Package ‘INLAspacetime’

December 21, 2025

Type Package

Title Spatial and Spatio-Temporal Models using 'INLA'

Version 0.1.13

Description Prepare objects to implement models over spatial and spacetime domains with the 'INLA' package (<https://www.r-inla.org>). These objects contain data to for the 'cgeneric' interface in 'INLA', enabling fast parallel computations.
We implemented the spatial barrier model, see Bakka et. al. (2019) <doi:10.1016/j.spasta.2019.01.002>, and some of the spatio-temporal models proposed in Lindgren et. al. (2024) <https://raco.cat/index.php/SORT/article/view/428665>.
Details are provided in the available vignettes and from the URL bellow.

URL https://github.com/eliaskrainski/INLAspacetime

Additional_repositories https://inla.r-inla-download.org/R/testing

BugReports https://github.com/eliaskrainski/INLAspacetime/issues

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.3.3

NeedsCompilation yes

Depends R (>= 4.3), Matrix, fmesher, INLAtools (>= 0.0.5)

Imports graphics, grDevices, methods, stats, utils, sp, sf, terra

Suggests INLA (>= 24.02.09), inlabru (>= 2.10.1), knitr, ggplot2, rmarkdown, parallel, data.table, rnaturalearth, rnaturalearthdata, ggpubr, DOYPAColors, s2, lubridate, ggOceanMaps, spdep

VignetteBuilder knitr

BuildVignettes true

Author Elias Teixeira Krainski [cre, aut, cph] (ORCID: <https://orcid.org/0000-0002-7063-2615>),
Finn Lindgren [aut] (ORCID: <https://orcid.org/0000-0002-5833-2011>),
Haavard Rue [aut] (ORCID: <https://orcid.org/0000-0002-0222-1881>)

Maintainer Elias Teixeira Krainski <eliaskrainski@gmail.com>

Repository CRAN

Date/Publication 2025-12-21 06:50:02 UTC

Contents

`ar2cov` *Illustrative code to compute the covariance of the second order autoregression (AR2) model.*

Description

Computes the auto-covariance for given coefficients.

Usage

```
ar2cov(a1, a2, k = 30, useC = FALSE)
```

Arguments

a1	the first auto-regression coefficient.
a2	the second auto-regression coefficient.
k	maximum lag for evaluating the auto-correlation.
useC	just a test (to use C code).

Value

the autocorrelation as a vector or matrix, whenever `a1` or `a2` are scalar or vector.

Details

Let the second order auto-regression model defined as `x_t + a_1 x_{t-1} + a_2 x_{t-2} = w_t` where `w_t` ~ `N(0, 1)`.

See Also

ar2precision

Examples

```
ar2cov(c(-1.7, -1.8), 0.963, k = 5)
plot(ar2cov(-1.7, 0.963), type = "o")
```

---

ar2precision — *Precision matrix for a stationary AR2 model.*

Description

Creates a precision matrix as a sparse matrix object considering the specification stated in Details.

Usage

```
ar2precision(n, a)
```

Arguments

n	integer with the size of the precision matrix.
a	numeric vector with length three with the coefficients.

Details

Let the second order auto-regression model be defined as

$$a_0 x_t + a_1 x_{t-1} + a_2 x_{t-2} = w_t, w_t\ N(0, 1).$$

The stationary assumption is to consider the variance of x_t to be the same for all $t = 1, \ldots, n$. This assumption gives the $n \times n$ symmetric precision matrix Q as a sparse matrix with the following non-zero elements:

$Q_{1,1} = Q_{n,n} = a_0^2$
$Q_{2,2} = Q_{n-1,n-1} = a_0^2 + a_1^2$
$Q_{1,2} = Q_{2,1} = Q_{n-1,n} = Q_{n,n-1} = a_0 a_1$
$Q_{t,t} = q_0 = a_0^2 + a_1^2 + a_2^2, t = 3, 4, ..., n-2$
$Q_{t,t-1} = Q_{t-1,t} = q_1 = a_1(a_0 + a_2), t = 3, 4, ..., n-1$
$Q_{t,t-2} = Q_{t-2,t} = q_2 = a_2 a_0, t = 3, 4, ..., n$

Value

sparse matrix.

See Also

ar2cov

Examples

```
ar2precision(7, c(1, -1.5, 0.9))
```

---

barrierModel.define — *Define a spacetime model object for the `f()` call.*

Description

Define a spacetime model object for the `f()` call.

Usage

```
barrierModel.define(
  mesh,
  barrier.triangles,
  prior.range,
  prior.sigma,
  range.fraction = 0.1,
  constr = FALSE,
  debug = FALSE,
  useINLAprecomp = TRUE,
  libpath = NULL
)
```

Arguments

`mesh`	a spatial mesh
`barrier.triangles`	a integer vector to specify which triangles centers are in the barrier domain, or a list with integer vector if more than one.
`prior.range`	numeric vector containing U and a to define the probability statements P(range < U) = a used to setup the PC-prior for range. If a = 0 or a = NA, then U is taken to be the fixed value for the range.
`prior.sigma`	numeric vector containing U and a to define the probability statements P(range > U) = a used to setup the PC-prior for sigma. If a = 0 or a = NA, then U is taken to be the fixed value for sigma.
`range.fraction`	numeric to specify the fraction of the range for the barrier domain. Default value is 0.1. This has to be specified with care in order to have it small enough to make it act as barrier but not too small in order to prevent numerical issues.
`constr`	logical, default is FALSE, to indicate if the integral of the field over the domain is to be constrained to zero.
`debug`	integer, default is zero, indicating the verbose level. Will be used as logical by INLA.
`useINLAprecomp`	logical, default is TRUE, indicating if it is to be used the shared object pre-compiled by INLA. This is not considered if 'libpath' is provided.
`libpath`	string, default is NULL, with the path to the shared object.

Details

See the paper.

Value

objects to be used in the f() formula term in INLA.

`bru_get_mapper.stModel_cgeneric`

Mapper object for automatic inlabru interface

Description

Return an `inlabru` `bru_mapper` object that can be used for computing model matrices for the space-time model components. The `bru_get_mapper()` function is called by the `inlabru` methods to automatically obtain the needed mapper object (from `inlabru 2.7.0.9001`; before that, use `mapper = bru_get_mapper(model)` explicitly).

Usage

```
## S3 method for class 'stModel_cgeneric'
bru_get_mapper(model, ...)
```

Arguments

model	The model object (of class stModel_cgeneric, from stModel.define or barrierModel_cgeneric, from barrierModel.define)
...	Unused.

Value

A bru_mapper object of class bru_mapper_multi with sub-mappers space and time based on the model smesh and tmesh or mesh objects.

See Also

inlabru::bru_get_mapper()

---

cgeneric_sspde — *Define the stationary SPDE cgeneric model for INLA.*

Description

Define the stationary SPDE cgeneric model for INLA.

Usage

```
cgeneric_sspde(mesh, alpha, control.priors, constr = FALSE, ...)
```

Arguments

mesh	triangulation mesh to discretize the model.
alpha	integer used to compute the smoothness parameter.
control.priors	named list with parameter priors. This shall contain prange and psigma each one as a length two vector with (U, a) to define the PC-prior parameters such that P(range<U)=a and P(sigma>U)=a, respectively. See Fuglstad et. al. (2019) <DOI: 10.1080/01621459.2017.1415907>. If a=0 or a=NA, then U is taken to be the fixed value of the parameter.
constr	logical, default is FALSE, to indicate if the integral of the field over the domain is to be constrained to zero.
...	additional arguments that will be passed on to INLAtools::cgenericBuilder(), such as: debug : logical/integer, default is FALSE/0. useINLAprecomp logical, default is TRUE, indicating if it is to be used the shared object pre-compiled by INLA.

Value

objects to be used in the f() formula term in INLA.

Note

This is the stationary case of `INLA::inla.spde2.pcmatern()` with slight change on the marginal variance when the domain is the sphere, following Eq. (23) in Lindgren et. al. (2024).

References

Geir-Arne Fuglstad, Daniel Simpson, Finn Lindgren & Håvard Rue (2019). Constructing Priors that Penalize the Complexity of Gaussian Random Fields. Journal of the American Statistical Association, V. 114, Issue 525.

Finn Lindgren, Haakon Bakka, David Bolin, Elias Krainski and Håvard Rue (2024). A diffusion-based spatio-temporal extension of Gaussian Matérn fields. SORT vol. 48, no. 1, pp. 3-66 <doi: 10.57645/20.8080.02.13>

`cWhittleMatern` *Computes the Whittle-Matern correlation function.*

Description

This computes the correlation function as derived in Matern model, see Matern (1960) eq. (2.4.7). For nu=1, see Whittle (1954) eq. (68). For the limiting case of nu=0, see Besag (1981) eq. (14-15).

Usage

```
cWhittleMatern(x, range, nu, kappa = sqrt(8 * nu)/range)
```

Arguments

`x`	distance.
`range`	practical range (our prefered parametrization) given as range = sqrt(8 * nu) / kappa, where kappa is the scale parameter in the specialized references.
`nu`	process smoothness parameter.
`kappa`	scale parameter, commonly considered in the specialized literature.

Value

the correlation.

Details

Whittle, P. (1954) On Stationary Processes in the Plane. Biometrika, Vol. 41, No. 3/4, pp. 434-449. http://www.jstor.org/stable/2332724

Matern, B. (1960) Spatial Variation: Stochastic models and their application to some problems in forest surveys and other sampling investigations. PhD Thesis.

Besag, J. (1981) On a System of Two-Dimensional Recurrence Equations. JRSS-B, Vol. 43 No. 3, pp. 302-309. https://www.jstor.org/stable/2984940

Examples

```
plot(function(x) cWhittleMatern(x, 1, 5),
  bty = "n", las = 1,
  xlab = "Distance", ylab = "Correlation"
)
plot(function(x) cWhittleMatern(x, 1, 1), add = TRUE, lty = 2)
plot(function(x) cWhittleMatern(x, 1, 0.5), add = TRUE, lty = 3)
abline(h = 0.139, lty = 3, col = gray(0.5, 0.5))
```

---

downloadUtilFiles *Download files from the NOAA's GHCN daily data*

Description

Download files from the NOAA's GHCN daily data

Usage

```
downloadUtilFiles(data.dir, year = 2022, force = FALSE)
```

Arguments

`data.dir`	the folder to store the files.
`year`	the year of the daily weather data.
`force`	logical indicating if it is to force the download. If FALSE each file will be downloaded if it does not exists locally yet.

Value

a named character vector with the local file names: daily.data, stations.all, elevation.

See Also

`ghcndSelect()`

Earth_poly — *Function to define the boundary Earch polygon in longlat projection for a given resolution.*

Description

Function to define the boundary Earch polygon in longlat projection for a given resolution.

Usage

```
Earth_poly(resol = 300, crs = "+proj=moll +units=km")
```

Arguments

`resol`	is the number of subdivisions along the latitude coordinates and half the number of subdivisions along the longitude coordinates.
`crs`	a string with the projection. Default is the Mollweide projection with units in kilometers.

Value

a 'st_sfc' object with the Earth polygon.

ghcndSelect — *Select data from the daily dataset*

Description

Select data from the daily dataset

Usage

```
ghcndSelect(
  gzfile,
  variable = c("TMIN", "TAVG", "TMAX"),
  station = NULL,
  qflag = "",
  verbose = TRUE,
  astype = as.integer
)
```

Arguments

`gzfile`	the local filename for the daily data file file. E.g. 2023.csv.gz from the daily GHCN data repository at NCEI-NOAA, at "https://www.ncei.noaa.gov/pub/data/ghcn/daily/by_year/". Please see the references bellow.
`variable`	string with the variable name(s) to be selected
`station`	string (vector) with the station(s) to be selected
`qflag`	a string with quality control flag(s)
`verbose`	logical indicating if progress is to be printed
`astype`	function to convert data to a class, default is set to convert the data to integer.

Value

if more than one variable, it returns an array whose dimentions are days, stations, variables. If one variable, then it returns a matrix whose dimentions are days, stations.

Details

The default selects TMIN, TAVG and TMAX and return it as integer because the original data is also integer with units in 10 Celcius degrees.

Warning

It can take time to execute if, for example, the data.table package is not available.

References

Menne, M., Durre, I., Vose, R., Gleason, B. and Houston, T. (2012) An overview of the global historical climatology network-daily database. Journal of Atmospheric and Oceanic Technology, 897–910.

`Heron` *Internal util functions for polygon properties.*

Description

This computes the area of a triangle given its three coordinates.

Usage

```
Heron(x, y)

Area(x, y)

s2trArea(tr, R = 1)

flatArea(tr)

Stiffness(tr)
```

Arguments

`x, y`	coordinate vectors.
`tr`	the triangle coordinates
`R`	the radius of the spherical domain

Details

Function used internally to compute the area of a triangle.

Value

the area of a 2d triangle

the area of a 2d polygon

the area of a triangle in S2

the area of a triangle

the stiffness matrix for a triangle

Warning

Internal functions, not exported.

`INLAspacetime` *Spatial and Spatio-Temporal Models using INLA*

Description

This package main purpose is to provide user friendly functions to fit temporal, spatial and space-time models using the INLA software available at www.r-inla.org as well the inlabru package available

Usage

```
INLAspacetime()
```

Value

opens the Vignettes directory on a browser

Jmatrices — *The 2nd order temporal matrices with boundary correction*

Description

The 2nd order temporal matrices with boundary correction

Usage

```
Jmatrices(tmesh)
```

Arguments

tmesh	Temporal mesh

Details

Temporal GMRF representation with stationary boundary conditions as in Appendix E in Lindgren et. al. (2024).

Value

return a list of temporal finite element method matrices for the supplied mesh.

mesh.dual — *Extracts the dual of a mesh object.*

Description

Extracts the dual of a mesh object.

Usage

```
mesh.dual(
  mesh,
  returnclass = c("list", "sf", "sv", "SP"),
  mc.cores = getOption("mc.cores", 2L)
)
```

Arguments

mesh	a 2d mesh object.
returnclass	if 'list' return a list of polygon coordinates, if "sf" return a 'sf' sfc_multipolygon object, if "sv" return a 'terra', SpatVector object, if "SP" return a 'sp' SpatialPolygons object.
mc.cores	number of threads to be used.

Value

one of the three in 'returnclass'

`mesh2d` *Illustrative code for building a mesh in 2d domain.*

Description

Creates a mesh object. This is just a test code. For efficient, reliable and general code use the **fmesher** package.

Usage

```
mesh2d(loc, domain, max.edge, offset, SP = TRUE)
```

Arguments

`loc`	a two column matrix with location coordinates.
`domain`	a two column matrix defining the domain.
`max.edge`	the maximum edge length.
`offset`	the length of the outer extension.
`SP`	logical indicating if the output will include the SpatialPolygons.

Value

a mesh object.

Warning

This is just for illustration purposes and one should consider the efficient function `fmesher::fm_mesh_2d()` (and other related functions) available a the fmesher package.

`mesh2fem` *Illustrative code for Finite Element matrices of a mesh in 2d domain.*

Description

Illustrative code for Finite Element matrices of a mesh in 2d domain.

Illustrative code for Finite Element matrices when some triangles are in a barrier domain.

Usage

```
mesh2fem(mesh, order = 2, barrier.triangles = NULL)

mesh2fem.barrier(mesh, barrier.triangles = NULL)
```

Arguments

`mesh`	a 2d mesh object.
`order`	the desired order.
`barrier.triangles`	integer index to specify the triangles in the barrier domain

Value

a list object containing the FE matrices.

a list object containing the FE matrices for the barrier problem.

`mesh2projector` *Illustrative code to build the projector matrix for SPDE models.*

Description

Creates a projector matrix object.

Usage

```
mesh2projector(
  mesh,
  loc = NULL,
  lattice = NULL,
  xlim = NULL,
  ylim = NULL,
  dims = c(100, 100)
)
```

Arguments

`mesh`	a 2d mesh object.
`loc`	a two columns matrix with the locations to project for.
`lattice`	Unused; feature not supported by this illustration.
`xlim`, `ylim`	vector with the boundary limits.
`dims`	the number of subdivisions over each boundary limits.

Value

the projector matrix as a list with sparse matrix object at `x$proj$A`..

Warning

This is just for illustration purpose and one should consider the efficient functions available in the fmesher package, e.g. `fmesher::fm_evaluator()` and `fmesher::fm_basis()`.

`outDetect` *Detect outliers in a time series considering the raw data and a smoothed version of it.*

Description

Detect outliers in a time series considering the raw data and a smoothed version of it.

Usage

```
outDetect(x, weights = NULL, ff = c(7, 7))
```

Arguments

`x`	numeric vector
`weights`	non-increasing numeric vector used as weights for computing a smoothed vector as a rooling window average. Default is null and then w_j is proportional to j in the equation in the Details below.
`ff`	numeric length two vector with the factors used to consider how many times the standard deviation one data point is out to be considered as an outlier.

Value

logical vector indicating if the data is an outlier with attributes as detailed bellow.

- attr(, 'm') is the mean of x.
- attr(, 's') is the standard devation of x.
- attr(, 'ss') is the standard deviation for the smoothed data y_t that is defined as

$y_t = \sum_{k=j}^{h} w_j * (x_{t-j} + x_{t+j})/2$

Both `s` and `ss` are used to define outliers if

$|x_t - m|/s > ff_1$ or $|x_t - y_t|/ss > ff_2$

- attr(, 'xs') the smoothed time series y_t

paramsUtils *User/internal parameters mapping functions*

Description

Functions to help converting from/to user/internal parametrization. The internal parameters are 'gamma_s, 'gamma_t', 'gamma_E' The user parameters are 'r_s', 'r_t', 'sigma'

Usage

```
lgsConstant(lg.s, alpha, smanifold)

params2gammas(
  lparams,
  alpha.t,
  alpha.s,
  alpha.e,
  smanifold = "R2",
  verbose = FALSE
)

gammas2params(lgammas, alpha.t, alpha.s, alpha.e, smanifold = "R2")
```

Arguments

`lg.s`	the logarithm of the SPDE parameter \gamma_s
`alpha`	the resulting spatial order.
`smanifold`	spatial domain manifold, which could be "S1", "S2", "R1", "R2" and "R3".
`lparams`	log(spatial range, temporal range, sigma)
`alpha.t`	temporal order of the SPDE
`alpha.s`	spatial order of the spatial differential operator in the non-separable part.
`alpha.e`	spatial order of the spatial differential operator in the separable part.
`verbose`	logical if it is to print internal variables
`lgammas`	numeric of length 3 with $log(\gamma_k)$ model parameters. The parameter order is log(gamma.s, gamma.t, gamma.e)

Details

See equation (23) in the paper.

See equations (19), (20) and (21) in the paper.

See equations (19), (20) and (21) in the paper.

Value

the part of `sigma` from the spatial constant and `\gamma_s`.

log(gamma.s, gamma.t, gamma.e)

log(spatial range, temporal range, sigma)

Examples

```
params2gammas(log(c(1, 1, 1)), 1, 2, 1, "R2")
gammas2params(log(c(0, 0, 0)), 1, 2, 1, "R2")
```

`pclrange` *Penalized Complexity (PC) prior for (log) range*

Description

Penalized Complexity (PC) prior for (log) range

Usage

```
pclrange(lrange, lam, d = 2, logdens = FALSE)

pcrange(range, lam, d = 2, logdens = FALSE)
```

Arguments

`lrange`	numeric with the log of the (practical) range
`lam`	numeric with the prior parameter
`d`	integer to specify the domain dimention
`logdens`	logical indicating if the density is to be returned in the log scale
`range`	numeric with the of the (practical) range

Examples

```
# P(range < 2.0) = 0.1
 lam <- -log(0.1) * 2.0
 plot(function(x) pcrange(x, lam), 1/100, 10, n = 100)
```

`spde2precision` *Illustrative code to build the precision matrix for SPDE kind models.*

Description

Creates a precision matrix as a sparse matrix object. For general code look at the functions in the INLA package.

Usage

```
spde2precision(kappa, fem, alpha)
```

Arguments

`kappa`	the scale parameter.
`fem`	a list containing the Finite Element matrices.
`alpha`	the smoothness parameter.

Value

the precision matrix as a sparse matrix object.

Warning

This is just for illustration purpose and one should consider the efficient function available a the INLA package.

`stats.inla` *To retrieve goodness of fit statistics for a specific model class.*

Description

Extracts dic, waic and log-cpo from an output returned by the inla function from the INLA package or by the bru function from the inlabru package, and computes log-po, mse, mae, crps and scrps for a given input. A summary is applied considering the user imputed function, which by default is the mean.

Usage

```
stats.inla(m, i = NULL, y, fsummarize = mean)
```

Arguments

`m`	an inla output object.
`i`	an index to subset the estimated values.
`y`	observed to compare against.
`fsummarize`	the summary function, the default is `base::mean()`.

Value

A named numeric vector with the extracted statistics.

Details

It assumes Gaussian posterior predictive distributions! Considering the defaults, for n observations, $y_i, i = 1, 2, ..., n$, we have

. dic

$$\sum_i d_i/n$$

where d_i is the dic computed for observation i.

. waic

$$\sum_i w_i/n$$

where w_i is the waic computed for observation i.

. lcpo

$$-\sum_i \log(p_i)/n$$

where p_i is the cpo computed for observation i.

For the log-po, crps, and scrps scores it assumes a Gaussian predictive distribution for each observation y_i which the following definitions: $z_i = (y_i - \mu_i)/\sigma_i$, μ_i is the posterior mean for the linear predictor, $\sigma_i = \sqrt{v_i + 1/\tau_y}$, τ_y is the observation posterior mean, v_i is the posterior variance of the linear predictor for y_i.

Then we consider $\phi()$ the density of a standard Gaussian variable and $\psi()$ the corresponding Cumulative Probability Distribution.

. lpo

$$-\sum_i \log(\phi(z_i))/n$$

. crps

$$\sum_i r_i/n$$

where

$$r_i = \sigma_i/\sqrt{\pi} - 2\sigma_i\phi(z_i) + (y_i - \mu_i)(1 - 2\psi(z_i))$$

. scrps

$$\sum_i s_i/n$$

where

$$s_i = -\log(2\sigma_i/\sqrt{\pi})/2 - \sqrt{\pi}(\phi(z_i) - \sigma_i z_i/2 + z_i\psi(z_i))$$

Warning

All the scores are negatively oriented which means that smaller scores are better.

References

Held, L. and Schrödle, B. and Rue, H. (2009). Posterior and Cross-validatory Predictive Checks: A Comparison of MCMC and INLA. Statistical Modelling and Regression Structures pp 91–110. https://link.springer.com/chapter/10.1007/978-3-7908-2413-1_6.

Bolin, D. and Wallin, J. (2022) Local scale invariance and robustness of proper scoring rules. Statistical Science. doi:10.1214/22STS864.

`stdSubs` *To check unusual low/high variance segments*

Description

To check unusual low/high variance segments

Usage

```
stdSubs(x, nsub = 12, fs = 15)
```

Arguments

`x`	numeric vector
`nsub`	number for the segments length
`fs`	numeric to use for detecting too hight or too low local standard deviations.

Value

logical indicating if any of the `st` are `fs` times lower/higher the average of `st`, where is returned as an attribute:

- attr(, 'st') numeric vector with the standard deviation at each segment of the data.

`stlines` *To visualize time series over space.*

Description

To visualize time series over space.

Usage

```
stlines(
  stdata,
  spatial,
  group = NULL,
  nmax.group = NULL,
  xscale = 1,
  yscale = 1,
  colour = NULL,
  ...
)

stpoints(
  stdata,
  spatial,
  group = NULL,
  nmax.group = NULL,
  xscale = 1,
  yscale = 1,
  colour = NULL,
  ...
)
```

Arguments

`stdata`	matrix with the data, each column is a location.
`spatial`	an object with one of class defined in the sp package.
`group`	an integer vector indicating to which spatial unit each time series belongs to. Default is NULL and them it is assumed that each time series belongs o each spatial unit.
`nmax.group`	an integer indicating the maximum number of time series to be plotted over each spatial unit. Default is NULL, so all will be drawn.
`xscale`	numeric to define a scaling factor in the horizontal direction.
`yscale`	numeric to define a scaling factor in the vertical direction.
`colour`	color (may be a vector, one for each time series). Default is NULL and it will generate colors considering the average of each time series. These automatic colors are defined using the `rgb()` function with `alpha=0.5`. It considers the relative rank of each time series mean, `r`. `r` is then used for red, `1-r` is used for blue and a triangular function, `1-2*|1-r/2|`, is considered for green. That is, time series with mean among the lowest time series averages are shown in blue and those among the highest temperatures are shown in red. The transition from blue to red goes so that the intermediate ones are shown in light green.
`...`	further arguments to be passed for the lines function.

Details

Scaling the times series is needed before drawing it over the map. The area of the bounding box for the spatial object divided by the number of locations is the standard scaling factor. This is further multiplied by the user given `xcale` and `yscale`.

Value

add lines to an existing plot

Functions

- `stlines()`: each time series over the map centered at the location.
- `stpoints()`: each time series over the map centered at the location.

Warning

if there are too many geographical locations, it will not look good

---

`stModel.define` *Define a spacetime model object for the `f()` call.*

Description

Define a spacetime model object for the `f()` call.

Usage

```
stModel.define(
  smesh,
  tmesh,
  model,
  control.priors,
  constr = FALSE,
  debug = FALSE,
  useINLAprecomp = TRUE,
  libpath = NULL
)
```

Arguments

`smesh`	a spatial mesh
`tmesh`	a temporal mesh
`model`	a three characters string to specify the smoothness alpha (each one as integer) parameters. Currently it considers the `102`, `121`, `202` and `220` models.

`control.priors`	a named list with parameter priors, named as `prs`, `prt` and `psigma`, each one as a vector with length two containing (U, a) to define the corresponding PC-prior such that, respectively, P(range.spatial<U)=a, P(range.temporal<U)=a or P(sigma>U)=a. If a=0 or a=NA, then U is taken to be the fixed value of the parameter.
`constr`	logical, default is FALSE, to indicate if the integral of the field over the domain is to be constrained to zero.
`debug`	integer, default is zero, indicating the verbose level. Will be used as logical by INLA.
`useINLAprecomp`	logical, default is TRUE, indicating if it is to be used the shared object pre-compiled by INLA. This is not considered if 'libpath' is provided.
`libpath`	string, default is NULL, with the path to the shared object.

Details

This function compute the matrices for computing the precision matrix. These are each one of the Kronecker products in Theorem 4.1 of Lindgren et. al. (2024) computed with the stModel.matrices and the parameters are as in Eq (19-21). We use the log of these parameters internally.

Value

objects to be used in the f() formula term in INLA.

References

Finn Lindgren, Haakon Bakka, David Bolin, Elias Krainski and Håvard Rue (2024). A diffusion-based spatio-temporal extension of Gaussian Matérn fields. SORT vol. 48, no. 1, pp. 3-66 <doi: 10.57645/20.8080.02.13>

`stModel.matrices` *Define the spacetime model matrices.*

Description

This function computes all the matrices needed to build the precision matrix for spatio-temporal model, as in Lindgren et. al. (2024)

Usage

```
stModel.matrices(smesh, tmesh, model, constr = FALSE)
```

Arguments

smesh	a mesh object over the spatial domain.
tmesh	a mesh object over the time domain.
model	a string identifying the model. So far we have the following models: '102', '121', '202' and '220' models.
constr	logical to indicate if the integral of the field over the domain is to be constrained to zero. Default value is FALSE.

Details

See the paper for details.

Value

a list containing needed objects for model definition.

1. 'manifold' to spedify the a string with the model identification
2. a length three vector with the constants `c1`, `c2` and `c3`
3. the vector `d`
4. the matrix `T`
5. the model matrices `M_1`, ..., `M_m`

`stModel.precision` *Spacetime precision matrix.*

Description

To build the the precision matrix for a spacetime model given the temporal and the spatial meshes.

Usage

```
stModel.precision(smesh, tmesh, model, theta, verbose = FALSE)
```

Arguments

smesh	a mesh object over the spatial domain.
tmesh	a mesh object over the time domain.
model	a string identifying the model. So far we have the following models: '102', '121', '202' and '220' models.
theta	numeric vector of length three with $log(gamma_s, gamma_t, gamma_e)$.
verbose	logical to print intermediate objects.

Value

a (sparse) precision matrix, as in Lindgren et. al. (2024)

worldMap — *Helper functions to retrieve the world map, a world polygon, and create grid centers.*

Description

Retrieve the map of the countries

Usage

```
worldMap(
  crs = "+proj=moll +units=km",
  scale = "medium",
  returnclass = c("sf", "sv")
)
```

Arguments

`crs`	a string with the projection. Default is the Mollweide projection with units in kilometers.
`scale`	The scale of map to return. Please see the help of 'ne_countries' function from the 'rnaturalearth' package.
`returnclass`	A string determining the class of the spatial object to return. Please see the help of 'ne_countries' function from the 'rnaturalearth' package.

References

The land and ocean maps are obtained with the 'rnaturalearth' package.

world_grid — *Define a regular grid in 'Mollweide' projection, with units in kilometers.*

Description

Define a regular grid in 'Mollweide' projection, with units in kilometers.

Usage

```
world_grid(size = 50, domain)
```

Arguments

`size`	the (in kilometers) of the grid cells.
`domain`	if provided it should be an `sf` or `sfc` object. In this case, the grid cells with centers falling inside will be retrieved.

Value

a 'sf' points object with the centers of a grid set within Earth (and the supplied domain)

Index