Package 'deform'

October 19, 2023

Type Package

Title Spatial Deformation and Dimension Expansion Gaussian Processes

Version 1.0.0

Date 2023-10-18

Maintainer Ben Youngman <b.youngman@exeter.ac.uk>

Description Methods for fitting nonstationary Gaussian process models by spatial deformation, as introduced by Sampson and Guttorp (1992) <doi:10.1080/01621459.1992.10475181>, and by dimension expansion, as introduced by Bornn et al. (2012) <doi:10.1080/01621459.2011.646919>. Low-rank thin-plate regression splines, as developed in Wood, S.N. (2003) <doi:10.1111/1467-9868.00374>, are used to either transform co-ordinates or create new latent dimensions.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.1

Imports Rcpp (>= 1.0.10), MASS

LinkingTo Rcpp, RcppArmadillo

Suggests lattice, gridExtra

Depends R (>= 3.5.0)

NeedsCompilation yes

Author Ben Youngman [aut, cre] (<https://orcid.org/0000-0003-0215-8189>)

Repository CRAN

Date/Publication 2023-10-19 08:10:02 UTC

R topics documented:

aniso *Fitting anisotropic spatial Gaussian process models*

Description

Function `aniso` fits a conventional 2-dimensional anisotropic Gaussian process, i.e. just with scalings in the x and y coordinates.

Usage

```
aniso(x, z, n, correlation = FALSE, cosine = FALSE, standardise = "together")
```

Arguments

`x`	a 2-column matrix comprising x and y coordinates column-wise, respectively, or a list; see Details for the latter
`z`	a variance-covariance matrix
`n`	an integer number of data
`correlation`	a logical defining whether z should be assumed to be a correlation matrix; defaults to `FALSE`
`cosine`	a logical defining whether the powered exponential covariance function should be multiplied by the cosine of scaled distances, i.e. giving a damped oscillation; defaults to `FALSE`
`standardise`	a character string that governs whether dimensions are scaled by a common (`"together"`) or dimension-specific factor; defaults to `"together"`

Details

If `x` is a list, then it wants elements `"x"`, `"z"` and `"n"` as described above.

Value

An object of class `deform` and then of class `anisotropic`

References

Sampson, P. D. and Guttorp, P. (1992) Nonparametric Estimation of Nonstationary Spatial Covariance Structure, Journal of the American Statistical Association, 87:417, 108-119, doi:10.1080/01621459.1992.10475181'

Examples

```
data(solar)
aniso(solar$x, solar$z, solar$n)
# equivalent to aniso(solar)
```

---

cencov *Correlation and covariance matrices from censored data*

Description

Correlation and covariance matrices from censored data

Usage

```
cencov(x, u)

cencor(x, u)
```

Arguments

x	a numeric matrix
u	a numeric matrix giving corresponding points of left-censoring

Details

For `cencov()` a covariance matrix is returned and for `cencor()` a correlation matrix is returned. Note that `cencov()` calls `cencor()`. Estimates are based on assuming values are from a multivariate Gaussian distribution.

Value

a matrix

See Also

cov and cor for uncensored estimates.

Examples

```
# generate some correlated data
n <- 1e2
x <- rnorm(n)
y <- 0.25 * x + sqrt(0.75) * rnorm(n)
xy <- cbind(x, y)
# threshold of zero for left-censoring
```

```
u <- matrix(0, n, 2)
# left-censored correlation matrix
cencor(xy, u) # could check with cor(xy)
# left-censored covariance matrix
cencov(xy, u)
```

`deform` *Fitting low-rank nonstationary spatial Gaussian process models through spatial deformation*

Description

Function `deform` fits a 2-dimensional deformation model, where typically x and y coordinates in geographic (G-) space will be provided and then deformed to give new coordinates in deformed (D-) space in which isotropy of a Gaussian process is optimally achieved.

Usage

```
deform(
  x,
  z,
  n,
  k = c(10, 10),
  lambda = c(-1, -1),
  lambda0 = rep(exp(3), length(k)),
  correlation = FALSE,
  cosine = FALSE,
  bijective = FALSE,
  bijective.args = NULL,
  trace = 0,
  standardise = "together"
)
```

Arguments

`x`	a 2-column matrix comprising x and y coordinates column-wise, respectively, or a list; see Details for the latter
`z`	a variance-covariance matrix
`n`	an integer number of data
`k`	an integer vector of ranks
`lambda`	specified lambda values; see Details
`lambda0`	initial lambda values
`correlation`	a logical defining whether `z` should be assumed to be a correlation matrix; defaults to `FALSE`

`cosine`	a logical defining whether the powered exponential covariance function should be multiplied by the cosine of scaled distances, i.e. giving a damped oscillation; defaults to `FALSE`
`bijective`	a logical for whether a bijective deformation should be imposed; defaults to FALSE
`bijective.args`	a list specifying quantities to ensure bijectivity, if bijective == TRUE; see Details
`trace`	an integer specifying the amount to report on optimisation (0, default, is nothing; 1 gives a bit)
`standardise`	a character string that governs whether dimensions are scaled by a common (`"together"`) or dimension-specific factor; defaults to `"together"`

Details

If `x` is a list, then it wants elements `"x"`, `"z"` and `"n"` as described above.

Values of `lambda` multiply the penalties placed on the wiggliness of the smooths that form the deformations. Larger values make things less wiggly. Values of `lambda0` specify initial values for `lambda`, which are still optimised.

`bijective.args()` is a 4-element list: `"mult"` is a penalty placed on the numerical approximation to identifying non-bijectivity, where larger values impose bijectivity more strictly; `"scl"` is a scaling placed on the grid used to numerically identify non-bijectivity, where smaller values will typically impose bijectivity more strictly; `"nx"` and `"ny"` specify the x and y dimensions of the grid used to numerically identify bijectivity. Defaults are `mult = 1e3`, `scl = 1`, `nx = 40` and `ny = 40`. It is advisable to use `"mult"` and not `"scl"` to control bijectivity, in the first instance.

Value

An object of class `deform` and then of class `deformation`

References

Sampson, P. D. and Guttorp, P. (1992) Nonparametric Estimation of Nonstationary Spatial Covariance Structure, Journal of the American Statistical Association, 87:417, 108-119, doi:10.1080/01621459.1992.10475181

Wood, S.N. (2003), Thin plate regression splines. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 65: 95-114. doi:10.1111/14679868.00374

Examples

```
data(solar)
deform(solar$x, solar$z, solar$n)
# equivalent to deform(solar)

# bijective deformation
deform(solar, bijective = TRUE)
```

```
# deformation with specified rank
deform(solar, k = c(10, 8))
```

---

expand — *Fitting low-rank nonstationary spatial Gaussian process models through dimension expansion*

Description

Function exapnd fits a multi-dimensional dimension expansion model, where typically x and y coordinates in geographic (G-) space will be provided and then scaled and combined with new latent dimensions (that a functions of x and y) to give new coordinates in deformed (D-) space in which isotropy of a Gaussian process is optimally achieved.

Usage

```
expand(
  x,
  z,
  n,
  k = 10,
  lambda = rep(-1, length(k)),
  lambda0 = rep(exp(3), length(k)),
  correlation = FALSE,
  cosine = FALSE,
  trace = 0,
  z0 = NULL,
  standardise = "together"
)
```

Arguments

`x`	a 2-column matrix comprising x and y coordinates column-wise, respectively, or a list; see Details for the latter
`z`	a variance-covariance matrix
`n`	an integer number of data
`k`	an integer vector of ranks
`lambda`	specified lambda values
`lambda0`	initial lambda values
`correlation`	a logical defining whether `z` should be assumed to be a correlation matrix; defaults to `FALSE`

`cosine`	a logical defining whether the powered exponential covariance function should be multiplied by the cosine of scaled distances, i.e. giving a damped oscillation; defaults to `FALSE`
`trace`	an integer specifying the amount to report on optimisation (0, default, is nothing; 1 gives a bit)
`z0`	a scalar giving initial values (which alternate `z0`, `-z0`, `z0`, `...` for latent dimensions
`standardise`	a character string that governs whether dimensions are scaled by a common (`"together"`) or dimension-specific factor; defaults to `"together"`

Details

If `x` is a list, then it wants elements `"x"`, `"z"` and `"n"` as described above.

Value

An object of class `deform` and then of class `expansion`

References

Bornn, L., Shaddick, G., & Zidek, J. V. (2012). Modeling nonstationary processes through dimension expansion. Journal of the American Statistical Association, 107(497), 281-289. doi:10.1080/01621459.2011.646919.

Examples

```
# one-dimensional expansion
data(solar)
expand(solar$x, solar$z, solar$n)
# equivalent to expand(solar)


# two-dimensional expansion with rank-8 and rank-5 dimensions
expand(solar$x, solar$z, solar$n, c(8, 5))
```

`plot.deform` *Plot a fitted* `deform` *object*

Description

Plot a fitted `deform` object

Usage

```
## S3 method for class 'deform'
plot(
  x,
  start = 1,
  graphics = "base",
  breaks = NULL,
  pal = function(n) hcl.colors(n, "YlOrRd", rev = TRUE),
  onepage = FALSE,
  nx = 10,
  ny = 10,
  xp = NULL,
  yp = NULL,
  xlab = NULL,
  ylab = NULL,
  ...
)
```

Arguments

`x`	a fitted `deform` object
`start`	an integer giving the starting dimension of plots of dimension expansion models; defaults to 1
`graphics`	a character string that is either `"graphics"` or `"lattice"` and states the graphics package to use for plots; defaults to `"graphics"`
`breaks`	an integer, vector or list; see Details
`pal`	a function specifying the colour palette to use for plots; defaults to `hcl.colors(..., 'YlOrRd', rev = TRUE)`
`onepage`	a logical specifying whether all plots should be put on one page; defaults to `FALSE`, which makes use of the current graphics state
`nx`	number of x points to use for plotting grid
`ny`	number of y points to use for plotting grid
`xp`	x points to use for plotting grid
`yp`	y points to use for plotting grid
`xlab`	x-axis label
`ylab`	y-axis label
`...`	extra arguments to pass to `plot()`

Details

If `breaks` is an integer then it specifies the number of breaks to use for colour scales; if it's a vector, then it's the breaks themselves; and if it's a list then it's different breaks for each dimension.

Value

Plots representing all one- or two-dimensional smooths

Examples

```
# deformations
data(solar)
m0 <- deform(solar$x, solar$z, solar$n)

# plot representation of deformation
plot(m0)

# as above with specified x and y grid
xvals <- seq(-123.3, -122.25, by = .05)
yvals <- seq(49, 49.4, by = .05)
plot(m0, xp = xvals, yp = yvals)



# one-dimensional expansion
data(solar)
m1 <- expand(solar$x, solar$z, solar$n)

# plot its three dimensions
op <- par(mfrow = c(1, 3))
plot(m1)
par(op)

# or plot using lattice::levelplot
plot(m1, graphics = 'lattice')
# or as above, but on one page
plot(m1, graphics = 'lattice', onepage = TRUE)



# two-dimensional expansion
m2 <- expand(solar$x, solar$z, solar$n, c(8, 5))
# plot of its third and fourth dimensions for given x and y values
op <- par(mfrow = c(1, 2))
plot(m2, start = 3, xp = xvals, yp = yvals)
par(op)

# using lattice::levelplot with common breaks across dimensions with
# a palette that gives latent dimensions in white where near zero
plot(m2, onepage = TRUE, graphics = 'lattice', breaks = seq(-0.35, 0.35, by = 0.1),
     pal = function(n) hcl.colors(n, 'Blue-Red 3'))
```

`predict.deform` *Predict from a fitted* `deform` *object*

Description

Predict from a fitted `deform` object

Usage

```
## S3 method for class 'deform'
predict(object, newdata = NULL, ...)
```

Arguments

`object`	a fitted `deform` object
`newdata`	a 2-column matrix of x and y coordinates
`...`	currently just a placeholder

Value

A 2-column matrix of predicted x and y points for deformations and a (2 + q)-column matrix for q-dimensional expansions.

Examples

```
# fit a deformation model
data(solar)
m0 <- deform(solar$x, solar$z, solar$n)

# predict D-space points for original locations
predict(m0)



# predictions for one-dimensional expansion model with specified locations
# and standard error estimates
data(solar)
m1 <- expand(solar$x, solar$z, solar$n)
xvals <- seq(-123.3, -122.2, by = .1)
yvals <- seq(49, 49.4, by = .1)
xyvals <- expand.grid(xvals, yvals)
predict(m1, xyvals, se.fit = TRUE)
```

`simulate.deform` *Simulate from a fitted* `deform` *object*

Description

Simulate from a fitted `deform` object

Usage

```
## S3 method for class 'deform'
simulate(object, nsim = 1, seed = NULL, newdata = NULL, ...)
```

Arguments

`object`	a fitted `deform` object
`nsim`	an integer giving the number of simulations
`seed`	an integer giving the seed for simulations
`newdata`	a 2-column matrix of x and y coordinates
`...`	extra arguments to pass to `predict.deform()`

Value

Plots representing all one- or two-dimensional smooths

Examples

```
# deformations
data(solar)
m0 <- deform(solar$x, solar$z, solar$n)
# Gaussian process simulations based on fitted deformation model
simulate(m0)



# one-dimensional expansion model with five simulations and specified locations
data(solar)
m1 <- expand(solar$x, solar$z, solar$n)
xvals <- seq(-123.3, -122.25, by = .05)
yvals <- seq(49, 49.4, by = .05)
xyvals <- expand.grid(xvals, yvals)
simulate(m1, 5, newdata = xyvals)
```

solar — *Variance-covariance matrix for British Columbia solar radiation data*

Description

Variance-covariance matrix for British Columbia solar radiation data

Format

A `list` with three elements, which are:

x a 12-row 2-column matrix of longitude-latitude coordinates for 12 stations

z a 12-row 12-column variance-covariance matrix

n an integer giving the original sample size

Source

These data were kindly provided by Alexandra Schmidt. They were originally published in Hay (1983) and then used in Sampson and Guttorp's (1992) pioneering deformation paper.

Hay, J. E. (1983) Solar energy system design: The impact of mesoscale variations in solar radiation, Atmosphere-Ocean, 21:2, 138-157, doi:10.1080/07055900.1983.9649161

Sampson, P. D. and Guttorp, P. (1992) Nonparametric Estimation of Nonstationary Spatial Covariance Structure, Journal of the American Statistical Association, 87:417, 108-119, doi:10.1080/01621459.1992.10475181

variogram — *Plot the variogram for a fitted* `deform` *object*

Description

Plot the variogram for a fitted `deform` object

Usage

```
variogram(object, bins = 20, bin.function = "pretty", trim = 0, ...)
```

Arguments

`object`	a fitted `deform` object
`bins`	an integer specifying the number of bins for plotting
`bin.function`	a character specifying a function to use to calculate bins; defaults to `pretty()`
`trim`	a scalar in [0, 0.5], which is passed to `mean()` when calculating binned variogram estimates; defaults to 0
`...`	extra arguments to pass to `plot()`

Value

Plot of variogram

Examples

```
# deformations
data(solar)
m0 <- deform(solar$x, solar$z, solar$n)

# empirical versus model-based variogram estimates against distance,
# where distance is based on D-space
variogram(m0)
# which is the default with approximately 20 bins, i.e. variogram(m0, bins = 20)



# variogram for one-dimensional expansion without binning
data(solar)
m1 <- expand(solar$x, solar$z, solar$n)
variogram(m1, bins = 0)
```

Index