A. and Stewart C. (2026). A Jensen–Shannon divergence based k–NN algorithm for missing value imputation in compositional data. Journal of Applied Statistics (Accepted for publication).

See Also

[knnimp](#)

Examples

```
x <- matrix( rgamma(100 * 7, runif(7, 4,6), 1), ncol = 7, byrow = TRUE )
x <- w <- x / Rfast::rowsums(x)  ## Dirichlet simulated values
xam <- matrix(nrow = 10, ncol =  3)
nu <- sample(100, 10)
for ( vim in 1:10)  xam[vim, ] <- sample(1:7, 3)
for ( vim in 1:10)  x[ nu[vim], xam[vim, ] ] <- NA
mod <- cv.knnimp(x, k = 2:10)
```

Imputation of missing values using the Jensen-Shannon based a-k–NN
algorithm

Imputation of missing values using the Jensen-Shannon based α–k–NN algorithm

---

Description

Imputation of missing values using the Jensen-Shannon based α–k–NN algorithm.

Usage

```
alfa.knnimp(x, k = 2:10, a = seq(0.1, 1, by = 0.1), econ = TRUE)
```

Arguments

x	A matrix with the compositional data, zeros are allowed.
k	The number of nearest neighbours to use. This can be a single number or a numerical vector.
a	A vector (or a single number) with the values of α to consider.
econ	This argument is for memory saving purposes. If this is TRUE, the function will return only the data with the imputed missing values, otherwise it will return the initial compositional datawith the imputed values. Evidently, the second option requires more memory.

Details

Missing value imputation is performed using the α–k–NN algorithm with the (squared) Jensen-Shanon divergence. The difference from the simple k–NN algorithm is that instead of computing the simple arithmetic mean to impute the missing values, we compute the Frechet mean (Tsagris, Preston and Wood, 2011).

Value

A list including:

index	A vector with the rows (indexes) that contain missing values.
xa	A list with as many elements as the number of the values of the argument "a" (α values), and inside of each element there is a sub-list with as many elements as the number of neighbours (length of k basically). If the argument "econ" is TRUE, then each element of the list contaions a matrix of the imputed compositions only. If the argument "econ" is FALSE, then each element contains the initial compositional data and the imputed missing values.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Tsagris M., Alenazi A. and Stewart C. (2026). A Jensen–Shannon divergence based k–NN algorithm for missing value imputation in compositional data. Journal of Applied Statistics (Accepted for publication).

Tsagris M.T., Preston S. and Wood A.T.A. (2011). A data-based power transformation for compositional data. In Proceedings of the 4th Compositional Data Analysis Workshop, Girona, Spain. https://arxiv.org/pdf/1106.1451.pdf

See Also

alfaknnimp.tune, knnimp

Examples

```
x <- matrix( rgamma(100 * 7, runif(7, 4,6), 1), ncol = 7, byrow = TRUE )
x <- w <- x / Rfast::rowsums(x)  ## Dirichlet simulated values
dm <- dim(x)    ;    n <- dm[1]    ;    p <- dm[2]
xam <- matrix(nrow = 10, ncol =  3)
nu <- sample(1:n, 10)
for ( vim in 1:10)  xam[vim, ] <- sample(1:p, 3)
for ( vim in 1:10)  x[ nu[vim], xam[vim, ] ] <- NA
mod <- alfa.knnimp( x, k = 2:4, a = c(0.5, 1) )
```

Imputation of missing values using the Jensen-Shannon based k–NN algorithm

Imputation of missing values using the Jensen-Shannon based k–NN algorithm

Description

Imputation of missing values using the Jensen-Shannon based k–NN algorithm.

Usage

```
knnimp(x, k = 2)
```

Arguments

x	A matrix with the compositional data, zeros are allowed.
k	The number of nearest neighbours to use. This can be a single number or a numerical vector.

Details

Missing value imputation is performed using the k–NN algorithm with the (squared) Jensen-Shanon divergence.

Value

A list including:

index	A vector with the rows (indexes) that contain missing values.
y	A list with as many elements as the number of neighbours (length of k basically). Each element of the list contains a matrix of the initial compositions, including the imputed missing values. These correspond to each value of k.

Author(s)

Michail Tsagris.

R implementation and documentation: Michail Tsagris <mtsagris@uoc.gr>.

References

Tsagris M., Alenazi A. and Stewart C. (2026). A Jensen–Shannon divergence based k–NN algorithm for missing value imputation in compositional data. Journal of Applied Statistics (Accepted for publication).

See Also

cv.knnimp, alfa.knnimp

Examples

```
x <- matrix( rgamma(100 * 7, runif(7, 4,6), 1), ncol = 7, byrow = TRUE )
x <- w <- x / Rfast::rowsums(x)  ## Dirichlet simulated values
dm <- dim(x)     ;     n <- dm[1]    ;     p <- dm[2]
xam <- matrix(nrow = 10, ncol =  3)
nu <- sample(1:n, 10)
for ( vim in 1:10)  xam[vim, ] <- sample(1:p, 3)
for ( vim in 1:10)  x[ nu[vim], xam[vim, ] ] <- NA
mod <- knnimp(x, k = 2)
```

Index