Package 'spBFA'

October 30, 2019

Type Package

Title Spatial Bayesian Factor Analysis

Version 1.0

Date 2019-10-16

Description Implements a spatial Bayesian non-parametric factor analysis model with inference in a Bayesian setting using Markov chain Monte Carlo (MCMC). Spatial correlation is introduced in the columns of the factor loadings matrix using a Bayesian non-parametric prior, the probit stick-breaking process. Areal spatial data is modeled using a conditional autoregressive (CAR) prior and point-referenced spatial data is treated using a Gaussian process. The response variable can be modeled as Gaussian, probit, Tobit, or Binomial (using Polya-Gamma augmentation). Temporal correlation is introduced for the latent factors through a hierarchical structure and can be specified as exponential or first-order autoregressive.

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

NeedsCompilation yes

Depends R (>= 3.0.2)

Imports graphics, grDevices, msm (>= 1.0.0), mvtnorm (>= 1.0-0), pgdraw (>= 1.0), Rcpp (>= 0.12.9), stats, utils

Suggests coda, classInt, knitr, rmarkdown, womblR (>= 1.0.3)

LinkingTo Rcpp, RcppArmadillo (>= 0.7.500.0.0)

VignetteBuilder knitr

Language en-US

Author Samuel I. Berchuck [aut, cre]

Maintainer Samuel I. Berchuck <sib2@duke.edu>

Repository CRAN

Date/Publication 2019-10-30 17:00:05 UTC

R topics documented:

bfa_sp 2
diagnostics 6
is.spBFA 8
predict.spBFA 9
reg.bfa_sp 10
spBFA 10

Index **11**

`bfa_sp` *Spatial factor analysis using a Bayesian hierarchical model.*

Description

`bfa_sp` is a Markov chain Monte Carlo (MCMC) sampler for a Bayesian spatial factor analysis model. The spatial component is introduced using a Probit stick-breaking process prior on the factor loadings. The model is implemented using a Bayesian hierarchical framework.

Usage

```
bfa_sp(formula, data, dist, time, K, L = Inf, trials = NULL,
  family = "normal", temporal.structure = "exponential",
  spatial.structure = "discrete", starting = NULL, hypers = NULL,
  tuning = NULL, mcmc = NULL, seed = 54, gamma.shrinkage = TRUE,
  include.space = TRUE, clustering = TRUE)
```

Arguments

`formula`	A `formula` object, corresponding to the spatial factor analysis model. The response must be on the left of a `~` operator, and the terms on the right must indicate the covariates to be included in the fixed effects. If no covariates are desired a zero should be used, `~ 0`.
`data`	A required `data.frame` containing the variables in the model. The data frame must contain `M x O x Nu` rows. Here, `M` represents the number of spatial locations, `O` the number of different observation types and `Nu` the number of temporal visits. The observations must be first be ordered spatially, second by observation type and then temporally. This means that the first `M x O` observations come from the first time point and the first `M` observations come the first spatial observation type.
`dist`	A `M x M` dimensional distance matrix. For a `discrete` spatial process the matrix contains binary adjacencies that dictate the spatial neighborhood structure and for `continuous` spatial processes the matrix should be a continuous distance matrix (e.g., Euclidean).
`time`	A `Nu` dimensional vector containing the observed time points in increasing order.
`K`	A scalar that indicates the dimension (i.e., quantity) of latent factors.

`L` The number of latent clusters. If finite, a scalar indicating the number of clusters for each column of the factor loadings matrix. By default `L` is set at `Inf` so that the Probit stick-breaking process becomes an infinite mixture model.

`trials` A variable in `data` that contains the number of trials for each of the binomial observations. If there is no count data, `trials` should be left missing.

`family` Character string indicating the distribution of the observed data. Options include: `"normal"`, `"probit"`, `"tobit"`, and `"binomial"`. `family` must have either `0` or `1` dimension(s) (the one populates the rest). Any combination of likelihoods can be used.

`temporal.structure`
Character string indicating the temporal kernel. Options include: `"exponential"` and `"ar1"`.

`spatial.structure`
Character string indicating the type of spatial process. Options include: `"continuous"` (i.e., Gaussian process with exponential kernel) and `"discrete"` (i.e., proper CAR).

`starting` Either `NULL` or a `list` containing starting values to be specified for the MCMC sampler. If `NULL` is not chosen then none, some or all of the starting values may be specified.

When `NULL` is chosen then default starting values are automatically generated. Otherwise a `list` must be provided with names `Beta`, `Delta`, `Sigma2`, `Kappa`, `Rho`, `Upsilon` or `Psi` containing appropriate objects. `Beta` (or `Delta`) must either be a `P` (or `K`) dimensional vector or a scalar (the scalar populates the entire vector). `Sigma2` must be either a `M x (O -C)` matrix or a scalar. `Kappa` must be a `O x O` dimensional matrix, `Rho` a scalar, `Upsilon` a `K x K` matrix, and `Psi` a scalar.

`hypers` Either `NULL` or a `list` containing hyperparameter values to be specified for the MCMC sampler. If `NULL` is not chosen then none, some or all of the hyperparameter values may be specified.

When `NULL` is chosen then default hyperparameter values are automatically generated. These default hyperparameters are described in detail in (Berchuck et al.). Otherwise a `list` must be provided with names `Beta`, `Delta`, `Sigma2`, `Kappa`, `Rho`, `Upsilon` or `Psi` containing further hyperparameter information. These objects are themselves `lists` and may be constructed as follows.

`Beta` is a `list` with two objects, `MuBeta` and `SigmaBeta`. These values represent the prior mean and variance parameters for the multivariate normal prior.

`Delta` is a `list` with two objects, `A1` and `A2`. These values represent the prior shape parameters for the multiplicative Gamma shrinkage prior.

`Sigma2` is a `list` with two objects, `A` and `B`. These values represent the shape and scale for the variance parameters.

`Kappa` is a `list` with two objects, `SmallUpsilon` and `BigTheta`. `SmallUpsilon` represents the degrees of freedom parameter for the inverse-Wishart hyperprior and must be a real number scalar, while `BigTheta` represents the scale matrix and must be a `O x O` dimensional positive definite matrix.

`Rho` is a `list` with two objects, `ARho` and `BRho`. `ARho` represents the lower bound for the uniform hyperprior, while `BRho` represents the upper bound. The bounds must be specified carefully. This is only specified for continuous spatial processes.

Upsilon is a list with two objects, Zeta and Omega. Zeta represents the degrees of freedom parameter for the inverse-Wishart hyperprior and must be a real number scalar, while Omega represents the scale matrix and must be a K x K dimensional positive definite matrix.

Psi is a list with two objects, dependent on if the temporal kernel is exponential or ar1. For exponential, the two objects are APsi and BPsi. APsi represents the lower bound for the uniform hyperprior, while BPsi represents the upper bound. The bounds must be specified carefully. For ar1, the two objects are Beta and Gamma, which are the two shape parameters of a Beta distribution shifted to have domain in (-1, 1).

tuning — Either NULL or a list containing tuning values to be specified for the MCMC Metropolis steps. If NULL is not chosen then all of the tuning values must be specified.

When NULL is chosen then default tuning values are automatically generated to 1. Otherwise a list must be provided with names Psi, or Rho. Each of these entries must be scalars containing tuning variances for their corresponding Metropolis updates. Rho is only specified for continuous spatial processes.

mcmc — Either NULL or a list containing input values to be used for implementing the MCMC sampler. If NULL is not chosen then all of the MCMC input values must be specified.

NBurn: The number of sampler scans included in the burn-in phase. (default = 10,000)

NSims: The number of post-burn-in scans for which to perform the sampler. (default = 10,000)

NThin: Value such that during the post-burn-in phase, only every NThin-th scan is recorded for use in posterior inference (For return values we define, NKeep = NSims / NThin (default = 1).

NPilot: The number of times during the burn-in phase that pilot adaptation is performed (default = 20)

seed — An integer value used to set the seed for the random number generator (default = 54).

gamma.shrinkage — A logical indicating whether a gamma shrinkage process prior is used for the variances of the factor loadings columns. If FALSE, the hyperparameters (A1 and A2) indicate the shape and rate for a gamma prior on the precisions. Default is TRUE.

include.space — A logical indicating whether a spatial process should be included. Default is TRUE, however if FALSE the spatial correlation matrix is fixed as an identity matrix. This specification overrides the spatial.structure input.

clustering — A logical indicating whether the Bayesian non-parametric process should be used, default is TRUE. If FALSE is specified each column is instead modeled with an independent spatial process.

Details

Details of the underlying statistical model proposed by Berchuck et al. 2019. are forthcoming.

Value

bfa_sp returns a list containing the following objects

lambda NKeep x (M x O x K) matrix of posterior samples for factor loadings matrix lambda. The labels for each column are Lambda_O_M_K.

eta NKeep x (Nu x K) matrix of posterior samples for the latent factors eta. The labels for each column are Eta_Nu_K.

beta NKeep x P matrix of posterior samples for beta.

sigma2 NKeep x (M * (O -C)) matrix of posterior samples for the variances sigma2. The labels for each column are Sigma2_O_M.

kappa NKeep x ((O * (O + 1)) / 2) matrix of posterior samples for kappa. The columns have names that describe the samples within them. The row is listed first, e.g., Kappa3_2 refers to the entry in row 3, column 2.

delta NKeep x K matrix of posterior samples for delta.

tau NKeep x K matrix of posterior samples for tau.

upsilon NKeep x ((K * (K + 1)) / 2) matrix of posterior samples for Upsilon. The columns have names that describe the samples within them. The row is listed first, e.g., Upsilon3_2 refers to the entry in row 3, column 2.

psi NKeep x 1 matrix of posterior samples for psi.

xi NKeep x (M x O x K) matrix of posterior samples for factor loadings cluster labels xi. The labels for each column are Xi_O_M_K.

rho NKeep x 1 matrix of posterior samples for rho.

metropolis 2 (or 1) x 3 matrix of metropolis acceptance rates, updated tuners, and original tuners that result from the pilot adaptation.

runtime A character string giving the runtime of the MCMC sampler.

datobj A list of data objects that are used in future bfa_sp functions and should be ignored by the user.

dataug A list of data augmentation objects that are used in future bfa_sp functions and should be ignored by the user.

References

Reference for Berchuck et al. 2019 is forthcoming.

Examples

```
###Load womblR for example visual field data
library(womblR)

###Format data for MCMC sampler
blind_spot <- c(26, 35) # define blind spot
VFSeries <- VFSeries[order(VFSeries$Location), ] # sort by location
VFSeries <- VFSeries[order(VFSeries$Visit), ] # sort by visit
VFSeries <- VFSeries[!VFSeries$Location %in% blind_spot, ] # remove blind spot locations
```

```
dat <- data.frame(Y = VFSeries$DLS / 10) # create data frame with scaled data
Time <- unique(VFSeries$Time) / 365 # years since baseline visit
W <- HFAII_Queen[-blind_spot, -blind_spot] # visual field adjacency matrix (data object from womblR)
M <- dim(W)[1] # number of locations

###Prior bounds for psi
TimeDist <- as.matrix(dist(Time))
BPsi <- log(0.025) / -min(TimeDist[TimeDist > 0])
APsi <- log(0.975) / -max(TimeDist)

###MCMC options
K <- 10 # number of latent factors
O <- 1 # number of spatial observation types
Hypers <- list(Sigma2 = list(A = 0.001, B = 0.001),
               Kappa = list(SmallUpsilon = O + 1, BigTheta = diag(O)),
               Delta = list(A1 = 1, A2 = 20),
               Psi = list(APsi = APsi, BPsi = BPsi),
               Upsilon = list(Zeta = K + 1, Omega = diag(K)))
Starting <- list(Sigma2 = 1,
                 Kappa = diag(O),
                 Delta = 2 * (1:K),
                 Psi = (APsi + BPsi) / 2,
                 Upsilon = diag(K))
Tuning <- list(Psi = 1)
MCMC <- list(NBurn = 1000, NSims = 1000, NThin = 2, NPilot = 5)

###Fit MCMC Sampler
reg.bfa_sp <- bfa_sp(Y ~ 0, data = dat, dist = W, time = Time,  K = 10,
                     starting = Starting, hypers = Hypers, tuning = Tuning, mcmc = MCMC,
                     L = Inf,
                     family = "tobit",
                     trials = NULL,
                     temporal.structure = "exponential",
                     spatial.structure = "discrete",
                     seed = 54,
                     gamma.shrinkage = TRUE,
                     include.space = TRUE,
                     clustering = TRUE)

###Note that this code produces the pre-computed data object "reg.bfa_sp"
###More details can be found in the vignette.
```

`diagnostics` *diagnostics*

Description

Calculates diagnostic metrics using output from the spBFA model.

Usage

```
diagnostics(object, diags = c("dic", "dinf", "waic"),
  keepDeviance = FALSE, keepPPD = FALSE, Verbose = TRUE, seed = 54)
```

Arguments

object	A spBFA model object for which diagnostics are desired from.
diags	A vector of character strings indicating the diagnostics to compute. Options include: Deviance Information Criterion ("dic"), d-infinity ("dinf") and Watanabe-Akaike information criterion ("waic"). At least one option must be included. Note: The probit model cannot compute the DIC or WAIC diagnostics due to computational issues with computing the multivariate normal CDF.
keepDeviance	A logical indicating whether the posterior deviance distribution is returned (default = FALSE).
keepPPD	A logical indicating whether the posterior predictive distribution at each observed location is returned (default = FALSE).
Verbose	A boolean logical indicating whether progress should be output (default = TRUE).
seed	An integer value used to set the seed for the random number generator (default = 54).

Details

To assess model fit, DIC, d-infinity and WAIC are used. DIC is based on the deviance statistic and penalizes for the complexity of a model with an effective number of parameters estimate pD (Spiegelhalter et al 2002). The d-infinity posterior predictive measure is an alternative diagnostic tool to DIC, where d-infinity=P+G. The G term decreases as goodness of fit increases, and P, the penalty term, inflates as the model becomes over-fit, so small values of both of these terms and, thus, small values of d-infinity are desirable (Gelfand and Ghosh 1998). WAIC is invariant to parametrization and is asymptotically equal to Bayesian cross-validation (Watanabe 2010). WAIC = -2 * (lppd - p_waic_2). Where lppd is the log pointwise predictive density and p_waic_2 is the estimated effective number of parameters based on the variance estimator from Vehtari et al. 2016. (p_waic_1 is the mean estimator).

Value

diagnostics returns a list containing the diagnostics requested and possibly the deviance and/or posterior predictive distribution objects.

Author(s)

Samuel I. Berchuck

References

Gelfand, A. E., & Ghosh, S. K. (1998). Model choice: a minimum posterior predictive loss approach. Biometrika, 1-11.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., & Van Der Linde, A. (2002). Bayesian measures of model complexity and fit. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 64(4), 583-639.

Vehtari, A., Gelman, A., & Gabry, J. (2016). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. Statistics and Computing, 1-20.

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. Journal of Machine Learning Research, 11(Dec), 3571-3594.

Examples

```
###Load pre-computed regression results
data(reg.bfa_sp)

###Compute and print diagnostics
diags <- diagnostics(reg.bfa_sp)
print(unlist(diags))
```

`is.spBFA` *is.spBFA*

Description

`is.spBFA` is a general test of an object being interpretable as a `spBFA` object.

Usage

```
is.spBFA(x)
```

Arguments

`x`	object to be tested.

Details

The `spBFA` class is defined as the regression object that results from the `spBFA` regression function.

Examples

```
###Load pre-computed results
data(reg.bfa_sp)

###Test function
is.spBFA(reg.bfa_sp)
```

predict.spBFA *predict.spBFA*

Description

Predicts future observations from the spBFA model.

Usage

```
## S3 method for class 'spBFA'
predict(object, NewTimes, NewX = NULL,
  NewTrials = NULL, Verbose = TRUE, type = "temporal", ...)
```

Arguments

object	A spBFA model object for which predictions are desired from.
NewTimes	A numeric vector including desired time(s) points for prediction.
NewX	A matrix including covariates at times NewTimes for prediction. NewX must have dimension (M x O x NNewVistis) x P. Where NNewVisits is the number of temporal locations being predicted. The default sets NewX to NULL, which assumes that the covariates for all predictions are the same as the final time point.
NewTrials	An array indicating the trials for categorical predictions. The array must have dimension M x C x NNewVisits and contain only non-negative integers. The default sets NewTrials to NULL, which assumes the trials for all predictions are the same as the final time point.
Verbose	A boolean logical indicating whether progress should be output.
type	A character string indicating the type of prediction, choices include "temporal" and "spatial". Spatial prediction has not been implemented yet.
...	other arguments.

Details

predict.spBFA uses Bayesian krigging to predict vectors at future time points. The function returns the krigged factors (Eta) and also the observed outcomes (Y).

Value

predict.spBFA returns a list containing the following objects.

Eta A list containing NNewVistis matrices, one for each new time prediction. Each matrix is dimension NKeep x K, where K is the number of latent factors Each matrix contains posterior samples obtained by Bayesian krigging.

Y A list containing NNewVistis posterior predictive distribution matrices. Each matrix is dimension NKeep x (M * O), where M is the number of spatial locations and O the number of observation types. Each matrix is obtained through Bayesian krigging.

Author(s)

Samuel I. Berchuck

Examples

```
###Load pre-computed regression results
data(reg.bfa_sp)

###Compute predictions
pred <- predict(reg.bfa_sp, NewTimes = 3)
pred.observations <- pred$Y$Y10 # observed data predictions
pred.krig <- pred$Eta$Eta10 # krigged parameters
```

---

reg.bfa_sp *Pre-computed regression results from* bfa_sp

Description

The data object reg.bfa_sp is a pre-computed spBFA data object for use in the package vignette and function examples.

Usage

```
data(reg.bfa_sp)
```

Format

The data object reg.bfa_sp is a spBFA data object that is the result of implementing the MCMC code in the vignette. It is presented here because the run-time would be unecessarily long when compiling the R package.

---

spBFA *spBFA*

Description

spBFA is a package for Bayesian spatial factor analysis. A corresponding manuscript is forthcoming.

Author(s)

Samuel I. Berchuck <sib2@duke.edu>

Index

∗Topic **datasets**
reg.bfa_sp, 10

bfa_sp, 2

diagnostics, 6

is.spBFA, 8

predict.spBFA, 9

reg.bfa_sp, 10

spBFA, *6–9*, 10
spBFA-package *(*spBFA*)*, 10