Package: mvInterpolation (via r-universe)

May 18, 2026

Title Performs interpolation in multidimensional settings

Version 0.0.0.9000

Description This package implements multivariate extensions of the interpolation algorithms from Shepard, Donald (1968): ``A two-dimensional interpolation function for irregularly-spaced data''. Proceedings of the 1968 ACM National Conference. pp. 517<96>524. doi:10.1145/800186.810616.

Depends R (>= 3.4.2)

License GPL-3

Encoding UTF-8

LazyData true

Imports geometry

RoxygenNote 6.0.1

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

Date/Publication 2018-01-19 14:44:12 UTC

RemoteUrl https://github.com/amaendle/mvInterpolation

RemoteRef HEAD

RemoteSha 38a2619d16dbe0a35407b23cd5cf7b30f4df5e53

Contents

`mvInterpolation-package` *Generate R documentation from inline comments.*

Description

Roxygen2 allows you to write documentation in comment blocks co-located with code.

Details

The only function Otherwise refer to the vignettes to see how to format the documentation.

Author(s)

Andreas Mändle, <maendle@uni-bremen.de>

References

Shepard, Donald (1968). "A two-dimensional interpolation function for irregularly-spaced data". Proceedings of the 1968 ACM National Conference. pp. 517–524. doi:10.1145/800186.810616.

`cosangmx` *Get cosinus of pairwise angles between data points and interpolation point as fixed point*

Description

This function is basically a helper function for mvInterpolation::ti_get and not intended for direct use.

Usage

```
cosangmx(data, xinp, distance = NULL)
```

Arguments

`data`	A matrix of data points
`xinp`	Coordinates of the fixed point of the angles; usually the point where interpolation has to be computed
`distance`	Distances between xinp and data can be given as an optional argument, if they have been coputed before, such that a new computation is not necessary

Value

The cosini of the pairwise angles between two data points with fixed point xinp

Examples

```
coords <- matrix(runif(10*4)*10, ncol=4)
cosangmx(coords,rep(4,4))
```

`di_get` *Get weights for interpolation by inverse distance weighting*

Description

Interpolation weights for inverse distance weighting.

Usage

```
di_get(data, xinp, p = 2)
```

Arguments

`data`	A matrix of the points for which the function is known
`xinp`	Coordinates, where interpolation has to be computed
`p`	additional parameter as in the Shepard paper

Value

The weights for the IDW interpolation

Examples

```
mydata <- matrix(runif(10*4)*10, ncol=4)
mydata <- cbind(mydata,abs(apply(mydata,1,sum)-3),abs(apply(mydata,1,prod)-4))
di_get(mydata, c(4,4,4,4))
```

`getR` *Get initial search-radius*

Description

This function determines a search radius such that on average 7 points lie within the ball with the radius.

Usage

```
getR(data, d = NULL)
```

Arguments

data	A matrix of the points for which the function is known
d	Dimension of the coordinate space

Value

The radius as a numeric

Examples

```
coords <- matrix(runif(10*4)*10, ncol=4)
getR(coords)
```

---

si_get *Get weights for interpolation by inverse distance weighting (IDW) for nearest neighbors*

Description

Interpolation weights for inverse distance weighting of the nearest neighbors (4 to 10 neighbors, depending on the distance)

Usage

```
si_get(data, xinp, rad = getR)
```

Arguments

data	A matrix of the points for which the function is known
xinp	Coordinates, where interpolation has to be computed
rad	Function which gives the initial radius for the observations considered in the interpolation

Value

The weights for the IDW interpolation based on the nearest neighbors

Examples

```
mydata <- matrix(runif(10*4)*10, ncol=4)
mydata <- cbind(mydata,abs(apply(mydata,1,sum)-3),abs(apply(mydata,1,prod)-4))
si_get(mydata, c(4,4,4,4))
```

slopezi_get *Get drift based on slopes*

Description

The drift is needed for interpolation by angular distance weighting extended by slopes. Interpolation weights are based on the nearest neighbors (4 to 10 neighbors, depending on the distance).

Usage

```
slopezi_get(data, xinp, rad = getR, vparam = 0.1)
```

Arguments

data	A matrix of the points for which the function is known
xinp	Coordinates, where interpolation has to be computed
rad	Function which gives the initial radius for the observations considered in the interpolation
vparam	Scalar, parameter for the slopes

Value

The weights for the ADW interpolation considering slopes and based on the nearest neighbors

Examples

```
mydata <- matrix(runif(10*4)*10, ncol=4)
mydata <- cbind(mydata,abs(apply(mydata,1,sum)-3),abs(apply(mydata,1,prod)-4))
slopezi_get(mydata, c(4,4,4,4))
```

ti_get *Get weights for interpolation by angular distance weighting (ADW)*

Description

Interpolation weights for angular distance weighting of the nearest neighbors (4 to 10 neighbors, depending on the distance)

Usage

```
ti_get(data, xinp, rad = getR)
```

Arguments

data	A matrix of the points for which the function is known
xinp	Coordinates, where interpolation has to be computed
rad	Function which gives the initial radius for the observations considered in the interpolation

Value

The weights for the ADW interpolation based on the nearest neighbors

Examples

```
mydata <- matrix(runif(10*4)*10, ncol=4)
mydata <- cbind(mydata,abs(apply(mydata,1,sum)-3),abs(apply(mydata,1,prod)-4))
ti_get(mydata, c(4,4,4,4))
```

---

wintpl *Do the multivaiate interpolation based on weights and drift functions*

Description

The drift is needed for interpolation by angular distance weighting extended by slopes. Interpolation weights include either all observations or are based on the nearest neighbors (4 to 10 neighbors, depending on the distance).

Usage

```
wintpl(data, xinp, weights, drift = 0, p = NULL, vparam = NULL)
```

Arguments

data	A matrix of the points for which the function is known
xinp	Coordinates, where interpolation has to be computed
weights	Vector of the weights for the interpolation
drift	Matrix of the drift for the interpolation
p	Scalar, parameter for the weights, currently not used
vparam	Scalar, parameter for the slopes, currently not used

Details

The interpoliation works for a multidimensional input space and also for multi-responses.

Value

The interpolated value(s) at xinp

Examples

```
mydata <- matrix(runif(10*4)*10, ncol=4)
mydata <- cbind(mydata,abs(apply(mydata,1,sum)-3),abs(apply(mydata,1,prod)-4))
inp <- rep(4,4)
#Inverse distance weighting (based on all observations)
wintpl(data=mydata, xinp=inp, weights=di_get(data=mydata, xinp=inp,2))
#Inverse distance weighting (based on nearest neighbors)
wintpl(data=mydata, xinp=inp, weights=si_get(data=mydata, xinp=inp)^2)
#Angular distance weighting
wintpl(data=mydata, xinp=inp, weights=ti_get(data=mydata, xinp=inp))
#Angular distance weighting under consideration of slopes
wintpl(data=mydata, xinp=inp, weights=ti_get(data=mydata, xinp=inp), drift=slopezi_get(mydata,inp))
```

Index