Package 'spBayesSurv'

May 17, 2023

Type Package

Title Bayesian Modeling and Analysis of Spatially Correlated Survival Data

Version 1.1.7

Date 2023-05-17

Description Provides several Bayesian survival models for spatial/non-spatial survival data: proportional hazards (PH), accelerated failure time (AFT), proportional odds (PO), and accelerated hazards (AH), a super model that includes PH, AFT, PO and AH as special cases, Bayesian nonparametric nonproportional hazards (LDDPM), generalized accelerated failure time (GAFT), and spatially smoothed Polya tree density estimation. The spatial dependence is modeled via frailties under PH, AFT, PO, AH and GAFT, and via copulas under LDDPM and PH. Model choice is carried out via the logarithm of the pseudo marginal likelihood (LPML), the deviance information criterion (DIC), and the Watanabe-Akaike information criterion (WAIC). See Zhou, Hanson and Zhang (2020) <doi:10.18637/jss.v092.i09>.

License GPL (>= 2)

Depends R (>= 4.0.0)

Imports Rcpp (>= 0.12.16), survival, coda, methods, MASS, fields, splines

LinkingTo Rcpp, RcppArmadillo (>= 0.8.500.0)

NeedsCompilation yes

Author Haiming Zhou [aut, cre, cph], Timothy Hanson [aut]

Maintainer Haiming Zhou <haiming2019@gmail.com>

Repository CRAN

Date/Publication 2023-05-17 19:00:02 UTC

R topics documented:

`anovaDDP` *Bayesian Nonparametric Survival Model*

Description

This function fits a Bayesian Nonparametric model (De Iorio et al., 2009) for non-spatial right censored time-to-event data. Note that the notations are different with those presented in the original paper; see Zhou, Hanson and Zhang (2018) for new examples.

Usage

```
anovaDDP(formula, data, na.action, prediction=NULL,
         mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
         prior=NULL, state=NULL, scale.designX=TRUE)
```

Arguments

`formula`	a formula expression with the response returned by the `Surv` function in the `survival` package. It currently only supports right-censoring.
`data`	a data frame in which to interpret the variables named in the `formula` argument.
`na.action`	a missing-data filter function, applied to the `model.frame`.
`prediction`	a list giving the information used to obtain conditional inferences. The list includes the following element: `xpred` giving the npred by p covariates matrix, used for prediction. If `prediction=NULL`, `xpred` will be set to be the design matrix used in `formula`.
`mcmc`	a list giving the MCMC parameters. The list must include the following elements: `nburn` an integer giving the number of burn-in scans, `nskip` an integer giving the thinning interval, `nsave` an integer giving the total number of scans to be saved, `ndisplay` an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every `ndisplay` iterations have been carried out).

prior	a list giving the prior information. See Zhou, Hanson and Zhang (2018) for more detailed hyperprior specifications.
state	a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.
scale.designX	flag to indicate wheter the design matrix X will be centered by column means and scaled by column standard deviations, where TRUE indicates yes. The default is TRUE for improving numerical stability. All returned posterior samples fit from scaled covariates. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when scale.designX=TRUE.

Details

This function fits a Bayesian Nonparametric model (De Iorio et al., 2009) for non-spatial right censored time-to-event data. Note that the notations are different with those presented in the original paper; see Zhou, Hanson and Zhang (2018) for new examples.

Value

The anovaDDP object is a list containing at least the following components:

n	the number of row observations used in fitting the model
p	the number of columns in the model matrix
Surv	the Surv object used
X.scaled	the n by p scaled design matrix
X	the n by p orginal design matrix
beta	the p+1 by N by nsave array of posterior samples for the coefficients
sigma2	the N by nsave matrix of posterior samples for sigma2 involved in the DDP.
w	the N by nsave matrix of posterior samples for weights involved in the DDP.
Tpred	the npred by nsave predicted survival times for covariates specified in the argument prediction.

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H., Hanson, T., and Zhang, J. (2020). spBayesSurv: Fitting Bayesian Spatial Survival Models Using R. *Journal of Statistical Software*, 92(9): 1-33.

Zhou, H., Hanson, T., and Knapp, R. (2015). Marginal Bayesian nonparametric model for time to disease arrival of threatened amphibian populations. *Biometrics*, 71(4): 1101-1110.

De Iorio, M., Johnson, W. O., Mueller, P., and Rosner, G. L. (2009). Bayesian nonparametric nonproportional hazards survival modeling. *Biometrics*, 65(3): 762-771.

See Also

spCopulaDDP, GetCurves

Examples

```
#################################################################
# A simulated data: mixture of two normals
#################################################################
rm(list=ls())
library(survival)
library(spBayesSurv)
library(coda)
## True parameters
betaT = cbind(c(3.5, 0.5), c(2.5, -1));
wT = c(0.4, 0.6);
sig2T = c(1^2, 0.5^2);
n=100;
## The Survival function for log survival times:
fiofy = function(y, xi, w=wT){
  nw = length(w);
  ny = length(y);
  res = matrix(0, ny, nw);
  Xi = c(1,xi);
  for (k in 1:nw){
    res[,k] = w[k]*dnorm(y, sum(Xi*betaT[,k]), sqrt(sig2T[k]) )
  }
  apply(res, 1, sum)
}
fioft = function(t, xi, w=wT) fiofy(log(t), xi, w)/t;
Fiofy = function(y, xi, w=wT){
  nw = length(w);
  ny = length(y);
  res = matrix(0, ny, nw);
  Xi = c(1,xi);
  for (k in 1:nw){
    res[,k] = w[k]*pnorm(y, sum(Xi*betaT[,k]), sqrt(sig2T[k]) )
  }
  apply(res, 1, sum)
}
Fioft = function(t, xi, w=wT) Fiofy(log(t), xi, w);
## The inverse for Fioft
Finv = function(u, x) uniroot(function (y) Fiofy(y,x)-u, lower=-250,
                              upper=250, extendInt ="yes", tol=1e-6)$root

## generate x
x1 = runif(n,-1.5,1.5); X = cbind(x1);
## generate survival times
u = runif(n);
tT = rep(0, n);
for (i in 1:n){
  tT[i] = exp(Finv(u[i], X[i,]));
}
```

```
### ----------- right-censored -------------###
t_obs=tT
Centime = runif(n, 20, 200);
delta = (tT<=Centime) +0 ;
length(which(delta==0))/n; # censoring rate
rcen = which(delta==0);
t_obs[rcen] = Centime[rcen]; ## observed time
## make a data frame
d = data.frame(tobs=t_obs, x1=x1, delta=delta, tT=tT);
table(d$delta)/n;

################################################################
# Independent DDP: Bayesian Nonparametric Survival Model
################################################################
# MCMC parameters
nburn=500; nsave=500; nskip=0;
# Note larger nburn, nsave and nskip should be used in practice.
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=1000);
prior = list(N=10, a0=2, b0=2);
# Fit the Cox PH model
res1 = anovaDDP(formula = Surv(tobs, delta)~x1, data=d,
                prior=prior, mcmc=mcmc);
## LPML
LPML = sum(log(res1$cpo)); LPML;
## Number of non-negligible components
quantile(colSums(res1$w>0.05))

#########################################
## Curves
#########################################
ygrid = seq(0,6.0,length=100); tgrid = exp(ygrid);
xpred = data.frame(x1=c(-1, 1))
plot(res1, xnewdata=xpred, tgrid=tgrid);
```

`baseline` *Stratification effects on baseline functions*

Description

This function allows one to add a simple baseline stratification term to the generalized AFT model.

Usage

```
baseline(...)
```

Arguments

`...`	stratification variables to be entered; see the example in `frailtyGAFT`.

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H., Hanson, T., and Zhang, J. (2020). spBayesSurv: Fitting Bayesian Spatial Survival Models Using R. *Journal of Statistical Software*, 92(9): 1-33.

Zhou, H., Hanson, T., and Zhang, J. (2017). Generalized accelerated failure time spatial frailty model for arbitrarily censored data. *Lifetime Data Analysis*, 23(3): 495-515..

`bspline` *Generate a Cubic B-Spline Basis Matrix*

Description

Generate the B-spline basis matrix for a cubic spline with the first and last columns dropped.

Usage

```
bspline(x, df=NULL, knots=NULL, Boundary.knots = range(x))
```

Arguments

`x`	the predictor variable. Missing values are allowed.
`df`	degrees of freedom; one can specify `df` rather than `knots`; the function chooses `df-2` inner knots at suitable quantile of `x` (which will ignore missing values). The default, `NULL`, corresponds to one inner knots, i.e. `df=3`.
`knots`	the internal breakpoints that define the spline. The default is `NULL`, which corresponds the median for one knot, quantiles for more knots.
`Boundary.knots`	boundary points at which to anchor the B-spline basis (default the range of the non-NA data).

Author(s)

Haiming Zhou and Timothy Hanson

References

Hastie, T. J. (1992) Generalized additive models. Chapter 7 of *Statistical Models in S* eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.

See Also

`predict.bspline`

Examples

```
require(stats)
basis <- bspline(women$height, df = 5)
newX <- seq(58, 72, length.out = 51)
# evaluate the basis at the new data
predict(basis, newX)
```

`cox.snell.survregbayes` *Cox-Snell Diagnostic Plot*

Description

This function provides the Cox-Snell diagnostic plot (Zhou and Hanson, 2018) for fitting for Bayesian semiparametric survival models.

Usage

```
cox.snell.survregbayes(x, ncurves = 10, PLOT = TRUE)
```

Arguments

`x`	an object obtained from the function `survregbayes`.
`ncurves`	the number of posterior draws.
`PLOT`	a logical value indicating whether the estimated survival curves will be plotted.

Value

The function returns the plot (if `PLOT = TRUE`) and a list with the following components:

`resid`	the `Surv` object using the everaged residuals
`resid*`	the `*` goes from 1 to `ncurves`, and each is the `Surv` object using the residuals obtained from each posterior draw.
`St1`	the `n` by `ncurves` matrix of survival functions evaluated at left endpoints of the time interval.
`St2`	the `n` by `ncurves` matrix of survival functions evaluated at right endpoints of the time interval.
`Delta`	The status indicator: 0=right censored, 1=event at time, 2=left censored, 3=interval censored.

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H. and Hanson, T. (2018). A unified framework for fitting Bayesian semiparametric models to arbitrarily censored survival data, including spatially-referenced data. *Journal of the American Statistical Association*, 113(522): 571-581.

See Also

survregbayes

`frailtyGAFT` *Generalized Accelerated Failure Time Frailty Model*

Description

This function fits a generalized accelerated failure time frailty model (Zhou, et al., 2017) for clustered and/or areal-level time-to-event data. Note that the function arguments are slightly different with those presented in the original paper; see Zhou, Hanson and Zhang (2018) for new examples.

Usage

```
frailtyGAFT(formula, data, na.action,
            mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
            prior=NULL, state=NULL, Proximity=NULL, Coordinates=NULL,
            DIST=NULL, scale.designX=TRUE)
```

Arguments

`formula`	a formula expression with the response returned by the `Surv` function in the `survival` package. It supports right-censoring, left-censoring, interval-censoring, and mixtures of them. To include CAR frailties, add `frailtyprior("car",ID)` to the formula, where `ID` is an n dimensional vector of cluster ID numbers. Furthermore, use `frailtyprior("iid",ID)` for Gaussian exchangeable frailties, use `frailtyprior("grf",ID)` for Gaussian random fields (GRF) frailties, and exclude the term `frailtyprior()` for non-frailty models. Note: the data need to be sorted by ID.
`data`	a data frame in which to interpret the variables named in the `formula` argument.
`na.action`	a missing-data filter function, applied to the `model.frame`.
`mcmc`	a list giving the MCMC parameters. The list must include the following elements: `nburn` an integer giving the number of burn-in scans, `nskip` an integer giving the thinning interval, `nsave` an integer giving the total number of scans to be saved, `ndisplay` an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every `ndisplay` iterations have been carried out).
`prior`	a list giving the prior information. See Zhou, Hanson and Zhang (2018) for more detailed hyperprior specifications.

state	a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.
Proximity	an m by m symetric adjacency matrix, where m is the number of clusters/regions. If CAR frailty model is specified in the formula, Proximity is required; otherwise it is ignored. Note: this matrix should be specified according to the data that have been sorted by ID.
Coordinates	an m by d coordinates matrix, where m is the number of clusters/regions, d is the dimension of coordiantes. If GRF frailty model is specified in the formula, Coordinates is required; otherwise it is ignored. Note: this matrix should be specified according to the data that have been sorted by ID.
DIST	This is a function argument, used to calculate the distance. The default is Euclidean distance (fields::rdist). This function should have two arguments (X1,X2), where X1 and X2 are matrices with coordinates as the rows. The returned value of this function should be the pairwise distance matrix. If nrow(X1)=m and nrow(X2)=n then the function should return an m by n matrix of all distances between these two sets of points.
scale.designX	flag to indicate whether the design matrix X and Xtf will be centered by column means and scaled by column standard deviations, where TRUE indicates yes. The default is TRUE for improving numerical stability. Even when it is scaled, the reported regression coefficients are in original scales. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when scale.designX=TRUE.

Details

This function fits a a generalized accelerated failure time frailty model (Zhou, et al., 2017) for clustered and/or areal-level time-to-event data. Note that the function arguments are slightly different with those presented in the original paper of Zhou, et al. (2017); see Zhou, Hanson and Zhang (2018) for new examples.

Value

The frailtyGAFT object is a list containing at least the following components:

modelname	the name of the fitted model
terms	the terms object used
coefficients	a named vector of coefficients. The last two elements are the estimates of scale parameter sigma and precision parameter alpha involved in the LDTFP prior.
call	the matched call
prior	the list of hyperparameters used in all priors.
mcmc	the list of MCMC parameters used
n	the number of row observations used in fitting the model
pce	the number of columns in the model matrix including the intercept
ptf	the number of columns in the model matrix used in the LDTFP baseline including the intercept

`Surv`	the `Surv` object used
`X.scaled`	the n by pce-1 scaled design matrix
`X`	the n by pce-1 orginal design matrix
`Xtf.scaled`	the n by ptf-1 scaled design matrix used in the LDTFP baseline
`Xtf`	the n by ptf-1 orginal design matrix used in the LDTFP baseline
`sigma2`	the vector of posterior samples for the variance parameter used in the LDTFP prior.
`beta`	the pce by nsave matrix of posterior samples for the coefficients in the `linear.predictors` which includes the intercept
`beta.scaled`	the pce by nsave matrix of posterior samples for the coefficients in the `linear.predictors`. Note that these posterior samples are based scaled design matrix.
`alpha`	the vector of posterior samples for the precision parameter alpha in the LDTFP prior.
`maxL`	the truncation level used in the LDTFP prior.
`logt`	the n by nsave matrix of posterior samples for log survival times.
`cpo`	the length n vector of the stabilized estiamte of CPO; used for calculating LPML
`accept_beta`	the acceptance rate in the posterior sampling of beta coefficient vector
`frail.prior`	the frailty prior used in `frailtyprior`
`BF`	the Bayes factors for testing necessariness of each stratification covariate.

The object will also have the following components when frailty models are fit:

`v`	the nID by nsave matrix of posterior samples for frailties, where nID is the number of clusters considered.
`tau2`	the vector of posterior samples for tau2 involved in the IID, GRF or CAR frailty prior.
`ID`	the cluster ID used in `frailtyprior`

If GRF frailties are used, the object will also have:

`Coordinates`	the `Coordinates` matrix used in `survregbayes`
`ratephi`	the acceptance rates in the posterior sampling of phi involved in the GRF prior
`phi`	the vector of posterior samples for phi involved in the GRF prior

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H., Hanson, T., and Zhang, J. (2020). spBayesSurv: Fitting Bayesian Spatial Survival Models Using R. *Journal of Statistical Software*, 92(9): 1-33.

Zhou, H., Hanson, T., and Zhang, J. (2017). Generalized accelerated failure time spatial frailty model for arbitrarily censored data. *Lifetime Data Analysis*, 23(3): 495-515.

See Also

baseline, frailtyprior, survregbayes, rdist

Examples

```
#################################################################
# A simulated data: GAFT spatial frailty model
#################################################################
rm(list=ls())
library(survival)
library(spBayesSurv)
library(coda)
library(MASS)

## True densities
set.seed(1)
Finvsingle = function(u, F) {
  res = uniroot(function (x) F(x)-u, lower=-1000, upper=1000,
                tol=.Machine$double.eps^0.5);
  res$root
}
Finv = function(u, F) {sapply(u, Finvsingle, F)};
f0 = function(x) dnorm(x, 0, 0.8);
F0 = function(x) pnorm(x, 0, 0.8);
shift=1
f1 = function(x) 0.5*dnorm(x, -shift, 0.5) + 0.5*dnorm(x, shift, 0.5)
F1 = function(x) 0.5*pnorm(x, -shift, 0.5) + 0.5*pnorm(x, shift, 0.5);
ff = function(x, xtf=0) {
  if(xtf==0) {res=f0(x)} else res=f1(x)
  res
}
FF = function(x, xtf=0){
  if(xtf==0) {res=F0(x)} else res=F1(x)
  res
}

# Simulation settings;
betaT = c(-1, 1, -0.5);
tau2T = 0.1;
m = 50; # blocks
mi = 2;
mis = rep(mi, m);
id = rep(1:m,mis);
n = length(id); # Total number of subjects
# Generate symmetric adjaceny matrix, W
wi = rep(0, m)
while(any(wi==0)){
  W = matrix(0,m,m)
  W[upper.tri(W,diag=FALSE)]<-rbinom(m*(m-1)/2,1,.1)
  W = W+t(W)
  wi = apply(W,1,sum)  # No. neighbors
}
```

```
# Spatial effects, v
Wstar = matrix(0, m-1, m-1);
Dstar = diag(wi[-m]);
for(i in 1:(m-1)){
  for(j in 1:(m-1)){
    Wstar[i,j] = W[j,i]-W[j,m]-W[m,i]-wi[m]
  }
}
Qstar = Dstar-Wstar;
covT = tau2T*solve(Qstar);
v0 = mvrnorm(1, mu=rep(0,m-1), Sigma=covT);
v = c(v0,-sum(v0));
vn = rep(v, mis);

# responses
x1 = rnorm(n, 0, 1);
x2 = rbinom(n, 1, 0.5);
xtf = x2; ptf = 2;
X = cbind(1,x1,x2); pce = ncol(X);
u = runif(n, 0, 1)
y = rep(0, n);
for(i in 1:n) {
  if(x2[i]==1) {
    y[i] = sum(betaT*X[i,]) + vn[i] + Finv(u[i], F1)
  }else{
    y[i] = sum(betaT*X[i,]) + vn[i] + Finv(u[i], F0)
  }
}

# generate responses
Cen = runif(n, 0.5, 1)
delta = (exp(y)<=Cen)+0;
sum(delta)/n
tTrue = exp(y);
tobs = cbind(tTrue, tTrue);
tobs[which(delta==0),] = cbind(Cen[which(delta==0)], NA);
dtotal = data.frame(tleft=tobs[,1], tright=tobs[,2], x1=x1,
                    x2=x2, xtf=x2, ID=id, tTrue=tTrue, censor=delta);
## sort the data by ID
d = dtotal[order(dtotal$ID),];

# Prior information and MCMC
fit0 <- survival::survreg(Surv(tleft, censor)~x1+x2, dist="lognormal", data=d);
prior = list(maxL = 4, a0 = 5, b0 = 1);
mcmc=list(nburn=200, nsave=200, nskip=0, ndisplay=100)
# Note larger nburn, nsave and nskip should be used in practice.

# Fit the model
ptm<-proc.time()
res = frailtyGAFT(Surv(tleft, tright, type="interval2")~x1+x2+baseline(x1, x2)+
                    frailtyprior(prior="car", ID),  data=d, mcmc=mcmc, prior=prior,
                  Proximity=W);
summary(res);
```

```
systime1=proc.time()-ptm; systime1;

### trace plots
par(mfrow = c(3,1))
traceplot(mcmc(res$beta[1,]), main="beta1");
traceplot(mcmc(res$beta[2,]), main="beta2");
traceplot(mcmc(res$beta[3,]), main="beta3");

#####################################################################
## Get curves
#####################################################################
par(mfrow = c(1,1))
xpred = data.frame(x1=c(1,1.5), x2=c(0,1))
xtfpred = xpred;
plot(res, xnewdata=xpred, xtfnewdata=xtfpred, CI=0.9);
```

`frailtyprior` *Frailty prior specification*

Description

This function allows one to add a frailty term to the linear predictor of semiparametric PH, PO and AFT models.

Usage

```
frailtyprior(prior="car", ...)
```

Arguments

`prior`	a name string to be entered, e.g, `"car"`, `"iid"`, or `"grf"`; see the example in `frailtyGAFT` and `survregbayes2`.
`...`	Cluster ID to be entered for clustered data or locations for point-referenced data; see the example in `frailtyGAFT` and `survregbayes2`.

Author(s)

Haiming Zhou and Timothy Hanson

GetCurves *Density, Survival, and Hazard Estimates*

Description

This function estimates density, survival, and hazard functions given covariates.

Usage

```
GetCurves(x, xnewdata, xtfnewdata, tgrid = NULL, ygrid = NULL,
          frail = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'survregbayes'
plot(x, xnewdata, tgrid = NULL,
     frail = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'frailtyGAFT'
plot(x, xnewdata, xtfnewdata, tgrid = NULL,
     frail = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'SuperSurvRegBayes'
plot(x, xnewdata, tgrid = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'indeptCoxph'
plot(x, xnewdata, tgrid = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'anovaDDP'
plot(x, xnewdata, tgrid = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'spCopulaCoxph'
plot(x, xnewdata, tgrid = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'spCopulaDDP'
plot(x, xnewdata, tgrid = NULL, CI = 0.95, PLOT = TRUE, ...)
## S3 method for class 'SpatDensReg'
plot(x, xnewdata, ygrid = NULL, CI = 0.95, PLOT = TRUE, ...)
```

Arguments

x	an object obtained from the functions survregbayes, frailtyGAFT, SuperSurvRegBayes, indeptCoxph, anovaDDP, spCopulaCoxph, spCopulaDDP and SpatDensReg.
xnewdata	A data frame in which to look for variables with which to obtain estimated curves.
xtfnewdata	A data frame in which to look for variables with which to obtain estimated curves, used only for frailtyGAFT.
tgrid	a vector of grid points indicating where the curves will be estimated.
ygrid	a vector of grid points indicating where the curves will be estimated, used only for SpatDensReg.
frail	an optional matrix of posterior frailty values for survregbayes and frailtyGAFT, where the rows refer to clusters/regions and the number of columns is the length of thined MCMC chain. The default is to set frailties to be zero.
CI	a numeric value indicating the level of credible interval.

PLOT	a logical value indicating whether the estimated survival curves will be plotted.
...	further arguments to be passed to or from other methods.

Details

This function estimates density, survival, and hazard functions given covariates.

Value

Use names to find out what they are, where fhat represents density, Shat represents survival, hhat represents hazard. The credible bands are also provided, e.g., Shatlow represents the lower band and Shatup represents the upper band.

Author(s)

Haiming Zhou and Timothy Hanson

See Also

survregbayes, frailtyGAFT, SuperSurvRegBayes, indeptCoxph, anovaDDP, spCopulaCoxph, spCopulaDDP and SpatDensReg

---

indeptCoxph *Bayesian Proportional Hazards Model*

Description

This function fits a Bayesian proportional hazards model (Zhou, Hanson and Zhang, 2018) for non-spatial right censored time-to-event data.

Usage

```
indeptCoxph(formula, data, na.action, prediction=NULL,
            mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
            prior=NULL, state=NULL, scale.designX=TRUE)
```

Arguments

formula	a formula expression with the response returned by the Surv function in the survival package. It currently only supports right-censoring.
data	a data frame in which to interpret the variables named in the formula argument.
na.action	a missing-data filter function, applied to the model.frame.
prediction	a list giving the information used to obtain conditional inferences. The list includes the following element: xpred giving the npred by p covariates matrix, used for prediction. If prediction=NULL, xpred will be set to be the design matrix used in formula.

mcmc	a list giving the MCMC parameters. The list must include the following elements: nburn an integer giving the number of burn-in scans, nskip an integer giving the thinning interval, nsave an integer giving the total number of scans to be saved, ndisplay an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every ndisplay iterations have been carried out).
prior	a list giving the prior information. See Zhou, Hanson and Zhang (2018) for more detailed hyperprior specifications.
state	a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.
scale.designX	flag to indicate wheter the design matrix X will be centered by column means and scaled by column standard deviations, where TRUE indicates yes. The default is TRUE for improving numerical stability. Even when it is scaled, the reported regression coefficients are in original scales. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when scale.designX=TRUE.

Details

This function fits a Bayesian proportional hazards model (Zhou, Hanson and Zhang, 2018) for non-spatial right censored time-to-event data.

Value

The indeptCoxph object is a list containing at least the following components:

modelname	the name of the fitted model
terms	the terms object used
coefficients	a named vector of coefficients.
call	the matched call
prior	the list of hyperparameters used in all priors.
mcmc	the list of MCMC parameters used
n	the number of row observations used in fitting the model
p	the number of columns in the model matrix
Surv	the Surv object used
X.scaled	the n by p scaled design matrix
X	the n by p orginal design matrix
beta	the p by nsave matrix of posterior samples for the coefficients in the linear.predictors
beta.scaled	the p by nsave matrix of posterior samples for the coefficients in the linear.predictors. Note that these posterior samples are based scaled design matrix.
ratebeta	the acceptance rate in the posterior sampling of beta coefficient vector
cpo	the length n vector of the stabilized estiamte of CPO; used for calculating LPML
Tpred	the npred by nsave predicted survival times for covariates specified in the argument prediction.

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H., Hanson, T., and Zhang, J. (2020). spBayesSurv: Fitting Bayesian Spatial Survival Models Using R. *Journal of Statistical Software*, 92(9): 1-33.

Zhou, H., Hanson, T., and Knapp, R. (2015). Marginal Bayesian nonparametric model for time to disease arrival of threatened amphibian populations. *Biometrics*, 71(4): 1101-1110.

See Also

spCopulaCoxph, GetCurves

Examples

```
#################################################################
# A simulated data: Cox PH
#################################################################
rm(list=ls())
library(survival)
library(spBayesSurv)
library(coda)
## True parameters
betaT = c(1,1);
n=100;
## Baseline Survival
f0oft = function(t) 0.5*dlnorm(t, -1, 0.5)+0.5*dlnorm(t,1,0.5);
S0oft = function(t) (0.5*plnorm(t, -1, 0.5, lower.tail=FALSE)+
                       0.5*plnorm(t, 1, 0.5, lower.tail=FALSE))
## The Survival function:
Sioft = function(t,x)  exp( log(S0oft(t))*exp(sum(x*betaT)) ) ;
fioft = function(t,x) exp(sum(x*betaT))*f0oft(t)/S0oft(t)*Sioft(t,x);
Fioft = function(t,x) 1-Sioft(t,x);
## The inverse for Fioft
Finv = function(u, x) uniroot(function (t) Fioft(t,x)-u, lower=1e-100,
                              upper=1e100, extendInt ="yes", tol=1e-6)$root

## generate x
x1 = rbinom(n, 1, 0.5); x2 = rnorm(n, 0, 1); X = cbind(x1, x2);
## generate survival times
u = runif(n);
tT = rep(0, n);
for (i in 1:n){
  tT[i] = Finv(u[i], X[i,]);
}

### ----------- right-censored -------------###
t_obs=tT
Centime = runif(n, 2, 6);
delta = (tT<=Centime) +0 ;
```

```
length(which(delta==0))/n; # censoring rate
rcen = which(delta==0);
t_obs[rcen] = Centime[rcen]; ## observed time
## make a data frame
d = data.frame(tobs=t_obs, x1=x1, x2=x2, delta=delta, tT=tT);
table(d$delta)/n;

#################################################################
# Independent Cox PH
#################################################################
# MCMC parameters
nburn=500; nsave=300; nskip=0;
# Note larger nburn, nsave and nskip should be used in practice.
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=1000);
prior = list(M=10, r0=1);
# Fit the Cox PH model
res1 = indeptCoxph(formula = Surv(tobs, delta)~x1+x2, data=d,
                   prior=prior, mcmc=mcmc);
sfit1=summary(res1); sfit1;
## traceplot
par(mfrow = c(2,2))
traceplot(mcmc(res1$beta[1,]), main="beta1")
traceplot(mcmc(res1$beta[2,]), main="beta2")
traceplot(mcmc(res1$h.scaled[2,]), main="h")
traceplot(mcmc(res1$h.scaled[3,]), main="h")

#########################################
## Curves
#########################################
par(mfrow = c(1,1))
tgrid = seq(1e-10,4,0.1);
xpred = data.frame(x1=c(0,0), x2=c(0,1));
plot(res1, xnewdata=xpred, tgrid=tgrid);
```

LeukSurv *The Leukemia Survival Data*

Description

A dataset on the survival of acute myeloid leukemia in 1,043 pateietns, first analyzed by Henderson et al. (2002). It is of interest to investigate possible spatial variation in survival after accounting for known subject-specific prognostic factors, which include age, sex, white blood cell count (wbc) at diagnosis, and the Townsend score (tpi) for which higher values indicates less affluent areas. Both exact residential locations of all patients and their administrative districts (24 districts that make up the whole region) are available.

Usage

```
data(LeukSurv)
```

Format

time:	survival time in days
cens:	right censoring status 0=censored, 1=dead
xcoord:	coordinates in x-axis of residence
ycoord:	coordinates in y-axis of residence
age:	age in years
sex:	male=1 female=0
wbc:	white blood cell count at diagnosis, truncated at 500
tpi:	the Townsend score for which higher values indicates less affluent areas
district:	administrative district of residence

Source

Henderson, R., Shimakura, S., and Gorst, D. (2002), Modeling spatial variation in leukemia survival data, *Journal of the American Statistical Association*, 97(460), 965-972.

Examples

```
data(LeukSurv)
head(LeukSurv)
```

`predict.bspline` *Evaluate a Cubic Spline Basis*

Description

Evaluate a predefined spline basis at given values.

Usage

```
## S3 method for class 'bspline'
predict(object, newx, ...)
```

Arguments

`object`	the result of a call to `bspline` having attributes describing knots, df, etc..
`newx`	the `x` values at which evaluations are required.
`...`	Optional additional arguments. At present no additional arguments are used.

Author(s)

Haiming Zhou and Timothy Hanson

See Also

`bspline`

Examples

```
require(stats)
basis <- bspline(women$height, df = 5)
newX <- seq(58, 72, length.out = 51)
# evaluate the basis at the new data
predict(basis, newX)
```

SpatDensReg *Bayesian Nonparametric Spatially Smoothed Density Estimation*

Description

This function provides a Bayesian nonparametric density estimator that changes smoothly in space. The estimator is built from the predictive rule for a marginalized Polya tree (PT), modified so that observations are spatially weighted by their distance from the location of interest.

Usage

```
SpatDensReg(formula, data, na.action, prior=NULL, state=NULL,
            mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
            permutation=TRUE, fix.theta=TRUE)
```

Arguments

formula	a formula expression with the response returned by the Surv function in the survival package. It supports right-censoring, left-censoring, interval-censoring, and mixtures of them. Note: for survival data, the input response should be log survival times.
data	a data frame in which to interpret the variables named in the formula argument.
na.action	a missing-data filter function, applied to the model.frame.
prior	a list giving the prior information. The list includes the following parameter: maxL an integer giving the maximum level of Polya trees, (a0, b0) parameters of the gamma prior for the precision parameter alpha, (theta0, V0) mean and variance of the normal prior for the centering distribution parameter vector theta, phiq0 the prior probability of phi=0, phib0 the rate parameter of the exponential prior of phi. The function itself provides all default priors. See Hanson, Zhou, and de Carvalho (2018).
state	a list giving the current value of the parameters. If NULL, all values are provided based on the centering parametric model.
mcmc	a list giving the MCMC parameters. The list must include the following elements: nburn an integer giving the number of burn-in scans, nskip an integer giving the thinning interval, nsave an integer giving the total number of scans to be saved, ndisplay an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every ndisplay iterations have been carried out).

permutation	flag to indicate whether a random data permutation will be implemented in the beginning of each iterate; default is TRUE.
fix.theta	flag to indicate whether the centering distribution parameters theta=(location, log(scale)) are fixed; default is TRUE indicating fixed.

Value

The SpatDensReg object is a list containing at least the following components:

modelname	the name of the fitted model
terms	the terms object used
call	the matched call
prior	the list of hyperparameters used in all priors.
mcmc	the list of MCMC parameters used
n	the number of row observations used in fitting the model
p	the number of columns in the model matrix
Surv	the Surv object used
X	the n by p design matrix
theta	the 2 by nsave matrix of posterior samples for location and log(scale) involved in the centering distribution of Polya tree prior.
phi	the vector of posterior samples for the phi parameter in the marginal PT definition.
alpha	the vector of posterior samples for the precision parameter alpha in the PT prior.
maxL	the truncation level used in the PT prior.
Surv.cox.snell	the Surv object used for Cox-Snell residual plot. This is not recommended for frailty models, for which please use the function cox.snell.survregbayes.
ratetheta	the acceptance rate in the posterior sampling of theta vector involved in the centering distribution
ratec	the acceptance rate in the posterior sampling of precision parameter alpha involved in the PT prior
ratephi	the acceptance rate in the posterior sampling of phi parameter
initial.values	the list of initial values used for the parameters
BF	the Bayes factor for comparing the spatial model vs. the exchangeable model

Author(s)

Haiming Zhou and Timothy Hanson

References

Hanson, T., Zhou, H., and de Carvalho, V. (2018). Bayesian nonparametric spatially smoothed density estimation. In *New Frontiers of Biostatistics and Bioinformatics* (pp 87-105). Springer.

Examples

```
## Simulated data
rm(list=ls())
library(survival)
library(spBayesSurv)
library(coda)

## True conditional density
fiofy_x = function(y, x){
  0.5*dnorm(y, -x, 1)+0.5*dnorm(y, x, 1);
}

## Generate data
n = 200;
x = runif(n, 0, 3)
y = rep(0, n);
uu = runif(n);
for(i in 1:n){
  if(uu[i]<0.5){
    y[i] = rnorm(1, -x[i], 1);
  }else{
    y[i] = rnorm(1, x[i], 1);
  }
}

## right censored
y1=y;y2=y;
Centime = runif(n, 2, 4);
delta = (y<=Centime) +0 ;
length(which(delta==0))/n; ## censoring rate
rcen = which(delta==0);
y1[rcen] = Centime[rcen];
y2[rcen] = NA;
## make a data frame
## Method 1: in the interval-censoring notation:
## y1 is the left endpoint and y2 is the right endpoint.
## This way we could use Surv(y1, y2, type="interval2")
## Method 2: Because we have right-censored data,
## we could use y1 as the observed survival times and delta as the indicator.
## This way we could use Surv(y1, delta). This is the same as above.
d = data.frame(y1=y1, y2=y2, x=x, delta=delta);

##-------------fit the model-------------------##
# MCMC parameters
nburn=50; nsave=50; nskip=0;
# Note larger nburn, nsave and nskip should be used in practice.
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=50);
prior = list(maxL=4, phiq0=0);
# Note please set 0<phiq0<1 for a valid Bayes factor of testing
# spatial model vs. exchangeable model.
# If the Bayes factor is not needed, setting phiq0=0 will speed up
# the computing time about seven times.
```

```
state = list(alpha=1);
ptm<-proc.time()
res1 = SpatDensReg(formula = Surv(y1, delta)~x, data=d,
                   prior=prior, state=state, mcmc=mcmc, permutation = TRUE,
                   fix.theta=FALSE);
## Or equivalently formula = Surv(y1, y2, type="interval2") can also be used.
sfit=summary(res1); sfit
systime1=proc.time()-ptm; systime1;
traceplot(mcmc(res1$theta[1,]))
traceplot(mcmc(res1$theta[2,]))
traceplot(mcmc(res1$alpha))
traceplot(mcmc(res1$phi))

## plots
ygrid = seq(-6, 6,length.out=100);
xpred = data.frame(x=c(0,1,2,3));
plot(res1, xnewdata=xpred, ygrid=ygrid);
```

---

spCopulaCoxph *Marginal Bayesian Proportional Hazards Model via Spatial Copula*

Description

This function fits a marginal Bayesian proportional hazards model (Zhou, Hanson and Zhang, 2018) for point-referenced right censored time-to-event data.

Usage

```
spCopulaCoxph(formula, data, na.action, prediction=NULL,
              mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
              prior=NULL, state=NULL, scale.designX=TRUE,
              Coordinates, DIST=NULL, Knots=NULL)
```

Arguments

formula	a formula expression with the response returned by the Surv function in the survival package. It currently only supports right-censoring.
data	a data frame in which to interpret the variables named in the formula argument.
na.action	a missing-data filter function, applied to the model.frame.
prediction	a list giving the information used to obtain conditional inferences. The list includes the following elements: spred and xpred giving the n by 2 new locations and corresponding npred by p covariates matrix, respectively, used for prediction. If prediction=NULL, xpred will be set to be the design matrix used in formula, and spred will be set to be in Coordinates.
mcmc	a list giving the MCMC parameters. The list must include the following elements: nburn an integer giving the number of burn-in scans, nskip an integer giving the thinning interval, nsave an integer giving the total number of scans to

be saved, ndisplay an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every ndisplay iterations have been carried out).

prior — a list giving the prior information. See Zhou, Hanson and Zhang (2018) for more detailed hyperprior specifications.

state — a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.

scale.designX — flag to indicate wheter the design matrix X will be centered by column means and scaled by column standard deviations, where TRUE indicates yes. The default is TRUE for improving numerical stability. Even when it is scaled, the reported regression coefficients are in original scales. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when scale.designX=TRUE.

Coordinates — an n by 2 coordinates matrix, where n is the sample size, 2 is the dimension of coordiantes. Note all cocordinates should be distinct.

DIST — This is a function argument, used to calculate the distance. The default is Euclidean distance (fields::rdist). This function should have two arguments (X1,X2), where X1 and X2 are matrices with coordinates as the rows. The returned value of this function should be the pairwise distance matrix. If nrow(X1)=m and nrow(X2)=n then the function should return an m by n matrix of all distances between these two sets of points.

Knots — an nknots by 2 matrix, where nknots is the number of selected knots for FSA, and 2 is the dimension of each location. If Knots is not specified, the space-filling algorithm will be used to find the knots.

Details

This function fits a marginal Bayesian proportional hazards model (Zhou, Hanson and Zhang, 2018) for point-referenced right censored time-to-event data.

Value

The spCopulaCoxph object is a list containing at least the following components:

modelname — the name of the fitted model

terms — the terms object used

coefficients — a named vector of coefficients.

call — the matched call

prior — the list of hyperparameters used in all priors.

mcmc — the list of MCMC parameters used

n — the number of row observations used in fitting the model

p — the number of columns in the model matrix

Surv — the Surv object used

X.scaled — the n by p scaled design matrix

X	the n by p orginal design matrix
beta	the p by nsave matrix of posterior samples for the coefficients in the linear.predictors
beta.scaled	the p by nsave matrix of posterior samples for the coefficients in the linear.predictors. Note that these posterior samples are based scaled design matrix.
theta	the 2 by nsave matrix of posterior samples for sill and range parameters
ratebeta	the acceptance rate in the posterior sampling of beta coefficient vector
ratetheta	the acceptance rate in the posterior sampling of theta
cpo	the length n vector of the stabilized estiamte of CPO; used for calculating LPML
Coordinates	the Coordinates matrix used in survregbayes
Tpred	the npred by nsave predicted survival times for covariates specified in the argument prediction.
Zpred	the npred by nsave predicted z values for covariates specified in the argument prediction.

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H., Hanson, T., and Zhang, J. (2020). spBayesSurv: Fitting Bayesian Spatial Survival Models Using R. *Journal of Statistical Software*, 92(9): 1-33.

Zhou, H., Hanson, T., and Knapp, R. (2015). Marginal Bayesian nonparametric model for time to disease arrival of threatened amphibian populations. *Biometrics*, 71(4): 1101-1110.

See Also

spCopulaDDP, GetCurves

Examples

```
#################################################################
# A simulated data: spatial Copula Cox PH
#################################################################
rm(list=ls())
library(survival)
library(spBayesSurv)
library(coda)
## True parameters
betaT = c(1,1);
theta1 = 0.98; theta2 = 0.1;
n=50; npred=3; ntot = n+npred;
## Baseline Survival
f0oft = function(t) 0.5*dlnorm(t, -1, 0.5)+0.5*dlnorm(t,1,0.5);
S0oft = function(t) (0.5*plnorm(t, -1, 0.5, lower.tail=FALSE)+
                       0.5*plnorm(t, 1, 0.5, lower.tail=FALSE))
## The Survival function:
Sioft = function(t,x)  exp( log(S0oft(t))*exp(sum(x*betaT)) ) ;
```

```
fioft = function(t,x) exp(sum(x*betaT))*f0oft(t)/S0oft(t)*Sioft(t,x);
Fioft = function(t,x) 1-Sioft(t,x);
## The inverse for Fioft
Finv = function(u, x) uniroot(function (t) Fioft(t,x)-u, lower=1e-100,
                              upper=1e100, extendInt ="yes", tol=1e-6)$root

## generate coordinates:
## npred is the # of locations for prediction
ldist = 100; wdist = 40;
s1 = runif(ntot, 0, wdist); s2 = runif(ntot, 0, ldist);
s = cbind(s1,s2); #plot(s[,1], s[,2]);
## Covariance matrix
corT = matrix(1, ntot, ntot);
for (i in 1:(ntot-1)){
  for (j in (i+1):ntot){
    dij = sqrt(sum( (s[i,]-s[j,])^2 ));
    corT[i,j] = theta1*exp(-theta2*dij);
    corT[j,i] = theta1*exp(-theta2*dij);
  }
}

## generate x
x1 = rbinom(ntot, 1, 0.5); x2 = rnorm(ntot, 0, 1); X = cbind(x1, x2);
## generate transformed log of survival times
z = MASS::mvrnorm(1, rep(0, ntot), corT);
## generate survival times
u = pnorm(z);
tT = rep(0, ntot);
for (i in 1:ntot){
  tT[i] = Finv(u[i], X[i,]);
}

### ----------- right-censored -------------###
t_obs=tT
Centime = runif(ntot, 2, 6);
delta = (tT<=Centime) +0 ;
length(which(delta==0))/ntot; # censoring rate
rcen = which(delta==0);
t_obs[rcen] = Centime[rcen]; ## observed time
## make a data frame
dtot = data.frame(tobs=t_obs, x1=x1, x2=x2, delta=delta, tT=tT,
                  s1=s1, s2=s2);
## Hold out npred for prediction purpose
predindex = sample(1:ntot, npred);
dpred = dtot[predindex,];
d = dtot[-predindex,];
# Prediction settings
prediction = list(xpred=cbind(dpred$x1, dpred$x2),
                  spred=cbind(dpred$s1, dpred$s2));

#################################################################
# Independent Cox PH
#################################################################
```

```
# MCMC parameters
nburn=500; nsave=500; nskip=0;
# Note larger nburn, nsave and nskip should be used in practice.
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=1000);
prior = list(M=10, r0=1, nknots=10, nblock=n);
# here nknots=10<n, so FSA will be used with nblock=n.
# As nknots is getting larger, the FSA is more accurate but slower
# As nblock is getting smaller, the FSA is more accurate but slower.
# In most applications, setting nblock=n works fine, which is the
# setting by not specifying nblock.
# If nknots is not specified or nknots=n, the exact covariance is used.
# Fit the Cox PH model
res1 = spCopulaCoxph(formula = Surv(tobs, delta)~x1+x2, data=d,
                     prior=prior, mcmc=mcmc, prediction=prediction,
                     Coordinates=cbind(d$s1,d$s2), Knots=NULL);
# here if prediction=NULL, prediction$xpred will be set as the design matrix
# in formula, and prediction$spred will be set as the Coordinates argument.
# Knots=NULL is the defualt setting, for which the knots will be generated
# using fields::cover.design() with number of knots equal to prior$nknots.
sfit1=summary(res1); sfit1;
## MSPE
mean((dpred$tT-apply(res1$Tpred, 1, median))^2);

## traceplot
par(mfrow = c(2,2))
traceplot(mcmc(res1$beta[1,]), main="beta1")
traceplot(mcmc(res1$beta[2,]), main="beta2")
traceplot(mcmc(res1$theta[1,]), main="sill")
traceplot(mcmc(res1$theta[2,]), main="range")

##########################################
## Curves
##########################################
par(mfrow = c(1,1))
tgrid = seq(1e-10,4,0.1);
xpred = data.frame(x1=c(0,0), x2=c(0,1));
plot(res1, xnewdata=xpred, tgrid=tgrid);
```

---

spCopulaDDP *Marginal Bayesian Nonparametric Survival Model via Spatial Copula*

Description

This function fits a marginal Bayesian Nonparametric model (Zhou, Hanson and Knapp, 2015) for point-referenced right censored time-to-event data. Note that the function arguments are slightly different with those presented in the original paper; see Zhou, Hanson and Zhang (2018) for new examples.

Usage

```
spCopulaDDP(formula, data, na.action, prediction=NULL,
            mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
            prior=NULL, state=NULL, scale.designX=TRUE,
            Coordinates, DIST=NULL, Knots=NULL)
```

Arguments

formula	a formula expression with the response returned by the Surv function in the survival package. It currently only supports right-censoring.
data	a data frame in which to interpret the variables named in the formula argument.
na.action	a missing-data filter function, applied to the model.frame.
prediction	a list giving the information used to obtain conditional inferences. The list includes the following elements: spred and xpred giving the n by 2 new locations and corresponding npred by p covariates matrix, respectively, used for prediction. If prediction=NULL, xpred will be set to be the design matrix used in formula, and spred will be set to be in Coordinates.
mcmc	a list giving the MCMC parameters. The list must include the following elements: nburn an integer giving the number of burn-in scans, nskip an integer giving the thinning interval, nsave an integer giving the total number of scans to be saved, ndisplay an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every ndisplay iterations have been carried out).
prior	a list giving the prior information. See Zhou, Hanson and Zhang (2018) for more detailed hyperprior specifications.
state	a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.
scale.designX	flag to indicate wheter the design matrix X will be centered by column means and scaled by column standard deviations, where TRUE indicates yes. The default is TRUE for improving numerical stability. Even when it is scaled, the reported regression coefficients are in original scales. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when scale.designX=TRUE.
Coordinates	an n by 2 coordinates matrix, where n is the sample size, 2 is the dimension of coordiantes. Note all cocordinates should be distinct.
DIST	This is a function argument, used to calculate the distance. The default is Euclidean distance (fields::rdist). This function should have two arguments (X1,X2), where X1 and X2 are matrices with coordinates as the rows. The returned value of this function should be the pairwise distance matrix. If nrow(X1)=m and nrow(X2)=n then the function should return an m by n matrix of all distances between these two sets of points.
Knots	an nknots by 2 matrix, where nknots is the number of selected knots for FSA, and 2 is the dimension of each location. If Knots is not specified, the space-filling algorithm will be used to find the knots.

Details

This function fits a marginal Bayesian Nonparametric model (Zhou, Hanson and Knapp, 2015) for point-referenced right censored time-to-event data. Note that the function arguments are slightly different with those presented in the original paper; see Zhou, Hanson and Zhang (2018) for new examples.

Value

The `spCopulaDDP` object is a list containing at least the following components:

`n`	the number of row observations used in fitting the model
`p`	the number of columns in the model matrix
`Surv`	the `Surv` object used
`X.scaled`	the n by p scaled design matrix
`X`	the n by p orginal design matrix
`beta`	the p+1 by N by nsave array of posterior samples for the coefficients
`sigma2`	the N by nsave matrix of posterior samples for sigma2 involved in the DDP.
`w`	the N by nsave matrix of posterior samples for weights involved in the DDP.
`theta`	the 2 by nsave matrix of posterior samples for partial sill and range involved in the Gaussian copula.
`Tpred`	the npred by nsave predicted survival times for covariates specified in the argument `prediction`.
`Zpred`	the npred by nsave predicted z values for covariates specified in the argument `prediction`.
`ratey`	the n-vector of acceptance rates for sampling censored survival times.
`ratebeta`	the N-vector of acceptance rates for sampling beta coefficients.
`ratesigma`	the N-vector of acceptance rates for sampling sigma2.
`ratetheta`	the acceptance rate for sampling theta.

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H., Hanson, T., and Zhang, J. (2020). spBayesSurv: Fitting Bayesian Spatial Survival Models Using R. *Journal of Statistical Software*, 92(9): 1-33.

Zhou, H., Hanson, T., and Knapp, R. (2015). Marginal Bayesian nonparametric model for time to disease arrival of threatened amphibian populations. *Biometrics*, 71(4): 1101-1110.

See Also

`anovaDDP`, `GetCurves`

Examples

```
#################################################################
# A simulated data: mixture of two normals with spatial dependence
#################################################################
rm(list=ls())
library(survival)
library(spBayesSurv)
library(coda)
## True parameters
betaT = cbind(c(3.5, 0.5), c(2.5, -1));
wT = c(0.4, 0.6);
sig2T = c(1^2, 0.5^2);
theta1 = 0.98; theta2 = 0.1;
n=30; npred=3; ntot = n+npred;
## The Survival function for log survival times:
fiofy = function(y, xi, w=wT){
  nw = length(w);
  ny = length(y);
  res = matrix(0, ny, nw);
  Xi = c(1,xi);
  for (k in 1:nw){
    res[,k] = w[k]*dnorm(y, sum(Xi*betaT[,k]), sqrt(sig2T[k]) )
  }
  apply(res, 1, sum)
}
fioft = function(t, xi, w=wT) fiofy(log(t), xi, w)/t;
Fiofy = function(y, xi, w=wT){
  nw = length(w);
  ny = length(y);
  res = matrix(0, ny, nw);
  Xi = c(1,xi);
  for (k in 1:nw){
    res[,k] = w[k]*pnorm(y, sum(Xi*betaT[,k]), sqrt(sig2T[k]) )
  }
  apply(res, 1, sum)
}
Fioft = function(t, xi, w=wT) Fiofy(log(t), xi, w);
## The inverse for Fioft
Finv = function(u, x) uniroot(function (y) Fiofy(y,x)-u, lower=-250,
                              upper=250, extendInt ="yes", tol=1e-6)$root

## generate coordinates:
## npred is the # of locations for prediction
ldist = 100; wdist = 40;
s1 = runif(ntot, 0, wdist); s2 = runif(ntot, 0, ldist);
s = cbind(s1,s2); #plot(s[,1], s[,2]);
## Covariance matrix
corT = matrix(1, ntot, ntot);
for (i in 1:(ntot-1)){
  for (j in (i+1):ntot){
    dij = sqrt(sum( (s[i,]-s[j,])^2 ));
    corT[i,j] = theta1*exp(-theta2*dij);
```

```
    corT[j,i] = theta1*exp(-theta2*dij);
  }
}

## generate x
x1 = runif(ntot,-1.5,1.5); X = cbind(x1);
## generate transformed log of survival times
z = MASS::mvrnorm(1, rep(0, ntot), corT);
## generate survival times
u = pnorm(z);
tT = rep(0, ntot);
for (i in 1:ntot){
  tT[i] = exp(Finv(u[i], X[i,]));
}

### ----------- right-censored -------------###
t_obs=tT
Centime = runif(ntot, 200, 500);
delta = (tT<=Centime) +0 ;
length(which(delta==0))/ntot; # censoring rate
rcen = which(delta==0);
t_obs[rcen] = Centime[rcen]; ## observed time
## make a data frame
dtot = data.frame(tobs=t_obs, x1=x1, delta=delta, tT=tT,
                  s1=s1, s2=s2);
## Hold out npred for prediction purpose
predindex = sample(1:ntot, npred);
dpred = dtot[predindex,];
d = dtot[-predindex,];
# Prediction settings
prediction = list(xpred=cbind(dpred$x1),
                  spred=cbind(dpred$s1, dpred$s2));

#################################################################
# Independent DDP: Bayesian Nonparametric Survival Model
#################################################################
# MCMC parameters
nburn=100; nsave=100; nskip=0;
# Note larger nburn, nsave and nskip should be used in practice.
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=1000);
prior = list(N=10, a0=2, b0=2, nknots=n, nblock=round(n/2));
# here nknots=n, so FSA is not used.
# If nknots<n, FSA will be used with nblock=round(n/2).
# As nknots is getting larger, the FSA is more accurate but slower
# As nblock is getting smaller, the FSA is more accurate but slower.
# In most applications, setting nblock=n works fine, which is the
# setting by not specifying nblock.
# If nknots is not specified or nknots=n, the exact covariance is used.
# Fit the Cox PH model
res1 = spCopulaDDP(formula = Surv(tobs, delta)~x1, data=d,
                     prior=prior, mcmc=mcmc, prediction=prediction,
                     Coordinates=cbind(d$s1,d$s2), Knots=NULL);
# here if prediction=NULL, prediction$xpred will be set as the design matrix
```

```
# in formula, and prediction$spred will be set as the Coordinates argument.
# Knots=NULL is the defualt setting, for which the knots will be generated
# using fields::cover.design() with number of knots equal to prior$nknots.
## LPML
LPML = sum(log(res1$cpo)); LPML;
## Number of non-negligible components
quantile(colSums(res1$w>0.05))
## MSPE
mean((log(dpred$tT)-apply(log(res1$Tpred), 1, median))^2);

## traceplot
par(mfrow = c(1,2))
traceplot(mcmc(res1$theta[1,]), main="sill")
traceplot(mcmc(res1$theta[2,]), main="range")

##########################################
## Curves
##########################################
ygrid = seq(0,6.0,length=100); tgrid = exp(ygrid);
ngrid = length(tgrid);
xpred = data.frame(x1=c(-1, 1));
plot(res1, xnewdata=xpred, tgrid=tgrid);
```

SuperSurvRegBayes *Bayesian Semiparametric Super Survival Model*

Description

This function fits a super survival model (Zhang, Hanson and Zhou, 2018). It can fit both Case I and Case II interval censored data, as well as standard right-censored, uncensored, and mixtures of these. The Bernstein Polynomial Prior is used for fitting the baseline survival function.

Usage

```
SuperSurvRegBayes(formula, data, na.action, dist="lognormal",
                  mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
                prior=NULL, state=NULL, truncation_time=NULL, subject.num=NULL,
                  InitParamMCMC=FALSE, scale.designX=TRUE)
```

Arguments

formula	a formula expression with the response returned by the Surv function in the survival package. It supports right-censoring, left-censoring, interval-censoring, and mixtures of them.
data	a data frame in which to interpret the variables named in the formula argument.
na.action	a missing-data filter function, applied to the model.frame.
dist	centering distribution for MPT. Choices include "loglogistic", "lognormal", and "weibull".

`mcmc` a list giving the MCMC parameters. The list must include the following elements: `nburn` an integer giving the number of burn-in scans, `nskip` an integer giving the thinning interval, `nsave` an integer giving the total number of scans to be saved, `ndisplay` an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every `ndisplay` iterations have been carried out).

`prior` a list giving the prior information. The list includes: `maxL` an integer giving the maximum level of MPT, `a0` and `b0` gamma prior of the precision parameter of MPT, `Shat` the initial covariance matrix of adaptive M-H for each of the beta_h, beta_o and beta_q, `Vhat` the initial covariance matrix of adaptive M-H for theta, `beta0` and `S0` the prior for each of the beta_h, beta_o and beta_q, for which the default is the g-prior, `theta0` and `V0` the prior for theta.

`state` a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.

`truncation_time`
a vector of left-trucation times with length n.

`subject.num` a vector of suject id numbers when time dependent covariates are considered. For example, for subject 1 time dependent covariates are recorded over [0,1), [1,3), and for subject 2 covariates are recorded over [0,2), [2,3), [3,4). Suppose we only have two subjects, i.e. n=2. In this case, we save the data in the long format, set truncation_time=c(0,1,0,2,3) and subject.num=c(1,1,2,2,2).

`InitParamMCMC` flag to indicate wheter an initial MCMC will be run based on the centering parametric model, where `TRUE` indicates yes.

`scale.designX` flag to indicate wheter the design matrix X will be centered by column means and scaled by column standard deviations, where `TRUE` indicates yes. The default is `TRUE` for improving numerical stability. Even when it is scaled, the reported regression coefficients are in original scales. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when `scale.designX=TRUE`.

Value

The `SuperSurvRegBayes` object is a list containing at least the following components:

`modelname` the name of the fitted model

`terms` the `terms` object used

`dist` the centering distribution used in the TBP prior on baseline survival function

`coefficients` a named vector of coefficients. The last two elements are the estimates of theta1 and theta2 involved in the centering baseline survival function.

`call` the matched call

`prior` the list of hyperparameters used in all priors.

`mcmc` the list of MCMC parameters used

`n` the number of row observations used in fitting the model

`p` the number of columns in the model matrix

nsubject	the number of subjects/individuals, which is equal to n in the absence of time-dependent covariates
subject.num	the vector of subject id numbers when time dependent covariates are considered
truncation_time	the vector of left-trucation times
Surv	the Surv object used
X.scaled	the n by p scaled design matrix
X	the n by p orginal design matrix
theta.scaled	the 2 by nsave matrix of posterior samples for theta1 and theta2 involved in the centering baseline survival function. Note that these posterior samples are based scaled design matrix.
alpha	the vector of posterior samples for the precision parameter alpha in the TBP prior.
maxL	the truncation level used in the TBP prior.
weight	the maxL by nsave matrix of posterior samples for the weights in the TBP prior.
cpo	the length n vector of the stabilized estiamte of CPO; used for calculating LPML
pD	the effective number of parameters involved in DIC
DIC	the deviance information criterion (DIC)
ratetheta	the acceptance rate in the posterior sampling of theta vector involved in the centering baseline survival function
ratebeta	the acceptance rate in the posterior sampling of beta coefficient vector
rateYs	the acceptance rate in the posterior sampling of weight vector involved in the TBP prior
ratec	the acceptance rate in the posterior sampling of precision parameter alpha involved in the TBP prior
BF	the Bayes factors for testing AFT, PH, PO, AH, EH and YP models.

Author(s)

Haiming Zhou

References

Zhang, J., Hanson, T., and Zhou, H. (2019). Bayes factors for choosing among six common survival models. *Lifetime Data Analysis*, 25(2): 361-379.

See Also

survregbayes

Examples

```
#####################################################################
# A simulated data based on PH_PO_AFT super model
#####################################################################
rm(list=ls())
library(coda)
library(survival)
library(spBayesSurv)

## True coeffs
betaT_h = c(1, 1);
betaT_o = c(0, 0);
betaT_q = c(1, 1);
## Baseline Survival
f0oft = function(t) 0.5*dlnorm(t, -1, 0.5)+0.5*dlnorm(t,1,0.5);
S0oft = function(t) {
  0.5*plnorm(t, -1, 0.5, lower.tail=FALSE)+
    0.5*plnorm(t, 1, 0.5, lower.tail=FALSE)
}
h0oft = function(t) f0oft(t)/S0oft(t);
## The Survival function:
Sioft = function(t,x){
  xibeta_h = sum(x*betaT_h);
  xibeta_o = sum(x*betaT_o);
  xibeta_q = sum(x*betaT_q);
  (1+exp(xibeta_o-xibeta_h+xibeta_q)*
     (1/S0oft(exp(xibeta_q)*t)-1))^(-exp(xibeta_h-xibeta_q));
}
Fioft = function(t,x) 1-Sioft(t,x);
## The inverse for Fioft
Finv = function(u, x) uniroot(function (t) Fioft(t,x)-u, lower=1e-100,
                              upper=1e100, extendInt ="yes", tol=1e-6)$root
### true plots
tt=seq(1e-10, 4, 0.02);
xpred1 = c(0,0);
xpred2 = c(0,1);
plot(tt, Sioft(tt, xpred1), "l", ylim=c(0,1));
lines(tt, Sioft(tt, xpred2), "l");

##-------------Generate data-------------------##
## generate x
n = 80;
x1 = rbinom(n, 1, 0.5); x2 = rnorm(n, 0, 1); X = cbind(x1, x2);
## generate survival times
u = runif(n);
tT = rep(0, n);
for (i in 1:n){
  tT[i] = Finv(u[i], X[i,]);
}

### ----------- right censored -------------###
t1=tT;t2=tT;
```

```
Centime = runif(n, 2, 6);
delta = (tT<=Centime) +0 ; length(which(delta==0))/n;
rcen = which(delta==0);
t1[rcen] = Centime[rcen];
t2[rcen] = NA;
## make a data frame
d = data.frame(t1=t1, t2=t2, x1=x1, x2=x2, delta=delta, tT=tT); table(d$delta)/n;

##-------------Fit the model-------------------##
# MCMC parameters
nburn=200; nsave=500; nskip=0; niter = nburn+nsave
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=1000);
prior = list(maxL=15, a0=1, b0=1, M=10, q=.9);
ptm<-proc.time()
res1 = SuperSurvRegBayes(formula = Surv(t1, t2, type="interval2")~x1+x2, data=d,
                         prior=prior, mcmc=mcmc, dist="lognormal");
sfit=summary(res1); sfit;
systime1=proc.time()-ptm; systime1;
par(mfrow = c(3,2))
traceplot(mcmc(res1$beta_h[1,]), main="beta_h for x1");
traceplot(mcmc(res1$beta_h[2,]), main="beta_h for x2");
traceplot(mcmc(res1$beta_o[1,]), main="beta_o for x1");
traceplot(mcmc(res1$beta_o[2,]), main="beta_o for x2");
traceplot(mcmc(res1$beta_q[1,]), main="beta_q for x1");
traceplot(mcmc(res1$beta_q[2,]), main="beta_q for x2");

#####################################################################
## Get curves
#####################################################################
par(mfrow = c(1,1))
tgrid = seq(1e-10,4,0.2);
xpred = data.frame(x1=c(0,0), x2=c(0,1));
plot(res1, xnewdata=xpred, tgrid=tgrid);
```

`survregbayes` *Bayesian Semiparametric Survival Models*

Description

This function fits semiparametric proportional hazards (PH), proportional odds (PO), accelerated failture time (AFT) and accelerated hazards (AH) models. Both georeferenced (location observed exactly) and areally observed (location known up to a geographic unit such as a county) spatial locations can be handled. Georeferenced data are modeled with Gaussian random field (GRF) frailties whereas areal data are modeled with a conditional autoregressive (CAR) prior on frailties. For non-spatial clustered data, an IID Gaussian frailties are assumed. Variable selection is also incorporated. The function can fit both Case I and Case II interval censored data, as well as standard right-censored, uncensored, and mixtures of these. The transformed Bernstein Polynomial (TBP) prior is used for fitting the baseline survival function. The full scale approximation (FSA) (Sang and Huang, 2012) could be used to inverse the spatial correlation matrix for georeferenced data. The

function also fits all these models without frailties. The logarithm of the pseudo marginal likelihood (LPML), the deviance information criterion (DIC), and the Watanabe-Akaike information criterion (WAIC) are provided for model comparison.

Usage

```
survregbayes(formula, data, na.action, survmodel="PH", dist="loglogistic",
             mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
             prior=NULL, state=NULL, selection=FALSE, Proximity=NULL,
             truncation_time=NULL, subject.num=NULL, Knots=NULL,
             Coordinates=NULL, DIST=NULL, InitParamMCMC=TRUE,
             scale.designX=TRUE)
```

Arguments

formula	a formula expression with the response returned by the Surv function in the survival package. It supports right-censoring, left-censoring, interval-censoring, and mixtures of them. To include CAR frailties, add frailtyprior("car",ID) to the formula, where ID is an n dimensional vector of cluster ID numbers. Furthermore, use frailtyprior("iid",ID) for Gaussian exchangeable frailties, use frailtyprior("grf",ID) for Gaussian random fields (GRF) frailties, and exclude the term frailtyprior() for non-frailty models. Note: the data need to be sorted by ID.
data	a data frame in which to interpret the variables named in the formula argument.
na.action	a missing-data filter function, applied to the model.frame.
survmodel	a character string for the assumed survival model. The options include "PH" for proportional hazards, "PO" for proportional odds, and "AFT" for accelerated failture time.
dist	centering distribution for TBP. Choices include "loglogistic", "lognormal", and "weibull".
mcmc	a list giving the MCMC parameters. The list must include the following elements: nburn an integer giving the number of burn-in scans, nskip an integer giving the thinning interval, nsave an integer giving the total number of scans to be saved, ndisplay an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every ndisplay iterations have been carried out).
prior	a list giving the prior information. The function itself provides all default priors. Note if FSA is used, the number of knots knots and the number of blocks nblock are specified here; see examples below. See Zhou, Hanson and Zhang (2018) for more detailed hyperprior specifications.
state	a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.
selection	flag to indicate whether variable selection is performed, where FALSE indicates that no variable selection will be performed.

`Proximity` an m by m symetric adjacency matrix, where m is the number of clusters/regions. If CAR frailty model is specified in the formula, `Proximity` is required; otherwise it is ignored. Note: this matrix should be specified according to the data that have been sorted by ID.

`truncation_time` a vector of left-trucation times with length n.

`subject.num` a vector of subject id numbers when time dependent covariates are considered. For example, for subject 1 time dependent covariates are recorded over [0,1), [1,3), and for subject 2 covariates are recorded over [0,2), [2,3), [3,4). Suppose we only have two subjects, i.e. n=2. In this case, we save the data in the long format, set truncation_time=c(0,1,0,2,3) and subject.num=c(1,1,2,2,2).

`Knots` an nknots by d matrix, where nknots is the number of selected knots for FSA, and d is the dimension of each location. If `Knots` is not specified, the space-filling algorithm will be used to find the knots.

`Coordinates` an m by d coordinates matrix, where m is the number of clusters/regions, d is the dimension of coordiantes. If GRF frailty model is specified in the formula, `Coordinates` is required; otherwise it is ignored. Note: this matrix should be specified according to the data that have been sorted by ID.

`DIST` This is a function argument, used to calculate the distance. The default is Euclidean distance (`fields::rdist`). This function should have two arguments (X1,X2), where X1 and X2 are matrices with coordinates as the rows. The returned value of this function should be the pairwise distance matrix. If nrow(X1)=m and nrow(X2)=n then the function should return an m by n matrix of all distances between these two sets of points.

`InitParamMCMC` flag to indicate whether an initial MCMC will be run based on the centering parametric model, where `TRUE` indicates yes.

`scale.designX` flag to indicate whether the design matrix X will be centered by column means and scaled by column standard deviations, where `TRUE` indicates yes. The default is `TRUE` for improving numerical stability. Even when it is scaled, the reported regression coefficients are in original scales. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when `scale.designX=TRUE`.

Value

This class of objects is returned by the `survregbayes` function to represent a fitted Bayesian semi-parametric survival model. Objects of this class have methods for the functions `print`, `summary` and `plot`.

The `survregbayes` object is a list containing at least the following components:

`modelname` the name of the fitted model

`terms` the `terms` object used

`dist` the centering distribution used in the TBP prior on baseline survival function

`survmodel` the model fitted

`coefficients` a named vector of coefficients. The last two elements are the estimates of theta1 and theta2 involved in the centering baseline survival function.

call	the matched call
prior	the list of hyperparameters used in all priors.
mcmc	the list of MCMC parameters used
n	the number of row observations used in fitting the model
p	the number of columns in the model matrix
nsubject	the number of subjects/individuals, which is equal to n in the absence of time-dependent covariates
subject.num	the vector of subject id numbers when time dependent covariates are considered
truncation_time	the vector of left-trucation times
Surv	the Surv object used
X.scaled	the n by p scaled design matrix
X	the n by p orginal design matrix
beta	the p by nsave matrix of posterior samples for the coefficients in the linear.predictors
theta.scaled	the 2 by nsave matrix of posterior samples for theta1 and theta2 involved in the centering baseline survival function. Note that these posterior samples are based scaled design matrix.
beta.scaled	the p by nsave matrix of posterior samples for the coefficients in the linear.predictors. Note that these posterior samples are based scaled design matrix.
alpha	the vector of posterior samples for the precision parameter alpha in the TBP prior.
maxL	the truncation level used in the TBP prior.
weight	the maxL by nsave matrix of posterior samples for the weights in the TBP prior.
cpo	the length n vector of the stabilized estiamte of CPO; used for calculating LPML
pD	the effective number of parameters involved in DIC
DIC	the deviance information criterion (DIC)
pW	the effective number of parameters involved in WAIC
WAIC	the Watanabe-Akaike information criterion (WAIC)
Surv.cox.snell	the Surv object used for Cox-Snell residual plot. This is not recommended for frailty models, for which please use the function cox.snell.survregbayes.
ratetheta	the acceptance rate in the posterior sampling of theta vector involved in the centering baseline survival function
ratebeta	the acceptance rate in the posterior sampling of beta coefficient vector
rateYs	the acceptance rate in the posterior sampling of weight vector involved in the TBP prior
ratec	the acceptance rate in the posterior sampling of precision parameter alpha involved in the TBP prior
frail.prior	the frailty prior used in frailtyprior
selection	whether the variable selection was performed
initial.values	the list of initial values used for the parameters

BF.baseline	the Bayes factor for comparing the parametric baseline vs. the TBP baseline
BF.bs	Bayes factors for testing linearity when bspline is added to the linear.predictors

The object will also have the following components when frailty models are fit:

v	the nID by nsave matrix of posterior samples for frailties, where nID is the number of clusters considered.
ratev	the vector of acceptance rates in the posterior sampling of frailties
tau2	the vector of posterior samples for tau2 involved in the IID, GRF or CAR frailty prior.
ID	the cluster ID used in frailtyprior

If GRF frailties are used, the object will also have:

Coordinates	the Coordinates matrix used in survregbayes
ratephi	the acceptance rates in the posterior sampling of phi involved in the GRF prior
phi	the vector of posterior samples for phi involved in the GRF prior
Knots	the Knots matrix used in survregbayes

If the variable selection is performed, the object will also include:

gamma	the p by nsave matrix of posterior samples for gamma involved in the variable selection

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H., Hanson, T., and Zhang, J. (2020). spBayesSurv: Fitting Bayesian Spatial Survival Models Using R. *Journal of Statistical Software*, 92(9): 1-33.

Zhou, H. and Hanson, T. (2018). A unified framework for fitting Bayesian semiparametric models to arbitrarily censored survival data, including spatially-referenced data. *Journal of the American Statistical Association*, 113(522): 571-581.

Sang, H. and Huang, J. Z. (2012). A full scale approximation of covariance functions for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 74(1), 111-132.

See Also

frailtyprior, cox.snell.survregbayes, rdist, rdist.earth

Examples

```
rm(list=ls())
library(survival)
library(spBayesSurv)
library(coda)
library(MASS)
library(fields)

## True coeffs
betaT = c(1,1);
## Baseline Survival
f0oft = function(t) 0.5*dlnorm(t, -1, 0.5)+0.5*dlnorm(t,1,0.5);
S0oft = function(t) (0.5*plnorm(t, -1, 0.5, lower.tail=FALSE)+
                       0.5*plnorm(t, 1, 0.5, lower.tail=FALSE))
## The Survival function:
Sioft = function(t,x,v=0)  exp( log(S0oft(t))*exp(sum(x*betaT)+v) ) ;
Fioft = function(t,x,v=0) 1-Sioft(t,x,v);
## The inverse for Fioft
Finv = function(u, x,v=0) uniroot(function (t) Fioft(t,x,v)-u, lower=1e-100,
                                  upper=1e100, extendInt ="yes", tol=1e-6)$root
## correlation function
rho_Exp = function(dis, phi) exp(-(dis*phi));

#############################################################################
############################ Start to simulation ############################
#############################################################################
phiT=1; sill=0.9999; ## phiT is the range parameter phi.
tau2T = 1; ## true frailty variance;
m = 50; mi=2
id=rep(1:m, each=mi)
mseq = rep(mi, m);
n = sum(mseq);
s1 = runif(m, 0, 10); s2 = runif(m, 0, 10);
locs = cbind(s1, s2);
ss = cbind(rep(locs[,1],each=mi), rep(locs[,2],each=mi)); ### the locations.
Dmm = .Call("DistMat", t(locs), t(locs), PACKAGE = "spBayesSurv");
Rmm = sill*rho_Exp(Dmm, phiT)+diag(1-sill, m, m);
v = mvrnorm(1, mu=rep(0,m), Sigma=tau2T*Rmm);
vn = rep(v, each=mi)
## generate x
x1 = rbinom(n, 1, 0.5); x2 = rnorm(n, 0, 1); X = cbind(x1, x2);
## generate survival times
u = runif(n);
tT = rep(0, n);
for (i in 1:n){
  tT[i] = Finv(u[i], X[i,], vn[i]);
}

### ----------- right censored -------------###
t1=tT;t2=tT;
## right censored
Centime = runif(n, 2,6);
```

```
delta = (tT<=Centime) +0 ; length(which(delta==0))/n;
rcen = which(delta==0);
t1[rcen] = Centime[rcen];
t2[rcen] = NA;
## make a data frame
## Method 1: in the interval-censoring notation:
## t1 is the left endpoint and t2 is the right endpoint.
## This way we could use Surv(t1, t2, type="interval2")
## Method 2: Because we have right-censored data,
## we could use t1 as the observed survival times and delta as the indicator.
## This way we could use Surv(t1, delta). This is the same as above.
## (s1, s2) are the locations.
d = data.frame(t1=t1, t2=t2, x1=x1, x2=x2, delta=delta,
               s1=ss[,1], s2=ss[,2], id=id);
table(d$delta)/n;

##-------------spBayesSurv-------------------##
# MCMC parameters
nburn=200; nsave=200; nskip=0;
# Note larger nburn, nsave and nskip should be used in practice.
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=1000);
prior = list(maxL=15, a0=1, b0=1, nknots=m, nblock=m, nu=1);
# here if nknots<m, FSA will be used with nblock=m.
cor.dist = function(x1, x2) rdist(x1,x2)
ptm<-proc.time()
res1 = survregbayes(formula = Surv(t1, delta)~x1+x2+
                      frailtyprior("grf", id), data=d, InitParamMCMC=FALSE,
                    survmodel="PH", prior=prior, mcmc=mcmc, DIST=cor.dist,
                    dist="loglogistic", Coordinates = locs);
## Or equivalently formula = Surv(t1, t2, type="interval2") can also be used.
## Note InitParamMCMC=FALSE is used for speeding,
## InitParamMCMC=TRUE is recommended in general.
sfit=summary(res1); sfit
systime1=proc.time()-ptm; systime1;

############################################
## Results
############################################
## acceptance rate of frailties
res1$ratev[1]
## traceplots;
par(mfrow=c(2,3));
traceplot(mcmc(res1$beta[1,]), main="beta1");
traceplot(mcmc(res1$beta[2,]), main="beta2");
traceplot(mcmc(res1$v[1,]), main="frailty");
traceplot(mcmc(res1$v[2,]), main="frailty");
traceplot(mcmc(res1$v[3,]), main="frailty");
#traceplot(mcmc(res1$v[4,]), main="frailty");
traceplot(mcmc(res1$phi), main="phi");

############################################
## Curves
############################################
```

```
par(mfrow=c(1,1));
wide=0.2;
tgrid = seq(1e-10,4,wide);
ngrid = length(tgrid);
p = length(betaT); # number of covariates
newdata = data.frame(x1=c(0,0), x2=c(0,1))
plot(res1, xnewdata=newdata, tgrid=tgrid, PLOT=TRUE);

## Cox-Snell plot
set.seed(1)
cox.snell.survregbayes(res1, ncurves=2, PLOT=TRUE);
```

`survregbayes2` *Bayesian Semiparametric Survival Models*

Description

This function fits mixtures of Polya trees (MPT) proportional hazards, proportional odds, and accelerated failture time models. It also allows to include exchangeable and CAR frailties for fitting clusterd survival data. The function can fit both Case I and Case II interval censored data, as well as standard right-censored, uncensored, and mixtures of these.

Usage

```
survregbayes2(formula, data, na.action, survmodel="PH", dist="loglogistic",
              mcmc=list(nburn=3000, nsave=2000, nskip=0, ndisplay=500),
              prior=NULL, state=NULL, selection=FALSE, Proximity=NULL,
              truncation_time=NULL, subject.num=NULL, InitParamMCMC=TRUE,
              scale.designX=TRUE)
```

Arguments

`formula`	a formula expression with the response returned by the `Surv` function in the `survival` package. It supports right-censoring, left-censoring, interval-censoring, and mixtures of them. To include CAR frailties, add `frailtyprior("car",ID)` to the formula, where `ID` is an n dimensional vector of cluster ID numbers. Furthermore, use `frailtyprior("iid",ID)` for Gaussian IID frailties, and exclude the term `frailtyprior()` for non-frailty models. Note: the data need to be sorted by ID.
`data`	a data frame in which to interpret the variables named in the `formula` argument.
`na.action`	a missing-data filter function, applied to the `model.frame`.
`survmodel`	a character string for the assumed survival model. The options include `"PH"` for proportional hazards, `"PO"` for proportional odds, and `"AFT"` for accelerated failture time.
`dist`	centering distribution for MPT. Choices include `"loglogistic"`, `"lognormal"`, and `"weibull"`.

mcmc	a list giving the MCMC parameters. The list must include the following elements: nburn an integer giving the number of burn-in scans, nskip an integer giving the thinning interval, nsave an integer giving the total number of scans to be saved, ndisplay an integer giving the number of saved scans to be displayed on screen (the function reports on the screen when every ndisplay iterations have been carried out).
prior	a list giving the prior information. The list includes the following parameter: maxL an integer giving the maximum number of mixtures of beta distributions. The function itself provides all default priors.
state	a list giving the current value of the parameters. This list is used if the current analysis is the continuation of a previous analysis.
selection	flag to indicate whether variable selection is performed, where FALSE indicates that no variable selection will be performed.
Proximity	an m by m symetric adjacency matrix, where m is the number of clusters/regions. If CAR frailty model is specified in the formula, Proximity is required; otherwise it is ignored. Note: this matrix should be specified according to the data that have been sorted by ID.
truncation_time	a vector of left-trucation times with length n.
subject.num	a vector of suject id numbers when time dependent covariates are considered. For example, for subject 1 time dependent covariates are recorded over [0,1), [1,3), and for subject 2 covariates are recorded over [0,2), [2,3), [3,4). Suppose we only have two subjects, i.e. n=2. In this case, we save the data in the long format, set truncation_time=c(0,1,0,2,3) and subject.num=c(1,1,2,2,2).
InitParamMCMC	flag to indicate wheter an initial MCMC will be run based on the centering parametric model, where TRUE indicates yes.
scale.designX	flag to indicate wheter the design matrix X will be centered by column means and scaled by column standard deviations, where TRUE indicates yes. The default is TRUE for improving numerical stability. Even when it is scaled, the reported regression coefficients are in original scales. Note if we want to specify informative priors for regression coefficients, these priors should correspond to scaled predictors when scale.designX=TRUE.

Value

The results include the MCMC chains for the parameters; use names to find out what they are.

Author(s)

Haiming Zhou and Timothy Hanson

References

Zhou, H. and Hanson, T. (2015). Bayesian spatial survival models. In Nonparametric Bayesian Inference in Biostatistics (pp 215-246). Springer International Publishing.

Zhao, L. and Hanson, T. (2011). Spatially dependent Polya tree modeling for survival data. *Biometrics*, 67(2), 391-403.

See Also

frailtyprior

Examples

```
rm(list=ls())
library(coda)
library(survival)
library(spBayesSurv)

## True coeffs
betaT = c(1,1);
## Baseline Survival
f0oft = function(t) 0.5*dlnorm(t, -1, 0.5)+0.5*dlnorm(t,1,0.5);
S0oft = function(t) (0.5*plnorm(t, -1, 0.5, lower.tail=FALSE)+
                       0.5*plnorm(t, 1, 0.5, lower.tail=FALSE));
## The Survival function:
Sioft = function(t,x,v=0)  exp( log(S0oft(t))*exp(sum(x*betaT)+v) ) ;
Fioft = function(t,x,v=0) 1-Sioft(t,x,v);
## The inverse for Fioft
Finv = function(u, x,v=0) uniroot(function (t) Fioft(t,x,v)-u,
                                  lower=1e-100, upper=1e100,
                                  extendInt ="yes")$root

##-------------Generate data-------------------##
## generate x
n = 100;
x1 = rbinom(n, 1, 0.5); x2 = rnorm(n, 0, 1); X = cbind(x1, x2);
## generate survival times
u = runif(n);
tT = rep(0, n);
for (i in 1:n){
  tT[i] = Finv(u[i], X[i,]);
}

### ----------- partly interval-censored -------------###
t1=rep(NA, n);t2=rep(NA, n); delta=rep(NA, n);
n1 = floor(0.5*n); ## right-censored part
n2 = n-n1; ## interval-censored part
# right-censored part
rcen = sample(1:n, n1);
t1_r=tT[rcen];t2_r=tT[rcen];
Centime = runif(n1, 2, 6);
delta_r = (tT[rcen]<=Centime) +0 ; length(which(delta_r==0))/n1;
t1_r[which(delta_r==0)] = Centime[which(delta_r==0)];
t2_r[which(delta_r==0)] = NA;
t1[rcen]=t1_r; t2[rcen]=t2_r; delta[rcen] = delta_r;
# interval-censored part
intcen = (1:n)[-rcen];
t1_int=rep(NA, n2);t2_int=rep(NA, n2); delta_int=rep(NA, n2);
npois = rpois(n2, 2)+1;
for(i in 1:n2){
```

```
  gaptime = cumsum(rexp(npois[i], 1));
  pp = Fioft(gaptime, X[intcen[i],]);
  ind = sum(u[intcen[i]]>pp);
  if (ind==0){
    delta_int[i] = 2;
    t2_int[i] = gaptime[1];
  }else if (ind==npois[i]){
    delta_int[i] = 0;
    t1_int[i] = gaptime[ind];
  }else{
    delta_int[i] = 3;
    t1_int[i] = gaptime[ind];
    t2_int[i] = gaptime[ind+1];
  }
}
t1[intcen]=t1_int; t2[intcen]=t2_int; delta[intcen] = delta_int;
## make a data frame
d = data.frame(t1=t1, t2=t2, x1=x1, x2=x2, delta=delta, tT=tT);
table(d$delta)/n;

##-------------spBayesSurv-------------------##
fit0=survreg(formula = Surv(t1, t2, type="interval2")~x1+x2,
             data=d, dist="loglogistic");
# MCMC parameters
nburn=500; nsave=500; nskip=0; niter = nburn+nsave
# Note larger nburn, nsave and nskip should be used in practice.
mcmc=list(nburn=nburn, nsave=nsave, nskip=nskip, ndisplay=500);
prior = list(maxL=4, a0=1, b0=1);
ptm<-proc.time()
res = survregbayes2(formula = Surv(t1, t2, type="interval2")~x1+x2, data=d,
                    survmodel="PH", prior=prior, mcmc=mcmc,
                   dist="loglogistic", InitParamMCMC=FALSE);
## Note InitParamMCMC=FALSE is used only speeding,
## InitParamMCMC=TRUE is recommended in general.
sfit=summary(res); sfit;
systime=proc.time()-ptm;

### trace plots
par(mfrow = c(2,2));
traceplot(mcmc(res$beta[1,]), main="beta1");
traceplot(mcmc(res$beta[2,]), main="beta2");

#####################################################################
## Get curves
#####################################################################
par(mfrow = c(1,1));
wide=0.01;
tgrid = seq(0.001,4,wide);
ngrid = length(tgrid);
xnew = c(0,1)
xpred = cbind(c(0,0), xnew);
nxpred = nrow(xpred);
estimates=plot(res, xpred, tgrid);
```

```
## plots
## survival function when x=(0,0)
i=2
par(cex=1.5,mar=c(4.1,4.1,1,1),cex.lab=1.4,cex.axis=1.1)
plot(tgrid, Sioft(tgrid, c(0,xnew[i])), "l", lwd=3,
     xlim=c(0,3), xlab="time", ylab="survival");
polygon(x=c(rev(tgrid),tgrid),
        y=c(rev(estimates$Shatlow[,i]),estimates$Shatup[,i]),
        border=NA,col="lightgray");
lines(tgrid, Sioft(tgrid, c(0,xnew[i])), "l", lwd=3);
lines(tgrid, estimates$Shat[,i], lty=3, lwd=3, col=1);
## survival function when x=(0,0)
i=1
par(cex=1.5,mar=c(4.1,4.1,1,1),cex.lab=1.4,cex.axis=1.1)
lines(tgrid, Sioft(tgrid, c(0,xnew[i])), "l", lwd=3,
     xlim=c(0,3), xlab="time", ylab="survival");
polygon(x=c(rev(tgrid),tgrid),
        y=c(rev(estimates$Shatlow[,i]),estimates$Shatup[,i]),
        border=NA,col="lightgray");
lines(tgrid, Sioft(tgrid, c(0,xnew[i])), "l", lwd=3);
lines(tgrid, estimates$Shat[,i], lty=3, lwd=3, col=1);
```

Index