PPMX documentation

of 'ppmx-package.Rd' etc.

November 27, 2008

R topics documented:

`ppmx-package` *Random partition model with covariates.*

Description

The package implements posterior inference for the PPMx model. The PPMx model is an extension of the product partition model to include a regression on covariates.

Details

Package:	ppmx
Type:	Package
Version:	1.0
Date:	2007-04-01
License:	GNU public domain software.

The function `ppmx` implements posterior MCMC simulation for the PPMx model assuming a normal sampling model and the random partition model of the Dirichlet process prior (Polya urn) as the underlying PPM. The function `pltS` plots posterior predictive inference summaries.

The package uses the model described in *Mueller, Quintana and Rosner (2008)*. Let $p(y_i \mid \rho, \mu, V) = N(\mu_i, V_i)$ denote the observed responses for experimental units $i = 1, \ldots, n$. Here $\rho = \{S_1, \ldots, S_k\}$ is a random partition of the experimental units, $(\mu_i, V_i) = (\mu_j^\star, V_j^\star)$ for all $i \in S_j$ and $(\mu^star_j, V_j^\star) \sim G^\star$, i.i.d. We model the random partition ρ as a PPM model with an additional similarity function.

Let $x_j^\star = (x_i,\ i \in S_j)$ denote the covariates arranged by clusters and let $g(x^star_j)$ denote a function that formalizes similarity of a set of covariates (similarity function). Let $c(S_j) = M(|S_j| - 1)!$ denote the cohesion function implied by a Dirichlet process prior with total mass parameter M. We assume

$$p(\rho) \propto \prod c(S_j) g(x_j^\star).$$

The base measure G^star is chosen conjugate to the normal sampling model, $G^\star(\mu, V^{-1}) = G_\mu^\star(\mu) G_V^\star(V^{-1})$ where

$$G_\mu^\star(\mu) = N(m, B) \text{ and } G_V^\star(V^{-1}) = Wishart(s, (sS)^{-1}).$$

The Wishart distribution is parametrized such that $E(V_{-1}) = S^{-1}$. We assume conjugate hyperpriors

$$m \sim N(a, A), B^{-1} \sim Wishart(c, (cC)^{-1}), S \sim Wishart(q, R/q).$$

The model is completed with a Gamma prior for the total mass paramter $\alpha \sim Ga(a_0, b_0)$.

Author(s)

Peter Mueller, Maintainer: Peter Mueller ⟨pm@wotan.mdacc.tmc.edu⟩

References

Müller, P., Quintana, F, and Rosner, G. (2008), "Bayesian Clustering with Regression", Technical report http://odin.mdacc.tmc.edu/~pm/

See Also

See also the package DPpackage, at

http://student.kuleuven.be/~s0166452/software.html.

`ppmx` *PPMX – Random partition with covariates*

Description

Fits the PPMx model for a random partition including a regression on covariates.

Usage

```
ppmx <-  function(Yin=NULL,p0=1,p1=NULL,p2=NULL,q2=NULL,p4=0,
     censoring=0,
     cov.Vj1=NULL, cov.mean1=NULL, cov.B1=NULL,
     cov.std1=1, cov.pi2=NULL, cov.a4=NULL,  cov.b4=NULL,
     n.iter=2000, n.discard=1000, n.reinit=10000,
     n.batch=50, m.prior=0, B.prior=0, S.prior=0, verbose=1,
     py=1, iy=0, ny=50, ygrid=NULL, xgrid=NULL,
     s=15, S.init=NULL, qq=5, R=NULL, B.scale=100, cc=5, m.init=NULL,
     a=NULL, A.inv=NULL, alpha=1, a0=1, b0=1, k0=NULL)
```

Arguments

Yin	response data as (n x p0) matrix of p0-dimensional responses for n experimental units. Either the data matrix itself or a file name for a text file with the data. If censoring=1 then Yin is a (n x 3) matrix with columns equal to the event time, an indicator for an observed event and the censoring time.
p0	dimension of the response vector for each data record
p1	number of continuous covariates (can be 0 for none)
p2	number of categorical covariates.
q2	vector of length p2; number of levels for each categorical covariate.
p4	number of count covariates.
censoring	indicator for censoring, censoring=1 is only allowed with p0=1.
cov.Vj1	vector of length p1, variance parameters for the similarity function for continuous covariates.
mean1	vector of length p1, prior mean of the location parameter for the similarity function for continuous covariates.
mean1	vector of length p1, prior variance for the location parameter in the similarity function for continuous covariates.
cov.std1	indicator for standardizing continuous covariates to sample mean 0 and standard deviation 1.
cov.pi2	array with p2 rows and q2[j] columns in row j. The j-th row defines the Dirichlet parameters for the similarity function for the j-th categorical covariate.
cov.a4	vector of length p4, defines the shape parameter of the Gamma prior in the similarity function for count covariates.
cov.b4	vector of length p4, defines the scale parameter of the Gamma prior in the similarity function for count covariates.
n.iter	number MCMC iterations
n.discard	initial transient
n.reinit	reinitialize every n.reinit iterations (not normally used)
n.batch	save imputed parameters every n.batch iterations
n.predupdate	save posterior predictive summaries every n.predupdate iterations
m.prior	indicator for resampling (1) versus fixing (0) the mean of the base measure, m
B.prior	indicator for resampling (1) versus fixing (0) the variance-covariance matrix of the base measure, B
S.prior	indicator for resampling (1) versus fixing (0) the sampling variance-covariance matrix.
verbose	level of comments
py	indicator for evaluating posterior predictive $p(y \mid data)$ for a future observation.
iy	The coordinate of the response for posterior predictive inference.
ny	Size of the grid on which to evaluate posterior predictive inference.

ygrid	vector of length 2, lower and upper bound of the grid on which to evaluate posterior predictive inference.
nx	number of covariate combinations for which to carry out the posterior predictive inference.
xgrid	$(nx \times p1 + p2 + p4)$ matrix of covariate values for which to carry out the posterior predictive inference.
s	degrees of freedom for the inverse Wishart prior on S
S.init	initial value for the kernel covariance matrix S
qq	degrees of freedom for the inverse Wishart base measure for $G_0(V_i)$
R	expectation of the inverse Wishart base measure $G_0(V_i)$
R	Describe R here
B.scale	The initial value for the covariance matrix B of the base measure $G_0(\mu) = N(m, B)$ is set to $B.scale \cdot S.init$.
cc	degrees of freedom of the inverse Wishart hyperprior for B
m.init	initial value for the mean m of the base measure $G_0(\mu) = N(m, B)$
a	hyperprior mean for m
A.inv	inverse of the hyperprior covariance matrix for m
alpha	initial value for the total mass parameter $alpha$
a0	hyperprior parameters for prior on total mass paramter α
b0	hyperprior parameters for prior on total mass paramter α
k0	initial number of distinct clusters

Details

See ppmx-package for a statement of the probability model. The function ppmx initializes and carries out MCMC posterior simulation. Simulation output is saved in the working directory. Change it by using setwd if desired.

The parameters cov.Vj1, cov.mean1, cov.B1, cov.pi2, k0, S.ini, R, m.init, A.inv, a, iy, ygrid, xgrid are set to default values if set to NULL. If NULL, the arguments cov.Vj1, cov.mean1, cov.B1, cov.pi2, S.ini, R, m.init, A.inv are replaced by suitable empirical moments of the data. The initial number of clustes k0 is set to n if there are no categorical covariates. If there are categorical covariates, then k0 is determined by the cross-tabulation of these categorical covariates. If $py = 1$ and xgrid is NULL, then xgrid is replaced by a default choice depending on p1+p2. If $p1 + p2 = 0$ then xgrid is simply chosen as the covariates of the first 5 data points. If $p1 + p2 > 0$ then xgrid is defined by all possible cross tabulations of the p2 categorical covariates and low, medium and high values of the p1 continuous covariates. See the file py-x.mdp for a copy of xgrid. Finally, if NULL then ygrid is set to the range of the response variable.

Output files:

par.mdp	imputed hyperparameters for each iteration of the MCMC. Columns are: iteration number, α, η, k
mu.mdp	imputed $\mu_j^\star$, $j = 1, \ldots, k$. Columns are iteration number, n_j, $\mu_j^\star$.

	(newlines are not alligned with j; n_j = size of j-th cluster)
V.mdp	imputed $V_j^\star$. Columns are it no, n_j, V_j (stacked as one row vector).
member.mdp	imputed cluster membership indicators $s_i,\ i = 1, \ldots, n;\ s_i \in \{0, \ldots, k-1\}$ Columns are iteration no, n clusters k, $s_i,\ i = 1, \ldots, n$
S.mdp	$(nx \times ny)$; estimated survival function $S(y_{iy} \mid x)$ one row per line of `xgrid`.
pdf.mdp	$(nx \times ny)$; estimated pdf $p(y_{iy} \mid x)$

Additional files are B.mdp (imputed variance-covariance matrix for cluster specific means), mean.mdp (imputed m), mj1.mdp (parameters for similarity function for continuous covariates), pij2.mdp (for categorical covariates), aj4.mdp (parameters for similarity function for count covariates), See the example in Mueller, Quintana and Rosner (2008, Section 6.1) for a definition of μ, V, m, B and other parameters. See Section 4 for a definition of the similarity functions. The underlying PPM is the PPM induced by a Dirichlet process prior with total mass parmaeter α. The parameter η is a latent variable used in the MCMC transition probability for α.

Value

The function returns no value. Simulation output is written to files.

Note

Careful, `ppmx` writes temporary files into the current working directory. The same files are used by `Splt` to plot posterior predictive distributions.

Author(s)

Peter Mueller ⟨pm@wotan.mdacc.tmc.edu⟩

References

the package uses the parametrization defined in:

Müller, P., Quintana, F, and Rosner, G. (2008), "Bayesian Clustering with Regression", Technical report http://odin.mdacc.tmc.edu/~pm/

See Also

See also the DPpackage, at

http://student.kuleuven.be/~s0166452/software.html.

Examples

```
## Not run:
require("ppmx")

#####################################################################
# prepare
#####################################################################
```

```
## prepare the data file:
data(dtasim)
N <- nrow(dtasim)
## now select a sub-sample of n=200 data points for the example
n <- 200                              # desired sample size
idx <- sample(1:N,n)
Yin <- as.matrix( dtasim[idx,] )

## parameters for posterior predictive inference
xgrid <- read.table("py-x.mdp")       # cov combinations for prediction
nx <- nrow(xgrid)                     # n of covariate combinations
ny <- 50                              # size of grid for prediction

######################################################################
# call ppmx
######################################################################

ppmx(Yin,n.discard=500,n.iter=3000,
     p0=1,                            # 1-dim response
     p1=1,p2=2,p4=0,                  # 1 cont and 2 categorical cov's
     B.prior=1,S.prior=1,n.batch=50,
     xgrid=xgrid)                     # request posterior pred inference

######################################################################
# some inference summaries
######################################################################

## read in some of the inference summaries
S <-   matrix(scan("py-S.mdp"),byrow=T,ncol=ny) # S is a (nx x ny) matrix
pdf <- matrix(scan("py-p.mdp"),byrow=T,ncol=ny) # pdf    (nx x ny) matrix
ygrid <- scan("py-y.mdp")                       # grid for predictive
nx <- nrow(xgrid)

matplot(ygrid,t(S),type="l",xlab="Y",ylab="SURVIVAL BY COVARIATE",
        bty="l")
legend(x=80,y=max(S),col=0:nx,lty=0:nx,
       legend=
       c("COVARIATES",paste(xgrid[,1],xgrid[,2],xgrid[,3],sep="\t")),
       bty="n",cex=0.5)

matplot(ygrid,t(pdf),type="l",xlab="Y",ylab="DIST BY COVARIATE",
        bty="l")
legend(x=80,y=max(pdf),col=0:nx,lty=0:nx,
       legend=
       c("COVARIATES",paste(xgrid[,1],xgrid[,2],xgrid[,3],sep="\t")),
       bty="n",cex=0.5)

## same summaries using pltS(.) function
pltS(idx=1:12,plt="p",lgd=T)          # plot post pred pdf
pltS(idx=1:12,plt="h")                # ... hazard
pltS(idx=1:12,plt="S",lgd=T)          # ... survival
pltS(idx=1:2,sd=T,plt="S",lgd=T)      # with pointwise post pred sd
```

```
## some summaries of the random clustering
## report on clustering of the first n0=50 data points
n0 <- 50
y <- Yin[1:n0,1]
X <- Yin[1:n0,2:4]                  # covariates
s <- read.table("member.mdp")[,2+(1:n0)]
     ## drop first two columns (it n, n clusters) and select first n0
M <- nrow(s)                        # n saved MC imputations
## empirical frquencies of pair-wise co-clustering
idx <- order(y)
sij <- matrix(0,nrow=n0,ncol=n0)
for(i in 1:(n0-1))
  for(j in (i+1):n0)
    sij[i,j] <- sum(s[,idx[i]]==s[,idx[j]])/M
for(i in 1:n0) sij[i,i] <- 1
image(sij,col=grey((1:12)/12))

## now try after ordering by covariates
idx <- order(X[,2],X[,3],X[,1])
sij <- matrix(0,nrow=n0,ncol=n0)
for(i in 1:(n0-1))
  for(j in (i+1):n0)
    sij[i,j] <- sum(s[,idx[i]]==s[,idx[j]])/M
for(i in 1:n0) sij[i,i] <- 1
image(sij,col=grey((1:12)/12))

## track cluster membership by iteration
image(1:n0,1:M,as.matrix(t(s)), col=rainbow(12),
      xlab="OBSERV",ylab="ITERATION")

## End(Not run)
```

`pltS` *Univariate posterior predictive inference*

Description

Plots marginal posterior predictive densities, hazard and survival function estimates based on $p(y \mid x) \equiv p(y_{n+1} = y \mid x_{n+1} = x, data)$, marginalizing w.r.t. all model parameters and the random clustering. The function plots the posterior predictive for one or more coordinates of a vector valued response y.

Usage

```
pltS <- function(idx=c(1,7),yl=NULL, xlim=NULL,
        col=NULL,sd=F,a=1.5, cex=0.7, lty=NULL, lwd=NULL,
        plt="p")
```

Arguments

idx	coordinates of a vector values response y to be plotted. Indices must be between 1 and p0.
yl	domain of the y-axis
xlim	domain of the x-axis
col	vector of as many colors as the size of idx.
lg	plot log density
sd	include plus/minus a posterior predictive standard deviation.
a	defines the step of standard deviation bounds. See previous item.
lty	vector of as many line type parameters as the size of idx.
lwd	vector of as many line width parameters as the size of idx.
plt	indicator for plotting the pdf ("p"), hazard ("h") or survival ("S") function.

Details

Need to call ppmx first to carry out the posterior Markov chain Monte Carlo simulation. The function pltS uses the simulation output to produce the desired posterior predictive distribution. The function assumes the simulation output is saved in the current working directory. Change it by using setwd if necessary.

The function is a simple R macro and can easily be modified for customized plots. See the use of the simulation output files py-S.mdp, py-pdf.mdp, py-pi.mdp and py-Si.mdp.

See ppmx-package for a statement of the probability model.

Value

The function returns no value.

Note

Careful, mdp writes temporary files into the current working directory. The same files are used by pltS, plt.ex and plt.exy to plot posterior predictive distributions and expectations.

Author(s)

Peter Mueller ⟨pm@wotan.mdacc.tmc.edu⟩

References

the package uses the parametrization defined in:

MacEachern, S.N. and Mueller, P. (1998). "Estimating Mixture of Dirichlet Process Models," Journal of Computational and Graphical Statistics, 7, 223–239.

See Also

See also the R library DPpackage, at

http://student.kuleuven.be/~s0166452/software.html.

Examples

```
## Not run:
require("ppmx")

#####################################################################
# prepare
#####################################################################

## prepare the data file:

data(dtasim)
N <- nrow(dtasim)
## now select a sub-sample of n=200 data points for the example
n <- 200                              # desired sample size
idx <- sample(1:N,n)
Yin <- as.matrix( dtasim[idx,] )

## parameters for posterior predictive inference
xgrid <- read.table("py-x.mdp")       # cov combinations for prediction
nx <- nrow(xgrid)                     # n of covariate combinations
ny <- 50                              # size of grid for prediction

#####################################################################
# call ppmx
#####################################################################

ppmx(Yin,n.discard=500,n.iter=3000,
     p0=1,                            # 1-dim response
     p1=1,p2=2,p4=0,                  # 1 cont and 2 categorical cov's
     B.prior=1,S.prior=1,n.batch=50,
     xgrid=xgrid)                     # request posterior pred inference

#####################################################################
# some inference summaries
#####################################################################

pltS(idx=1:12,plt="p",lgd=T)          # plot post pred pdf
pltS(idx=1:12,plt="h")                # ... hazard
pltS(idx=1:12,plt="S",lgd=T)          # ... survival
pltS(idx=1:2,sd=T,plt="S",lgd=T)      # with pointwise post pred sd

## same thing using basic R plot commands.
S <-   matrix(scan("py-S.mdp"),byrow=T,ncol=ny) # S is a (nx x ny) matrix
pdf <- matrix(scan("py-p.mdp"),byrow=T,ncol=ny) # pdf    (nx x ny) matrix
ygrid <- scan("py-y.mdp")                       # grid for predictive
nx <- nrow(xgrid)

matplot(ygrid,t(S),type="l",xlab="Y",ylab="SURVIVAL BY COVARIATE",
        bty="l")
legend(x=80,y=max(S),col=0:nx,lty=0:nx,
       legend=
       c("COVARIATES",paste(xgrid[,1],xgrid[,2],xgrid[,3],sep="\t")),
```

```
       bty="n",cex=0.5)

matplot(ygrid,t(pdf),type="l",xlab="Y",ylab="DIST BY COVARIATE",
        bty="l")
legend(x=80,y=max(pdf),col=0:nx,lty=0:nx,
       legend=
       c("COVARIATES",paste(xgrid[,1],xgrid[,2],xgrid[,3],sep="\t")),
       bty="n",cex=0.5)

## same summaries using pltS(.) funcitons
## End(Not run)
```

Index