Package 'spBPS'

May 14, 2026

Title Bayesian Predictive Stacking for Scalable Geospatial Transfer Learning

Version 2.0-1

Maintainer Luca Presicce <l.presicce@campus.unimib.it>

Author Luca Presicce [aut, cre] (ORCID: <https://orcid.org/0009-0005-7062-3523>), Sudipto Banerjee [aut]

Description Provides functions for Bayesian Predictive Stacking within the Bayesian transfer learning framework for geospatial artificial systems, as introduced in ``Bayesian Transfer Learning for Artificially Intelligent Geospatial Systems: A Predictive Stacking Approach" (Presicce and Banerjee, 2025) <doi:10.48550/arXiv.2410.09504>. This methodology enables efficient Bayesian geostatistical modeling, utilizing predictive stacking to improve inference across spatial datasets. The core functions leverage 'C++' for high-performance computation, making the framework well-suited for large-scale spatial data analysis in parallel and distributed computing environments. Designed for scalability, it allows seamless application in computationally demanding scenarios.

Depends R (>= 1.8.0)

Imports Rcpp, CVXR (>= 1.8.1), mniw

LinkingTo Rcpp, RcppArmadillo

Suggests knitr, rmarkdown, abind, mvnfast, ECOSolveR, foreach, parallel, doParallel, tictoc, MBA, RColorBrewer, classInt, sp, fields, testthat (>= 3.0.0)

Config/testthat/edition 3

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

VignetteBuilder knitr

URL https://lucapresicce.github.io/spBPS/

NeedsCompilation yes

Repository CRAN

Date/Publication 2026-05-14 14:50:02 UTC

Contents

`arma_dist` *Compute the Euclidean distance matrix*

Description

Compute the Euclidean distance matrix

Usage

```
arma_dist(X)
```

Arguments

`X`	matrix (tipically of N coordindates on $\mathbb{R}^2$)

Value

matrix distance matrix of the elements of X

`bayesMvLMconjugate` *Gibbs sampler for Conjugate Bayesian Multivariate Linear Models*

Description

Gibbs sampler for Conjugate Bayesian Multivariate Linear Models

Usage

```
bayesMvLMconjugate(Y, X, mu_B, V_B, nu, Psi, n_iter = 1000, burn_in = 500)
```

Arguments

`Y`	matrix $n \times q$ of response variables
`X`	matrix $n \times p$ of predictors
`mu_B`	matrix $p \times q$ prior mean for β
`V_B`	matrix $p \times p$ prior row covariance for β
`nu`	double prior parameter for Σ
`Psi`	matrix prior parameter for Σ
`n_iter`	integer iteration number for Gibbs sampler
`burn_in`	integer number of burn-in iteration

Value

B_samples array of posterior sample for β

Sigma_samples array of posterior samples for Σ

Examples

```
## Generate data
n <- 100
p <- 3
q <- 2
Y <- matrix(rnorm(n*q), nrow = n, ncol = q)
X <- matrix(rnorm(n*p), nrow = n, ncol = p)

## Prior parameters
mu_B <- matrix(0, p, q)
V_B <- diag(10, p)
nu <- 3
Psi <- diag(q)

## Samples from posteriors
```

```
n_iter <- 1000
burn_in <- 500
set.seed(1234)
samples <- spBPS::bayesMvLMconjugate(Y, X, mu_B, V_B, nu, Psi, n_iter, burn_in)
```

`BPS_combine` *Combine subset models wiht BPS*

Description

Combine subset models wiht BPS

Usage

```
BPS_combine(fit_list, K, rp)
```

Arguments

`fit_list`	list K fitted model outputs composed by two elements each: first named epd, second named W
`K`	integer number of folds
`rp`	double percentage of observations to take into account for optimization (`default=1`)

Value

matrix posterior predictive density evaluations (each columns represent a different model)

`BPS_postdraws_MvT` *Compute the BPS posterior samples given a set of stacking weights*

Description

Compute the BPS posterior samples given a set of stacking weights

Usage

```
BPS_postdraws_MvT(data, priors, coords, hyperpar, W, R, par)
```

Arguments

`data`	list two elements: first named Y, second named X
`priors`	list priors: named μ_B,V_r,Ψ,ν
`coords`	matrix sample coordinates for X and Y
`hyperpar`	list two elemets: first named α, second named ϕ
`W`	matrix set of stacking weights
`R`	integer number of desired samples
`par`	if `TRUE` only β and Σ are sampled (ω is omitted)

Value

matrix BPS posterior samples

`BPS_post_MvT` *Perform the BPS sampling from posterior and posterior predictive given a set of stacking weights*

Description

Perform the BPS sampling from posterior and posterior predictive given a set of stacking weights

Usage

```
BPS_post_MvT(data, X_u, priors, coords, crd_u, hyperpar, W, R)
```

Arguments

`data`	list two elements: first named Y, second named X
`X_u`	matrix unobserved instances covariate matrix
`priors`	list priors: named μ_B,V_r,Ψ,ν
`coords`	matrix sample coordinates for X and Y
`crd_u`	matrix unboserved instances coordinates
`hyperpar`	list two elemets: first named α, second named ϕ
`W`	matrix set of stacking weights
`R`	integer number of desired samples

Value

list BPS posterior predictive samples

BPS_pred_MvT — *Compute the BPS spatial prediction given a set of stacking weights*

Description

Compute the BPS spatial prediction given a set of stacking weights

Usage

```
BPS_pred_MvT(data, X_u, priors, coords, crd_u, hyperpar, W, R)
```

Arguments

data	list two elements: first named Y, second named X
X_u	matrix unobserved instances covariate matrix
priors	list priors: named μ_B,V_r,Ψ,ν
coords	matrix sample coordinates for X and Y
crd_u	matrix unboserved instances coordinates
hyperpar	list two elemets: first named α, second named ϕ
W	matrix set of stacking weights
R	integer number of desired samples

Value

list BPS posterior predictive samples

BPS_PseudoBMA — *Combine subset models wiht Pseudo-BMA*

Description

Combine subset models wiht Pseudo-BMA

Usage

```
BPS_PseudoBMA(fit_list)
```

Arguments

fit_list	list K fitted model outputs composed by two elements each: first named epd, second named W

Value

matrix posterior predictive density evaluations (each columns represent a different model)

BPS_weights_MvT *Compute the BPS weights by convex optimization*

Description

Compute the BPS weights by convex optimization

Usage

```
BPS_weights_MvT(data, priors, coords, hyperpar, K)
```

Arguments

data	list two elements: first named Y, second named X
priors	list priors: named μ_B,V_r,Ψ,ν
coords	matrix sample coordinates for X and Y
hyperpar	list two elemets: first named α, second named ϕ
K	integer number of folds

Value

matrix posterior predictive density evaluations (each columns represent a different model)

conv_opt *Solver for Bayesian Predictive Stacking of Predictive densities convex optimization problem*

Description

Solver for Bayesian Predictive Stacking of Predictive densities convex optimization problem

Usage

```
conv_opt(scores)
```

Arguments

scores	matrix $N \times K$ of expected predictive density evaluations for the K models considered

Value

W matrix of Bayesian Predictive Stacking weights for the K models considered

---

`CVXR_opt` *Compute the BPS weights by convex optimization*

Description

Compute the BPS weights by convex optimization

Usage

```
CVXR_opt(scores)
```

Arguments

`scores`	matrix $N \times K$ of expected predictive density evaluations for the K models considered

Value

conv_opt function to perform convex optimiazion with CVXR R package

---

`dens_kcv_MvT` *Compute the KCV of the density evaluations for fixed values of the hyperparameters*

Description

Compute the KCV of the density evaluations for fixed values of the hyperparameters

Usage

```
dens_kcv_MvT(data, priors, coords, hyperpar, K)
```

Arguments

`data`	list two elements: first named Y, second named X
`priors`	list priors: named μ_B,V_r,Ψ,ν
`coords`	matrix sample coordinates for X and Y
`hyperpar`	list two elemets: first named α, second named ϕ
`K`	integer number of folds

Value

vector posterior predictive density evaluations

`dens_loocv_MvT` *Compute the LOOCV of the density evaluations for fixed values of the hyperparameters*

Description

Compute the LOOCV of the density evaluations for fixed values of the hyperparameters

Usage

```
dens_loocv_MvT(data, priors, coords, hyperpar)
```

Arguments

`data`	list two elements: first named Y, second named X
`priors`	list priors: named μ_B,V_r,Ψ,ν
`coords`	matrix sample coordinates for X and Y
`hyperpar`	list two elemets: first named α, second named ϕ

Value

vector posterior predictive density evaluations

`d_pred_cpp_MvT` *Evaluate the density of a set of unobserved response with respect to the conditional posterior predictive*

Description

Evaluate the density of a set of unobserved response with respect to the conditional posterior predictive

Usage

```
d_pred_cpp_MvT(data, X_u, Y_u, d_u, d_us, hyperpar, poster)
```

Arguments

`data`	list two elements: first named Y, second named X
`X_u`	matrix unobserved instances covariate matrix
`Y_u`	matrix unobserved instances response matrix
`d_u`	matrix unobserved instances distance matrix
`d_us`	matrix cross-distance between unobserved and observed instances matrix
`hyperpar`	list two elemets: first named α, second named ϕ
`poster`	list output from `fit_cpp` function

Value

double posterior predictive density evaluation

`expand_grid_cpp` *Build a grid from two vector (i.e. equivalent to* `expand.grid()` *in* R*)*

Description

Build a grid from two vector (i.e. equivalent to `expand.grid()` in R)

Usage

```
expand_grid_cpp(x, y)
```

Arguments

`x`	vector first vector of numeric elements
`y`	vector second vector of numeric elements

Value

matrix expanded grid of combinations

`fit_cpp_MvT` *Compute the parameters for the posteriors distribution of* β *and* Σ *(i.e. updated parameters)*

Description

Compute the parameters for the posteriors distribution of β and Σ (i.e. updated parameters)

Usage

```
fit_cpp_MvT(data, priors, coords, hyperpar)
```

Arguments

`data`	list two elements: first named Y, second named X
`priors`	list priors: named μ_B,V_r,Ψ,ν
`coords`	matrix sample coordinates for X and Y
`hyperpar`	list two elemets: first named α, second named ϕ

Value

list posterior update parameters

forceSymmetry_cpp *Function to subset data for meta-analysis*

Description

Function to subset data for meta-analysis

Usage

```
forceSymmetry_cpp(mat)
```

Arguments

mat	matrix not-symmetric matrix

Value

matrix symmetric matrix (lower triangular of mat is used)

models_dens_MvT *Return the CV predictive density evaluations for all the model combinations*

Description

Return the CV predictive density evaluations for all the model combinations

Usage

```
models_dens_MvT(data, priors, coords, hyperpar, useKCV, K)
```

Arguments

data	list two elements: first named Y, second named X
priors	list priors: named μ_B,V_r,Ψ,ν
coords	matrix sample coordinates for X and Y
hyperpar	list two elemets: first named α, second named ϕ
useKCV	if TRUE K-fold cross validation is used instead of LOOCV (no default)
K	integer number of folds

Value

matrix posterior predictive density evaluations (each columns represent a different model)

post_draws_MvT *Sample R draws from the posterior distributions*

Description

Sample R draws from the posterior distributions

Usage

```
post_draws_MvT(poster, R, par, p)
```

Arguments

poster	list output from fit_cpp function
R	integer number of posterior samples
par	if TRUE only β and Σ are sampled (ω is omitted)
p	integer if par = TRUE, it specifies the column number of X

Value

list posterior samples

predict.spBPS *Predict at new locations using a fitted spBPS model*

Description

Generates posterior predictive samples at new spatial locations using a fitted spBPS object. Memory usage is controlled via pred_batch_size

Usage

```
## S3 method for class 'spBPS'
predict(
  object,
  newdata,
  draws = 200L,
  pred_batch_size = 200L,
  n_cores = 1L,
  return_summary = FALSE,
  probs = c(0.025, 0.975),
  ...
)
```

Arguments

object	Object of class "spBPS" returned by spBPS().
newdata	List with elements X (u x p covariate matrix) and coords (u x d coordinate matrix).
draws	Integer. Number of posterior predictive draws. Default 200.
pred_batch_size	Integer. Number of prediction sites processed per batch. Default 200. Smaller values use less RAM; larger values may be faster.
n_cores	Integer. Number of OMP threads. Default 1.
return_summary	Logical. If FALSE (default) return full draw arrays. If TRUE return summary statistics.
probs	Numeric vector of length 2. Quantile probabilities for credible intervals when return_summary = TRUE. Default c(0.025, 0.975).
...	Currently unused.

Value

When return_summary = FALSE, a list with arrays Wu, Yu, MY of dimension u x q x R. When return_summary = TRUE, a list with u x q matrices mu_Yu, sd_Yu, q_lo, q_hi, mu_Wu.

See Also

spBPS

Examples

```
n <- 1000
p <- 2
q <- 1

Y <- matrix(rnorm(n * q), ncol = q)
X <- matrix(rnorm(n * p), ncol = p)
coords <- matrix(runif(n * 2), ncol = 2)

data    <- list(Y = Y, X = X)
priors  <- list(mu_B = matrix(0, nrow = p, ncol = q),
                V_r  = diag(10, p),
                Psi  = diag(1, q),
                nu   = 3)
hyperpar <- list(alpha = 0.5, phi = 1)

res <- spBPS(data, priors, coords, hyperpar, subset_size = 200)

u <- 500
p <- 2
q <- 1

X_u <- matrix(rnorm(u * p), ncol = p)
```

```
crd_u <- matrix(runif(u * 2), ncol = 2)

# Predictive sampling
pred <- predict(res, newdata=list(X=X_u, coords=crd_u),
                draws=200L, pred_batch_size=500L)

# Summary statistics only (memory-efficient for large u)
pred_sum <- predict(res, newdata=list(X=X_u, coords=crd_u),
                    draws=500L, return_summary=TRUE,
                    probs=c(0.025, 0.975))
```

`pred_bayesMvLMconjugate` *Predictive sampler for Conjugate Bayesian Multivariate Linear Models*

Description

Predictive sampler for Conjugate Bayesian Multivariate Linear Models

Usage

```
pred_bayesMvLMconjugate(X_new, B_samples, Sigma_samples)
```

Arguments

`X_new`	matrix $n_new \times p$ of predictors for new data points
`B_samples`	array of posterior sample for β
`Sigma_samples`	array of posterior samples for Σ

Value

Y_pred matrix of posterior mean for response matrix Y predictions

Y_pred_samples array of posterior predictive sample for response matrix Y

Examples

```
## Generate data
n <- 100
p <- 3
q <- 2
Y <- matrix(rnorm(n*q), nrow = n, ncol = q)
X <- matrix(rnorm(n*p), nrow = n, ncol = p)

## Prior parameters
```

```
mu_B <- matrix(0, p, q)
V_B <- diag(10, p)
nu <- 3
Psi <- diag(q)

## Samples from posteriors
n_iter <- 1000
burn_in <- 500
set.seed(1234)
samples <- spBPS::bayesMvLMconjugate(Y, X, mu_B, V_B, nu, Psi, n_iter, burn_in)

## Extract posterior samples
B_samples <- samples$B_samples
Sigma_samples <- samples$Sigma_samples

## Samples from predictive posterior (based posterior samples)
m <- 50
X_new <- matrix(rnorm(m*p), nrow = m, ncol = p)
pred <- spBPS::pred_bayesMvLMconjugate(X_new, B_samples, Sigma_samples)
```

`r_pred_cond_MvT` *Draw from the conditional posterior predictive for a set of unobserved covariates*

Description

Draw from the conditional posterior predictive for a set of unobserved covariates

Usage

```
r_pred_cond_MvT(data, X_u, d_u, d_us, hyperpar, poster, post)
```

Arguments

`data`	list two elements: first named Y, second named X
`X_u`	matrix unobserved instances covariate matrix
`d_u`	matrix unobserved instances distance matrix
`d_us`	matrix cross-distance between unobserved and observed instances matrix
`hyperpar`	list two elemets: first named α, second named ϕ
`poster`	list output from `fit_cpp_MvT` function
`post`	list output from `post_draws_MvT` function

Value

list posterior predictive samples

`r_pred_joint_MvT` *Draw from the joint posterior predictive for a set of unobserved covariates*

Description

Draw from the joint posterior predictive for a set of unobserved covariates

Usage

```
r_pred_joint_MvT(data, X_u, d_u, d_us, hyperpar, poster, R)
```

Arguments

`data`	list two elements: first named Y, second named X
`X_u`	matrix unobserved instances covariate matrix
`d_u`	matrix unobserved instances distance matrix
`d_us`	matrix cross-distance between unobserved and observed instances matrix
`hyperpar`	list two elemets: first named α, second named ϕ
`poster`	list output from `fit_cpp` function
`R`	integer number of posterior predictive samples

Value

list posterior predictive samples

`r_pred_marg_MvT` *Draw from the joint posterior predictive for a set of unobserved covariates*

Description

Draw from the joint posterior predictive for a set of unobserved covariates

Usage

```
r_pred_marg_MvT(data, X_u, d_u, d_us, hyperpar, poster, R)
```

Arguments

`data`	list two elements: first named Y, second named X
`X_u`	matrix unobserved instances covariate matrix
`d_u`	matrix unobserved instances distance matrix
`d_us`	matrix cross-distance between unobserved and observed instances matrix
`hyperpar`	list two elemets: first named α, second named ϕ
`poster`	list output from `fit_cpp` function
`R`	integer number of posterior predictive samples

Value

list posterior predictive samples

`sample_index` *Function to sample integers (index)*

Description

Function to sample integers (index)

Usage

```
sample_index(size, length, p)
```

Arguments

`size`	integer dimension of the set to sample
`length`	integer number of elements to sample
`p`	vector sampling probabilities

Value

vector sample of integers

spBPS *Unified spatial BPS workflow (multivariate path, works for q = 1)*

Description

Orchestrates subsetting, local stacking weight estimation, global stacking combination, and optional posterior or predictive simulation using Double Bayesian Predictive Stacking for latent spatial regression. Works for both multivariate outcomes and the univariate case via `q = 1`.

Usage

```
spBPS(
  data,
  priors,
  coords,
  hyperpar,
  subset_size = 500L,
  K = NULL,
  cv_folds = 5L,
  rp = 1,
  combine_method = c("bps", "pseudoBMA"),
  draws = 0L,
  newdata = NULL,
  include_latent = FALSE,
  n_cores = 1L,
  pred_batch_size = 200L
)
```

Arguments

`data`	List with matrices `Y` (response, n x q) and `X` (covariates, n x p).
`priors`	List of priors for the multivariate model: `mu_B` (p x q mean matrix), `V_r` (p x p covariance), `Psi` (q x q scale), `nu` (degrees of freedom).
`coords`	Matrix of observation coordinates (n x d).
`hyperpar`	List with elements `alpha` and `phi` (vectors allowed); defines the grid of models over which stacking weights are computed.
`subset_size`	Target subset size when `K` is not provided. Default 500.
`K`	Optional number of subsets. When `NULL`, computed as `ceiling(nrow(Y) / subset_size)` and lower-bounded at 1.
`cv_folds`	Number of folds for local cross-validation. Default 5.
`rp`	Fraction of rows used when recomputing global stacking weights (passed to `BPS_combine`). Ignored when `combine_method = "pseudoBMA"`.
`combine_method`	Global combination method: Bayesian Predictive Stacking (`"bps"`) or pseudo-BMA (`"pseudoBMA"`).

draws	Number of posterior/predictive draws to return (0 to skip sampling). When positive and newdata is supplied, joint posterior and predictive draws are returned. When positive and newdata is NULL, only posterior draws (beta, sigma) are returned.
newdata	Optional list with X (u x p) and coords (u x d) for prediction locations. Required when draws > 0 and predictions are desired. For large u, use pred_batch_size to control memory usage.
include_latent	Unused; kept for compatibility.
n_cores	Number of cores for parallel computation. Controls both the local weight estimation (parallel over K subsets) and predictive sampling. Default 1.
pred_batch_size	Batch size for streaming prediction. Controls the maximum number of prediction sites processed at once, hence the peak memory usage. Default 200. Rule of thumb: peak RAM (MB) ? batch_size x q x draws x 8 / 1e6. Set NULL for automatic selection (min(200, u)).

Value

Object of class "spBPS" ? a list with components:

subsets Partition information: Y_list, X_list, crd_list.

weights_global K-vector of global stacking weights.

weights_local K-list of local stacking weight vectors.

epd K-list of n_k x J log-density matrices.

priors, hyperpar Stored for use by predict.spBPS.

timings Named numeric vector: fitting, combination, sampling (seconds).

posterior (if draws > 0) List with three-dimensional arrays: beta (p x q x R), sigma (q x q x R), model (R).

predictive (if draws > 0 and newdata supplied) List with three-dimensional arrays: Wu, Yu, MY (each u x q x R).

See Also

predict.spBPS for generating predictions on new data after fitting.

Examples

```
n <- 1000
p <- 2
q <- 1

Y <- matrix(rnorm(n * q), ncol = q)
X <- matrix(rnorm(n * p), ncol = p)
coords <- matrix(runif(n * 2), ncol = 2)

data    <- list(Y = Y, X = X)
priors  <- list(mu_B = matrix(0, nrow = p, ncol = q),
```

```
                V_r  = diag(10, p),
                Psi  = diag(1, q),
                nu   = 3)
hyperpar <- list(alpha = 0.5, phi = 1)

res <- spBPS(data, priors, coords, hyperpar, subset_size = 200)
```

---

spBPS_old *Unified spatial BPS workflow (multivariate path, works for q = 1)*

Description

Orchestrates subsetting, local stacking weight estimation, global stacking combination, and optional posterior or predictive simulation using the multivariate Student-t spatial model. Works for both multivariate outcomes and the univariate case via `q = 1`.

Usage

```
spBPS_old(
  data,
  priors,
  coords,
  hyperpar,
  subset_size = 500L,
  K = NULL,
  cv_folds = 5L,
  rp = 1,
  combine_method = c("bps", "pseudoBMA"),
  draws = 0L,
  newdata = NULL,
  include_latent = FALSE,
  cores = NULL
)
```

Arguments

`data`	List with matrices `Y` (response) and `X` (covariates).
`priors`	List of priors for the multivariate model (`mu_B`, `V_r`, `Psi`, `nu`).
`coords`	Matrix of observation coordinates.
`hyperpar`	List with elements `alpha` and `phi` (vectors allowed).
`subset_size`	Target subset size when `K` is not provided. Default 500.
`K`	Optional number of subsets. When `NULL`, computed as `ceiling(nrow(Y) / subset_size)` and lower-bounded at 1.
`cv_folds`	Number of folds for local cross-validation (default 5).

rp	Fraction of rows used when recomputing global stacking weights (passed to BPS_combine). Ignored when combine_method = "pseudoBMA".
combine_method	Choose between Bayesian Predictive Stacking ("bps") or pseudo-BMA ("pseudoBMA") for combining subsets.
draws	Number of joint posterior/predictive draws to return (0 to skip). When positive, newdata must be supplied because draws are obtained via BPS_post_MvT which jointly samples posterior and predictive.
newdata	Optional list with X and coords for prediction locations; required when either draw count is positive.
include_latent	Logical; if TRUE, posterior draws include latent processes.
cores	Optional integer; when >1 a parallel backend is registered internally via doParallel::registerDoParal for the fit and draw loops. When NULL, the existing foreach backend (if any) is used.

Value

List with components subsets, weights_global, weights_local, epd, and optional posterior and predictive draws.

Examples

```
n <- 1000
p <- 2
q <- 1

Y <- matrix(rnorm(n*q), ncol = q)
X <- matrix(rnorm(n*p), ncol = p)
coords <- matrix(runif(n*2), ncol = 2)

data <- list(Y = Y, X = X)
priors <- list(mu_B = matrix(0, nrow = p, ncol = q),
                             V_r = diag(10, p),
                             Psi = diag(1, q),
                             nu = 3)
hyperpar <- list(alpha = 0.5, phi = 1)
subset_size <- 200

res <- spBPS(data, priors, coords, hyperpar, subset_size = subset_size)
```

---

subset_data *Function to subset data for meta-analysis*

Description

Function to subset data for meta-analysis

Usage

```
subset_data(data, K)
```

Arguments

data	list three elements: first named Y, second named X, third named crd
K	integer number of desired subsets

Value

list subsets of data, and the set of indexes

Index