Package ‘ppmSuite’

June 15, 2022

Type Package

Title A Collection of Models that Employ a Product Partition Distribution as a Prior on Partitions

Version 0.2.4

Maintainer Garritt L. Page <page@stat.byu.edu>

Description Provides a suite of functions that fit models that use PPM type priors for partitions. Models include hierarchical Gaussian and probit ordinal models with a (covariate dependent) PPM. If a covariate dependent product partition model is selected, then all the options detailed in Page, G.L.; Quintana, F.A. (2018) <doi:10.1007/s11222-017-9777-z> are available. If covariate values are missing, then the approach detailed in Page, G.L.; Quintana, F.A.; Mueller, P (2020) <doi:10.1080/10618600.2021.1999824> is employed. Also included in the package is a function that fits a Gaussian likelihood spatial product partition model that is detailed in Page, G.L.; Quintana, F.A. (2016) <doi:10.1214/15-BA971>, and multivariate PPM change point models that are detailed in Quinlan, J.J.; Page, G.L.; Castro, L.M. (2021) <arXiv:2201.07830>.

Depends R (>= 3.5.0)

License GPL

Encoding UTF-8

LazyData true

Suggests cluster

NeedsCompilation yes

Author Garritt L. Page [aut, cre, cph],
Jose J. Quinlan [aut, cph],
S. McKay Curtis [ctb, cph],
Radford M. Neal [ctb, cph]

Repository CRAN

Date/Publication 2022-06-15 09:30:06 UTC

R topics documented:

`bear` *Bear dataset*

Description

Number of physiological measurements from 54 bears.

Format

data: A data frame with 54 rows and the following 9 variables:

age

length

sex

weight

chest

headlength

headwid

month

neck

`ccp_ppm` *Function that fits a multivariate correlated product partition change point model*

Description

`ccp_ppm` is a function that fits a Bayesian product partition change point model, where the set of change point indicators between time series are correlated.

Usage

```
ccp_ppm(ydata,
         nu0, mu0, sigma0,
         mltypes, thetas,
         devs,
         nburn, nskip, nsave,
         verbose = FALSE)
```

Arguments

`ydata` An $L \times n$ data matrix, where L is the number of time series and n, the number of time points.

`nu0` Degrees of freedom of the multivariate Student's t-distribution (see section Details).

`mu0` Location vector of dimension L (see section Details).

`sigma0` Positive definite scale matrix of order $L \times L$ (see section Details).

`mltypes` Type of marginal likelihood. Currently only available is:

- `mltypes = 1`. Observations within a block are conditionally independent $Normal(\mu, \sigma^2)$ variates with mean μ and variance σ^2. The desired marginal likelihood is obtained after integrating (μ, σ^2) with respect to a $Normal-Inverse-Gamma(\mu_0, \kappa_0, \alpha_0, \beta_0)$ prior.

`thetas` An $L \times q$ matrix containing hyperparameters associated with the marginal likelihood. The number of rows (L) corresponds to the number of series. The number of columns (q) depend on the marginal likelihood:

- If `mltypes = 1`, then $q = 4$ and `thetas` equals the hyperparameter $(\mu_0, \kappa_0, \alpha_0, \beta_0)$ of the Normal-Inverse-Gamma prior.

`devs` An $L \times (n-1)$ matrix containing the standard deviations of the candidate density associated with the random walk Metropolis-Hastings steps for updating change point probabilities.

`nburn` The number of initial MCMC iterates to be discarded as burn-in.

`nskip` The amount to thinning that should be applied to the MCMC chain.

`nsave` Then number of MCMC iterates to be stored.

`verbose` Logical indicating whether to print to screen the MCMC progression. The default value is `verbose = FALSE`.

Details

As described in Quinlan et al. (add cite), for each time series $\boldsymbol{y}_i = (y_{i,1}, \ldots, y_{i,n})'$:

$$\boldsymbol{y}_i \mid \rho_i \sim \prod_{j=1}^{b_i} \mathcal{F}(\boldsymbol{y}_{i,j} \mid \boldsymbol{\theta}_i)$$

$$\rho_i \mid (p_{i,1}, \ldots, p_{i,n-1})' \sim \prod_{t \in T_i} p_{i,t} \prod_{t \notin T_i} (1 - p_{i,t}) : T_i = \{\tau_{i,1}, \ldots, \tau_{i,b_i-1}\}$$

$$(p_{1,t}, \ldots, p_{L,t})' \sim logit - t(\nu_0, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0).$$

Here, $\rho_i = \{S_{i,1}, \ldots, S_{i,b_i}\}$ is a partition of the set $\{1, \ldots, n\}$ into b_i contiguous blocks, and $\boldsymbol{y}_{i,j} = (y_{i,t} : t \in S_{i,j})'$. Also, $\tau_{i,j} = \max(S_{i,j})$ and $\mathcal{F}(\cdot \mid \boldsymbol{\theta}_i)$ is a marginal likelihood function which depends on the nature of $\boldsymbol{y}_i$, indexed by a hyperparameter $\boldsymbol{\theta}_i$. In addition, $logit - t(\nu_0, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ is the logit of a multivariate Student's t-distribution with degrees of freedom ν_0, location vector $\boldsymbol{\mu}_0$ and scale matrix $\boldsymbol{\Sigma}_0$.

Value

The function returns a list containing arrays filled with MCMC iterates corresponding to model parameters. In order to provide more detail, in what follows let M be the number of MCMC iterates collected. The output list contains the following:

- C. An $M \times \{L(n-1)\}$ matrix containing MCMC iterates associated with each series indicators of a change point. The mth row in `C` is divided into L blocks; the first $(n-1)$ change point indicators for time series 1, the next $(n-1)$ change point indicators for time series 2, and so on.
- P. An $M \times \{L(n-1)\}$ matrix containing MCMC iterates associated with each series probability of a change point. The mth row in `P` is divided into L blocks; the first $(n-1)$ change point probabilities for time series 1, the next $(n-1)$ change point probabilities for time series 2, and so on.

Examples

```
# Generate data that has two series, each with 100 observations
y1 <- replicate(25, rnorm(4, c(-1, 0, 1, 2), c(0.1, 0.25, 0.5, 0.75)))
y2 <- replicate(25, rnorm(4, c(2, 1, 0, -2), c(0.1, 0.25, 0.5, 0.75)))
y <- rbind(c(t(y1)), c(t(y2)))

# Marginal likelihood parameters
thetas <- matrix(1, nrow = 2,ncol = 4)
thetas[1,] <- c(0, 1, 2, 1)
thetas[2,] <- c(0, 1, 2, 1)

# M-H candidate density standard deviations
devs = matrix(0.1, nrow = 2, ncol = (dim(y)[2] - 1))

# Prior parameters for logit-t distribution
L <- nrow(y)
pivar <- 10
picorr <- 0.9
pimu <- rep(-6, L) # mean associated with logit of p_i
piSigma <- pivar*picorr*(rep(1, L) %*% t(rep(1, L))) +
           pivar*(1 - picorr)*diag(L)
nu0 = 3
mu0 = pimu
sigma0 = piSigma
```

```
# Fit the bayesian ppm change point model
fit <- ccp_ppm(nburn = 1000, nskip = 1, nsave = 1000, ydata = y, nu0 = nu0,
               mu0 = mu0, sigma0 = sigma0, mltypes = c(1, 1), thetas = thetas,
               devs = devs)
```

`gaussian_ppmx` *Function that fits Gaussian PPMx model*

Description

`gaussian_ppmx` is the main function used to fit Gaussian PPMx model.

Usage

```
gaussian_ppmx(y, X=NULL, Xpred=NULL,
                    meanModel=1,
                    cohesion=1,
                    M=1,
                    PPM = FALSE,
                    similarity_function=1,
                    consim=1,
                    calibrate=0,
                    simParms=c(0.0, 1.0, 0.1, 1.0, 2.0, 0.1, 1),
                    modelPriors=c(0, 100^2, 1, 1),
                    mh=c(0.5, 0.5),
                    draws=1100,burn=100,thin=1,
                    verbose=FALSE)
```

Arguments

`y`	numeric vector for the response variable
`X`	a data frame whose columns consist of covariates that will be incorporated in the partition model. Those with class of "character" or "factor" will be treated as categorical covariates. All others will be treated as continuous covariates.
`Xpred`	a data frame containing covariate values for which out-of-sample predictions are desired. The format of and order of Xpred must be the same as that found in X.
`meanModel`	Type of mean model included in the likelihood that is to be used. Options are 1 or 2 with • 1 - cluster-specific means with no covariates in likelihood. • 2 - cluster-specific intercepts and a global regression of the type Xbeta is included in the likelihood.
`cohesion`	Type of cohesion function to use in the PPMx prior. Options are 1 or 2 with • 1 - Dirichlet process style of cohesion c(S) = M x (|S| - 1)!

- 2 - Uniform cohesion c(S) = 1

`M` Precision parameter. Default is 1.

`PPM` Logical argument that indicates if the PPM or PPMx partition model should be employed. If PPM = FALSE, then an X matrix must be supplied.

`similarity_function`
Type of similarity function that is employed for the PPMx prior on partitions. Options are 1-4 with

- 1 - Auxilliary similarity
- 2 - Double dipper similarity
- 3 - Cluster variance or entropy for categorical covariates
- 4 - Mean Gower dissimilarity (Gower dissimilarity is not available if missing values are present in X)

`consim` If similarity_function is either set to 1 or 2, then this specifies the type of marginal likelihood used as the similarity function. Options are 1 or 2 with

- 1 - N-N(m0, s20, v) (v variance of "likelihood", m0 and s20 "prior" parameters),
- 2 - N-NIG(m0,s20, k0, nu0) (m0 and s20 center and scale of Gaussian, k0 and nu0 shape and scale of Inverse Gamma)

`calibrate` Indicates if the similarity should be calibrated. Options are 0-2 with

- 0 - no calibration
- 1 - standardize similarity value for each covariate
- 2 - coarsening is applied so that each similarity is raised to the 1/p power

`simParms` Vector of parameter values employed in the similarity function of the PPMx. Entries of the vector correspond to

- m0 - center continuous similarity with default 0,
- s20 - spread of 'prior' continuous similarity with default 1,
- v2 - spread of 'likelihood' for conitnuous similarity (smaller values place more weight partitions with clusters that contain homogeneous covariate values)
- k0 - degrees of freedom upper for v (only used for N-NIG similarity model)
- nu0 - scale for v (only used for N-NIG similarity model)
- a0 - dirichlet weight for categorical similarity with default of 0.1 (smaller more weight placed on this variable)
- alpha - weight associated with cluster-variance and Gower disimilarity

`modelPriors` Vector of prior parameter values used in the PPMx prior.

- m - prior mean for mu0 with default equal to 0,
- s2 - prior variance mu0 with default equal to 100^2,
- A - upper bound on sigma2*_j with default equal to 10
- B - upper bound on sig20 with default equal to 10

`mh` two dimensional vector containing values for tunning parameter associated with MH update for sigma2 and sigma20

`draws` number of MCMC iterates to be collected. default is 1100

`burn`	number of MCMC iterates discared as burn-in. default is 100
`thin`	number by which the MCMC chain is thinne. default is 1. Thin must be selected so that it is a multilple of (draws - thin)
`verbose`	Logical indicating if information regarding data and MCMC iterate should be printed to screen

Details

This function is able to fit a Gaussian PPM or PPMx model as detailed in (Mueller, Quintana, and Rosner, 2011). The data model is a Gaussian distribution with cluster-specific means and variances. If meanModel = 2, then a "global" regression component is added to the mean. Conjugate priors are used for cluster-specific means while uniform priors are used for variance components. A variety of options associated with the similarity function of the PPMx are available. See Page, Quintana 2018; Mueller, Quintana, Rosner 2011 for more details.

If covariate matrix contains missing values, then the approach described in Page, Quintana, Mueller (2020) is automatically employed. Missing values must be denoted using "NA". Currently, NAs cannot be accommodated if a "global" regression is desired.

All continuous X's are standardized to have mean 0 and unit standard deviation before being passed to the PPMx partition model. However, for meanModel = 2 the regression coefficients are estimated on the original scale and are ordered such that the continuous covariates appear first and the categorical covariates come after.

The computational implementation of the model is based on algorithm 8 found in Neal 2000.

Value

The function returns a list containing arrays filled with MCMC iterates corresponding to model parameters and model fit metrics. In order to provide more detail, in what follows let

"T" - be the number of MCMC iterates collected,

"N" - be the number of observations,

"P" - be the number of predictions.

"C" - be the total number of covariates

The output list contains the following

- mu - a matrix of dimension (T, N) containing MCMC iterates associated with each subjects mean parameter (mu*_c_i).
- sig2 - a matrix of dimension (T, N) containing MCMC iterates associated with each subjects variance parameter (sigma2*_c_i)
- beta - if meanModel = 2, then this is a matrix of dimension (T,C) containing MCMC iterates associated coefficients in the global regression
- Si - a matrix of dimension (T, N) containing MCMC iterates assocated with each subjects cluster label.
- mu0 - vector of length T containing MCMC iterates for mu0 parameter
- sig20 - vector of length T containing MCMC iterates for sig20
- nclus - vector of length T containing number of clusters at each MCMC iterate

- like - a matrix of dimension (T, N) containing likelihood values at each MCMC iterate.
- WAIC - scalar containing the WAIC value
- lpml - scalar containing lpml value
- fitted.values - a matrix of dimension (T, N) containing fitted (or in sample predictions) for each subject at each MCMC iterate
- ppred - a matrix of dimension (T, P) containing out of sample preditions for each "new" subject at each MCMC iterate of the posterior predictive distribution
- predclass - a matrix of dimension (T, P) containing MCMC iterates of cluster two which "new" subject is allocated
- rbpred - a matrix of dimension (T, P) containing out of sample preditions for each "new" subject at each MCMC iterate based on the rao-blackwellized prediction
- predclass_prob - a matrix of dimension (T, P*N) that contains the cluster allocation probabilities. They are organized so that each row corresponds to an MCMC iterate. Letting nclus represent the number of components at the t-th MCMC sample, the the first nclus columns of row t correspond to probabilities for new subject 1. Then columns (N+1):(N+nclus) correspond to probabilities associated with new subject 2, etc.

Examples

```
data(bear)

# plot length, sex, and weight of bears
ck <- c(4,3,2)
pairs(bear[,ck])


# response is length
Y <- bear$weight

# Continuous Covariate is chest
# Categorical covariate is sex
X <- bear[,c("length", "sex")]
X$sex <- as.factor(X$sex)

# Randomly partition data into 44 training and 10 testing
set.seed(1)
trainObs <- sample(1:length(Y),44, replace=FALSE)

Ytrain <- Y[trainObs]
Ytest <- Y[-trainObs]

Xtrain <- X[trainObs,,drop=FALSE]
Xtest <- X[-trainObs,,drop=FALSE]

simParms <- c(0.0, 1.0, 0.1, 1.0, 2.0, 0.1)
modelPriors <- c(0, 100^2, 0.5*sd(Y), 100)
M <- 1.0
```

```
niter <- 100000
nburn <- 50000
nthin <- 50

nout <- (niter - nburn)/nthin

mh <- c(1,10)

# Run MCMC algorithm for Gaussian PPMx model
out1 <- gaussian_ppmx(y=Ytrain, X=Xtrain, Xpred=Xtest, M=M, PPM=FALSE,
              meanModel = 1,
          similarity_function=1,
          consim=1,
          calibrate=0,
          simParms=simParms,
          modelPriors = modelPriors,
          draws=niter, burn=nburn, thin=nthin,
          mh=mh)



# plot a select few posterior distributions
plot(density(out1$mu[,1])) # first observation's mean
plot(density(out1$sig2[,1])) # first observation's variance
plot(table(out1$nc)/nout,type='h') # distribution
plot(density(out1$mu0), type='l')
plot(density(out1$sig20))


# The first partition iterate is used for plotting
# purposes only. We recommended using the salso
# R-package to estimate the partition based on Si
pairs(bear[trainObs,ck],col=out1$Si[1,], pch=out1$Si[1,])


# Compare fit and predictions when covariates are not included
# in the partition model.  That is, refit data with PPM rather than PPMx
out2 <- gaussian_ppmx(y=Ytrain, X=Xtrain, Xpred=Xtest, M=M, PPM=TRUE,
              meanModel = 1,
          similarity_function=1,
          consim=1,
          calibrate=0,
          simParms=simParms,
          modelPriors = modelPriors,
          draws=niter, burn=nburn, thin=nthin,
          mh=mh)



oldpar <- par(no.readonly = TRUE)

par(mfrow=c(1,2))
```

```
plot(Xtrain[,1], Ytrain, ylab="weight", xlab="length", pch=20)
points(Xtrain[,1], apply(out2$fitted,2,mean), col='red',pch=2, cex=1)
points(Xtrain[,1], apply(out1$fitted,2,mean), col='blue',pch=3, cex=1)
legend(x="topleft",legend=c("Observed","PPM","PPMx"),
          col=c("black","red","blue", "green"),pch=c(20,2,3,4))


plot(Xtest[,1], Ytest, ylab="weight", xlab="length",pch=20)
points(Xtest[,1], apply(out2$ppred,2,mean), col='red',pch=2, cex=1)
points(Xtest[,1], apply(out1$ppred,2,mean), col='blue',pch=3, cex=1)
legend(x="topleft",legend=c("Observed","PPM","PPMx"),
          col=c("black","red","blue","green"),pch=c(20,2,3,4))



par(oldpar)
```

`icp_ppm` *Function that fits the multivariate independent product partition change point model*

Description

`icp_ppm` is a function that fits a Bayesian product partition change point model. Each series is treated independently.

Usage

```
icp_ppm(ydata,
          a0, b0,
          mltypes,
          thetas,
          nburn, nskip, nsave,
          verbose = FALSE)
```

Arguments

`ydata`	An $L \times n$ data matrix, where L is the number of time series and n, the number of time points.
`a0`	Vector of dimension L with shape 1 Beta parameters (see Details).
`b0`	Vector of dimension L with shape 2 Beta parameters (see Details).
`mltypes`	Type of marginal likelihood. Currently only available is:

- `mltypes = 1`. Observations within a block are conditionally independent $Normal(\mu, \sigma^2)$ variates with mean μ and variance σ^2. The desired marginal likelihood is obtained after integrating (μ, σ^2) with respect to a $Normal-Inverse-Gamma(\mu_0, \kappa_0, \alpha_0, \beta_0)$ prior.

`thetas` An $L \times q$ matrix containing hyperparameters associated with the marginal likelihood. The number of rows (L) corresponds to the number of series. The number of columns (q) depend on the marginal likelihood:

- If `mltypes = 1`, then $q = 4$ and `thetas` equals the hyperparameter $(\mu_0, \kappa_0, \alpha_0, \beta_0)$ of the Normal-Inverse-Gamma prior.

`nburn` The number of initial MCMC iterates to be discarded as burn-in.

`nskip` The amount to thinning that should be applied to the MCMC chain.

`nsave` Then number of MCMC iterates to be stored.

`verbose` Logical indicating whether to print to screen the MCMC progression. The default value is `verbose = FALSE`.

Details

As described in Barry and Hartigan (1992) and Loschi and Cruz (2002), for each time series $\boldsymbol{y}_i = (y_{i,1}, \ldots, y_{i,n})'$:

$$\boldsymbol{y}_i \mid \rho_i \sim \prod_{j=1}^{b_i} \mathcal{F}(\boldsymbol{y}_{i,j} \mid \boldsymbol{\theta}_i)$$

$$\rho_i \mid p_i \sim p_i^{b_i-1}(1-p_i)^{n-b_i}$$

$$p_i \sim Beta(a_{i,0}, b_{i,0}).$$

Here, $\rho_i = \{S_{i,1}, \ldots, S_{i,b_i}\}$ is a partition of the set $\{1, \ldots, n\}$ into b_i contiguous blocks, and $\boldsymbol{y}_{i,j} = (y_{i,t} : t \in S_{i,j})'$. Also, $\mathcal{F}(\cdot \mid \boldsymbol{\theta}_i)$ is a marginal likelihood function which depends on the nature of $\boldsymbol{y}_i$, indexed by a hyperparameter $\boldsymbol{\theta}_i$. Notice that p_i is the probability of observing a change point in series i, at each time $t \in \{2, \ldots, n\}$.

Value

The function returns a list containing arrays filled with MCMC iterates corresponding to model parameters. In order to provide more detail, in what follows let M be the number of MCMC iterates collected. The output list contains the following:

- C. An $M \times \{L(n-1)\}$ matrix containing MCMC iterates associated with each series indicators of a change point. The mth row in `C` is divided into L blocks; the first $(n-1)$ change point indicators for time series 1, the next $(n-1)$ change point indicators for time series 2, and so on.
- P. An $M \times \{L(n-1)\}$ matrix containing MCMC iterates associated with each series probability of a change point. The mth row in `P` is divided into L blocks; the first $(n-1)$ change point probabilities for time series 1, the next $(n-1)$ change point probabilities for time series 2, and so on.

Examples

```
# Generate data that has two series, each with 100 observations
y1 <- replicate(25, rnorm(4, c(-1, 0, 1, 2), c(0.1, 0.25, 0.5, 0.75)))
y2 <- replicate(25, rnorm(4, c(2, 1, 0, -2), c(0.1, 0.25, 0.5, 0.75)))
y <- rbind(c(t(y1)), c(t(y2)))
n <- ncol(y)
# Marginal likelihood parameters
thetas <- matrix(1, nrow = 2, ncol = 4)
thetas[1,] <- c(0, 1, 2, 1)
thetas[2,] <- c(0, 1, 2, 1)

# Fit the Bayesian ppm change point model
fit <- icp_ppm(ydata = y,
               a0 = c(1, 1),
               b0 = c(1, 1),
               mltypes = c(1, 1),
               thetas = thetas,
               nburn = 1000, nskip = 1, nsave = 1000)

cpprobsL <- matrix(apply(fit$C,2,mean), nrow=n-1, byrow=FALSE)
```

---

ordinal_ppmx *Function that fits ordinal probit model with a PPMx as a prior on partitions*

Description

ordinal_ppmx is the main function used to fit ordinal probit model with a PPMx as a prior on partitions.

Usage

```
ordinal_ppmx(y, co, X=NULL,Xpred=NULL,
               meanModel=1,
               cohesion=1,
               M=1,
               PPM = FALSE,
               similarity_function=1,
               consim=1,
               calibrate=0,
               simParms=c(0.0, 1.0, 0.1, 1.0, 2.0, 0.1, 1),
               modelPriors=c(0, 10, 1, 1),
               mh=c(0.5, 0.5),
               draws=1100,burn=100,thin=1,
               verbose=FALSE)
```

Arguments

`y` Response vector containing ordinal categories that have been mapped to natural numbers beginning with 0

`co` Vector specifying the boundaries associated with auxiliary variables of the probit model. If the number of ordinal categories is c, then the dimension of this vector must be c+1.

`X` a data frame whose columns consist of covariates that will be incorporated in the partition model. Those with class of "character" or "factor" will be treated as categorical covariates. All others will be treated as continuous covariates.

`Xpred` a data frame containing covariate values for which out of sample predictions are desired. The format of Xpred must be the same as for X.

`meanModel` Type of mean model included in the likelihood that is to be used

- 1 - cluster-specific means with no covariates in likelihood.
- 2 - cluster-specific intercepts and a global regression of the type Xbeta is included in the likelihood.

`cohesion` Type of cohesion function to use in the PPMx prior.

- 1 - Dirichlet process style of cohesion c(S) = M x (|S| - 1)!
- 2 - Uniform cohesion c(S) = 1

`M` Precision parameter of the PPMx if a DP style cohesion is used. See above. Default is 1.

`PPM` Logical argument that indicates if the PPM or PPMx partition model should be employed. If PPM = FALSE, then an X matrix must be supplied.

`similarity_function`
Type of similarity function that is employed for the PPMx prior on partitions. Options are

- 1 - Auxilliary similarity
- 2 - Double dipper similarity
- 3 - Cluster variance or entropy for categorical covariates
- 4 - Mean Gower disimilarity (this one not available if missing values are present in X)

`consim` If similarity_function is set to either 1 or 2, then consim specifies the type of marginal likelihood used as the similarity function. Options are

- 1 - N-N(m0, s20, v) (v variance of ”likelihood”, m0 and s20 ”prior” parameters),
- 2 - N-NIG(m0,s20, k0, nu0) (m0 and s20 center and scale of Gaussian, k0 and nu0)

`calibrate` This argument determines if the similarity should be calibrated. Options are

- 0 - no calibration
- 1 - standardize similarity value for each covariate
- 2 - coarsening is applied so that each similarity is raised to the 1/p power

`simParms` Vector of parameter values employed in the similarity function of the PPMx. Entries of the vector correspond to

- m0 - center continuous similarity with default 0,
- s20 - spread of continuous similarity with default 1,
- v2 - spread of 'likelihood' for conitnuous similarity (smaller values place more weight on partitions with clusters that contain homogeneous covariate values)
- k0 - degrees of freedom upper for v (only used for N-NIG similarity model)
- nu0 - scale for v (only used for N-NIG similarity model)
- a0 - dirichlet weight for categorical similarity with default of 0.1 (smaller values place more weight on partitions with individuals that are in the same category.)
- alpha - weight associated with cluster-variance and Gower disimilarity

`modelPriors` Vector of prior parameter values used in the PPMx prior.

- m - prior mean for mu0 with default equal to 0,
- s2 - prior variance mu0 with default equal to 100^2,
- A - upper bound on sigma2*_j with default equal to 10
- B - upper bound on sig20 with default equal to 10

`mh` two dimensional vector containing values for tunning parameter associated with MH update for sigma2 and sigma20

`draws` number of MCMC iterates to be collected. default is 1100

`burn` number of MCMC iterates discared as burn-in. default is 100

`thin` number by which the MCMC chain is thinne. default is 1. Thin must be selected so that it is a multilple of (draws - thin)

`verbose` Logical indicating if information regarding data and MCMC iterate should be printed to screen

Details

This function fits an ordinal probit model with either a PPM or PPMx prior on partitions. For details on the ordinal probit model see Kottas et al (2005) and Page, Quintana, Rosner (2020). Cutpoints listed in the "co" argument can be arbitrarily selected, but values that are too far from zero will result in an algorithm that will require more burn-in. Based on these cutpoints latent variables are introduced. The latent variables are assumed to follow a Gaussian distribution with cluster-specific means and variances. If meanModel = 2, then a "global" regression component is added to the mean resulting in a model with cluster-specific parallel regression lines. Commonly used conjugate priors are then employed in the regression component.

If covariates contain missing values, then the approach developed in Page, Quintana, Mueller (2020) is automatically employed. Missing values must be denoted using "NA". Currently, NAs cannot be accommodated if a "global" regression is desired (i.e., meanMode = 2).

We recommend standardizing covariates so thay they have mean zero and standard deviation one. This makes the default values provided for the similarity function reasonable in most cases. If covariates are standardized and meanModel = 2 the regression coefficients are estimated on the original scale and are ordered such that the continuous covariates appear first and the categorical covariates come after.

The MCMC algorithm used to sample from the joint posterior distribution is based on algorithm 8 found in Neal 2000.

Value

The function returns a list containing arrays filled with MCMC iterates corresponding to model parameters and also returns a couple of model fit metrics. In order to provide more detail, in what follows let

"T" - be the number of MCMC iterates collected,

"N" - be the number of observations,

"P" - be the number of predictions.

"C" - be the total number of covariates

The output list contains the following

- mu - a matrix of dimension (T, N) containing MCMC iterates associated with each subjects mean parameter (mu*_c_i).
- sig2 - a matrix of dimension (T, N) containing MCMC iterates associated with each sujbects variance parameter (sigma2*_c_i)
- beta - available only if meanModel = 2, then this is a matrix of dimension (T,C) containing MCMC iterates associated coefficients in the global regression
- Si - a matrix of dimension (T, N) containing MCMC iterates assocated with each subjects cluster label.
- zi - a matrix of dimension (T, N) containing MCMC iterates assocated with each subjects latent variable.
- mu0 - vector of length T containing MCMC iterates for mu0 parameter
- sig20 - vector of length T containing MCMC iterates for sig20
- nclus - vector of length T containing number of clusters at each MCMC iterate
- like - a matrix of dimension (T, N) containing likelihood values at each MCMC iterate.
- WAIC - scalar containing the WAIC value
- lpml - scalar containing lpml value
- fitted.values - a matrix of dimension (T, N) containing fitted values at the latent variable level for each subject at each MCMC iterate
- ppred - a matrix of dimension (T, P) containing out of sample preditions at the latent variable level for each “new” subject at each MCMC iterate
- predclass - a matrix of dimension (T, P) containing MCMC iterates of cluster two which "new" subject is allocated
- rbpred - a matrix of dimension (T, P) containing out of sample preditions at the latent variable level for each "new" subject at each MCMC iterate based on the rao-blackwellized prediction
- predclass_prob - a matrix of dimension (T, P*N) that contains the cluster allocation probabilities. They are organized so that each row corresponds to an MCMC iterate. Letting nclus represent the number of components at the t-th MCMC sample, the first nclus columns of row t correspond to probabilities for new subject 1. Then columns (N+1):(N+nclus) correspond to probabilities associated with new subject 2, etc.
- ord.fitted.values - a matrix of dimension (T, N) containing fitted values on the ordinal variable scale for each subject at each MCMC iterate.

- ord.ppred - a matrix of dimension (T, P) containing out of sample preditions on the ordinal variable scale for each “new” subject at each MCMC iterate from the posterior predictive distribution.
- ord.rbpred - a matrix of dimension (T, P) containing out of sample preditions on the ordinal variable scale for each "new" subject at each MCMC iterate based on the rao-blackwellized prediction.

Examples

```
n <- 100
# Continuous Covariate
X1 <- runif(n, 0,1)

# Binary Covariate
X2 <- rbinom(n, 1, 0.5)

pi <- exp(2*X1 + -2*X2)/(exp(2*X1 + -2*X2) + 1)

# Binary response
Y <- rbinom(n, 1, pi)


keep <- 1:(n-25)

X <- data.frame(X1=X1, X2=as.factor(X2))

Xtn <- X[keep,]
ytn <- Y[keep]
Xtt <- X[-keep,]
ytt <- Y[-keep]


# Since we have a binary response there are two "latent states".
# The boundaries of the latent states can be selected arbitrarily.
# Below I essentially use (-Inf, 0, Inf) to define the two latent spaces.
co <- c(-100000, 0, 100000)


#             m0   s20  v    k0   n0   a0   alpha
simParms <- c(0.0, 1.0, 0.5, 1.0, 2.0, 0.1, 1)
#                m  s2  s  s0
modelPriors <- c(0, 10, 1, 1)


draws <- 50000
burn <- 25000
thin <- 25
nout <- (draws - burn)/thin


# Takes about 20 seconds to run
```

```
fit <- ordinal_ppmx(y = ytn, co=co, X=Xtn, Xpred=Xtt,
                     meanModel=1,
                     similarity_function=1, consim=1,
                     calibrate=0,
                     simParms=simParms,
                     modelPriors=modelPriors,
                     draws=draws, burn=burn, thin=thin, verbose=FALSE)

# The first partition iterate is used for plotting
# purposes only. We recommended using the salso
# R-package to estimate the partition based on Si
pairs(cbind(Y, X), col=fit$Si[1,])
```

`ozone` *Ozone data*

Description

data set consists of 112 measurements of maximum daily ozone in Rennes. In addition, temperature (T), nebulosity (Ne), and projection of wind speed vectors (Vx) were measured three times daily (9:00, 12:00, and 15:00 hours) resulting in nine covariates.

Format

data: A data frame with 112 rows and the following variables:

num observed number of cancer cases

maxO3 max daily ozone

T9-T15 temperature measured at 9:00, 12:00, and 15:00 hours

Ne9-Ne15 nebulosity measured at 9:00, 12:00, and 15:00 hours

Vx9-Vx15 projection of wind speed vectors measured at 9:00, 12:00, and 15:00 hours

max03v max daily ozone of previous day.

WindDirection The wind direction

Source

Source of data: https://github.com/njtierney/user2018-missing-data-tutorial/

rppmx *Function generates random realizations from a PPM or PPMx*

Description

rppmx Employes the ploya urn sampling scheme to randomly generate a partition from the PPM or PPMx.

Usage

```
rppmx(m, X=NULL,
      similarity,
      simparm,
      M=1,
      m0=0,s20=1,v=2,k0=10,v0=1,alpha=1)
```

Arguments

m	Number of unites that are allocated to partitions
X	a data frame whose columns consist of covariates that will be incorporated in the partition model. Those with class of "character" or "factor" will be treated as categorical covaraites. All others will be treated as continuous covariates. If NULL, then a PPM partition is produced.
similarity	Type of similarity function that is employed for covariates. Options are 1 - Auxilliary similarity, 2 - Double dipper similarity 3 - variance similarity
simparm	Type of similarty model employed for continuous covariates. Options are 1 - N-N(m0, s20, v) (v variance of ”likelihood”, m0 and s20 ”prior” parameters), 2 - N-NIG(m0,k0, k0, v0, s20) (m0 and k0 center and scale of Gaussian, n0 and s20 shape and scale of IG)
M	Precision parameter. Default is 1.
m0	Continuous similarity function value (see above)
s20	Continuous similarity function value (see above)
v	Continuous similarity function value (see above)
k0	Continuous similarity function value (see above)
v0	Continuous similarity function value (see above)
alpha	Penalty value when using the variance similarity

Details

Use polya urn scheme to sample from the PPM or the PPMx

Value

The function returns randomly generated partition

Examples

```
X <- cbind(rnorm(100), rbinom(100,1,0.5))
p <- rppmx(m=100, X=X, similarity=1, simparm=1, M=1)
p
```

---

scallops *Scallops data*

Description

Data set that provides the location and scallop catches in the Atlantic waters off the coasts of New Jersey and Long Island, New York.

Format

data: A data frame with 148 rows and the variables are the following:

strata

sample

lat

long

tcatch

prerec

recruits

Source

Banerjee, S; Carline, B. P.; Gelfand, A. E.; (2015) Hierarchical Modeling and Analysis for Spatial Data 2nd Ed. CRC. Press

SIMCE *Standardized testing data in Chile*

Description

Average standard testing results with average mother's and father's education level for schools in the greater Santiago area of Chile. Measurements are recorded from 2005-2011 and spatial coordinates of the schools are provided.

Format

data: A data frame with 1072 rows and the following variables:

coords.x1 longitude coordinates of school

coords.x2 lattitude coordinates of school

Schoole Unique school identifier

COMUNA Name of the commune in which the school resides

SIMCE05-SIMCE11 Math score of standardized test in 2005-2011

EDpad05-EDpad11 Average level of father's education of students that attended school 2005-2011

EDmad05-EDmad11 Average level of mother's education of students that attended school 2005-2011

Source

Page, G. L. and Quintana, F. A. (2016) Spatial Product Partition Models, Bayesian Anal., Volume 11, Number 1, 265-298.

sppm *Function that fits spatial product partition model with Gaussian likelihood*

Description

sppm is the main function used to fit model with Guassian likelihood and spatial PPM as prior on partitions.

Usage

```
sppm(y,s,
    s.pred=NULL,
    cohesion,
    M=1,
    modelPriors=c(0, 100^2, 10, 10),
    cParms=c(1, 1.5, 0, 1, 2, 2),
    mh=c(0.5, 0.5),
    draws=1100,burn=100,thin=1)
```

Arguments

`y`	numeric vector containing response variable
`s`	Two-column matrix containing spatial locations (i.e., longitude and lattitude).
`s.pred`	Two-column matrix containing spatial locations at which out-of-sample predictions will be collected.
`cohesion`	Scalar that indicates which cohesion to use. 1. distance from centroids 2. upper bound 3. auxiliary similarity 4. double dipper similarity
`M`	Parameter related to Dirichlet process scale or dispersion parameter.
`modelPriors`	Vector containing model prior values (see below for more details)
`cParms`	Vector containing partition model prior values (see below for more details)
`mh`	Tuning standard deviations for metropolis updates for sigma2 and sigma20
`draws`	Number of MCMC samples to collect
`burn`	Number of the MCMC samples discarded in the burn-in phase of the sampler
`thin`	The amount of thinning desired for the chain

Details

The vector `modelPriors = c(m0, s20, ms, ms0)` where each prior parameter is listed in the model description below.

The cParm vector contains values associated with the cohesion function.

```
cParm = c(
  epsilon value - cohesion 1 only,
  distance bound - cohesion 2 only,
  mu0 - center of NNIG for cohesion 3 and 4
  k0 - scale parm of gaussian part of NNIG for cohesion 3 and 4
  v0 - degrees of freedom IG part of NNIG for cohesion 3 and 4
 L0 - scale parm (scalar of identity matrix) IG part of NNIG for cohesion 3 and 4).
```

The model this function fits is Gaussian likelihood model using the sPPM prior on partitions (Page and Quintana, 2016). Specific model details are

$$y_i|\mu^*, \sigma^{2*}, c_i \sim N(\mu^*_{c_i}, \sigma^{2*}_{c_i}), i = 1, \dots, n$$

$$\mu^*_j|\mu_0, \sigma^2_0 \sim N(\mu_0, \sigma^2_0)$$

$$\sigma^*_j|A \sim UN(0, ms)$$

$$\rho|M, \xi \sim sPPM$$

To complete the model specification, the folloing hyperpriors are assumed,

$$\mu_0 | m, s^2 \sim N(m0, s0^2)$$

$$\sigma_0 | B \sim UN(0, ms0)$$

Note that we employ uniform prior distributions on variance components as suggest in Gelman's 2006 Bayesian paper. "sPPM" in the model specificaiton denotes the the spatial product partition model. The computational implementation of the model is based algorithm 8 found in Neal's 2000 JCGS paper.

Value

This function returns in a list all MCMC interates for each model parameter, posterior predictive, and fitted values. In addition the LPML model fit metric is provided.

Examples

```
data(scallops)

Y<-log(scallops[,5]+1)
s_coords <- scallops[,3:4] #lat and long
m <- dim(s_coords)[1]



# standardize spatial coordinates
smn <- apply(s_coords,2,mean)
ssd <- apply(s_coords,2,sd)
s_std <- t((t(s_coords) - smn)/ssd)

# Create a grid of prediction locations
np <- 10

sp <- expand.grid(seq(min(s_coords[,1]), max(s_coords[,1]),length=np),
                  seq(min(s_coords[,2]), max(s_coords[,2]), length=np))

sp_std <- t((t(sp) - smn)/ssd) # standardized prediction spatial coordinates


niter <- 20000
nburn <- 10000
nthin <- 10
nout <- (niter - nburn)/nthin


out <- sppm(y=Y,s=s_std,s.pred=sp_std,cohesion=4, M=1, draws=niter, burn=nburn, thin=nthin)
```

```
# fitted values
fitted.values <- out$fitted
fv.mn <- apply(fitted.values, 2,mean)
mean((Y - fv.mn)^2) # MSE
out$lpml #lpml value

ppred <- out$ppred
predmn <- apply(ppred,2,mean)

# The first partition iterate is used for plotting
# purposes only. We recommended using the salso
# R-package to estimate the partition based on Si
Si <- out$Si
plot(s_coords[,1], s_coords[,2], col=Si[1,])
```

Index