Package ‘spatsurv’

March 31, 2021

Type Package
Title Bayesian Spatial Survival Analysis with Parametric Proportional Hazards Models
Version 1.6
Date 2021-03-26
Author Benjamin M. Taylor and Barry S. Rowlingson
Additional contributions
Ziyu Zheng
Maintainer Benjamin M. Taylor <benjamin.taylor.software@gmail.com>
License GPL-3
Imports survival, sp, spatstat.geom, spatstat.core, spatstat (>= 2.0-0), raster, iterators, RandomFields, fields, Matrix, stringr, rgeos, RColorBrewer, methods, lubridate
Suggests rgdal, rgl, OpenStreetMap
Encoding UTF-8
RoxygenNote 7.1.1
NeedsCompilation no
Depends R (>= 2.10)
Repository CRAN
Date/Publication 2021-03-31 16:30:03 UTC

R topics documented:

  spatsurv-package ................................................. 5
  .onAttach ......................................................... 10
  allocate ......................................................... 11
  alpha ......................................................... 12
  B .............................................................. 12
R topics documented:

basehazard                    .............................................. 13
basehazard.basehazardspec     ........................................... 13
baseHazST                     .............................................. 14
baselinehazard               ............................................... 15
baselinehazard_multiWay       .......................................... 16
betapriorGauss               ............................................... 17
blockDiag                    ............................................... 17
boxplotRisk                  ............................................... 18
Bspline.construct            .............................................. 18
BsplineHaz                   ............................................... 19
checkSurvivalData            .............................................. 20
circulant                   ............................................... 21
circulant.matrix            ............................................... 21
circulant.numeric           ............................................... 22
circulantij                 ............................................... 22
covmodel                   ............................................... 23
CSplot                     ............................................... 23
cumbasehazard               ............................................... 24
cumbasehazard.basehazardspec ........................................... 24
cumulativeBspline.construct ............................................. 25
densityquantile            ............................................... 25
densityquantile.basehazardspec ........................................... 26
densityquantile_PP          ............................................... 27
density_P                  ............................................... 27
derivindepGaussianprior       ........................................... 28
derivindepGaussianpriorST    ........................................... 28
derivpsplineprior            ............................................... 29
distinfo                   ............................................... 30
distinfo.basehazardspec     .............................................. 30
estimateY                  ............................................... 31
etapriorGauss              ............................................... 31
Et_PP                       ............................................... 32
EvalCov                     ............................................... 32
ExponentialCovFct           ............................................... 33
exponentialHaz              ............................................... 33
FFTgrid                    ............................................... 35
fixedpars                  ............................................... 35
fixmatrix                  ............................................... 36
fixParHaz                  ............................................... 36
frailtylag1                ............................................... 37
fs                         ............................................... 37
fstimes                   ............................................... 38
gamma2risk                 ............................................... 39
GammafromY                 ............................................... 39
GammaFromY_SPDE            ............................................... 40
gencens                    ............................................... 40
getBackground              ............................................... 41
getbb                      ............................................... 42
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>getBbasis</td>
<td>42</td>
</tr>
<tr>
<td>getcov</td>
<td>43</td>
</tr>
<tr>
<td>getgrid</td>
<td>43</td>
</tr>
<tr>
<td>getGrid</td>
<td>44</td>
</tr>
<tr>
<td>getleneta</td>
<td>44</td>
</tr>
<tr>
<td>getOptCellwidth</td>
<td>45</td>
</tr>
<tr>
<td>getparranges</td>
<td>45</td>
</tr>
<tr>
<td>getsurvdata</td>
<td>46</td>
</tr>
<tr>
<td>gompertzHaz</td>
<td>46</td>
</tr>
<tr>
<td>gradbasehazard</td>
<td>48</td>
</tr>
<tr>
<td>gradbasehazard.basehazardspec</td>
<td>48</td>
</tr>
<tr>
<td>gradcumbasehazard</td>
<td>49</td>
</tr>
<tr>
<td>gradcumbasehazard.basehazardspec</td>
<td>49</td>
</tr>
<tr>
<td>grid2spdf</td>
<td>50</td>
</tr>
<tr>
<td>grid2spix</td>
<td>50</td>
</tr>
<tr>
<td>grid2spts</td>
<td>51</td>
</tr>
<tr>
<td>gridY</td>
<td>51</td>
</tr>
<tr>
<td>gridY_polygonal</td>
<td>52</td>
</tr>
<tr>
<td>guess_t</td>
<td>52</td>
</tr>
<tr>
<td>hasNext</td>
<td>53</td>
</tr>
<tr>
<td>hasNext.iter</td>
<td>53</td>
</tr>
<tr>
<td>hazardexceedance</td>
<td>53</td>
</tr>
<tr>
<td>hazardpars</td>
<td>54</td>
</tr>
<tr>
<td>hazard_PP</td>
<td>55</td>
</tr>
<tr>
<td>hessbasehazard</td>
<td>55</td>
</tr>
<tr>
<td>hessbasehazard.basehazardspec</td>
<td>56</td>
</tr>
<tr>
<td>hesscumbasehazard</td>
<td>56</td>
</tr>
<tr>
<td>hesscumbasehazard.basehazardspec</td>
<td>57</td>
</tr>
<tr>
<td>imputationModel</td>
<td>57</td>
</tr>
<tr>
<td>Independent</td>
<td>58</td>
</tr>
<tr>
<td>indepGaussianprior</td>
<td>58</td>
</tr>
<tr>
<td>indepGaussianpriorST</td>
<td>59</td>
</tr>
<tr>
<td>inference.control</td>
<td>60</td>
</tr>
<tr>
<td>insert</td>
<td>61</td>
</tr>
<tr>
<td>invtransformweibull</td>
<td>62</td>
</tr>
<tr>
<td>is.burnin</td>
<td>62</td>
</tr>
<tr>
<td>is.retain</td>
<td>63</td>
</tr>
<tr>
<td>iteration</td>
<td>63</td>
</tr>
<tr>
<td>logPosterior</td>
<td>64</td>
</tr>
<tr>
<td>logPosterior_gridded</td>
<td>65</td>
</tr>
<tr>
<td>logPosterior_polygonal</td>
<td>66</td>
</tr>
<tr>
<td>logPosterior_SPDE</td>
<td>67</td>
</tr>
<tr>
<td>loop.mcmc</td>
<td>68</td>
</tr>
<tr>
<td>makehamHaz</td>
<td>69</td>
</tr>
<tr>
<td>maxlikparamPHsurv</td>
<td>70</td>
</tr>
<tr>
<td>MCE</td>
<td>71</td>
</tr>
<tr>
<td>mcmcLoop</td>
<td>71</td>
</tr>
<tr>
<td>mcmc pars</td>
<td>72</td>
</tr>
</tbody>
</table>
R topics documented:

mcmcPriors .......................................................... 72
mcmcProgressNone ................................................. 73
mcmcProgressPrint ................................................. 74
mcmcProgressTextBar .............................................. 74
midpts ................................................................. 75
multiWayHaz ........................................................ 75
neighLocs ............................................................ 76
nextStep ............................................................... 77
NonSpatialLogLikelihood_or_gradient ......................... 77
omegapriorGauss .................................................... 78
omegapriorGaussST .................................................. 79
optifix ............................................................... 79
plot.FFTgrid .......................................................... 80
plotsurv .............................................................. 81
polyadd ............................................................... 82
polymult .............................................................. 83
posteriorcov .......................................................... 83
predict.mcmcspatsurv ............................................ 83
print.mcmc ........................................................... 84
print.mcmcspatsurv ............................................... 85
print.mlspatsurv .................................................... 86
print.mlspatsurv .................................................... 86
print.textSummary ................................................... 87
priorposterior ........................................................ 88
proposalVariance .................................................... 89
proposalVariance_gridded ........................................ 90
proposalVariance_polygonal ..................................... 91
proposalVariance_SPDE .......................................... 92
PsplineHaz ........................................................... 93
psplineprior .......................................................... 94
psplineRWprior ........................................................ 95
QuadApprox .......................................................... 96
quantile.mcmcspatsurv ........................................... 96
quantile.mlspatsurv ............................................... 97
randompars .......................................................... 98
reconstruct.bs ....................................................... 98
reconstruct.bs.mcmcspatsurv ................................... 99
resetLoop ............................................................ 100
residuals.mcmcspatsurv ......................................... 101
rootWeibullHaz ....................................................... 102
setTxtProgressBar2 ............................................... 103
setupHazard .......................................................... 103
setupPrecMatStruct ............................................... 104
showGrid ............................................................ 105
simsurv ............................................................. 105
spatialpars ........................................................... 106
spatsurvVignette .................................................... 107
Description

An R package for spatially correlated parametric proportional hazards survival analysis.

Usage

spatsurv

Format

An object of class `logical` of length 1.
Details

Package: spatsurv
Type: Package
Title: Bayesian Spatial Survival Analysis with Parametric Proportional Hazards Models
Version: 1.6
Date: 2021-03-26
Author: Benjamin M. Taylor and Barry S. Rowlingson Additional contributions Ziyu Zheng
Maintainer: Benjamin M. Taylor <benjamin.taylor.software@gmail.com>
License: GPL-3
Imports: survival, sp, spatstat.geom, spatstat.core, spatstat (>= 2.0-0), raster, iterators, RandomFields, fields, Matrix, stringr, rgeos, RColorBrewer, methods, lubridate
Suggests: rgdal, rgl, OpenStreetMap
Encoding: UTF-8
RoxygenNote: 7.1.1
NeedsCompilation: no
Packaged: 2021-03-26 13:51:11 UTC; ben
Depends: R (>= 2.10)
Repository: CRAN
Date/Publication: 2020-04-26 07:00:02 UTC

Index of help topics:

.onAttach .onAttach function
B B function
Bspline.construct Bspline.construct function
BsplineHaz BsplineHaz function
CSplot CSplot function
Et_PP Et_PP function
EvalCov EvalCov function
ExponentialCovFct ExponentialCovFct function
FFTgrid FFTgrid function
GammaFromY_SPDE GammaFromY_SPDE function
GammafromY GammafromY function
Independent Independent function
MCE MCE function
NonSpatialLogLikelihood_or_gradient NonSpatialLogLikelihood_or_gradient function
PsplineHaz PsplineHaz function
QuadApprox QuadApprox function
SPDE SPDE function
SPDEprec SPDEprec function
SpikedExponentialCovFct SpikedExponentialCovFct function
Summarise Summarise function
TwoWayHazAdditive TwoWayHazAdditive function
YFromGamma_SPDE YFromGamma_SPDE function
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YfromGamma</td>
<td>YfromGamma function</td>
</tr>
<tr>
<td>allocate</td>
<td>allocate function</td>
</tr>
<tr>
<td>alpha</td>
<td>alpha function</td>
</tr>
<tr>
<td>baseHazST</td>
<td>baseHazST function</td>
</tr>
<tr>
<td>basehazard</td>
<td>basehazard function</td>
</tr>
<tr>
<td>basehazard.basehazardspec</td>
<td>basehazard.basehazardspec function</td>
</tr>
<tr>
<td>baselinehazard</td>
<td>baselinehazard function</td>
</tr>
<tr>
<td>baselinehazard_multiWay</td>
<td>baselinehazard_multiWay function</td>
</tr>
<tr>
<td>betapriorGauss</td>
<td>betapriorGauss function</td>
</tr>
<tr>
<td>blockDiag</td>
<td>A function to</td>
</tr>
<tr>
<td>boxplotRisk</td>
<td>boxplotRisk function</td>
</tr>
<tr>
<td>checkSurvivalData</td>
<td>checkSurvivalData function</td>
</tr>
<tr>
<td>circulant</td>
<td>circulant function</td>
</tr>
<tr>
<td>circulant.matrix</td>
<td>circulant.matrix function</td>
</tr>
<tr>
<td>circulant.numeric</td>
<td>circulant.numeric function</td>
</tr>
<tr>
<td>circulantij</td>
<td>circulantij function</td>
</tr>
<tr>
<td>covmodel</td>
<td>covmodel function</td>
</tr>
<tr>
<td>cumbasehazard</td>
<td>cumbasehazard function</td>
</tr>
<tr>
<td>cumbasehazard.basehazardspec</td>
<td>cumbasehazard.basehazardspec function</td>
</tr>
<tr>
<td>cumulativeBspline.construct</td>
<td>cumulativeBspline.construct function</td>
</tr>
<tr>
<td>density_PP</td>
<td>density_PP function</td>
</tr>
<tr>
<td>densityquantile</td>
<td>densityquantile function</td>
</tr>
<tr>
<td>densityquantile.basehazardspec</td>
<td>densityquantile.basehazardspec function</td>
</tr>
<tr>
<td>densityquantile_PP</td>
<td>densityquantile_PP function</td>
</tr>
<tr>
<td>derivindepGaussianprior</td>
<td>derivindepGaussianprior function</td>
</tr>
<tr>
<td>derivindepGaussianpriorST</td>
<td>derivindepGaussianpriorST function</td>
</tr>
<tr>
<td>derivpsplineprior</td>
<td>derivpsplineprior function</td>
</tr>
<tr>
<td>distinfo</td>
<td>distinfo function</td>
</tr>
<tr>
<td>distinfo.basehazardspec</td>
<td>distinfo.basehazardspec function</td>
</tr>
<tr>
<td>estimateY</td>
<td>estimateY function</td>
</tr>
<tr>
<td>etapriorGauss</td>
<td>etapriorGauss function</td>
</tr>
<tr>
<td>exponentialHaz</td>
<td>exponentialHaz function</td>
</tr>
<tr>
<td>fixParHaz</td>
<td>fixParHaz function</td>
</tr>
<tr>
<td>fixedpars</td>
<td>fixedpars function</td>
</tr>
<tr>
<td>fixmatrix</td>
<td>fixmatrix function</td>
</tr>
<tr>
<td>frailtylag1</td>
<td>frailtylag1 function</td>
</tr>
<tr>
<td>fs</td>
<td>London Fire Brigade property</td>
</tr>
<tr>
<td>fstimes</td>
<td>London Fire Brigade response times to dwelling fires, 2009</td>
</tr>
<tr>
<td>gamma2risk</td>
<td>gamma2risk function</td>
</tr>
</tbody>
</table>
gencens  gencens function
getBackground  getBackground function
getBbasis  getBbasis function
getGrid  getGrid function
getOptCellwidth  getOptCellwidth function
getbb  getbb function
getcov  getcov function
getgrd  getgrd function
getleneta  getleneta function
getparranges  getparranges function
getsurvdata  getsurvdata function
gompertzHaz  gompertzHaz function
gradbasehazard  gradbasehazard function
gradbasehazard.basehazardspec
  gradbasehazard.basehazardspec function
gradcumbasehazard  gradcumbasehazard function
gradcumbasehazard.basehazardspec
  gradcumbasehazard.basehazardspec function
grid2spdf  grid2spdf function
grid2spix  grid2spix function
grid2spts  grid2spts function
gridY  gridY function
gridY_polygonal  gridY_polygonal function
guess_t  guess_t function
hasNext  generic hasNext method
hasNext.iter  hasNext.iter function
hazard_PP  hazard_PP function
hazardexceedance  hazardexceedance function
hazardpars  hazardpars function
hessbasehazard  hessbasehazard function
hessbasehazard.basehazardspec
  hessbasehazard.basehazardspec function
hesscumbasehazard  hesscumbasehazard function
hesscumbasehazard.basehazardspec
  hesscumbasehazard.basehazardspec function
imputationModel  imputationModel function
indepGaussianprior  indepGaussianprior function
indepGaussianpriorST  indepGaussianpriorST function
inference.control  inference.control function
insert  insert function
invtransformweibull  invtransformweibull function
is.burnin  is this a burn-in iteration?
is.retain  do we retain this iteration?
iteration  iteration number
logPosterior  logPosterior function
logPosterior_SPDE  logPosterior_SPDE function
logPosterior_gridded  logPosterior_gridded function
logPosterior_polygonal
spatsurv-package

logPosterior_polygonal function
loop.mcmc loop over an iterator
makehamHaz makehamHaz function
maxlikparamPHsurv maxlikparamPHsurv function
mcmcLoop iterator for MCMC loops
mcmcPriors mcmcPriors function
mcmcProgressNone null progress monitor
mcmcProgressPrint printing progress monitor
mcmcProgressTextBar text bar progress monitor
mcmcpars mcmcpars function
midpts midpts function
multiWayHaz multiWayHaz function
neighLocs neighLocs function
neighOrder neighOrder function
nextStep next step of an MCMC chain
omegapriorGauss omegapriorGauss function
omegapriorGaussST omegapriorGaussST function
optifix optifix function
plot.FFTgrid plot.FFTgrid function
plotsurv plotsurv function
polyadd polyadd function
polymult polymult function
posteriorcov posteriorcov function
predict.mcmcspatsurv predict.mcmcspatsurv function
print.mcmc print.mcmc function
print.mcmcspatsurv print.mcmcspatsurv function
print.mlspatsurv print.mlspatsurv function
print.textSummary print.textSummary function
priorposterior priorposterior function
proposalVariance proposalVariance function
proposalVariance_SPDE proposalVariance_SPDE function
proposalVariance_gridded proposalVariance_gridded function
proposalVariance_polygonal proposalVariance_polygonal function
psplineRWprior psplineRWprior function
psplineprior psplineprior function
quantile.mcmcspatsurv quantile.mcmcspatsurv function
quantile.mlspatsurv quantile.mlspatsurv function
randompars randompars function
reconstruct.bs reconstruct.bs function
reconstruct.bs.coxph reconstruct.bs.coxph function
reconstruct.bs.mcmcspatsurv reconstruct.bs.mcmcspatsurv function
resetLoop reset iterator
residuals.mcmcspatsurv residuals.mcmcspatsurv function
rootWeibullHaz rootWeibullHaz function
The package `spatsurv` depends upon some other important contributions to CRAN in order to operate; their uses here are indicated:

- survival, sp, spatstat, raster, iterators, RandomFields, fields, rgl, Matrix, stringr, RColorBrewer, geostatsp, rgeos.

To cite use of `spatsurv`, the user may refer to the following work:


Author(s)

Benjamin Taylor, Health and Medicine, Lancaster University, Barry Rowlingson, Health and Medicine, Lancaster University
allocate

Description
A function to print a welcome message on loading package

Usage
.onAttach(libname, pkgname)

Arguments
libname libname argument
pkgname pkgname argument

Value
...

allocate allocate function

Description
A function to allocate coordinates to an observation whose spatial location is known to the regional level

Usage
allocate(poly, popden, survdat, pid, sid, n = 2, wid = 2000)

Arguments
poly a SpatialPolygonsDataFrame, on which the survival data exist in aggregate form
popden a sub-polygon raster image of population density
survdat data.frame containing the survival data
pid name of the variable in the survival data that gives the region identifier in poly
sid the name of the variable in poly to match the region identifier in survdat to
n the number of different allocations to make. e.g. if n is 2 (the default) two candidate sets of locations are available.
wid The default is 2000, interpreted in metres ie 2Km. size of buffer to add to window for raster cropping purposes: this ensures that for each polygon, the cropped raster covers it completely.

Value
matrices x and y, both of size (number of observations in survdat x n) giving n potential candidate locations of points in the columns of x and y.
**alpha**  

*alpha function*

---

**Description**

A function used in calculating the coefficients of a B-spline curve

**Usage**

\[ \text{alpha}(i, j, \text{knots}, \text{knotidx}) \]

**Arguments**

- \(i\): index \(i\)
- \(j\): index \(j\)
- \(\text{knots}\): knot vector
- \(\text{knotidx}\): knot index

**Value**

a vector

---

**B**  

*B function*

---

**Description**

A recursive function used in calculating the coefficients of a B-spline curve

**Usage**

\[ \text{B}(x, i, j, \text{knots}) \]

**Arguments**

- \(x\): locations at which to evaluate the B-spline
- \(i\): index \(i\)
- \(j\): index \(j\)
- \(\text{knots}\): a knot vector

**Value**

a vector of polynomial coefficients
basehazard

Description
Generic function for computing the baseline hazard

Usage
basehazard(obj, ...)

Arguments

obj an object

... additional arguments – currently there are none, but this is for extensibility

Value

method basehazard

See Also

basehazard.basehazardspec, exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

basehazard.basehazardspec

Description
A function to retrieve the baseline hazard function

Usage

## S3 method for class 'basehazardspec'
basehazard(obj, ...)

Arguments

obj an object of class basehazardspec

... additional arguments – currently there are none, but this is for extensibility

Value

a function returning the baseline hazard
See Also

exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

---

**baseHazST**  
*baseHazST function*

**Description**

A function to

**Usage**

```r
baseHazST(
  bh1 = NULL,
  survobj,
  t0,
  nbbreaks = 5,
  breakmethod = "quantile",
  MLinits = NULL
)
```

**Arguments**

- `bh1`  
- `survobj`  
- `t0`  
- `nbbreaks`  
- `breakmethod`  
- `MLinits`  

**Value**

...
baselinehazard

Description

A function to compute quantiles of the posterior baseline hazard or cumulative baseline hazard.

Usage

baselinehazard(
  x,
  t = NULL,
  n = 100,
  probs = c(0.025, 0.5, 0.975),
  cumulative = FALSE,
  plot = TRUE,
  bw = FALSE,
  ...
)

Arguments

x an object inheriting class mcmcspatsurv

t optional vector of times at which to compute the quantiles, Default is NULL, in which case a uniformly spaced vector of length n from 0 to the maximum time is used

n the number of points at which to compute the quantiles if t is NULL

probs vector of probabilities

cumulative logical, whether to return the baseline hazard (default i.e. FALSE) or cumulative baseline hazard

plot whether to plot the result

bw Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.

... additional arguments to be passed to plot

Value

the vector of times and quantiles of the baseline or cumulative baseline hazard at those times

See Also

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, predict.mcmcspatsurv, priorposterior, posteriororcov, MCE, hazardexceedance
Baseline hazard multiWay

Description

A function to

Usage

baselinehazard_multiWay(
  x,
  probs = c(0.025, 0.5, 0.975),
  cumulative = FALSE,
  plot = TRUE,
  joint = FALSE,
  xlims = NULL,
  ylims = NULL,
  ...
)

Arguments

x X
probs X
cumulative X
plot X
joint X
xlims X
ylims X
...
X

Value

...
**betapriorGauss**

**Description**
A function to define Gaussian priors for beta. This function simply stores a vector of means and standard deviations to be passed to the main MCMC function, survspat.

**Usage**

```r
betapriorGauss(mean, sd)
```

**Arguments**

- `mean`:
  - the prior mean, a vector of length 1 or more. 1 implies a common mean.

- `sd`:
  - the prior standard deviation, a vector of length 1 or more. 1 implies a common standard deviation.

**Value**

an object of class "betapriorGauss"

**See Also**

- `survspat`, `betapriorGauss`, `omegapriorGauss`, `etapriorGauss`, `indepGaussianprior`, `derivindepGaussianprior`

---

**blockDiag**

**A function to**

**Description**
A function to

**Usage**

```r
blockDiag(matlist)
```

**Arguments**

- `matlist`:
  - X

**Value**

...
boxplotRisk

**Description**

A function to

**Usage**

`boxplotRisk(g2r)`

**Arguments**

- `g2r` X

**Value**

...

Bspline.construct

**Description**

A function to construct a B-spline basis matrix for given data and basis coefficients. Used in evaluating the baseline hazard.

**Usage**

`Bspline.construct(x, basis)`

**Arguments**

- `x` a vector, the data
- `basis` an object created by the `getBbasis` function

**Value**

a basis matrix
BsplineHaz function

Description

A function to define a parametric proportional hazards model where the baseline hazard is modelled by a basis spline. This function returns an object inheriting class 'basehazardspec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard', and 'densityquantile'.

Usage

BsplineHaz(times, knots = quantile(times), degree = 3, MLinits = NULL)

Arguments

times vector of survival times (both censored and uncensored)
knots vector of knots in ascending order, must include minimum and maximum values of 'times'
degree degree of the spline basis, default is 3
MLinits optional starting values for the non-spatial maximisation routine using optim. Note that we are working with the log of the parameters. Default is -10 for each parameter.

Details

The distinfo function is used to provide basic distribution specific information to other spatsurv functions. The user is required to provide the following information in the returned list: npars, the number of parameters in this distribution; parnames, the names of the parameters; trans, the transformation scale on which the priors will be provided; itrans, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; jacobian, the derivative of the inverse transformation function with respect to each of the parameters; and hessian, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package spatsurv only allows the use of functions where the parameters are transformed independently.

The basehazard function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a vector.

The gradbasehazard function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.

The hessbasehazard function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.

The cumbasehazard function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a vector.
The `gradcumbasehazard` function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, \( t \), and returns a matrix.

The `hesscumbasehazard` function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, \( t \) and returns a list of hessian matrices corresponding to each \( t \).

The `densityquantile` function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the `predict` function where `type` is 'densityquantile'. In the case of the Weibull model for the baseline hazard, it can be shown that the q-th quantile is:

\[
\text{Value}
\]

an object inheriting class 'basehazardspec'

See Also

`exponentialHaz`, `gompertzHaz`, `makehamHaz`, `weibullHaz`

---

**Description**

A function to check whether the survival data to be passed to survspat is in the correct format

**Usage**

`checkSurvivalData(s)`

**Arguments**

\( s \) an object of class Surv, from the survival package

**Value**

if there are any issues with data format, these are returned with the data an error message explaining any issues with the data
**circulant**  

circulant function

**Description**

generic function for constructing circulant matrices

**Usage**

circulant(x, ...)

**Arguments**

- `x` an object
- `...` additional arguments

**Value**

method circulant

---

**circulant.matrix**  
circulant.matrix function

---

**Description**

If `x` is a matrix whose columns are the bases of the sub-blocks of a block circulant matrix, then this function returns the block circulant matrix of interest.

**Usage**

```r
## S3 method for class 'matrix'
circulant(x, ...)
```

**Arguments**

- `x` a matrix object
- `...` additional arguments

**Value**

If `x` is a matrix whose columns are the bases of the sub-blocks of a block circulant matrix, then this function returns the block circulant matrix of interest.
## circulant.numeric

**Description**

returns a circulant matrix with base x

**Usage**

```r
## S3 method for class 'numeric'
circulant(x, ...)
```

**Arguments**

- `x` an numeric object
- `...` additional arguments

**Value**

a circulant matrix with base x

---

## circulantij

**Description**

A function to return the "idx" i.e. c(i,j) element of a circulant matrix with base "base".

**Usage**

```r
circulantij(idx, base)
```

**Arguments**

- `idx` vector of length 2 th (i,j) (row,column) index to return
- `base` the base matrix of a circulant matrix

**Value**

the ij element of the full circulant
covmodel

covmodel function

Description
A function to define the spatial covariance model, see also ?CovarianceFct. Note that the parameters defined by the 'pars' argument are fixed, i.e. not estimated by the MCMC algorithm. To have spatsurv estimate these parameters, the user must construct a new covariance function to do so, see the spatsurv vignette.

Usage
covmodel(model, pars)

Arguments

  model  correlation type, a string see ?CovarianceFct
  pars   vector of additional parameters for certain classes of covariance function (eg Matern), these must be supplied in the order given in ?CovarianceFct and are not estimated

Value
an object of class covmodel

See Also
CovarianceFct

CSplot

CSplot function

Description
A function to produce a diagnostic plot for model fit using the Cox-Snell residuals.

Usage
CSplot(mod, plot = TRUE, bw = FALSE, ...)

Arguments

  mod       an object produced by the function survspat
  plot      whether to plot the result, default is TRUE
  bw        Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.
  ...      other arguments to pass to plot
**cumbasehazard**

**cumbasehazard function**

---

**Description**

Generic function for computing the cumulative baseline hazard

**Usage**

`cumbasehazard(obj, ...)`

**Arguments**

- `obj`: an object
- `...`: additional arguments – currently there are none, but this is for extensibility

**Value**

method `cumbasehazard`

**See Also**

`cumbasehazard.basehazardspec`, `exponentialHaz`, `weibullHaz`, `gompertzHaz`, `makehamHaz`, `tpowHaz`

---

**cumbasehazard.basehazardspec**

**cumbasehazard.basehazardspec function**

---

**Description**

A function to retrieve the cumulative baseline hazard function

**Usage**

```
## S3 method for class 'basehazardspec'
cumbasehazard(obj, ...)
```

**Arguments**

- `obj`: an object of class `basehazardspec`
- `...`: additional arguments – currently there are none, but this is for extensibility

---

**Value**

the x and y values used in the plot
Value

a function returning the cumulative baseline hazard

See Also

exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

cumulativeBspline.construct

cumulativeBspline.construct function

Description

A function to construct the integral of a B-spline curve given data and basis coefficients. Used in evaluating the cumulative baseline hazard.

Usage

cumulativeBspline.construct(x, basis)

Arguments

x a vector, the data
basis an object created by the getBbasis function

Value

an object that allows the integral of a given B-spline curve to be computed

densityquantile densityquantile function

Description

Generic function for computing quantiles of the density function for a given baseline hazard. This may not be analytically tractable.

Usage

densityquantile(obj, ...)

Arguments

obj an object
... additional arguments – currently there are none, but this is for extensibility
Value

method densityquantile

See Also

densityquantile.basehazardspec, exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

densityquantile.basehazardspec

densityquantile.basehazardspec function

Description

A function to retrieve the quantiles of the density function

Usage

```r
## S3 method for class 'basehazardspec'
densityquantile(obj, ...)
```

Arguments

- `obj` an object of class basehazardspec
- `...` additional arguments – currently there are none, but this is for extensibility

Value

a function returning the density quantiles

See Also

exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz
densityquantile_PP  

**densityquantile_PP function**

**Description**

A function to compute quantiles of the density function

**Usage**

densityquantile_PP(inputs)

**Arguments**

- **inputs**: inputs for the function including the model matrix, frailties, fixed effects and the parameters of the baseline hazard derived from this model

**Value**

quantiles of the density function for the individual


density_PP  

**density_PP function**

**Description**

A function to compute an individual’s density function

**Usage**

density_PP(inputs)

**Arguments**

- **inputs**: inputs for the function including the model matrix, frailties, fixed effects and the parameters of the baseline hazard derived from this model

**Value**

the density function for the individual
derivindepGaussianprior

Description
A function for evaluating the first and second derivatives of the log of an independent Gaussian prior

Usage
derivindepGaussianprior(beta = NULL, omega = NULL, eta = NULL, priors)

Arguments
- beta: a vector, the parameter beta
- omega: a vector, the parameter omega
- eta: a vector, the parameter eta
- priors: an object of class `mcmcPrior`, see ?mcmcPrior

Value
returns the first and second derivatives of the prior

See Also
survspat, betapriorGauss, omegapriorGauss, etapriorGauss, indepGaussianprior, derivindepGaussianpriorST

derivindepGaussianpriorST

Description
A function to

Usage
derivindepGaussianpriorST(beta = NULL, omega = NULL, eta = NULL, priors)
**Arguments**

- beta X
- omega X
- eta X
- priors X

**Value**

...

---

**derivpsplineprior**  
*derivpsplineprior function*

**Description**

A function for evaluating the first and second derivatives of the log of an independent Gaussian prior

**Usage**

`derivpsplineprior(beta = NULL, omega = NULL, eta = NULL, priors)`

**Arguments**

- beta a vector, the parameter beta
- omega a vector, the parameter omega
- eta a vector, the parameter eta
- priors an object of class `mcmcPrior`, see `?mcmcPrior`

**Value**

returns the first and second derivatives of the prior

**See Also**

`survspat`, `betapriorGauss`, `omegapriorGauss`, `etapriorGauss`, `indepGaussianprior`, `derivindepGaussianprior`
distinfo

**Description**
Generic function for returning information about the class of baseline hazard functions employed.

**Usage**

```r
distinfo(obj, ...)
```

**Arguments**

- `obj` an object
- `...` additional argument – currently there are none, but this is for extensibility

**Value**

method distinfo

**See Also**

- `distinfo.basehazardspec`, `exponentialHaz`, `weibullHaz`, `gompertzHaz`, `makehamHaz`, `tpowHaz`

distinfo.basehazardspec

**Description**

A function to retrieve information on the baseline hazard distribution of choice

**Usage**

```r
## S3 method for class 'basehazardspec'
distinfo(obj, ...)
```

**Arguments**

- `obj` an object of class basehazardspec
- `...` additional arguments – currently there are none, but this is for extensibility

**Value**

a function returning information on the baseline hazard distribution of choice
**estimateY**

*estimateY function*

**Description**

A function to get an initial estimate of Y, to be used in calibrating the MCMC. Not for general use.

**Usage**

`estimateY(X, betahat, omegahat, surv, control)`

**Arguments**

- **X**: the design matrix containing covariate information.
- **betahat**: an estimate of beta.
- **omegahat**: an estimate of omega.
- **surv**: an object of class Surv.
- **control**: a list containing various control parameters for the MCMC and post-processing routines.

**Value**

An estimate of Y, to be used in calibrating the MCMC.

**etapriorGauss**

*etapriorGauss function*

**Description**

A function to define Gaussian priors for eta. This function simply stores a vector of means and standard deviations to be passed to the main MCMC function, survspat.

**Usage**

`etapriorGauss(mean, sd)`

**Arguments**

- **mean**: the prior mean, a vector of length 1 or more. 1 implies a common mean.
- **sd**: the prior standard deviation, a vector of length 1 or more. 1 implies a common standard deviation.
**EvalCov**

**Value**

an object of class "etapriorGauss"

**See Also**

survspat, betapriorGauss, omegapriorGauss, etapriorGauss, indepGaussianprior, derivindepGaussianprior

---

**Et_PP**

**Et_PP function**

**Description**

A function to compute an individual’s approximate expected survival time using numerical integration. Note this appears to be unstable; the function is based on R’s integrate function. Not intended for general use (yet!).

**Usage**

Et_PP(inputs)

**Arguments**

inputs inputs for the function including the model matrix, frailties, fixed effects and the parameters of the baseline hazard derived from this model

**Value**

the expected survival time for the individual, obtained by numerical integration of the density function.

---

**EvalCov**

**EvalCov function**

**Description**

This function is used to evaluate the covariance function within the MCMC run. Not intended for general use.

**Usage**

EvalCov(cov.model, u, parameters)
ExponentialCovFct

**Arguments**

- **cov.model** an object of class `covmodel`
- **u** vector of distances
- **parameters** vector of parameters

**Value**

method `EvalCov`

---

**ExponentialCovFct function**

**Description**

A function to declare and also evaluate an exponential covariance function.

**Usage**

```r
ExponentialCovFct()
```

**Value**

the exponential covariance function

**See Also**

`SpikedExponentialCovFct`, `covmodel`, `CovarianceFct`

---

**exponentialHaz function**

**Description**

A function to define a parametric proportional hazards model where the baseline hazard is taken from the exponential model. This function returns an object inheriting class 'basehazardspec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard' and 'densityquantile'

**Usage**

```r
exponentialHaz()
```
Details

The distinfo function is used to provide basic distribution specific information to other spatsurv functions. The user is required to provide the following information in the returned list: npars, the number of parameters in this distribution; parnames, the names of the parameters; trans, the transformation scale on which the priors will be provided; itrans, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; jacobian, the derivative of the inverse transformation function with respect to each of the parameters; and hessian, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package spatsurv only allows the use of functions where the parameters are transformed independently.

The basehazard function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, \( t \) and returns a vector.

The gradbasehazard function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, \( t \), and returns a matrix.

The hessbasehazard function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, \( t \) and returns a list of hessian matrices corresponding to each \( t \).

The cumbasehazard function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, \( t \) and returns a vector.

The gradcumbasehazard function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, \( t \), and returns a matrix.

The hesscumbasehazard function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, \( t \) and returns a list of hessian matrices corresponding to each \( t \).

The densityquantile function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the predict function where type is 'densityquantile'. In the case of the Weibull model for the baseline hazard, it can be shown that the q-th quantile is:

\[
\text{Value} \quad \text{an object inheriting class 'basehazardspec'}
\]

See Also

tpowHaz, gompertzHaz, makehamHaz, weibullHaz
**FFTgrid**

**FFTgrid function**

**Description**

A function to generate an FFT grid and associated quantities including cell dimensions, size of extended grid, centroids,

**Usage**

`FFTgrid(spatialdata, cellwidth, ext, boundingbox = NULL)`

**Arguments**

- `spatialdata` a SpatialPixelsDataFrame object
- `cellwidth` width of computational cells
- `ext` multiplying constant: the size of the extended grid: ext*M by ext*N
- `boundingbox` optional bounding box over which to construct computational grid, supplied as an object on which the function `bbox` returns the bounding box

**Value**

a list

**fixedpars**

**fixedpars function**

**Description**

A function to return the mcmc chains for the covariate effects

**Usage**

`fixedpars(x)`

**Arguments**

- `x` an object of class mcmcspatsurv

**Value**

the beta mcmc chains
See Also

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailty-lag1, spatialpars, hazardpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

---

### fixmatrix

#### Description

!! THIS FUNCTION IS NOT INTENDED FOR GENERAL USE !!

#### Usage

fixmatrix(mat)

#### Arguments

- **mat**: a matrix

#### Details

A function to fix up an estimated covariance matrix using a VERY ad-hoc method.

#### Value

the fixed matrix

---

### fixParHaz

#### Description

A function to

#### Usage

fixParHaz(bh, idx, fixval)

#### Arguments

- **bh**: XX
- **idx**: XX
- **fixval**: XX

#### Value

...
**frailtylag1**

**frailtylag1 function**

**Description**

A function to produce a plot of, and return, the lag 1 (or higher, see argument 'lag') autocorrelation for each of the spatially correlated frailty chains

**Usage**

```r
frailtylag1(object, plot = TRUE, lag = 1, ...)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>object</code></td>
<td>an object inheriting class mcmcspatsurv</td>
</tr>
<tr>
<td><code>plot</code></td>
<td>logical whether to plot the result, default is TRUE</td>
</tr>
<tr>
<td><code>lag</code></td>
<td>the lag to plot, the default is 1</td>
</tr>
<tr>
<td><code>...</code></td>
<td>other arguments to be passed to the plot function</td>
</tr>
</tbody>
</table>

**Value**

the lag 1 autocorrelation for each of the spatially correlated frailty chains

**See Also**

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

---

**fs**

**London Fire Brigade property**

**Description**

London Fire Brigade property

**Usage**

```r
data(fs)
```

**Format**

data.frame
Source

https://data.london.gov.uk/

References


Examples

fire <- data(fs)

---

**fstimes**

*London Fire Brigade response times to dwelling fires, 2009*

Description

London Fire Brigade response times to dwelling fires, 2009

Usage

data(fstimes)

Format

data.frame

Source

https://data.london.gov.uk/

References


Examples

firetimes <- data(fstimes)
gamma2risk function

Description
A function to

Usage
gamma2risk(mod)

Arguments
mod X

Value
...

GammafromY function

Description
A function to change Ys (spatially correlated noise) into Gammas (white noise). Used in the MALA algorithm.

Usage
GammafromY(Y, rootQeigs, mu)

Arguments
Y Y matrix
rootQeigs square root of the eigenvectors of the precision matrix
mu parameter of the latent Gaussian field

Value
Gamma
GammaFromY_SPDE  

Description

A function to go from Y to Gamma

Usage

GammaFromY_SPDE(Y, U, mu)

Arguments

Y  
U  upper Cholesky matrix
mu  the mean

Value

the value of Gamma for the given Y

References


gencens  

Description

A function to generate observed times given a vector of true survival times and a vector of censoring times. Used in the simulation of survival data.

Usage

gencens(survtimes, censtimes, type = "right")
**getBackground**

**Arguments**

- **survtimes**: a vector of survival times
- **censtimes**: a vector of censoring times for left or right censored data, 2-column matrix of censoring times for interval censoring (number of rows equal to the number of observations).
- **type**: the type of censoring to generate can be 'right' (default), 'left' or 'interval'

**Value**

an object of class 'Surv', the censoring indicator is equal to 1 if the event is uncensored and 0 otherwise for right/left censored data, or for interval censored data, the indicator is 0 uncensored, 1 right censored, 2 left censored, or 3 interval censored.

---

**getBackground**

**getBackground function**

**Description**

A function to

**Usage**

```r
getBackground(poly, type = "stamen-toner")
```

**Arguments**

- **poly**: a spatial object that can be transformed and the extent obtained using the bbox function.
- **type**: see ?openmap

**Value**

...
### getbb function

**Description**

A function to get the bounding box of a Spatial object

**Usage**

`getbb(obj)`

**Arguments**

- `obj`  
  a spatial object e.g. a `SpatialPolygonsDataFrame`, `SpatialPolygons`, etc ... anything with a bounding box that can be computed with `bbox(obj)`

**Value**

a `SpatialPolygons` object: the bounding box

### getBbasis function

**Description**

A function returning the piecewise polynomial coefficients for a B-spline basis function i.e. the basis functions.

**Usage**

`getBbasis(x, knots, degree, force = FALSE)`

**Arguments**

- `x`  
  a vector of data
- `knots`  
  a vector of knots in ascending order. The first and last knots must be respectively the minimum and maximum of x.
- `degree`  
  the degree of the spline
- `force`  
  logical: skip check on knots? (not recommended!)

**Value**

the knots and the piecewise polynomial coefficients for a B-spline basis function i.e. the basis functions.
getcov

Description

A function to return the covariance from a model based on the randomFields covariance functions. Not intended for general use.

Usage

getcov(u, sigma, phi, model, pars)

Arguments

u       distance
sigma   variance parameter
phi     scale parameter
model   correlation type, see ?CovarianceFct
pars    vector of additional parameters for certain classes of covariance function (eg Matern), these must be supplied in the order given in ?CovarianceFct and are not estimated

Value

this is just a wrapper for CovarianceFct

getgrd

Description

A function to create a regular grid over an observation window in order to model the spatial random effects as a Gaussian Markov random field.

Usage

getgrd(shape, cellwidth)

Arguments

shape    an object of class SpatialPolygons or SpatialPolygonsDataFrame
cellwidth a scalar, the width of the grid cells
Value

A function to extract and return the computational grid from a gridded analysis.

Usage

getGrid(mod, returnclass = "SpatialPolygonsDataFrame")

Arguments

mod an object of class mcmcspatsurv, returned by the function survspat
returnclass the class of object to return, default is a 'SpatialPolygonsDataFrame'. Other options are 'raster', which returns a raster brick; or 'SpatialPixelsDataFrame'

Value

a SpatialPolygonsDataFrame in which Monte Carlo expectations can be stored and later plotted.

describe

describe function

Description

A function to compute the length of eta

Usage

geleneta(cov.model)

Arguments

cov.model a covariance model
getOptCellwidth

Value

the length of eta

gOptCellwidth  getOptCellwidth function

Description

A function to compute an optimal cellwidth close to an initial suggestion. This maximises the efficiency of the MCMC algorithm when in the control argument of the function survspat, the option gridded is set to TRUE.

Usage

gOptCellwidth(dat, cellwidth, ext = 2, plot = TRUE, boundingbox = NULL)

Arguments

dat  any spatial data object whose bounding box can be computed using the function bbox.
cellwidth  an initial suggested cellwidth
ext  the extension parameter for the FFT transform, set to 2 by default
plot  whether to plot the grid and data to illustrate the optimal grid
boundingbox  optional bounding box over which to construct computational grid, supplied as an object on which the function 'bbox' returns the bounding box

Value

the optimum cell width

gparranges  gparranges function

Description

A function to extract parameter ranges for creating a grid on which to evaluate the log-posterior, used in calibrating the MCMC. This function is not intended for general use.

Usage

gparranges(priors, leneta, mult = 1.96)
Arguments

- **priors**: an object of class mcmcPriors
- **leneta**: the length of eta passed to the function
- **mult**: defaults to 1.96 so the grid formed will be mean plus/minus 1.96 times the standard deviation

Value

an appropriate range used to calibrate the MCMC: the mean of the prior for eta plus/minus 1.96 times the standard deviation

---

**getsurvdata**  
*_getsurvdata function_*

**Description**

A function to return the survival data from an object of class mcmcspatsurv. This function is not intended for general use.

**Usage**

`getsurvdata(x)`

**Arguments**

- **x**: an object of class mcmcspatsurv

**Value**

the survival data from an object of class mcmcspatsurv

---

**gompertzHaz**  
*_gompertzHaz function_*

**Description**

A function to define a parametric proportional hazards model where the baseline hazard is taken from a Gompertz model. This function returns an object inheriting class 'basehazardspec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard' and 'densityquantile'

**Usage**

`gompertzHaz()`
Details

The `distinfo` function is used to provide basic distribution specific information to other `spatsurv` functions. The user is required to provide the following information in the returned list: `npars`, the number of parameters in this distribution; `parnames`, the names of the parameters; `trans`, the transformation scale on which the priors will be provided; `itrans`, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; `jacobian`, the derivative of the inverse transformation function with respect to each of the parameters; and `hessian`, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package `spatsurv` only allows the use of functions where the parameters are transformed independently.

The `basehazard` function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, `t`, and returns a vector.

The `gradbasehazard` function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, `t`, and returns a vector.

The `hessbasehazard` function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, `t`, and returns a list of hessian matrices corresponding to each `t`.

The `cumbasehazard` function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, `t`, and returns a vector.

The `gradcumbasehazard` function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, `t`, and returns a matrix.

The `hesscumbasehazard` function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, `t`, and returns a list of hessian matrices corresponding to each `t`.

The `densityquantile` function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the `predict` function where type is 'densityquantile'. In the case of the Weibull model for the baseline hazard, it can be shown that the q-th quantile is:

\[
\text{Value}
\]

an object inheriting class 'basehazardspec'

See Also

tpowHaz, exponentialHaz, makehamHaz, weibullHaz
Description

Generic function for computing the gradient of the baseline hazard

Usage

gradbasehazard(obj, ...)

Arguments

obj an object
... additional arguments – currently there are none, but this is for extensibility

Value

method gradbasehazard

See Also

gradbasehazard.basehazardspec, exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

Description

A function to retrieve the gradient of the baseline hazard function

Usage

## S3 method for class 'basehazardspec'
gradbasehazard(obj, ...)

Arguments

obj an object of class basehazardspec
... additional arguments – currently there are none, but this is for extensibility

Value

a function returning the gradient of the baseline hazard
See Also

exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

gradcumbasehazard

gradcumbasehazard function

Description

Generic function for computing the gradient of the cumulative baseline hazard

Usage

gradcumbasehazard(obj, ...)

Arguments

obj an object

... additional arguments – currently there are none, but this is for extensibility

Value

method gradcumbasehazard

See Also

gradcumbasehazard.basehazardspec, exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

gradcumbasehazard.basehazardspec

gradcumbasehazard.basehazardspec function

Description

A function to retrieve the gradient of the cumulative baseline hazard function

Usage

## S3 method for class 'basehazardspec'
gradcumbasehazard(obj, ...)

Arguments

obj an object of class basehazardspec

... additional arguments – currently there are none, but this is for extensibility
Value

a function returning the gradient of the cumulative baseline hazard

See Also

exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

grid2spdf

grid2spdf function

Description

A function to convert a regular (x,y) grid of centroids into a SpatialPoints object

Usage

grid2spdf(xgrid, ygrid, proj4string = CRS(as.character(NA)))

Arguments

xgrid vector of x centroids (equally spaced)
ygrid vector of x centroids (equally spaced)
proj4string an optional proj4string, projection string for the grid, set using the function CRS

Value

a SpatialPolygonsDataFrame

grid2spix

grid2spix function

Description

A function to convert a regular (x,y) grid of centroids into a SpatialPixels object

Usage

grid2spix(xgrid, ygrid, proj4string = CRS(as.character(NA)))

Arguments

xgrid vector of x centroids (equally spaced)
ygrid vector of x centroids (equally spaced)
proj4string an optional proj4string, projection string for the grid, set using the function CRS

Value

a SpatialPixels object
grid2spts function

Description

A function to convert a regular (x,y) grid of centroids into a SpatialPoints object

Usage

`grid2spts(xgrid, ygrid, proj4string = CRS(as.character(NA)))`

Arguments

- `xgrid`: vector of x centroids (equally spaced)
- `ygrid`: vector of y centroids (equally spaced)
- `proj4string`: an optional proj4string, projection string for the grid, set using the function CRS

Value

a SpatialPoints object

gridY function

Description

A function to put estimated individual Y’s onto a grid

Usage

`gridY(Y, control)`

Arguments

- `Y`: estimate of Y
- `control`: control parameters

Value

...
gridY_polygonal function

Description
A function to put estimated individual Y’s onto a grid

Usage
gridY_polygonal(Y, control)

Arguments
Y estimate of Y
control control parameters

Value
...

guess_t function

Description
A function to get an initial guess of the failure time t, to be used in calibrating the MCMC. Not for general use

Usage
guess_t(surv)

Arguments
surv an object of class Surv

Value
a guess at the failure times
hasNext

**generic hasNext method**

**Description**

test if an iterator has any more values to go

**Usage**

```r
hasNext(obj)
```

**Arguments**

- `obj`: an iterator

hasNext.iter

**hasNext.iter function**

**Description**

method for iter objects test if an iterator has any more values to go

**Usage**

```r
## S3 method for class 'iter'
hasNext(obj)
```

**Arguments**

- `obj`: an iterator

hazardexceedance

**hazardexceedance function**

**Description**

A function to compute exceedance probabilities for the spatially correlated frailties.

**Usage**

```r
hazardexceedance(threshold, direction = "upper")
```
hazardpars

Arguments

threshold vector of thresholds
direction default is "upper" which will calculate P(Y>threshold), alternative is "lower", which will calculate P(Y<threshold)

Value

a function that can be passed to the function MCE in order to compute the exceedance probabilities

See Also

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailty-lag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, prior-posterior, posteriorcov, MCE,

Description

A function to return the mcmc chains for the hazard function parameters

Usage

hazardpars(x)

Arguments

x an object of class mcmcspatsurv

Value

the omega mcmc chains

See Also

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailty-lag1, spatialpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance
**hazard_PP**

### hazard_PP function

**Description**

A function to compute an individual’s hazard function.

**Usage**

hazard_PP(inputs)

**Arguments**

- `inputs` inputs for the function including the model matrix, frailties, fixed effects and the parameters of the baseline hazard derived from this model

**Value**

the hazard function for the individual

---

**hessbasehazard**

### hessbasehazard function

**Description**

Generic function for computing the hessian of the baseline hazard

**Usage**

hessbasehazard(obj, ...)

**Arguments**

- `obj` an object
- `...` additional arguments – currently there are none, but this is for extensibility

**Value**

method hessbasehazard

**See Also**

hessbasehazard.basehazardspec, exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz
hessbasehazard.basehazardspec

*hessbasehazard.basehazardspec function*

Description
A function to retrieve the Hessian of the baseline hazard function

Usage
```
## S3 method for class 'basehazardspec'
hessbasehazard(obj, ...)
```

Arguments
- `obj` an object of class `basehazardspec`
- `...` additional arguments – currently there are none, but this is for extensibility

Value
a function returning the Hessian of the baseline hazard

See Also
- `exponentialHaz`, `weibullHaz`, `gompertzHaz`, `makehamHaz`, `tpowHaz`

hesscumbasehazard

* hesscumbasehazard function

Description
Generic function for computing the Hessian of the cumulative baseline hazard

Usage
```
hesscumbasehazard(obj, ...)
```

Arguments
- `obj` an object
- `...` additional arguments – currently there are none, but this is for extensibility

Value
method `hesscumbasehazard`
**hesscumbasehazard.basehazardspec**

**See Also**

hesscumbasehazard.basehazardspec, exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

---

**Description**

A function to retrieve the hessian of the cumulative baseline hazard function

**Usage**

```r
## S3 method for class 'basehazardspec'
hesscumbasehazard(obj, ...)```

**Arguments**

- **obj**
  - an object of class basehazardspec

- **...**
  - additional arguments – currently there are none, but this is for extensibility

**Value**

a function returning the hessian of the cumulative baseline hazard

**See Also**

exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz

---

**imputationModel**

**imputationModel function**

---

**Description**

A function to

**Usage**

```r
imputationModel(formula, offset, covariateData, priors)```
indepGaussianprior

Arguments

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>formula</td>
<td>X</td>
</tr>
<tr>
<td>offset</td>
<td>X</td>
</tr>
<tr>
<td>covariateData</td>
<td>X</td>
</tr>
<tr>
<td>priors</td>
<td>X</td>
</tr>
</tbody>
</table>

Value

...

Independent  Independent function

Description

A function to declare and also evaluate an exponential covariance function.

Usage

Independent()

Value

the exponential covariance function

See Also

SpikedExponentialCovFct, covmodel, CovarianceFct

indepGaussianprior  indepGaussianprior.function

Description

A function for evaluating the log of an independent Gaussian prior for a given set of parameter values.

Usage

indepGaussianprior(beta = NULL, omega = NULL, eta = NULL, priors)
Arguments

- **beta**: parameter beta at which prior is to be evaluated
- **omega**: parameter omega at which prior is to be evaluated
- **eta**: parameter eta at which prior is to be evaluated
- **priors**: an object of class mcmcPriors, see ?mcmcPriors

Value

the log of the prior evaluated at the given parameter values

See Also

- survspat, betapriorGauss, omegapriorGauss, etapriorGauss, indepGaussianprior, derivindepGaussianprior

Description

A function to

Usage

```r
indepGaussianpriorST(beta = NULL, omega = NULL, eta = NULL, priors)
```

Arguments

- **beta**: X
- **omega**: X
- **eta**: X
- **priors**: X

Value

...
inference.control

inference.control function

Description

A function to control inferential settings. This function is used to set parameters for more advanced use of spatsurv.

Usage

inference.control(
  gridded = FALSE,
  cellwidth = NULL,
  ext = 2,
  imputation = NULL,
  optimcontrol = NULL,
  hessian = FALSE,
  plotcal = FALSE,
  timeonlyMCMC = FALSE,
  nugget = FALSE,
  savenugget = FALSE,
  split = 0.5,
  logUsigma_priormean = 0,
  logUsigma_priorsd = 0.5,
  nis = NULL,
  olinfo = NULL
)

Arguments

gridded logical. Whether to perform computation on a grid. Default is FALSE.
cellwidth the width of computational cells to use
ext integer the number of times to extend the computational grid by in order to perform computation. The default is 2.
imputation for polygonal data, an optional model for inference at the sub-polygonal level, see function imputationModel
optimcontrol a list of optional arguments to be passed to optim for non-spatial models
hessian whether to return a numerical hessian. Set this to TRUE for non-spatial models. equal to the number of parameters of the baseline hazard
plotcal logical, whether to produce plots of the MCMC calibration process, this is a technical option and should only be set to TRUE if poor mixing is evident (the printed h is low), then it is also useful to use a graphics device with multiple plotting windows.
timeonlyMCMC logical, whether to only time the MCMC part of the algorithm, or whether to include in the reported running time the time taken to calibrate the method (default)
nugget whether to include a nugget effect in the estimation. Note that only the mean and variance of the nugget effect is returned.
savenugget whether to save the MCMC chain for the nugget effect
split how to split the spatial and nugget proposal variance as a the proportion of variance assigned to the spatial effect apriori. Default is 0.5
logUsigma_priormean prior mean for log standard deviation of nugget effect
logUsigma_priorsd prior sd for log standard deviation of nugget effect
nis list of cell counts, each element being a matrix, with attributes "x" and "y" giving grid centroids in x and y directions. Used to impute locations of aggregated data:
olinfo to be supplied with nis, if continuous inference from aggregated data is required

Value
returns parameters to be used in the function survspat

See Also
survspat

insert

Description
A function to

Usage
insert(pars, idx, val)

Arguments
pars X
idx X
val X

Value
...

insert  insert function
invtransformweibull  *invtransformweibull function*

**Description**
A function to transform estimates of the (alpha, lambda) parameters of the weibull baseline hazard function, so they are commensurate with R’s inbuilt density functions, (shape, scale).

**Usage**
```r
invtransformweibull(x)
```

**Arguments**
- `x`  
a vector of parameters

**Value**
the transformed parameters. For the weibull model, this transforms 'shape' 'scale' (see ?dweibull) to 'alpha' and 'lambda' for the MCMC

---

is.burnin  *is this a burn-in iteration?*

**Description**
if this mcmc iteration is in the burn-in period, return TRUE

**Usage**
```r
is.burnin(obj)
```

**Arguments**
- `obj`  
an mcmc iterator

**Value**
TRUE or FALSE
is.retain  

*do we retain this iteration?*

**Description**

if this mcmc iteration is one not thinned out, this is true

**Usage**

```r
is.retain(obj)
```

**Arguments**

- `obj`: an mcmc iterator

**Value**

TRUE or FALSE

---

**iteration**  

*iteration number*

**Description**

within a loop, this is the iteration number we are currently doing.

**Usage**

```r
iteration(obj)
```

**Arguments**

- `obj`: an mcmc iterator

**Details**

- get the iteration number

**Value**

integer iteration number, starting from 1.
logPosterior

logPosterior ( \textit{logPosterior function} \\

Description

A function to evaluate the log-posterior of a spatial parametric proportional hazards model. Not intended for general use.

Usage

\begin{verbatim}
logPosterior(
  surv,
  X,
  beta,
  omega,
  eta,
  gamma,
  priors,
  cov.model,
  u,
  control,
  gradient = FALSE,
  hessian = FALSE
)
\end{verbatim}

Arguments

\begin{tabular}{ll}
\textbf{surv} & an object of class Surv \\
\textbf{X} & the design matrix, containing covariate information \\
\textbf{beta} & parameter beta \\
\textbf{omega} & parameter omega \\
\textbf{eta} & parameter eta \\
\textbf{gamma} & parameter gamma \\
\textbf{priors} & the priors, an object of class 'mcmcPriors' \\
\textbf{cov.model} & the spatial covariance model \\
\textbf{u} & vector of interpoint distances \\
\textbf{control} & a list containing various control parameters for the MCMC and post-processing routines \\
\textbf{gradient} & logical whether to evaluate the gradient \\
\textbf{hessian} & logical whether to evaluate the Hessian \\
\end{tabular}

Value

evaluates the log-posterior and the gradient and hessian, if required.
logPosterior_gridded

References


logPosterior_gridded  logPosterior_gridded function

Description

A function to evaluate the log-posterior of a spatial parametric proportional hazards model using gridded Y. Not intended for general use.

Usage

logPosterior_gridded(
  surv, 
  X, 
  beta, 
  omega, 
  eta, 
  gamma, 
  priors, 
  cov.model, 
  u, 
  control, 
  gradient = FALSE, 
  hessian = FALSE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>surv</td>
<td>an object of class Surv</td>
</tr>
<tr>
<td>X</td>
<td>the design matrix, containing covariate information</td>
</tr>
<tr>
<td>beta</td>
<td>parameter beta</td>
</tr>
<tr>
<td>omega</td>
<td>parameter omega</td>
</tr>
<tr>
<td>eta</td>
<td>parameter eta</td>
</tr>
<tr>
<td>gamma</td>
<td>parameter gamma</td>
</tr>
<tr>
<td>priors</td>
<td>the priors, an object of class 'mcmcPriors'</td>
</tr>
<tr>
<td>cov.model</td>
<td>the spatial covariance model</td>
</tr>
<tr>
<td>u</td>
<td>vector of interpoint distances</td>
</tr>
<tr>
<td>control</td>
<td>a list containing various control parameters for the MCMC and post-processing routines</td>
</tr>
<tr>
<td>gradient</td>
<td>logical whether to evaluate the gradient</td>
</tr>
<tr>
<td>hessian</td>
<td>logical whether to evaluate the Hessian</td>
</tr>
</tbody>
</table>
logPosterior_polygonal

Value

evaluates the log-posterior and the gradient and hessian, if required.

References


logPosterior_polygonal

logPosterior_polygonal function

Description

A function to evaluate the log-posterior of a spatial parametric proportional hazards model. Not
intended for general use.

Usage

logPosterior_polygonal(
   surv, X, beta, omega, eta, gamma, priors, cov.model, u, control, gradient = FALSE, hessian = FALSE
)

Arguments

surv an object of class Surv
X the design matrix, containing covariate information
beta parameter beta
omega parameter omega
eta parameter eta
gamma parameter gamma
priors the priors, an object of class 'mcmcPriors'
cov.model the spatial covariance model
logPosterior_SPDE function

Description

A function to evaluate the log-posterior of a spatial parametric proportional hazards model. Not intended for general use.

Usage

logPosterior_SPDE(
  surv,
  X,
  beta,
  omega,
  eta,
  gamma,
  priors,
  cov.model,
  u,
  control,
  gradient = FALSE,
  hessian = FALSE
)

Arguments

surv  an object of class Surv
X     the design matrix, containing covariate information
beta  parameter beta
omega parameter omega
eta parameter eta

gamma parameter gamma

priors the priors, an object of class 'mcmcPriors'
cov.model the spatial covariance model
u vector of interpoint distances
control a list containing various control parameters for the MCMC and post-processing routines
gradient logical whether to evaluate the gradient
hessian logical whether to evaluate the Hessian

Value

evaluates the log-posterior and the gradient and hessian, if required.

References


loop.mcmc loop over an iterator

Description

useful for testing progress bars

Usage

loop.mcmc(object, sleep = 1)

Arguments

object an mcmc iterator
sleep pause between iterations in seconds
makehamHaz

makehamHaz function

Description

A function to define a parametric proportional hazards model where the baseline hazard is taken from the Gompertz-Makeham model. This function returns an object inheriting class 'basehazard-spec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard' and 'densityquantile'

Usage

makehamHaz()

Details

The distinfo function is used to provide basic distribution specific information to other spatsurv functions. The user is required to provide the following information in the returned list: npars, the number of parameters in this distribution; parnames, the names of the parameters; trans, the transformation scale on which the priors will be provided; itrans, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; jacobian, the derivative of the inverse transformation function with respect to each of the parameters; and hessian, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package spatsurv only allows the use of functions where the parameters are transformed independently.

The basehazard function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a vector.

The gradbasehazard function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.

The hessbasehazard function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.

The cumbasehazard function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a vector.

The gradcumbasehazard function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.

The hesscumbasehazard function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.

The densityquantile function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the predict function where type is 'densityquantile'. In the case of the Weibull model for the baseline hazard, it can be shown that the q-th quantile is:
maxlikparamPHsurv

Value

an object inheriting class 'basehazardspec'

See Also

tpowHaz, exponentialHaz, gompertzHaz, weibullHaz

Description

A function to get initial estimates of model parameters using maximum likelihood. Not intended for general purport use.

Usage

maxlikparamPHsurv(surv, X, control)

Arguments

surv an object of class Surv
X the design matrix, containing covariate information
control a list containing various control parameters for the MCMC and post-processing routines

Value

initial estimates of the parameters

References

**MCE**

**MCE function**

---

**Description**

A function to compute Monte Carlo expectations from an object inheriting class mcmcspatsurv

**Usage**

```r
MCE(object, fun)
```

**Arguments**

- **object** an object inheriting class mcmcspatsurv
- **fun** a function with arguments beta, omega, eta and Y

**Value**

the Monte Carlo mean of the function over the posterior.

**See Also**

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, hazardexceedance

---

**mcmcLoop**

**iterator for MCMC loops**

---

**Description**

control an MCMC loop with this iterator

**Usage**

```r
mcmcLoop(N, burnin, thin, trim = TRUE, progressor = mcmcProgressPrint)
```

**Arguments**

- **N** number of iterations
- **burnin** length of burn-in
- **thin** frequency of thinning
- **trim** whether to cut off iterations after the last retained iteration
- **progressor** a function that returns a progress object
mcmcpars  mcmcpars function

Description

A function for setting MCMC options.

Usage

mcmcpars(nits, burn, thin, inits = NULL, adaptivescheme = NULL)

Arguments

nits          number of iterations,
burn         length of burnin
thin         thinning parameter eg operated on chain every 'thin' iteration (eg store output
or compute some posterior functional)
inits         NOT CURRENTLY IN USE
adaptivescheme NOT CURRENTLY IN USE

Value

mcmc parameters

mcmcPriors  mcmcPriors function

Description

A function to define priors for the MCMC.

Usage

mcmcPriors(
  betaprior = NULL,
  omegaprior = NULL,
  etaprior = NULL,
  call = NULL,
  derivative = NULL
)
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>betaprior</td>
<td>prior for beta, the covariate effects</td>
</tr>
<tr>
<td>omegaprior</td>
<td>prior for omega, the parameters of the baseline hazard</td>
</tr>
<tr>
<td>etaprior</td>
<td>prior for eta, the parameters of the latent field</td>
</tr>
<tr>
<td>call</td>
<td>function to evaluate the log-prior e.g. logindepGaussianprior</td>
</tr>
<tr>
<td>derivative</td>
<td>function to evaluate the first and second derivatives of the prior</td>
</tr>
</tbody>
</table>

Details

The package \texttt{spatsurv} only provides functionality for the built-in Gaussian priors. However, the choice of prior is extensible by the user by creating functions similar to the functions \texttt{betapriorGauss}, \texttt{omegapriorGauss}, \texttt{etapriorGauss}, \texttt{indepGaussianprior} and \texttt{derivindepGaussianprior}: the first three of which provide a mechanism for storing and retrieving the parameters of the priors; the fourth, a function for evaluating the log of the prior for a given set of parameter values; and the fifth, a function for evaluating the first and second derivatives of the log of the prior. It is assumed that parameters are a priori independent. The user interested in using other priors is encouraged to look at the structure of the five functions mentioned above.

Value

an object of class \texttt{mcmcPriors}

See Also

\texttt{survspar}, \texttt{betapriorGauss}, \texttt{omegapriorGauss}, \texttt{etapriorGauss}, \texttt{indepGaussianprior}, \texttt{derivindepGaussianprior}

\begin{verbatim}

mcmcProgressNone

null progress monitor

\end{verbatim}

Description

a progress monitor that does nothing

Usage

\begin{verbatim}

mcmcProgressNone(mcmcloop)

\end{verbatim}

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mcmcloop</td>
<td>an mcmc loop iterator</td>
</tr>
</tbody>
</table>

Value

a progress monitor
mcmcProgressTextBar

---

mcmcProgressPrint

**Description**

A progress monitor that prints each iteration

**Usage**

mcmcProgressPrint(mcmcloop)

**Arguments**

mcmcloop: an mcmc loop iterator

**Value**

A progress monitor

---

mcmcProgressTextBar

**Description**

A progress monitor that uses a text progress bar

**Usage**

mcmcProgressTextBar(mcmcloop)

**Arguments**

mcmcloop: an mcmc loop iterator

**Value**

A progress monitor
## midpts

**Description**

A function to compute the midpoints of a vector

**Usage**

midpts(x)

**Arguments**

- **x**: a vector

**Value**

The midpoints, a vector of length `length(x)-1`

## multiWayHaz

**multiWayHaz function**

**Description**

A function to

**Usage**

multiWayHaz(bhlist, bhtime, bhfix, MLinits = NULL)

**Arguments**

- **bhlist**: X
- **bhtime**: X
- **bhfix**: X
- **MLinits**: X

**Value**

...
neighLocs function

Description
A function used in the computation of neighbours on non-rectangular grids. Not intended for general use.

Usage
neighLocs(coord, cellwidth, order)

Arguments
- **coord**: coordinate of interest
- **cellwidth**: a scalar, the width of the grid cells
- **order**: the order of the SPDE approximation: see Lindgren et al 2011 for details

Value
coordinates of centroids of neighbours

References

neighOrder function

Description
A function to compute the order of a set of neighbours. Not intended for general use.

Usage
neighOrder(neighlocs)

Arguments
- **neighlocs**: an object created by the function neighLocs
Value

the neighbour orders

References


Description

just a wrapper for nextElem really.

Usage

nextStep(object)

Arguments

object an mcmc loop object

NonSpatialLogLikelihood_or_gradient

NonSpatialLogLikelihood_or_gradient function

Description

A function to evaluate the log-likelihood of a non-spatial parametric proportional hazards model. Not intended for general use.

Usage

NonSpatialLogLikelihood_or_gradient(
  surv,
  X,
  beta,
  omega,
  control,
  loglikelihood,
  gradient
)
omegapriorGauss

Arguments

- **surv**: an object of class Surv
- **X**: the design matrix, containing covariate information
- **beta**: parameter beta
- **omega**: parameter omega
- **control**: a list containing various control parameters for the MCMC and post-processing routines
- **loglikelihood**: logical whether to evaluate the log-likelihood
- **gradient**: logical whether to evaluate the gradient

Value

...

References


omegapriorGauss  omegapriorGauss function

Description

A function to define Gaussian priors for omega. This function simply stores a vector of means and standard deviations to be passed to the main MCMC function, survspat.

Usage

omegapriorGauss(mean, sd)

Arguments

- **mean**: the prior mean, a vector of length 1 or more. 1 implies a common mean.
- **sd**: the prior standard deviation, a vector of length 1 or more. 1 implies a common standard deviation.

Value

an object of class "omegapriorGauss"

See Also

survspat, betapriorGauss, omegapriorGauss, etapriorGauss, indepGaussianprior, derivindepGaussianprior
omegapriorGaussST

**Description**

A function to

**Usage**

```r
omegapriorGaussST(basehaz, fmean, fsd, taumean, tausd, thetamean, thetasd)
```

**Arguments**

- `basehaz`: X
- `fmean`: X
- `fsd`: X
- `taumean`: X
- `tausd`: X
- `thetamean`: X
- `thetasd`: X

**Value**

...

---

optifix

**Description**

optifix. Optimise with fixed parameters

**Usage**

```r
optifix(par, fixed, fn, gr = NULL, ...,
method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "SANN"),
lower = -Inf,
upper = Inf,
control = list(),
hessian = FALSE
)
```
Arguments

- **par**
- **fixed**
- **fn**
- **gr**
- **...**
- **method**
- **lower**
- **upper**
- **control**
- **hessian**

Details

It's like `optim`, but with fixed parameters.

Specify a second argument 'fixed', a vector of TRUE/FALSE values. If TRUE, the corresponding parameter in `fn()` is fixed. Otherwise, its variable and optimised over.

The return thing is the return thing from `optim()` but with a couple of extra bits - a vector of all the parameters and a vector copy of the 'fixed' argument.

Written by Barry Rowlingson <b.rowlingson@lancaster.ac.uk> October 2011

This file released under a CC By-SA license: http://creativecommons.org/licenses/by-sa/3.0/

and must retain the text: "Originally written by Barry Rowlingson" in comments.

Value

...
plotsurv

Value

...

plotsurv  plotsurv function

Description

A function to produce a 2-D plot of right censored spatial survival data.

Usage

plotsurv(
  spp,
  ss,
  maxcex = 1,
  transform = identity,
  background = NULL,
  eventpt = 19,
  eventcol = "red",
  censpt = "+",
  censcol = "black",
  xlim = NULL,
  ylim = NULL,
  xlab = NULL,
  ylab = NULL,
  add = FALSE,
  ...
)

Arguments

spp  A spatial points data frame
ss   A Surv object (with right-censoring)
maxcex maximum size of dots default is equivalent to setting cex equal to 1
transform optional transformation to apply to the data, a function, for example 'sqrt'
background a background object to plot default is null, which gives a blank background note that if non-null, the parameters xlim and ylim will be derived from this object.
eventpt  The type of point to illustrate events, default is 19 (see ?pch)
eventcol the colour of events, default is black
censpt   The type of point to illustrate events, default is "+" (see ?pch)
censcol  the colour of censored observations, default is red
xlim     optional x-limits of plot, default is to choose this automatically
polyadd

ylim  optional y-limits of plot, default is to choose this automatically
xlab  label for x-axis
ylab  label for y-axis
add   logical, whether to add the survival plot on top of an existing plot, default is FALSE, which produces a plot in a new device
...  other arguments to pass to plot

Value

Plots the survival data non-censored observations appear as dots and censored observations as crosses. The size of the dot is proportional to the observed time.

polyadd  polyadd function

Description

A function to add two polynomials in the form of vectors of coefficients. The first element of the vector being the constant (order 0) term

Usage

cpolyadd(poly1, poly2)

Arguments

cpoly1  a vector of coefficients for the first polynomial of length degree plus 1
cpoly2  a vector of coefficients for the second polynomial of length degree plus 1

Value

cthe coefficients of the sum of poly1 and poly2
polymult

polymult function

Description

A function to multiply two polynomials in the form of vectors of coefficients. The first element of
the vector being the constant (order 0) term

Usage

polymult(poly1, poly2)

Arguments

poly1 a vector of coefficients for the first polynomial of length degree plus 1
poly2 a vector of coefficients for the second polynomial of length degree plus 1

Value

the coefficients of the product of poly1 and poly2

posteriorcov

posteriorcov function

Description

A function to produce a plot of the posterior covariance function with upper and lower quantiles.

Usage

posteriorcov(
  x,
  probs = c(0.025, 0.5, 0.975),
  rmax = NULL,
  n = 100,
  plot = TRUE,
  bw = FALSE,
  corr = FALSE,
  ...
)

predict.mcmcspatsurv

Arguments

- **x**: an object of class `mcmcspatsurv`
- **probs**: vector of probabilities to be fed to quantile function
- **rmax**: maximum distance in space to compute this distance up to
- **n**: the number of points at which to evaluate the posterior covariance.
- **plot**: whether to plot the result
- **bw**: Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.
- **corr**: logical whether to return the correlation function, default is FALSE i.e. returns the covariance function
- **...**: other arguments to be passed to matplot function

Value

produces a plot of the posterior spatial covariance function.

See Also

- `print.mcmcspatsurv`, `quantile.mcmcspatsurv`, `summary.mcmcspatsurv`, `vcov.mcmcspatsurv`, `frailtylag1`, `spatialpars`, `hazardpars`, `fixedpars`, `randompars`, `baselinehazard`, `predict.mcmcspatsurv`, `priorposterior`, `MCE`, `hazardexceedance`

predict.mcmcspatsurv  predict.mcmcspatsurv function

Description

A function to produce predictions from MCMC output. These could include quantiles of the individual density, survival or hazard functions or quantiles of the density function (if available analytically).

Usage

```r
# S3 method for class 'mcmcspatsurv'
predict(
  object,
  type = "density",
  t = NULL,
  n = 110,
  indx = NULL,
  probs = c(0.025, 0.5, 0.975),
  plot = TRUE,
  pause = TRUE,
  bw = FALSE,
  ...
)
```

Arguments

- **object**: an object of class `mcmcspatsurv`
- **type**: can be "density", "hazard", "survival" or "densityquantile". Default is "density". Note that "densityquantile" is not always analytically tractable for some choices of baseline hazard function.
- **t**: optional vector of times at which to compute the quantiles. Default is NULL, in which case a uniformly spaced vector of length n from 0 to the maximum time is used.
- **n**: the number of points at which to compute the quantiles if t is NULL
- **indx**: the index number of a particular individual or vector of indices of individuals for which the quantiles should be produced
- **probs**: vector of probabilities
- **plot**: whether to plot the result
- **pause**: logical whether to pause between plots, the default is TRUE
- **bw**: Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.
- **...**: other arguments, not used here

Value

The required predictions

See Also

`print.mcmcspatsurv`, `quantile.mcmcspatsurv`, `summary.mcmcspatsurv`, `vcov.mcmcspatsurv`, `frailtylag1`, `spatialpars`, `hazardpars`, `fixedpars`, `randompars`, `baselinehazard`, `priorposterior`, `posteriorcov`, `MCE`, `hazardexceedance`
print.mcmcspatsurv

Description
A function to print summary tables from an MCMC run

Usage

## S3 method for class 'mcmcspatsurv'
print(x, probs = c(0.5, 0.025, 0.975), digits = 3, scientific = -3, ...)

Arguments

  x  an object inheriting class mcmcspatsurv
  probs  vector of quantiles to return
  digits  see help file ?format
  scientific  see help file ?format
  ...  additional arguments, not used here

Value

prints summary tables to the console

See Also

quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

print.mlspatsurv

Description
A function to print summary tables from an MCMC run

Usage

## S3 method for class 'mlspatsurv'
print(x, probs = c(0.5, 0.025, 0.975), digits = 3, scientific = -3, ...)
print.textSummary

Arguments

  x          an object inheriting class mcmcspatsurv
  probs      vector of quantiles to return
  digits     see help file ?format
  scientific see help file ?format
  ...        additional arguments, not used here

Value

  prints summary tables to the console

See Also

quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

print.textSummary  print.textSummary function

Description

  A function to print summary tables from an MCMC run

Usage

  ## S3 method for class 'textSummary'
  print(x, ...)

Arguments

  x          an object inheriting class textSummary
  ...        additional arguments, not used here

Value

  prints a text summary of 'x' to the console
priorposterior  

priorposterior function

Description

A function to produce plots of the prior (which shows as a red line) and posterior (showing as a histogram).

Usage

priorposterior(
  x, 
  breaks = 30, 
  ylab = "Density", 
  main = "", 
  pause = TRUE, 
  bw = FALSE, 
  ... 
)

Arguments

x        an object inheriting class mcmcspatsurv
breaks   see ?hist
ylab     optional y label
main     optional title
pause    logical whether to pause between plots, the default is TRUE
bw       Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.
...      other arguments passed to the hist function

Value

plots of the prior (red line) and posterior (histogram).

See Also

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, posteriorcov, MCE, hazardexceedance
**Description**

A function to compute an approximate scaling matrix for the MCMC algorithm. Not intended for general use.

**Usage**

```r
proposalVariance(
  X, surv, betahat, omegahat, Yhat, priors, cov.model, u, control
)
```

**Arguments**

- `X`: the design matrix, containing covariate information
- `surv`: an object of class Surv
- `betahat`: an estimate of beta
- `omegahat`: an estimate of omega
- `Yhat`: an estimate of Y
- `priors`: the priors
- `cov.model`: the spatial covariance model
- `u`: a vector of pairwise distances
- `control`: a list containing various control parameters for the MCMC and post-processing routines

**Value**

an estimate of eta and also an approximate scaling matrix for the MCMC
Description

A function to compute an approximate scaling matrix for the MCMC algorithm. Not intended for general use.

Usage

proposalVariance_gridded(
  X,
  surv,
  betahat,
  omegahat,
  Yhat,
  priors,
  cov.model,
  u,
  control
)

Arguments

- X: the design matrix, containing covariate information
- surv: an object of class Surv
- betahat: an estimate of beta
- omegahat: an estimate of omega
- Yhat: an estimate of Y
- priors: the priors
- cov.model: the spatial covariance model
- u: a vector of pairwise distances
- control: a list containing various control parameters for the MCMC and post-processing routines

Value

an estimate of eta and also an approximate scaling matrix for the MCMC
**Description**

A function to compute an approximate scaling matrix for the MCMC algorithm. Not intended for general use.

**Usage**

```r
proposalVariance_polygonal(
  X,
  surv,
  betahat,
  omegahat,
  Yhat,
  priors,
  cov.model,
  u,
  control
)
```

**Arguments**

- **X** the design matrix, containing covariate information
- **surv** an object of class Surv
- **betahat** an estimate of beta
- **omegahat** an estimate of omega
- **Yhat** an estimate of Y
- **priors** the priors
- **cov.model** the spatial covariance model
- **u** a vector of pairwise distances
- **control** a list containing various control parameters for the MCMC and post-processing routines

**Value**

an estimate of eta and also an approximate scaling matrix for the MCMC
Description

A function to compute an approximate scaling matrix for the MCMC algorithm. Not intended for general use.

Usage

```
proposalVariance_SPDE(
  X,
  surv,
  betahat,
  omegahat,
  Yhat,
  priors,
  cov.model,
  u,
  control
)
```

Arguments

- **X**: the design matrix, containing covariate information
- **surv**: an object of class Surv
- **betahat**: an estimate of beta
- **omegahat**: an estimate of omega
- **Yhat**: an estimate of Y
- **priors**: the priors
- **cov.model**: the spatial covariance model
- **u**: a vector of pairwise distances
- **control**: a list containing various control parameters for the MCMC and post-processing routines

Value

- an estimate of eta and also an approximate scaling matrix for the MCMC
### PsplineHaz

**PsplineHaz function**

**Description**

A function to define a parametric proportional hazards model where the baseline hazard is modelled by a basis spline and where the coefficients of the model follow a partially improper random walk prior. This function returns an object inheriting class 'basehazardspec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard' and 'densityquantile'.

**Usage**

```r
PsplineHaz(times, knots = quantile(times), degree = 3, MLinits = NULL)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>times</code></td>
<td>vector of survival times (both censored and uncensored)</td>
</tr>
<tr>
<td><code>knots</code></td>
<td>vector of knots in ascending order, must include minimum and maximum values of 'times'</td>
</tr>
<tr>
<td><code>degree</code></td>
<td>degree of the spline basis, default is 3</td>
</tr>
<tr>
<td><code>MLinits</code></td>
<td>optional starting values for the non-spatial maximisation routine using optim. Note that we are working with the log of the parameters. Default is -10 for each parameter.</td>
</tr>
</tbody>
</table>

**Details**

The `distinfo` function is used to provide basic distribution specific information to other `spatsurv` functions. The user is required to provide the following information in the returned list: `npars`, the number of parameters in this distribution; `parnames`, the names of the parameters; `trans`, the transformation scale on which the priors will be provided; `itrans`, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; `jacobian`, the derivative of the inverse transformation function with respect to each of the parameters; and `hessian`, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package `spatsurv` only allows the use of functions where the parameters are transformed independently.

The `basehazard` function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, `t` and returns a vector.

The `gradbasehazard` function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, `t`, and returns a matrix.

The `hessbasehazard` function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, `t` and returns a list of hessian matrices corresponding to each `t`. 
The `cumbasehazard` function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, \( t \) and returns a vector.

The `gradcumbasehazard` function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, \( t \), and returns a matrix.

The `hesscumbasehazard` function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, \( t \), and returns a list of hessian matrices corresponding to each \( t \).

The `densityquantile` function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the predict function where type is 'densityquantile'. In the case of the Weibull model for the baseline hazard, it can be shown that the q-th quantile is:

Value

an object inheriting class 'basehazardspec'

See Also

`exponentialHaz`, `gompertzHaz`, `makehamHaz`, `weibullHaz`

---

### psplineprior function

**Description**

A function for evaluating the log of an independent Gaussian prior for a given set of parameter values.

**Usage**

`psplineprior(beta = NULL, omega = NULL, eta = NULL, priors)`

**Arguments**

- `beta`: parameter beta at which prior is to be evaluated
- `omega`: parameter omega at which prior is to be evaluated
- `eta`: parameter eta at which prior is to be evaluated
- `priors`: an object of class `mcmcPriors`, see ?mcmcPriors

**Value**

the log of the prior evaluated at the given parameter values
**psplineRWprior**

**See Also**

survspat, betapriorGauss, omegapriorGauss, etapriorGauss, indepGaussianprior, derivindepGaussianprior

---

**psplineRWprior**  
*psplineRWprior function*

---

**Description**

A function to define Gaussian priors for omega. This function simply stores a vector of means and standard deviations to be passed to the main MCMC function, survspat.

**Usage**

```
psplineRWprior(taumean, tausd, basehaz, order = 2)
```

**Arguments**

- `taumean`: the prior mean, a vector of length 1 or more. 1 implies a common mean.
- `tausd`: the prior standard deviation, a vector of length 1 or more. 1 implies a common standard deviation.
- `basehaz`: an object inheriting class "basehazardspec", specifically, this function was used for such objects created by a call to the function PsplineHaz
- `order`: the order of the random walk, default is 2

**Value**

an object of class "omegapriorGauss"

**See Also**

survspat, betapriorGauss, omegapriorGauss, etapriorGauss, indepGaussianprior, derivindepGaussianprior
QuadApprox  

QuadApprox function

Description

A function to compute the second derivative of a function (of several real variables) using a quadratic approximation on a grid of points defined by the list argRanges. Also returns the local maximum.

Usage

QuadApprox(fun, npts, argRanges, plot = FALSE, ...)

Arguments

- **fun**: a function
- **npts**: integer number of points in each direction
- **argRanges**: a list of ranges on which to construct the grid for each parameter
- **plot**: whether to plot the quadratic approximation of the posterior (for two-dimensional parameters only)
- **...**: other arguments to be passed to fun

Value

a 2 by 2 matrix containing the curvature at the maximum and the (x,y) value at which the maximum occurs

quantile.mcmcspatsurv  

quantile.mcmcspatsurv function

Description

A function to extract quantiles of the parameters from an mcmc run

Usage

```r
## S3 method for class 'mcmcspatsurv'
quantile(x, probs = c(0.025, 0.5, 0.975), ...)
```

Arguments

- **x**: an object inheriting class mcmcspatsurv
- **probs**: vector of probabilities
- **...**: other arguments to be passed to the function, not used here
quantile.mlspatsurv

Value

quantiles of model parameters

See Also

print.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

quantile.mlspatsurv  quantile.mlspatsurv function

Description

A function to extract quantiles of the parameters from an mcmc run

Usage

## S3 method for class 'mlspatsurv'
quantile(x, probs = c(0.025, 0.5, 0.975), ...)

Arguments

x an object inheriting class mcmcspatsurv
probs vector of probabilities
... other arguments to be passed to the function, not used here

Value

quantiles of model parameters

See Also

print.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance
randompars function

Description
A function to return the mcmc chains for the spatially correlated frailties

Usage
randompars(x)

Arguments
x an object of class mcmcspatsurv

Value
the Y mcmc chains

See Also
print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, vcov.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

reconstruct.bs function

Description
Generic function for reconstructing B-spline covariate effects. See ?reconstruct.bs.mcmcspatsurv and ?reconstruct.bs.coxph

Usage
reconstruct.bs(mod, ...)

Arguments
mod an object
... additional arguments

Value
method reconstruct.bs
**Description**

When bs(varname) has been used in the formula of a coxph model, this function can be used to reconstruct the predicted relative risk of that parameter over time.

**Usage**

```r
## S3 method for class 'coxh'
reconstruct.bs(  
  mod,  
  varname,  
  fun = NULL,  
  probs = c(0.025, 0.975),  
  bw = FALSE,  
  xlab = NULL,  
  ylab = NULL,  
  plot = TRUE,  
  ...  
)
```

**Arguments**

- `mod`: model output, created by function survspat
- `varname`: name of the variable modelled by a B-spline
- `fun`: optional function to feed in. Default is to plot relative risk against the covariate of interest. Useful choices include "identity" (but with no quotes), which plots the non-linear effect on the scale of the linear predictor.
- `probs`: upper and lower quantiles for confidence regions to plot. The default is `c(0.025, 0.975)`.
- `bw`: Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.
- `xlab`: label for x axis, there is a sensible default
- `ylab`: label for y axis, there is a sensible default
- `plot`: logical, whether to plot the effect of varname over time
- `...`: other arguments to be passed to the plotting function.

**Value**

median, upper and lower confidence bands for the effect of varname over time; the function also produces a plot.
Description

When `bs(varname)` has been used in the formula of a model, this function can be used to reconstruct the posterior relative risk of that parameter over time.

Usage

```r
## S3 method for class 'mcmcspatsurv'
reconstruct.bs(  
  mod,  
  varname,  
  probs = c(0.025, 0.975),  
  bw = FALSE,  
  xlab = NULL,  
  ylab = NULL,  
  plot = TRUE,  
  ...  
)
```

Arguments

- `mod`: model output, created by function `survspat`
- `varname`: name of the variable modelled by a B-spline
- `probs`: upper and lower quantiles for confidence regions to plot. The default is `c(0.025, 0.975)`.  
- `bw`: Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.  
- `xlab`: label for x axis, there is a sensible default  
- `ylab`: label for y axis, there is a sensible default  
- `plot`: logical, whether to plot the effect of `varname` over time  
- `...`: other arguments to be passed to the plotting function.

Value

median, upper and lower confidence bands for the effect of `varname` over time; the function also produces a plot.
resetLoop

reset Loop

reset iterator

Description

call this to reset an iterator's state to the initial

Usage

resetLoop(obj)

Arguments

obj an mcmc iterator

residuals.mcmcspatsurv

residuals.mcmcspatsurv function

Description

A function to compute Cox-Snell / modified Cox-Snell / Martingale or Deviance residuals

Usage

## S3 method for class 'mcmcspatsurv'
residuals(object, type = "Cox-Snell", ...)

Arguments

object an object produced by the function survspat
type type of residuals to return. Possible choices are 'Cox-Snell', 'modified-Cox-Snell', 'Martingale' or 'deviance'.
... other arguments (not used here)

Value

the residuals
rootWeibullHaz

rootWeibullHaz function

Description

A function to define a parametric proportional hazards model where the baseline hazard is taken from the Weibull model. This function returns an object inheriting class 'basehazardspec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard', 'densityquantile'

Usage

rootWeibullHaz(MLinits = NULL)

Arguments

MLinits initial values for optim, default is NULL

Details

The distinfo function is used to provide basic distribution specific information to other spatsurv functions. The user is required to provide the following information in the returned list: npars, the number of parameters in this distribution; parnames, the names of the parameters; trans, the transformation scale on which the priors will be provided; itrans, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; jacobian, the derivative of the inverse transformation function with respect to each of the parameters; and hessian, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package spatsurv only allows the use of functions where the parameters are transformed independently.

The basehazard function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a vector.

The gradbasehazard function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.

The hessbasehazard function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.

The cumbasehazard function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.

The gradcumbasehazard function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.

The hesscumbasehazard function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.
The `densityquantile` function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the `predict` function where `type` is 'densityquantile'. In the case of the Weibull model for the baseline hazard, it can be shown that the \( q \)-th quantile is:

**Value**

an object inheriting class 'basehazardspec'

**See Also**

tpowHaz, exponentialHaz, gompertzHaz, makehamHaz

---

**Description**

update a text progress bar. See help(txtProgressBar) for more info.

**Usage**

```r
setTxtProgressBar2(pb, value, title = NULL, label = NULL)
```

**Arguments**

- `pb` text progress bar object
- `value` new value
- `title` ignored
- `label` text for end of progress bar

---

**Description**

A function to set up the baseline hazard, cumulative hazard and derivative functions for use in evaluating the log posterior. This function is not intended for general use.

**Usage**

```r
setupHazard(dist, pars, grad = FALSE, hess = FALSE)
```
**setupPrecMatStruct**

**Arguments**

- **dist**: an object of class 'basehazardspec'
- **pars**: parameters with which to create the functions necessary to evaluate the log posterior
- **grad**: logical, whether to create gradient functions for the baseline hazard and cumulative hazard
- **hess**: logical, whether to create hessian functions for the baseline hazard and cumulative hazard

**Value**

a list of functions used in evaluating the log posterior

**Description**

A function to set up the computational grid and precision matrix structure for SPDE models.

**Usage**

```r
setupPrecMatStruct(shape, cellwidth, no)
```

**Arguments**

- **shape**: an object of class SpatialPolygons or SpatialPolygonsDataFrame
- **cellwidth**: a scalar, the width of the grid cells
- **no**: the order of the SPDE approximation: see Lindgren et al 2011 for details

**Value**

the computational grid and a function for constructing the precision matrix

**References**

**showGrid**

**showGrid function**

**Description**

A function to show the grid that will be used for a given cellwidth

**Usage**

```r
showGrid(dat, cellwidth, ext = 2, boundingbox = NULL)
```

**Arguments**

- **dat**: any spatial data object whose bounding box can be computed using the function `bbox`.
- **cellwidth**: an initial suggested cellwidth
- **ext**: the extension parameter for the FFT transform, set to 2 by default
- **boundingbox**: optional bounding box over which to construct computational grid, supplied as an object on which the function `bbox` returns the bounding box

**Value**

a plot showing the grid and the data. Ideally the data should only just fit inside the grid.

---

**simsurv**

**simsurv function**

**Description**

A function to simulate spatial parametric proportional hazards model. The function works by simulating candidate survival times using MCMC in parallel for each individual based on each individual’s covariates and the common parameter effects, beta.

**Usage**

```r
simsurv(
  X = cbind(age = runif(100, 5, 50), sex = rbinom(100, 1, 0.5), cancer = rbinom(100, 1, 0.2)),
  beta = c(0.0296, 0.0261, 0.035),
  omega = 1,
  dist = exponentialHaz(),
  coords = matrix(runif(2 * nrow(X)), nrow(X), 2),
  cov.parameters = c(1, 0.1),
  cov.model = covmodel(model = "exponential", pars = NULL),
  mcmc.control = mcmcpars(nits = 1e+05, burn = 10000, thin = 90),
  savechains = TRUE
)
```
Arguments

- `X`: a matrix of covariate information
- `beta`: the parameter effects
- `omega`: vector of parameters for the baseline hazard model
- `dist`: the distribution choice: `exp` or `weibull` at present
- `coords`: matrix with 2 columns giving the coordinates at which to simulate data
- `cov.parameters`: a vector: the parameters for the covariance function
- `cov.model`: an object of class `covmodel`, see ?`covmodel`
- `mcmc.control`: mcmc control parameters, see ?`mcmcpars`
- `savechains`: save all chains? runs faster if set to `FALSE`, but then you’ll be unable to conduct convergence/mixing diagnostics

Value

in list element `survtimes`, a vector of simulated survival times (the last simulated value from the MCMC chains) in list element `T` the MCMC chains

See Also

- `covmodel`, `survspat`, `tpowHaz`, `exponentialHaz`, `gompertzHaz`, `makehamHaz`, `weibullHaz`

---

**spatialpars**

**spatialpars function**

**Description**

A function to return the mcmc chains for the spatial covariance function parameters

**Usage**

`spatialpars(x)`

**Arguments**

- `x`: an object of class `mcmcspatSurv`

**Value**

the eta mcmc chains

**See Also**

- `print.mcmcspatSurv`, `quantile.mcmcspatSurv`, `summary.mcmcspatSurv`, `vcov.mcmcspatSurv`, `frailty.lag1`, `hazardpars`, `fixedpars`, `randompars`, `baselinehazard`, `predict.mcmcspatSurv`, `priorposterior`, `posteriorcov`, `MCE`, `hazardexceedance`
spatsurvVignette  

**spatsurvVignette function**

**Description**
Display the introductory vignette for the spatsurv package.

**Usage**
spatsurvVignette()

**Value**
displays the vignette by calling browseURL

---

**SPDE  

**SPDE function**

**Description**
A function to declare and evaluate an SPDE covariance function.

**Usage**
SPDE(ord)

**Arguments**
ord the order of the model to be used, currently an integer between 1 an 3. See Lindgren 2011 paper.

**Value**
an covariance function based on the SPDE model

**See Also**
ExponentialCovFct, covmodel, CovarianceFct
SPDEprec

Description

A function to used in entering elements into the precision matrix of an SPDE model. Not intended for general use.

Usage

SPDEprec(a, ord)

Arguments

a
parameter a, see Lindgren et al 2011.

ord
the order of the SPDE model, see Lindgren et al 2011.

Value

a function used for creating the precision matrix

References


SpikedExponentialCovFct

Description

A function to declare and also evaluate a spiked exponential covariance function. This is an exponential covariance function with a nugget.

Usage

SpikedExponentialCovFct()

Value

the spiked exponential covariance function
See Also

ExponentialCovFct, covmodel, CovarianceFct

spplot1

spplot1 function

Description

A function to provide spplot-like plotting capability but NOT using trellis graphics. This function also acts as an interface for fast plotting of SpatialPolygonsDataFrame or SpatialPixelsDataFrame objects using leaflet HTML plotting capabilities to get zoomable plots with real-world context: transformation to the correct projection is done automatically.

Usage

spplot1(
  x,
  what,
  palette = brewer.pal(5, "Oranges"),
  breaks = NULL,
  legpos = "topleft",
  fun = identity,
  include.lowest = TRUE,
  bty = "n",
  bg = NULL,
  printlegend = TRUE,
  bw = FALSE,
  useLeaflet = FALSE,
  urltemplate = urlTemplate("Stamen_Toner"),
  fillOpacity = 0.5,
  legendOpacity = 0.5,
  OSMbg = NULL,
  leafletLegend = TRUE,
  alpha.f = 0.5,
  plotinorder = FALSE,
  legendText = NULL,
  legendFun = NULL,
  ...
)

Arguments

x a SpatialPolygonsDataFrame or a SpatialPointsDataFrame
what the name of the variable to plot
palette the palette, can either be a vector of names of colours, or a vector of colours produced for example by the brewer.pal function.
breaks  optional breaks for the legend, a vector of length 1 + length(palette)
legpos  the position of the legend, options are 'topleft', 'topright', 'bottomleft', 'bottomright'
fun  an optional function of the data to plot, default is the identity function
include.lowest  see ?cut
bty  see ?legend
bg  see ?legend
printlegend  logical: print the legend?
bw  Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.
useLeaflet  whether to use leaflet to produce a zoomable map this requires the leaflet package, available by issuing the command "devtools::install_github('rstudio/leaflet')"
urltemplate  template for leaflet map background, default is urlTemplate('Stamen-Toner'), but any valid web address for leaflet templates will work here. See ?urlTemplate.
fillOpacity  see ?addPolygons
legendOpacity  see opacity argument in function addLegend
OSMbg  optional OpenStreetMap background to add to plot, obtain this using the function getBackground
leafletLegend  logical, display the leaflet legend?
alpha.f  point transparency, see ?adjustcolor, default is 0.5
plotinorder  whether to plot in order of the size of the variable being plotted, useful for overlapping windows where small counts may obscure big counts
legendText  X
legendFun  X
...  other arguments to be passed to plot

Details

See http://leaflet-extras.github.io/leaflet-providers/preview/ for examples of leaflet templates.
Instructions on installing the leaflet R package are available from https://rstudio.github.io/leaflet/

Value

either produces a plot or if useLeaflet is TRUE, returns a leaflet map widget to which further layers can be added

See Also

urlTemplate, getBackground, brewer.pal
spplot_compare

spplot_compare function

Description

A function to compare two SpatialPolygonsDataFrame or SpatialPointsDataFrame objects using a unified legend for the variable of interest in both.

Usage

spplot_compare(
  x,
  y,
  what,
  what1 = what,
  palette = brewer.pal(9, "Oranges"),
  legpos = "topleft",
  border = NA,
  fun = identity,
  t1 = "",
  t2 = "",
  bw = FALSE,
  ...
)

Arguments

x a SpatialPolygonsDataFrame or a SpatialPointsDataFrame
y a SpatialPolygonsDataFrame or a SpatialPointsDataFrame
what the name of the variable from x to plot
what1 the name of the variable from y to plot. default is to plot the variable of the same name
palette the palette, can either be a vector of names of colours, or a vector of colours produced for example by the brewer.pal function.
legpos the position of the legend, options are 'topleft', 'topright', 'bottomleft', 'bottomright'
border see ?spplot
fun an optional function of the data to plot, default is the identity function
t1 title for the plot of x
t2 title for the plot of y
bw Logical. Plot in black/white/greyscale? Default is to produce a colour plot. Useful for producing plots for journals that do not accept colour plots.
... other arguments to be passed to the plot function
Summarise

Value

produces a plot comparing x[[what]] and y[[what1]]

---

**Summarise**

**Summarise function**

---

**Description**

A function to completely summarise the output of an object of class mcmcspatsurv.

**Usage**

```r
Summarise(
  obj,
  digits = 3,
  scientific = -3,
  inclIntercept = FALSE,
  printmode = "LaTeX",
  displaymode = "console",
  ...
)
```

**Arguments**

- `obj`: an object produced by a call to lgcpPredictSpatialPlusPars, lgcpPredictAggregateSpatialPlusPars, lgcpPredictSpatioTemporalPlusPars or lgcpPredictMultitypeSpatialPlusPars
- `digits`: see the option "digits" in ?format
- `scientific`: see the option "scientific" in ?format
- `inclIntercept`: logical: whether to summarise the intercept term, default is FALSE.
- `printmode`: the format of the text to return, can be 'LaTeX' (the default) or 'text' for plain text.
- `displaymode`: default is 'console' alternative is 'rstudio'
- `...`: other arguments passed to the function "format"

**Value**

A text summary, that can be pasted into a LaTeX document and later edited.
Summary

The `summary.mcmc` function provides a summary of an mcmc iterator by printing out values of an iterator and then resetting it. It is recommended not to call this function in a loop that uses this iterator as it will reset it and break.

Usage

```r
## S3 method for class 'mcmc'
summary(object, ...)  
```

Arguments

- `object`: an mcmc iterator
- `...`: other arguments

Summary.mcmcspatsurv

The `summary.mcmcspatsurv` function returns summary tables from an MCMC run.

Usage

```r
## S3 method for class 'mcmcspatsurv'
summary(object, probs = c(0.5, 0.025, 0.975), ...)  
```

Arguments

- `object`: an object inheriting class mcmcspatsurv
- `probs`: vector of quantiles to return
- `...`: additional arguments

Value

Summary tables to the console

See Also

- `print.mcmcspatsurv`, `quantile.mcmcspatsurv`, `vcov.mcmcspatsurv`, `frailtylag1`, `spatialpars`, `hazardpars`, `fixedpars`, `randompars`, `baselinehazard`, `predict.mcmcspatsurv`, `priorposterior`, `posteriorcov`, `MCE`, `hazardexceedance`
surv3d

**Spatial Survival Plot in 3D**

**Description**

Do a 3d plot of spatial survival data

**Usage**

```r
surv3d(
spp,
ss,
lwd = 2,
lcol = "black",
lalpha = 1,
pstyle = c("point", "text"),
psize = c(20, 10),
pcol = c("red", "black"),
pextent = c("x", ""),
palpha = 1,
title = "Spatial Survival",
basegrid = TRUE,
baseplane = TRUE
)
```

**Arguments**

- `spp` A spatial points data frame
- `ss` A `Surv` object (with right-censoring)
- `lwd` Line width for stems
- `lcol` Line colour for stems
- `lalpha` Opacity for stems
- `pstyle` Point style "point" or "text"
- `psize` Vector of length 2 for uncensored/censored points size
- `pcol` Vector of length 2 for uncensored/censored points colours
- `pextent` Vector of length 2 for uncensored/censored text characters
- `palpha` Opacity for points/text
- `title` Main title for plot
- `basegrid` add a grid at t=0
- `baseplane` add a plane at t=0

**Details**

Uses rgl graphics to make a spinny zoomy plot
survival_PP

Value
nothing

Author(s)
Barry S Rowlingson

Examples
```r
## Not run:
require(sp)
require(survival)
d = data.frame(
  x=runif(40)*1.5,
  y = runif(40),
  age=as.integer(20+30*runif(40)),
  sex = sample(c("M","F"),40,TRUE)
)
coordinates(d)==x+y
d$surv = Surv(as.integer(5+20*runif(40)),runif(40)>0.9)
clear3d();surv3d(d,d$surv,baseplane=TRUE,basegrid=TRUE)
clear3d();surv3d(d,d$surv,baseplane=TRUE,basegrid=TRUE,pstyle="t",lalpha=0.5,lwd=3,palpha=1)
## End(Not run)
```

survival_PP survival_PP function

Description
A function to compute an individual’s survival function

Usage
```
survival_PP(inputs)
```

Arguments
inputs inputs for the function including the model matrix, frailties, fixed effects and the parameters of the baseline hazard derived from this model

Value
the survival function for the individual
survspat  

**survspat function**

**Description**

A function to run a Bayesian analysis on censored spatial survival data assuming a proportional hazards model using an adaptive Metropolis-adjusted Langevin algorithm.

**Usage**

```r
survspat(
  formula,
  data,
  dist,
  cov.model,
  mcmc.control,
  priors,
  shape = NULL,
  ids = list(shpid = NULL, dataid = NULL),
  control = inference.control(gridded = FALSE),
  boundingbox = NULL
)
```

**Arguments**

- `formula`: the model formula in a format compatible with the function `flexsurvreg` from the `flexsurv` package
- `data`: a `SpatialPointsDataFrame` object containing the survival data as one of the columns OR for polygonal data a `data.frame`, in which case, the argument `shape` must also be supplied
- `dist`: choice of distribution function for baseline hazard. Current options are: `expoHaz`, `weibullHaz`, `gompertzHaz`, `makehamHaz`, `tpowHaz`
- `cov.model`: an object of class `covmodel`, see `?covmodel` ?ExponentialCovFct or ?SpikedExponentialCovFct
- `mcmc.control`: mcmc control parameters, see `?mcmcpars`
- `priors`: an object of class `Priors`, see `?mcmcPriors`
- `shape`: when data is a `data.frame`, this can be a `SpatialPolygonsDataFrame`, or a `SpatialPointsDataFrame`, used to model spatial variation at the small region level. The regions are the polygons, or they represent the (possibly weighted) centroids of the polygons.
- `ids`: named list entry `shpid` character string giving name of variable in `shape` to be matched to variable `dataid` in `data`. `dataid` is the second entry of the named list.
- `control`: additional control parameters, see `?inference.control`
- `boundingbox`: optional bounding box over which to construct computational grid, supplied as an object on which the function `bbox` returns the bounding box
survspatNS

Value

an object inheriting class ‘mcmcspatsurv’ for which there exist methods for printing, summarising and making inference from.

References


See Also
tpowHaz, exponentialHaz, gompertzHaz, makehamHaz, weibullHaz, covmodel, linkExponential-CovFct, SpikedExponentialCovFct, mcmcpars, mcmcPriors, inference.control

survspatNS function

Description

A function to perform maximum likelihood inference for non-spatial survival data.

Usage

survspatNS(formula, data, dist, control = inference.control())

Arguments

formula the model formula in a format compatible with the function flexsurvreg from the flexsurv package
data a SpatialPointsDataFrame object containing the survival data as one of the columns
dist choice of distribution function for baseline hazard. Current options are: exponentialHaz, weibullHaz, gompertzHaz, makehamHaz, tpowHaz
control additional control parameters, see ?inference.control

Value

an object inheriting class ‘mcmcspatsurv’ for which there exist methods for printing, summarising and making inference from.

References


See Also
tpowHaz, exponentialHaz, gompertzHaz, makehamHaz, weibullHaz, covmodel, linkExponential-CovFct, SpikedExponentialCovFct, mcmcpars, mcmcPriors, inference.control
Description

A function to print a text description of the inferred parameters beta and eta from a call to the function `lgcpPredictSpatialPlusPars`, `lgcpPredictAggregateSpatialPlusPars`, `lgcpPredictSpatioTemporalPlusPars` or `lgcpPredictMultitypeSpatialPlusPars`

Usage

```
textSummary(
  obj,
  digits = 3,
  scientific = -3,
  inclIntercept = FALSE,
  printmode = "LaTeX",
  ...
)
```

Arguments

- `obj`: an object produced by a call to `lgcpPredictSpatialPlusPars`, `lgcpPredictAggregateSpatialPlusPars`, `lgcpPredictSpatioTemporalPlusPars` or `lgcpPredictMultitypeSpatialPlusPars`
- `digits`: see the option "digits" in `?format`
- `scientific`: see the option "scientific" in `?format`
- `inclIntercept`: logical: whether to summarise the intercept term, default is FALSE.
- `printmode`: the format of the text to return, can be 'LaTeX' (the default) or 'text' for plain text.
- `...`: other arguments passed to the function "format"

Value

A text summary, that can be pasted into a LaTeX document and later edited.
**timevaryingPL**

**timevaryingPL function**

---

**Description**

A function to

**Usage**

```r
timevaryingPL(
    formula,
    t0,
    t,
    delta,
    dist,
    data,
    ties = "Efron",
    optimcontrol = NULL
)
```

**Arguments**

- `formula`: a formula of the form `S ~ coef1 + coef2` etc the object S will be created
- `t0`, `t`: X
- `delta`: censoring indicator a vector of 1 for an event and 0 for censoring
- `dist`, `data`: X
- `ties`: X default is Efron
- `optimcontrol`: X

**Value**

...
Description

A function to define a parametric proportional hazards model where the baseline hazard is taken from the 'powers of t' model. This function returns an object inheriting class 'basehazardspec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard' and 'densityquantile'

Usage

tpowHaz(powers)

Arguments

powers a vector of powers of t. These are powers are treated as fixed in estimation routines and it is assumed that the log cumulative baseline hazard is a linear combination of these powers of t

Details

The distinfo function is used to provide basic distribution specific information to other spatsurv functions. The user is required to provide the following information in the returned list: npars, the number of parameters in this distribution; parnames, the names of the parameters; trans, the transformation scale on which the priors will be provided; itrans, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; jacobian, the derivative of the inverse transformation function with respect to each of the parameters; and hessian, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package spatsurv only allows the use of functions where the parameters are transformed independently.

The basehazard function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a vector.

The gradbasehazard function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.

The hessbasehazard function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, t, and returns a list of hessian matrices corresponding to each t.

The cumbasehazard function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.

The gradcumbasehazard function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.
The `hesscumbasehazard` function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, `t` and returns a list of hessian matrices corresponding to each `t`.

The `densityquantile` function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the `predict` function where `type` is 'densityquantile'. In the case of the Weibull model for the baseline hazard, it can be shown that the q-th quantile is:

\[
\text{Value}
\]

an object inheriting class 'basehazardspec'

**See Also**

`exponentialHaz`, `gompertzHaz`, `makehamHaz`, `weibullHaz`

---

```r
transformweibull
```

**transformweibull function**

**Description**

A function to back-transform estimates of the parameters of the weibull baseline hazard function, so they are commensurate with R’s inbuilt density functions. Transforms from (shape, scale) to (alpha, lambda)

**Usage**

`transformweibull(x)`

**Arguments**

- `x` a vector of parameters

**Value**

the transformed parameters. For the weibull model, this is the back-transform from 'alpha' and 'lambda' to 'shape' 'scale' (see 'dweibull').
TwoWayHazAdditive function

Description
A function to

Usage
TwoWayHazAdditive(bhlist, bhtime, bhfix, MLinits = NULL)

Arguments

- `bhlist`: X
- `bhtime`: X
- `bhfix`: X
- `MLinits`: X

Value
...

txtProgressBar2

Description
This is the base txtProgressBar but with a little modification to implement the label parameter for style=3. For full info see txtProgressBar

Usage
txtProgressBar2(
  min = 0,
  max = 1,
  initial = 0,
  char = "=",
  width = NA,
  title = "",
  label = "",
  style = 1
)
urlTemplate

Arguments

min  min value for bar
max  max value for bar
initial initial value for bar
char the character (or character string) to form the progress bar.
width progress bar width
title ignored
label text to put at the end of the bar
style bar style

urlTemplate  urlTemplate function

Description

A function to return a url for a leaflet template for use as map backgrounds with the spplot1 function.

Usage

urlTemplate(name = "Stamen_Toner")

Arguments

name  name of the template to use, the default is 'Stamen_Toner'

Details


See http://leaflet-extras.github.io/leaflet-providers/preview/ for other leaflet templates

Value

url for the leaflet template
vcov.mcmcspatsurv function

Description
A function to return the variance covariance matrix of the parameters beta, omega and eta

Usage
## S3 method for class 'mcmcspatsurv'
vcov(object, ...)

Arguments

object an object inheriting class mcmcspatsurv
...
other arguments, not used here

Value
the variance covariance matrix of the parameters beta, omega and eta

See Also
print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

vcov.mlspatsurv function

Description
A function to return the variance covariance matrix of the parameters beta, omega and eta

Usage
## S3 method for class 'mlspatsurv'
vcov(object, ...)

Arguments

object an object inheriting class mlspatsurv
...
other arguments, not used here
weibullHaz

Value

the variance covariance matrix of the parameters beta, omega and eta

See Also

print.mcmcspatsurv, quantile.mcmcspatsurv, summary.mcmcspatsurv, frailtylag1, spatialpars, hazardpars, fixedpars, randompars, baselinehazard, predict.mcmcspatsurv, priorposterior, posteriorcov, MCE, hazardexceedance

weibullHaz

weibullHaz function

Description

A function to define a parametric proportional hazards model where the baseline hazard is taken from the Weibull model. This function returns an object inheriting class 'basehazardspec', list of functions 'distinfo', 'basehazard', 'gradbasehazard', 'hessbasehazard', 'cumbasehazard', 'gradcumbasehazard', 'hesscumbasehazard' and 'densityquantile'

Usage

weibullHaz(MLinits = NULL)

Arguments

MLinits initial values for optim, default is NULL

Details

The distinfo function is used to provide basic distribution specific information to other spatsurv functions. The user is required to provide the following information in the returned list: npars, the number of parameters in this distribution; parnames, the names of the parameters; trans, the transformation scale on which the priors will be provided; itrans, the inverse transformation function that will be applied to the parameters before the hazard, and other functions are evaluated; jacobian, the derivative of the inverse transformation function with respect to each of the parameters; and hessian, the second derivatives of the inverse transformation function with respect to each of the parameters – note that currently the package spatsurv only allows the use of functions where the parameters are transformed independently.

The basehazard function is used to evaluate the baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, t and returns a vector.

The gradbasehazard function is used to evaluate the gradient of the baseline hazard function with respect to the parameters, this typically returns a vector. It returns a function that accepts as input a vector of times, t, and returns a matrix.

The hessbasehazard function is used to evaluate the Hessian of the baseline hazard function. It returns a function that accepts as input a vector of times, t and returns a list of hessian matrices corresponding to each t.
The `cumbasehazard` function is used to evaluate the cumulative baseline hazard function for the distribution of interest. It returns a function that accepts as input a vector of times, `t` and returns a vector.

The `gradcumbasehazard` function is used to evaluate the gradient of the cumulative baseline hazard function with respect to the parameters. This typically returns a vector. It returns a function that accepts as input a vector of times, `t`, and returns a matrix.

The `hesscumbasehazard` function is used to evaluate the Hessian of the cumulative baseline hazard function. It returns a function that accepts as input a vector of times, `t` and returns a list of hessian matrices corresponding to each `t`.

The `densityquantile` function is used to return quantiles of the density function. This is NOT REQUIRED for running the MCMC, merely for us in post-processing with the `predict` function where `type` is `densityquantile`. In the case of the Weibull model for the baseline hazard, it can be shown that the q-th quantile is:

Value

an object inheriting class 'basehazardspec'

See Also

tpowHaz, exponentialHaz, gompertzHaz, makehamHaz

---

### YfromGamma

**YfromGamma function**

Description

A function to change Gammas (white noise) into Ys (spatially correlated noise). Used in the MALA algorithm.

Usage

`YfromGamma(Gamma, invrootQeigs, mu)`

Arguments

- **Gamma**: Gamma matrix
- **invrootQeigs**: inverse square root of the eigenvectors of the precision matrix
- **mu**: parameter of the latent Gaussian field

Value

Y
YFromGamma_SPDE

YFromGamma_SPDE function

Description
A function to go from Gamma to Y

Usage
YFromGamma_SPDE(gamma, U, mu)

Arguments
- gamma: Gamma
- U: upper Cholesky matrix
- mu: the mean

Value
the value of Y for the given Gamma

References
Index

* datasets
  fs, 37
  fstimes, 38

* package
  spatsurv-package, 5
  .onAttach, 10
  allocate, 11
  alpha, 12
  B, 12
  basehazard, 13
  basehazard.basehazardspec, 13, 13
  baseHazST, 14
  baselinehazard, 15, 36, 37, 54, 71, 84–88, 97, 98, 106, 113, 124, 125
  baselinehazard_multiWay, 16
  betapriorGauss, 17, 17, 28, 29, 32, 59, 73, 78, 95
  blockDiag, 17
  boxplotRisk, 18
  Bspline.construct, 18
  BsplineHaz, 19
  checkSurvivalData, 20
  circulant, 21
  circulant.matrix, 21
  circulant.numeric, 22
  circulantij, 22
  covmodel, 23, 33, 58, 106, 107, 109, 117
  CSpot, 23
  cumbasehazard, 24
  cumbasehazard.basehazardspec, 24, 24
  cumulativeBspline.construct, 25
  density_PP, 27
  densityquantile, 25
  densityquantile.basehazardspec, 26, 26
  densityquantile_PP, 27
  derivindepGaussianprior, 17, 28, 28, 29, 32, 59, 73, 78, 95
  derivindepGaussianpriorST, 28
  derivpsplineprior, 29
  distinfo, 30
  distinfo.basehazardspec, 30, 30
  estimateY, 31
  Et_PP, 32
  etapriorGauss, 17, 28, 29, 31, 32, 59, 73, 78, 95
  EvalCov, 32
  ExponentialCovFct, 33, 107, 109
  exponentialHaz, 13, 14, 20, 24–26, 30, 31, 33, 47–50, 55–57, 70, 94, 103, 106, 117, 121, 126
  FFTgrid, 35
  fixedpars, 15, 35, 37, 54, 71, 84–88, 97, 98, 106, 113, 124, 125
  fixmatrix, 36
  fixParHaz, 36
  frailtylag1, 15, 36, 37, 54, 71, 84–88, 97, 98, 106, 113, 124, 125
  fs, 37
  fstimes, 38
  gamma2risk, 39
  GammafromY, 39
  GammaFromY_SPDE, 40
  gencens, 40
  getBackground, 41, 110
  getbb, 42
  getBbasis, 42
  getcov, 43
  getgrd, 43
  getGrid, 44
  getleneta, 44
  getOptCellwidth, 45
  getsurvdata, 46
quantile.mlspatsurv, 97
randompars, 15, 36, 37, 54, 71, 84–88, 97, 98, 106, 113, 124, 125
reconstruct.bs, 98
reconstruct.bs.coxph, 99
reconstruct.bs.mcmcspatsurv, 100
resetLoop, 101
residuals.mcmcspatsurv, 101
rootWeibullHaz, 102

setTextProgressBar2, 103
setupHazard, 103
setupPrecMatStruct, 104
showGrid, 105
simsurv, 105
spatialpars, 15, 36, 37, 54, 71, 84–88, 97, 98, 106, 113, 124, 125
spatsurv(spatsurv-package), 5
spatsurv-package, 5
spatsurvVignette, 107
SPDE, 107
SPDEprec, 108
SpikedExponentialCovFct, 33, 58, 108
spplot1, 109
spplot_compare, 111
Summarise, 112
summary.mcmc, 113
summary.mcmcspatsurv, 15, 36, 37, 54, 71, 84–88, 97, 98, 106, 113, 124, 125
surv3d, 114
survival_PP, 115
survspat, 17, 28, 29, 32, 59, 61, 73, 78, 95, 106, 116
survspatNS, 117
textSummary, 118
timevaryingPL, 119
tpowHaz, 13, 14, 24–26, 30, 31, 34, 47–50, 55–57, 70, 103, 106, 117, 120, 126
transformweibull, 121
TwoWayHazAdditive, 122
txtProgressBar2, 122

urlTemplate, 110, 123

vcov.mcmcspatsurv, 15, 36, 37, 54, 71, 84–88, 97, 98, 106, 113, 124
vcov.mlspatsurv, 124

weibullHaz, 13, 14, 20, 24–26, 30, 31, 34, 47–50, 55–57, 70, 94, 106, 117, 121, 125
YfromGamma, 126
YFromGamma_SPDE, 127