Package: BayesFM (via r-universe)

October 29, 2026

Title Bayesian Inference for Factor Modeling

Type Package

Version 0.1.7

Description Collection of procedures to perform Bayesian analysis on a variety of factor models. Currently, it includes: ``Bayesian Exploratory Factor Analysis'' (befa) from G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014) <doi:10.1016/j.jeconom.2014.06.008>, an approach to dedicated factor analysis with stochastic search on the structure of the factor loading matrix. The number of latent factors, as well as the allocation of the manifest variables to the factors, are not fixed a priori but determined during MCMC sampling.

Depends R (>= 3.0.0)

Imports checkmate (>= 1.8.0), coda, ggplot2 (>= 2.1.0), gridExtra, plyr (>= 1.8.0)

SystemRequirements gfortran (>= 4.6.3)

License GPL-3

NeedsCompilation yes

Encoding UTF-8

RoxygenNote 7.3.1

Author Rémi Piatek [aut, cre] (<https://orcid.org/0000-0002-3474-1304>)

Maintainer Rémi Piatek <remi.piatek@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2024-06-14 02:42:30 UTC

RemoteUrl https://github.com/cran/BayesFM

RemoteRef HEAD

RemoteSha 296a80a1a76714a99fee9316ee7669ead836ca26

Contents

BayesFM 2
befa 3
plot.befa 9
post.column.switch 11
post.sign.switch 12
simul.dedic.facmod 13
simul.nfac.prior 16
simul.R.prior 17
summary.befa 18

Index **22**

`BayesFM` *BayesFM: Package for Bayesian Factor Modeling*

Description

The long-term goal of this package is to provide a collection of procedures to perform Bayesian inference on a variety of factor models.

Details

Currently, this package includes: Bayesian Exploratory Factor Analysis (`befa`), as developed in Conti et al. (2014), an approach to dedicated factor analysis with stochastic search on the structure of the factor loading matrix. The number of latent factors, as well as the allocation of the observed variables to the factors, are not fixed a priori but determined during MCMC sampling. More approaches will be included in future releases of this package.

Note

You are very welcome to send me any comments or suggestions for improvements, and to share with me any problems you may encounter with the use of this package.

Author(s)

Rémi Piatek `<remi.piatek@gmail.com>`

References

G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014): "Bayesian Exploratory Factor Analysis", *Journal of Econometrics*, 183(1), pages 31-57, doi:10.1016/j.jeconom.2014.06.008.

befa *Bayesian Exploratory Factor Analysis*

Description

This function implements the Bayesian Exploratory Factor Analysis (befa) approach developed in Conti et al. (CFSHP, 2014). It runs a MCMC sampler for a factor model with dedicated factors, where each manifest variable is allowed to load on at most one latent factor. The allocation of the manifest variables to the latent factors is not fixed *a priori* but determined stochastically during sampling. The minimum number of variables dedicated to each factor can be controlled by the user to achieve the desired level of identification. The manifest variables can be continuous or dichotomous, and control variables can be introduced as covariates.

Usage

```
befa(model, data, burnin = 1000, iter = 10000, Nid = 3, Kmax, A0 = 10,
  B0 = 10, c0 = 2, C0 = 1, HW.prior = TRUE, nu0 = Kmax + 1, S0 = 1,
  kappa0 = 2, xi0 = 1, kappa = 1/Kmax, indp.tau0 = TRUE,
  rnd.step = TRUE, n.step = 5, search.delay = min(burnin, 10),
  R.delay = min(burnin, 100), dedic.start, alpha.start, sigma.start,
  beta.start, R.start, verbose = TRUE)
```

Arguments

model	This argument specifies the manifest variables and the covariates used in the model (if any). It can be specified in two different ways: • A numeric matrix or a data frame containing the manifest variables. This corresponds to a model without covariates, and the argument data is not required. • A list of model formulas. Each element of the list specifies a manifest variable and its corresponding control variables (e.g., 'Y1 ~ X1 + X2' to use X1 and X2 as control variables for Y1). If a formula has no left-hand side variable, covariates on the right-hand side are included in all equations (e.g., '~ X3' means that X3 is used as a control variable for all the manifest variables). Argument data can be passed to the function in that case, otherwise parent data frame is used. Binary manifest variables should be specified as logical vectors in the data frame to be treated as dichotomous. NA values are accepted in manifest variables only.
data	Data frame. If missing, parent data frame if used.
burnin	Burn-in period of the MCMC sampler.
iter	Number of MCMC iterations saved for posterior inference (after burn-in).
Nid	Minimum number of manifest variables dedicated to each latent factor for identification.

`Kmax` Maximum number of latent factors. If missing, the maximum number of factors that satisfies the identification condition determined by `Nid` and the Ledermann bound is specified (see CFSHP, section 2.2).

`A0` Scaling parameters of the variance of the Normal prior on the nonzero factor loadings. Either a scalar or a numeric vector of length equal to the number of manifest variables.

`B0` Variances of the Normal prior on the regression coefficients. Either a scalar or a numeric vector of length equal to the number of manifest variables.

`c0` Shape parameters of the Inverse-Gamma prior on the idiosyncratic variances. Either a scalar or a numeric vector of length equal to the number of manifest variables.

`C0` Scale parameters of the Inverse-Gamma prior on the idiosyncratic variances. Either a scalar or a numeric vector of length equal to the number of manifest variables.

`HW.prior` If `TRUE`, implement Huang-Wand (2013) prior on the covariance matrix of the factors in the expanded model, otherwise use an Inverse-Wishart prior if `FALSE`, see CFSHP section 2.3.5.

`nu0` Degrees of freedom of the Inverse-Wishart prior on the covariance matrix of the latent factors in the expanded model.

`S0` Scale parameters of the Inverse-Wishart prior on the covariance matrix of latent factors in the expanded model:

- if `HW.prior` = `TRUE`, scale parameter of the Gamma hyperprior distribution on the individual scales of the Inverse-Wishart prior.
- if `HW.prior` = `FALSE`, diagonal elements of the scale matrix of the Inverse-Wishart prior on the covariance matrix of the latent factors in the expanded model.

Either a scalar or a numeric vector of length equal to `Kmax`.

`kappa0` First shape parameter of the Beta prior distribution on the probability τ_0 that a manifest variable does not load on any factor.

`xi0` Second shape parameter of the Beta prior distribution on the probability τ_0 that a manifest variable does not load on any factor.

`kappa` Concentration parameters of the Dirichlet prior distribution on the indicators. Either a scalar or a numeric vector of length equal to `Kmax`.

`indp.tau0` If `TRUE`, specify the alternative prior specification with independent parameters τ_{0m} across manifest variables $m = 1, ..., M$, otherwise use a common parameter τ_0 if `FALSE`.

`rnd.step` If `TRUE`, select randomly the number of intermediate steps in non-identified models at each MCMC iteration, otherwise use a fixed number of steps if `FALSE`.

`n.step` Controls the number of intermediate steps in non-identified models:

- if `rnd.step` = `TRUE`, average number of steps. The number of steps is sampled at each MCMC iteration from 1+Poisson(`n.step`-1).
- if `rnd.step` = `FALSE`, fixed number of steps.

`search.delay` Number of MCMC iterations run with fixed indicator matrix (specified in `dedic.start`) at beginning of MCMC sampling.

R.delay	Number of MCMC iterations run with fixed correlation matrix (specified in dedic.start) at beginning of MCMC sampling.
dedic.start	Starting values for the indicators. Vector of integers of length equal to the number of manifest variables. Each element takes a value among 0, 1, ..., Kmax. If missing, random allocation of the manifest variables to the maximum number of factors Kmax, with a minimum of Nid manifest variables loading on each factor.
alpha.start	Starting values for the factor loadings. Numeric vector of length equal to the number of manifest variables. If missing, sampled from a Normal distribution with zero mean and variance A0.
sigma.start	Starting values for the idiosyncratic variances. Numeric vector of length equal to the number of manifest variables. Sampled from prior if missing.
beta.start	Starting values for the regression coefficients. Numeric vector of length equal to the total number of regression coefficients, concatenated for all the manifest variables. Sampled from prior if missing.
R.start	Starting values for the correlation matrix of the latent factors. Numeric matrix with Kmax rows and columns, and unit diagonal elements. If missing, identity matrix is used.
verbose	If TRUE, display information on the progress of the function.

Details

Model specification. The model is specified as follows, for each observation $i = 1, ..., N$:

$$Y_i^* = \beta X_i + \alpha\theta_i + \epsilon_i$$

$$\theta_i \sim \mathcal{N}(0, R)$$

$$\epsilon_i \sim \mathcal{N}(0, \Sigma)$$

$$\Sigma = diag(\sigma_1^2, ..., \sigma_M^2)$$

where Y_i^* is the M-vector containing the latent variables underlying the corresponding M manifest variables Y_i, which can be continuous such that $Y_{im} = Y_{im}^*$, or binary with $Y_{im} = 1[Y_{im}^* > 0]$, for $m = 1, ..., M$. The K-vector θ_i contains the latent factors, and α is the $(M \times K)$-matrix of factor loadings. The M-vector ϵ_i is the vector of error terms. Covariates can be included in the Q-vector X_i and are related to the manifest variables through the $(M \times Q)$-matrix of regression coefficients β. Intercept terms are automatically included, but can be omitted in some or all equations using the usual syntax for R formulae (e.g., 'Y1 ~ X1 - 1' specifies that that Y1 is regressed on X1 and no intercept is included in the corresponding equation).

The number of latent factors K is specified as Kmax. However, during MCMC sampling the stochastic search process on the matrix α may produce zero columns, thereby reducing the number of active factors.

The covariance matrix R of the latent factors is assumed to be a correlation matrix for identification.

Each row of the factor loading matrix α contains at most one nonzero element (dedicated factor model). The allocation of the manifest variables to the latent factors is indicated by the binary matrix Δ with same dimensions as α, such that each row Δ_m indicates which factor loading is different from zero, e.g.:

$$\Delta_m = (0, .., 0, 1, 0, ..., 0) \equiv e_k$$

indicates that variable m loads on the kth factor, where e_k is a K-vector that contains only zeros, besides its kth element that equals 1.

Identification. The function verifies that the maximum number of latent factors `Kmax` does not exceed the Ledermann bound. It also checks that `Kmax` is consistent with the identification restriction specified with `Nid` (enough variables should be available to load on the factors when `Kmax` is reached). The default value for `Kmax` is the minimum between the Ledermann bound and the maximum number of factors that can be loaded by `Nid` variables. The user is free to select the level of identification, see CFSHP section 2.2 (non-identified models are allowed with `Nid = 1`).

Note that identification is achieved only with respect to the scale of the latent factors. Non-identifiability problems may affect the posterior sample because of column switching and sign switching of the factor loadings. These issues can be addressed *a posteriori* with the functions `post.column.switch` and `post.sign.switch`.

Prior specification. The indicators are assumed to have the following probabilities, for $k = 1, ..., K$:

$$Prob(\Delta_m = e_k \mid \tau_k) = \tau_k$$
$$\tau = (\tau_0, \tau_1, ..., \tau_K)$$

If `indp.tau0 = FALSE`, the probabilities are specified as:

$$\tau = [\tau_0, (1-\tau_0)\tau_1^*, ..., (1-\tau_0)\tau_K^*]$$
$$\tau_0 \sim \mathcal{B}eta(\kappa_0, \xi_0)$$
$$\tau^* = (\tau_1^*, ..., \tau_K^*) \sim \mathcal{D}ir(\kappa)$$

with κ_0 = `kappa0`, ξ_0 = `xi0` and κ = `kappa`. Alternatively, if `indp.tau0 = TRUE`, the probabilities are specified as:

$$\tau_m = [\tau_{0m}, (1-\tau_{0m})\tau_1^*, ..., (1-\tau_{0m})\tau_K^*]$$
$$\tau_{0m} \sim \mathcal{B}eta(\kappa_0, \xi_0)$$

for each manifest variable $m = 1, ..., M$.

A normal-inverse-Gamma prior distribution is assumed on the nonzero factor loadings and on the idiosyncratic variances:

$$\sigma_m^2 \sim \mathcal{I}nv - \mathcal{G}amma(c_{0m}, C_{0m})$$
$$\alpha_m^{\Delta} \mid \sigma_m^2 \sim \mathcal{N}(0, A_{0m}\sigma_m^2)$$

where α_m^{Δ} denotes the only nonzero loading in the mth row of α.

For the regression coefficients, a multivariate Normal prior distribution is assumed on each row $m = 1, ..., M$ of β:

$$\beta_m \sim \mathcal{N}(0, B_0 I_Q)$$

The covariates can be different across manifest variables, implying zero restrictions on the matrix β. To specify covariates, use a list of formulas as `model` (see example below). Intercept terms can be introduced using

To sample the correlation matrix R of the latent factors, marginal data augmentation is implemented (van Dyk and Meng, 2001), see CFSHP section 2.2. Using the transformation $\Omega = \Lambda^{1/2} R \Lambda^{1/2}$, the parameters $\Lambda = diag(\lambda_1, ..., \lambda_K)$ are used as *working parameters*. These parameters correspond to the variances of the latent factors in an expanded version of the model where the factors do not have unit variances. Two prior distributions can be specified on the covariance matrix Ω in the expanded model:

- If HW.prior = FALSE, inverse-Wishart distribution:

$$\Omega \sim \mathcal{I}nv - \mathcal{W}ishart(\nu_0, diag(S_0))$$

with ν_0 = nu0 and S_0 = S0.

- If HW.prior = TRUE, Huang-Wand (2013) prior:

$$\Omega \sim \mathcal{I}nv - \mathcal{W}ishart(\nu_0, W), \qquad W = diag(w_1, ..., w_K)$$

$$w_k \sim \mathcal{G}amma\left(\frac{1}{2}, \frac{1}{2\nu^* S_{0k}}\right)$$

with ν^* = nu0 - Kmax + 1, and the shape and rate parameters are specified such that the mean of the gamma distribution is equal to $\nu^* S_{0k}$, for each $k = 1, ..., K$.

Missing values. Missing values (NA) are allowed in the manifest variables. They are drawn from their corresponding conditional distributions during MCMC sampling. Control variables with missing values can be passed to the function. However, all the observations with at least one missing value in the covariates are discarded from the sample (a warning message is issued in that case).

Value

The function returns an object of class 'befa' containing the MCMC draws of the model parameters saved in the following matrices (each matrix has 'iter' rows):

- alpha: Factor loadings.
- sigma: Idiosyncratic variances.
- R: Correlation matrix of the latent factors (off-diagonal elements only).
- beta: regression coefficients (if any).
- dedic: indicators (integers indicating on which factors the manifest variable load).

The returned object also contains:

- nfac: Vector of number of 'active' factors across MCMC iterations (i.e., factors loaded by at least Nid manifest variables).
- MHacc: Logical vector indicating accepted proposals of Metropolis-Hastings algorithm.

The parameters Kmax and Nid are saved as object attributes, as well as the function call and the number of mcmc iterations (burnin and iter), and two logical variables indicating if the returned object has been post processed to address the column switching problem (post.column.switch) and the sign switching problem (post.sign.switch).

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

References

G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014): "Bayesian Exploratory Factor Analysis", *Journal of Econometrics*, 183(1), pages 31-57, doi:10.1016/j.jeconom.2014.06.008.

A. Huang, M.P. Wand (2013): "Simple Marginally Noninformative Prior Distributions for Covariance Matrices", *Bayesian Analysis*, 8(2), pages 439-452, doi:10.1214/13BA815.

D.A. van Dyk, X.-L. Meng (2001): "The Art of Data Augmentation", *Journal of Computational and Graphical Statistics*, 10(1), pages 1-50, doi:10.1198/10618600152418584.

See Also

`post.column.switch` and `post.sign.switch` for column switching and sign switching of the factor loading matrix and of the correlation matrix of the latent factors to restore identification *a posteriori*.

`summary.befa` and `plot.befa` to summarize and plot the posterior results.

`simul.R.prior` and `simul.nfac.prior` to simulate the prior distribution of the correlation matrix of the factors and the prior distribution of the indicator matrix, respectively. This is useful to perform prior sensitivity analysis and to understand the role of the corresponding parameters in the factor search.

Examples

```
#### model without covariates

set.seed(6)

# generate fake data with 15 manifest variables and 3 factors
N <- 100    # number of observations
Y <- simul.dedic.facmod(N, dedic = rep(1:3, each = 5))

# run MCMC sampler
# notice: 1000 MCMC iterations for illustration purposes only,
#  increase this number to obtain reliable posterior results!
mcmc <- befa(Y, Kmax = 5, iter = 1000)

# post process MCMC draws to restore identification
mcmc <- post.column.switch(mcmc)
mcmc <- post.sign.switch(mcmc)

summary(mcmc)  # summarize posterior results
plot(mcmc)     # plot posterior results

# summarize highest posterior probability (HPP) model
summary(mcmc, what = 'hppm')

#### model with covariates

# generate covariates and regression coefficients
Xcov <- cbind(1, matrix(rnorm(4*N), ncol = 4))
colnames(Xcov) <- c('(Intercept)', paste0('X', 1:4))
```

```
beta <- rbind(rnorm(15), rnorm(15), diag(3) %x% t(rnorm(5)))

# add covariates to previous model
Y <- Y + Xcov %*% beta

# specify model
model <- c('~ X1',                          # X1 covariate in all equations
           paste0('Y', 1:5,   ' ~ X2'),   # X2 covariate for Y1-Y5 only
           paste0('Y', 6:10,  ' ~ X3'),   # X3 covariate for Y6-Y10 only
           paste0('Y', 11:15, ' ~ X4'))   # X4 covariate for Y11-Y15 only
model <- lapply(model, as.formula)        # make list of formulas

# run MCMC sampler, post process and summarize
mcmc <- befa(model, data = data.frame(Y, Xcov), Kmax = 5, iter = 1000)
mcmc <- post.column.switch(mcmc)
mcmc <- post.sign.switch(mcmc)
mcmc.sum <- summary(mcmc)
mcmc.sum

# compare posterior mean of regression coefficients to true values
beta.comp <- cbind(beta[beta != 0], mcmc.sum$beta[, 'mean'])
colnames(beta.comp) <- c('true', 'mcmc')
print(beta.comp, digits = 3)
```

`plot.befa` *Plot object of class 'befa'*

Description

This function makes different plots that are useful to assess the posterior results: a trace plot of the number of latent factors (also showing Metropolis-Hastings acceptance across MCMC replications), a histogram of the posterior probabilities of the number of factors, heatmaps for the inficator probabilities, the factor loading matrix, and the correlation matrix of the latent factors.

Usage

```
## S3 method for class 'befa'
plot(x, ...)
```

Arguments

`x`	Object of class 'befa'.
`...`	The following extra arguments can be specified:

- `what`: How to summarize the posterior distribution?
  - `what = 'maxp'` (default): Only factor loadings with highest posterior probability of being different from zero or discarded from the model (if `dedic = 0`) are summarized.

  - what = 'all': All factor loadings with corresponding posterior probability to be allocated to a given factor (or to be discarded from the model) larger than min.prob are summarized.
  - what = 'hppm': Highest posterior probability models with probability larger than min.prob are summarized.
- byfac: Sort factor loadings by factors if TRUE, otherwise by manifest variables if FALSE (default).
- hpd.prob: Probability used to compute the highest posterior density intervals of the posterior distribution of the model parameters (default: 0.95).
- min.prob: If what = 'all', only factor loadings with posterior probability of being dedicated to a given factor (or discarded from the model) larger than this value are displayed. If what = 'hppm', only highest posterior probability models with probability larger than this value are displayed. (default: 0.20)

Details

This function makes graphs based on the summary results returned by summary.befa. It therefore accepts the same optional arguments as this function.

Value

No return value, called for side effects (plots the posterior results returned by summary.befa).

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

See Also

summary.befa to summarize posterior results.

Examples

```
set.seed(6)

# generate fake data with 15 manifest variables and 3 factors
Y <- simul.dedic.facmod(N = 100, dedic = rep(1:3, each = 5))

# run MCMC sampler and post process output
# notice: 1000 MCMC iterations for illustration purposes only,
#  increase this number to obtain reliable posterior results!
mcmc <- befa(Y, Kmax = 5, iter = 1000)
mcmc <- post.column.switch(mcmc)
mcmc <- post.sign.switch(mcmc)

# plot results for highest posterior probability model
plot(mcmc, what = 'hppm')
```

`post.column.switch` *Perform column switchting on posterior MCMC sample*

Description

This function reorders the columns of the factor loading matrix for each MCMC draw, as well as the rows and columns of the correlation matrix of the factors, to restore the identification of the model *a posteriori* with respect to the column switching problem.

Usage

```
post.column.switch(mcmc)
```

Arguments

`mcmc`	Object of class '`befa`'.

Details

The reordering of the columns of the factor loading matrix is based on the top elements of the columns (i.e., the first row containing a nonzero factor loading in each nonzero column of α, starting from the top of the matrix). At each MCMC iteration, the nonzero columns of α are reordered such that the top elements appear in increasing order. The rows and columns of the correlation matrix R of the factors are switched accordingly. See section 4.3 of CFSHP (p.42) for more details.

Value

Same '`befa`' object as the one passed to the function, where the indicators in the matrix `dedic`, as well as the rows and columns of the correlation matrix of the factors saved in `draws`, have been switched appropriately to restore the identification of the factor model with respect to column switching.

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

References

G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014): "Bayesian Exploratory Factor Analysis", *Journal of Econometrics*, 183(1), pages 31-57, doi:10.1016/j.jeconom.2014.06.008.

See Also

`post.sign.switch` to restore identification a posteriori with respect to the sign switching problem.

Examples

```
set.seed(6)
Y <- simul.dedic.facmod(N = 100, dedic = rep(1:3, each = 5))
mcmc <- befa(Y, Kmax = 5, iter = 1000)
mcmc <- post.column.switch(mcmc)
```

`post.sign.switch` *Perform sign switchting on posterior MCMC sample*

Description

This function performs a sign switch on the MCMC draws to restore the consistency of the signs of the factors loadings and of the correlations of the latent factors *a posteriori*.

Usage

```
post.sign.switch(mcmc, benchmark = NULL, benchmark.threshold = 0.5)
```

Arguments

`mcmc`	Object of class 'befa'.
`benchmark`	Vector of integers of length equal to the maximum number of latent factors. Each element indicates which factor loading is used as a benchmark for the sign switch. If `NULL`, the factor loadings with the highest posterior probabilities of being different from zero in each column of the factor loading matrix are used as benchmarks.
`benchmark.threshold`	Minimum posterior probability for a factor loading to be considered as a benchmark.

Details

The signs of the factor loadings, as well as of the corresponding correlations of the latent factors, are switched for each MCMC iteration such that the factor loadings defined as `benchmarks` are positive. The sign switch can only be performed if `post.column.switch` has been run before. See section 4.3 (p.42) of CFSHP for more details.

If a latent factor has no benchmarks, or if its benchmark is equal to zero at some MCMC iteration, then no sign switch is performed on the corresponding loadings and correlations for this particular factor or MCMC iteration.

Note that in complicated models where the sampler visits several models with different numbers of latent factors, it may not be relevant to use the default value of `benchmark`, as the posterior probabilities that the factor loadings are different from zero would be computed across models. Instead, the user might consider finding the highest posterior probability model first, and use its top elements in each column of the factor loading matrix as benchmarks to perform the sign switch.

Value

This function returns the same 'befa' object, where the signs of the factor loadings and of the factor correlations have been switched appropriately to restore the identification of the factor model with respect to sign switching.

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

References

G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014): "Bayesian Exploratory Factor Analysis", *Journal of Econometrics*, 183(1), pages 31-57, doi:10.1016/j.jeconom.2014.06.008.

See Also

post.column.switch for column switching of the factor loading matrix and of the correlation matrix of the latent factors to restore identification *a posteriori*.

Examples

```
set.seed(6)
Y <- simul.dedic.facmod(N = 100, dedic = rep(1:3, each = 5))
mcmc <- befa(Y, Kmax = 5, iter = 1000)
mcmc <- post.column.switch(mcmc)

# factor loadings corresponding to variables 1, 6, 11, 12 and 13 are
# used as benchmarks:
mcmc1 <- post.sign.switch(mcmc, benchmark = c(1, 6, 11, 12, 13))

# factor loadings with the highest posterior probability of being different
# from zero in each column are used as benchmark:
mcmc2 <- post.sign.switch(mcmc)
```

---

simul.dedic.facmod *Generate synthetic data from a dedicated factor model*

Description

This function simulates data from a dedicated factor model. The parameters of the model are either passed by the user or simulated by the function.

Usage

```
simul.dedic.facmod(N, dedic, alpha, sigma, R, R.corr = TRUE,
  max.corr = 0.85, R.max.trial = 1000)
```

Arguments

N	Number of observations in data set.
dedic	Vector of indicators. The number of manifest variables is equal to the length of this vector, and the number of factors is equal to the number of unique nonzero elements. Each integer element indicates on which latent factor the corresponding variable loads uniquely.
alpha	Vector of factor loadings, should be of same length as dedic. If missing, values are simulated (see details below).
sigma	Idiosyncratic variances, should be of same length as dedic. If missing, values are simulated (see details below).
R	Covariance matrix of the latent factors. If missing, values are simulated (see details below).
R.corr	If TRUE, covariance matrix R is rescaled to be a correlation matrix.
max.corr	Maximum correlation allowed between the latent factors.
R.max.trial	Maximum number of trials allowed to sample from the truncated distribution of the covariance matrix of the latent factors (accept/reject sampling scheme, to make sure max.corr is not exceeded).

Details

The function simulates data from the following dedicated factor model, for $i = 1, ..., N$:

$$Y_i = \alpha\theta_i + \epsilon_i$$

$$\theta_i \sim \mathcal{N}(0, R)$$

$$\epsilon_i \sim \mathcal{N}(0, \Sigma)$$

where the K-vector θ_i contains the latent factors, and α is the $(M \times K)$-matrix of factor loadings. Each row m of α contains only zeros, besides its element indicated by the mth element of dedic that is equal to the mth element of alpha (denoted α_m^{Δ} below). The M-vector ϵ_i is the vector of error terms, with $\Sigma = diag(\texttt{sigma})$. M is equal to the length of the vector dedic, and K is equal to the maximum value of this vector.

Only N and dedic are required, all the other parameters can be missing, completely or partially. Missing values (NA) are independently sampled from the following distributions, for each manifest variable $m = 1, ..., M$:

Factor loadings:

$$\alpha_m^{\Delta} = (-1)^{\phi_m}\sqrt{a_m}$$

$$\phi_m \sim \mathcal{B}er(0.5)$$

$$a_m \sim \mathcal{U}nif(0.04, 0.64)$$

Idiosyncratic variances:

$$\sigma_m^2 \sim \mathcal{U}nif(0.2, 0.8)$$

For the variables that do not load on any factors (i.e., for which the corresponding elements of dedic are equal to 0), it is specified that $\alpha_m^{\Delta} = 0$ and $\sigma_m^2 = 1$.

Covariance matrix of the latent factors:

$$\Omega \sim \mathcal{I}nv - \mathcal{W}ishart(K + 5, I_K)$$

which is rescaled to be a correlation matrix if `R.corr = TRUE`:

$$R = \Lambda^{-1/2}\Omega\Lambda^{-1/2}$$

$$\Lambda = diag(\Omega)$$

Note that the distribution of the covariance matrix is truncated such that all the off-diagonal elements of the implied correlation matrix R are below `max.corr` in absolute value. The truncation is also applied if the covariance matrix is used instead of the correlation matrix (i.e., if `R.corr = FALSE`).

The distributions and the corresponding default values used to simulate the model parameters are specified as in the Monte Carlo study of CFSHP, see section 4.1 (p.43).

Value

The function returns a data frame with `N` observations simulated from the corresponding dedicated factor model. The parameters used to generate the data are saved as attributes: `dedic`, `alpha`, `sigma` and `R`.

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

References

G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014): "Bayesian Exploratory Factor Analysis", *Journal of Econometrics*, 183(1), pages 31-57, doi:10.1016/j.jeconom.2014.06.008.

Examples

```
# generate 1000 observations from model with 4 factors and 20 variables
# (5 variables loading on each factor)
dat <- simul.dedic.facmod(N = 1000, dedic = rep(1:4, each = 5))

# generate data set with 5000 observations from the following model:
dedic <- rep(1:3, each = 4)        # 3 factors and 12 manifest variables
alpha <- rep(c(1, NA, NA, NA), 3)  # set first loading to 1 for each factor,
                                   #   sample remaining loadings from default
sigma <- rep(0.5, 12)              # idiosyncratic variances all set to 0.5
R <- toeplitz(c(1, .6, .3))        # Toeplitz matrix
dat <- simul.dedic.facmod(N = 5000, dedic, alpha, sigma, R)
```

`simul.nfac.prior` *Simulate prior distribution of number of latent factors*

Description

This function produces a sample from the prior distribution of the number of latent factors. It depends on the prior parameters used for the distribution of the indicators, on the size of the model (number of manifest variables and maximum number of latent factors), and on the identification restriction (minimum number of manifest variables dedicated to each factor).

Usage

```
simul.nfac.prior(nvar, Kmax, Nid = 3, kappa = 1/Kmax, nrep = 10^6)
```

Arguments

`nvar`	Number of manifest variables.
`Kmax`	Maximum number of latent factors.
`Nid`	Minimum number of manifest variables dedicated to each latent factor for identification.
`kappa`	Concentration parameter of the Dirichlet prior distribution on the indicators.
`nrep`	Number of Monte Carlo replications.

Details

This function simulates the prior distribution of the number of latent factors for models that fulfill the identification restriction restriction that at least `Nid` manifest variables (or no variables) are loading on each latent factor. Several (scalar) parameters `kappa` can be passed to the function to simulate the prior for different prior parameter values and compare the results.

An accept/reject sampling scheme is used: a vector of probabilities is drawn from a Dirichlet distribution with concentration parameter `kappa`, and the `nvar` manifest variables are randomly allocated to the `Kmax` latent factors. If each latent factor has at least `Nid` dedicated variables or no variables at all, the identification requirement is fulfilled and the draw is accepted. The number of factors loaded by at least `Nid` manifest variables is returned as a draw from the prior distribution.

Note that this function does not use the two-level prior distribution implemented in CFSHP, where manifest variables can be discarded from the model according to a given probability. Therefore, this function only help understand the prior distribution conditional on all the manifest variables being included into the model.

Value

A list of length equal to the number of parameters specified in `kappa` is returned, where each element of the list contains:

- `nfac`: Vector of integers of length equal to the number of accepted draws.
- `acc`: Acceptance rate of the accept/reject sampling scheme.

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

References

G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014): "Bayesian Exploratory Factor Analysis", *Journal of Econometrics*, 183(1), pages 31-57, doi:10.1016/j.jeconom.2014.06.008.

Examples

```
# replicate first row of table 2 in CFSHP (p.44)
# note: use larger number of replications nrep to improve accuracy
prior.nfac <- simul.nfac.prior(nvar = 15, Kmax = 5, kappa = c(.3, .7, 1),
                               nrep = 10000)
summary(prior.nfac)
plot(prior.nfac)
```

---

simul.R.prior *Simulate prior distribution of factor correlation matrix*

Description

This function produces a sample of correlation matrices drawn from their prior distribution induced in the identified version of the factor model, given the prior distribution specified on the corresponding covariance matrices of the factors in the expanded model.

Usage

```
simul.R.prior(Kmax, nu0 = Kmax + 1, S0 = 1, HW.prior = TRUE,
  nrep = 10^5, verbose = TRUE)
```

Arguments

Kmax	Maximum number of latent factors.
nu0	Degrees of freedom of the Inverse-Wishart prior on the covariance matrix of the latent factors in the expanded model.
S0	Scale parameters of the Inverse-Wishart prior on the covariance matrix of latent factors in the expanded model: • if HW.prior = TRUE, scale parameter of the Gamma hyperprior distribution on the individual scales of the Inverse-Wishart prior. • if HW.prior = FALSE, diagonal elements of the scale matrix of the Inverse-Wishart prior on the covariance matrix of the latent factors in the expanded model. Either a scalar or a numeric vector of length equal to Kmax.

HW.prior	If TRUE, implement Huang-Wand (2013) prior on the covariance matrix of the factors in the expanded model, otherwise use an Inverse-Wishart prior if FALSE, see CFSHP section 2.3.5.
nrep	Number of Monte Carlo replications.
verbose	If TRUE, display information on the progress of the function.

Details

Covariance matrices are sampled from the prior distribution in the expanded model, and transformed to produce the corresponding correlation matrices. See section 2.3.5 of CFSHP (p.36-37), as well as the details of the function befa.

To compare several prior specifications, different values of the parameters nu0 and S0 can be specified. The function then simulates for each pair of these parameters. nu0 and S0 should therefore be scalars or vectors of same length.

Value

A list of length equal to the number of pairs of parameters nu0 and S0, where each element of the list is an array of dimension (Kmax, Kmax, nrep) that contains the correlation matrices of the latent factors drawn from the prior.

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

References

G. Conti, S. Frühwirth-Schnatter, J.J. Heckman, R. Piatek (2014): "Bayesian Exploratory Factor Analysis", *Journal of Econometrics*, 183(1), pages 31-57, doi:10.1016/j.jeconom.2014.06.008.

Examples

```
# partial reproduction of figure 1 in CFSHP (p.38)
# note: use larger number of replications nrep to increase smoothness
Kmax <- 10
Rsim <- simul.R.prior(Kmax, nu0 = Kmax + c(1, 2, 5), S0 = .5, nrep = 1000)
summary(Rsim)
plot(Rsim)
```

---

summary.befa *Summarize 'befa' object*

Description

Generic function summarizing the posterior results of a 'befa' object. Optional arguments can be specified to customize the summary.

Usage

```
## S3 method for class 'befa'
summary(object, ...)
```

Arguments

`object`	Object of class 'befa'.
`...`	The following extra arguments can be specified:

- `what`: How to summarize the posterior distribution?
  - `what = 'maxp'` (default): Only factor loadings with highest posterior probability of being different from zero or discarded from the model (if `dedic = 0`) are summarized.
  - `what = 'all'`: All factor loadings with corresponding posterior probability to be allocated to a given factor (or to be discarded from the model) larger than `min.prob` are summarized.
  - `what = 'hppm'`: Highest posterior probability models with probability larger than `min.prob` are summarized.
- `byfac`: Sort factor loadings by factors if `TRUE`, otherwise by manifest variables if `FALSE` (default).
- `hpd.prob`: Probability used to compute the highest posterior density intervals of the posterior distribution of the model parameters (default: 0.95).
- `min.prob`: If `what = 'all'`, only factor loadings with posterior probability of being dedicated to a given factor (or discarded from the model) larger than this value are displayed. If `what = 'hppm'`, only highest posterior probability models with probability larger than this value are displayed. (default: 0.20)

Details

This function summarizes the posterior distribution of the parameters. The algorithm may visit different configurations of the indicator matrix Δ during sampling, where the manifest variables are allocated to different latent factors. When the posterior distribution of the factor loadings is summarized separately for each manifest variable (`what = 'maxp'` or `what = 'all'`), the function provides the latent factor each manifest variable is allocated to (`dedic`), and the corresponding posterior probability (`prob`). If `dedic = 0`, then `prob` corresponds to the posterior probability that the manifest variable is discarded. Discarded variables are listed last if `byfac = TRUE`. Low probability cases can be discarded by setting `min.prob` appropriately (default is 0.20).

Idiosyncratic variances, factor correlation matrix and regression coefficients (if any) are summarized across all MCMC iterations if `what = 'all'` or `what = 'maxp'`, and within each HPP model if `what = 'hppm'`.

Highest posterior probability model. The HPP model is the model with a given allocation of the measurements to the latent factors (i.e., a given indicator matrix Δ) that is visited most often by the algorithm.

When specifying `what = 'hppm'`, the function sorts the models according to the posterior frequencies of their indicator matrices in decreasing order. Therefore, the first model returned (labeled 'm1') corresponds to the HPP model. Low probability models can be discarded by setting `min.prob`

appropriately(default is 0.20, implying that only models with a posterior probability larger than 0.20 are displayed).

HPP models can only be found if identification with respect to column switching has been restored *a posteriori*. An error message is returned if this is not the case.

Value

If called directly, the summary is formatted and displayed on the standard output. Otherwise if saved in an object, a list of the following elements is returned:

- `MHacc`: Metropolis-Hastings acceptance rate.
- `alpha`: Data frame (or list of data frames if `what = 'hppm'`) containing posterior summary statistics for the factor loadings.
- `sigma`: Data frame (or list of matrices if `what = 'hppm'`) containing posterior summary statistics for the idiosyncratic variances.
- `R`: Data frame (or list of data frames if `what = 'hppm'`) containing posterior summary statistics for the factor correlations.
- `beta`: Data frame (or list of data frames if `what = 'hppm'`) containing posterior summary statistics for the regression coefficients (if any).
- `nfac` (only if `what = 'maxp'` or `what = 'all'`): Table of posterior frequencies of numbers of factors.
- `hppm` (only if `what = 'hppm'`): List of the following elements summarizing the different HPP models, sorted in decreasing order of their posterior probabilities:
  - `prob`: Vector of posterior probabilities.
  - `nfac`: Vector of numbers of factors.
  - `dedic`: Data frame of factor indicators.

Data frames of posterior summary statistics include the means (`mean`), standard deviations (`sd`) and highest posterior density intervals (`hpd.lo` and `hpd.up`, for the probability specified in `hpd.prob`) of the corresponding parameters.

For the factor loadings, the matrix may also include a column labeled '`dedic`' indicating to which factors the corresponding manifest variables are dedicated (a zero value means that the manifest variable does not load on any factor), as well as a column labeled '`prob`' showing the corresponding posterior probabilities that the manifest variables load on these factors.

Summary results are returned as lists of data frames for HPP models, where the elements of the list are labeled as '`m1`, '`m2`', etc.

Author(s)

Rémi Piatek <remi.piatek@gmail.com>

See Also

`plot.befa` to plot posterior results.

Examples

```
set.seed(6)

# generate fake data with 15 manifest variables and 3 factors
Y <- simul.dedic.facmod(N = 100, dedic = rep(1:3, each = 5))

# run MCMC sampler and post process output
# notice: 1000 MCMC iterations for illustration purposes only,
#  increase this number to obtain reliable posterior results!
mcmc <- befa(Y, Kmax = 5, iter = 1000)
mcmc <- post.column.switch(mcmc)
mcmc <- post.sign.switch(mcmc)

# summarize posterior results
summary(mcmc)

# summarize highest posterior probability (HPP) model
hppm.sum <- summary(mcmc, what = 'hppm')

# print summary with 6-digit precision
print(hppm.sum, digits = 6)

# extract posterior means of the factor loadings in HPP model
alpha.mean <- hppm.sum$alpha$m1$mean
print(alpha.mean)
```

Index

BayesFM, 2
BayesFM-package (BayesFM), 2
befa, 3, *18*

plot.befa, *8*, 9, *20*
post.column.switch, *6*, *8*, 11, *12*, *13*
post.sign.switch, *6*, *8*, *11*, 12

simul.dedic.facmod, 13
simul.nfac.prior, *8*, 16
simul.R.prior, *8*, 17
summary.befa, *8*, *10*, 18