Package: tensorBF (via r-universe)

October 31, 2024

Type Package

Title Bayesian Tensor Factorization

Version 1.0.2

Date 2018-10-01

Author Suleiman A Khan [aut, cre], Muhammad Ammad-ud-din [aut]

Maintainer Suleiman A Khan <khan.suleiman@gmail.com>

Description Bayesian Tensor Factorization for decomposition of tensor data sets using the trilinear CANDECOMP/PARAFAC (CP) factorization, with automatic component selection. The complete data analysis pipeline is provided, including functions and recommendations for data normalization and model definition, as well as missing value prediction and model visualization. The method performs factorization for three-way tensor datasets and the inference is implemented with Gibbs sampling.

License MIT + file LICENSE

Imports tensor, methods

RoxygenNote 6.1.0

NeedsCompilation no

Date/Publication 2018-10-02 18:10:03 UTC

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

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

RemoteRef HEAD

RemoteSha bf051a0197eeab6971c959530c1e51cfe311dcc4

Contents

`getDefaultOpts` *A function for generating a default set of parameters for Bayesian Tensor Factorization methods*

Description

`getDefaultOpts` returns the default choices for model parameters.

Usage

```
getDefaultOpts(method = "CP")
```

Arguments

`method`	the factorization method for which options are required. Currently only "CP" (default) is supported.

Details

This function returns options for defining the model's high-level structure (sparsity priors), the hyperparameters, and the uninformative priors. We recommend keeping these as provided.

Value

A list with the following model options:

`ARDX`	TRUE: use elementwise ARD prior for X, resulting in sparse X's. FALSE: use guassian prior for a dense X (default).
`ARDW`	TRUE: use elementwise ARD prior for W, resulting in sparse W's (default). FALSE: use guassian prior for a dense W.
`ARDU`	TRUE: use elementwise ARD prior for U, resulting in sparse U's. FALSE: use guassian prior for a dense U (default).
`iter.burnin`	The number of burn-in samples (default 5000).
`iter.sampling`	The number of saved posterior samples (default 50).
`iter.thinning`	The thinning factor to use in saving posterior samples (default 10).
`prior.alpha_0t`	The shape parameter for residual noise (tau's) prior (default 1).
`prior.beta_0t`	The rate parameter for residual noise (tau's) prior (default 1).
`prior.alpha_0`	The shape parameter for the ARD precisions (default 1e-3).
`prior.beta_0`	The rate parameter for the ARD precisions (default 1e-3).

prior.betaW1	Bernoulli prior for component activiations, prior.betaW1 < prior.betaW2: sparsity inducing (default: 1).
prior.betaW2	Bernoulli prior for component activation, (default: 1).
init.tau	The initial value for noise precision (default 1e3).
verbose	The verbosity level. 0=no printing, 1=moderate printing, 2=maximal printing (default 1).
checkConvergence	Check for the convergence of the data reconstruction, based on the Geweke diagnostic (default TRUE).

Examples

```
#To run the algorithm with other values:
opts <- getDefaultOpts()
opts$ARDW <- FALSE #Switch off Feature-level Sparsity on W's
 ## Not run: res <- tensorBF(Y=Y,opts=opts)
```

---

normFiberCentering *Preprocessing: fiber Centering*

Description

normFiberCentering center the fibers of the o^{th} mode of the tensor to zero mean.

Usage

```
normFiberCentering(Y, o)
```

Arguments

Y	the tensor data. See function tensorBF for details.
o	the o^{th} (default: 1) mode of the tensor in which the fibers are to be centered to zero mean.

Value

a list containing the following elements:

data	The data after performing the required centering operation.
pre	The centering values used for preprocessing.

References

Kolda, Tamara G., and Brett W. Bader. "Tensor decompositions and applications." SIAM review 51.3 (2009): 455-500.

Examples

```
#Data generation
K <- 3
X <- matrix(rnorm(20*K),20,K)
W <- matrix(rnorm(30*K),30,K)
U <- matrix(rnorm(3*K),3,K)
Y = 0
for(k in 1:K) Y <- Y + outer(outer(X[,k],W[,k]),U[,k])
 Y <- Y + array(rnorm(20*30*3),dim=c(20,30,3))

#center the fibers in first mode of tensor Y
res <- normFiberCentering(Y=Y,o=1)
dim(res$data) #the centered data
```

`normSlabScaling` *Preprocessing: Slab Scaling*

Description

`normSlabScaling` scales the slabs of the o^{th} mode of the tensor to unit variance.

Usage

```
normSlabScaling(Y, o = 2)
```

Arguments

`Y`	the tensor data. See function `tensorBF` for details.
`o`	the o^{th} (default: 2) mode of the tensor in which the slabs are to be scaled to unit variance.

Value

a list containing the following elements:

`data`	The data after performing the required scaling operation.
`pre`	The scale's used for preprocessing.

References

Kolda, Tamara G., and Brett W. Bader. "Tensor decompositions and applications." SIAM review 51.3 (2009): 455-500.

Examples

```
#Data generation
K <- 3
X <- matrix(rnorm(20*K),20,K)
W <- matrix(rnorm(30*K),30,K)
U <- matrix(rnorm(3*K),3,K)
Y = 0
for(k in 1:K) Y <- Y + outer(outer(X[,k],W[,k]),U[,k])
 Y <- Y + array(rnorm(20*30*3),dim=c(20,30,3))

#scale the slabs in second mode of tensor Y
res <- normSlabScaling(Y=Y,o=2)
dim(res$data) #the scaled data
```

---

plotTensorBF *Plot Tensor Components*

Description

plotTensorBF shows the heatmap of components inferred by tensorBF.

Usage

```
plotTensorBF(res, Y = NULL, k = 1, modesOnAxis = c(1, 2, 3),
  nTopFeatures = c(5, 15, 3), margins = c(4, 4, 4, 12), cex.axis = 1,
  cols = colorRampPalette(c("blue", "white", "red"))(101), key = TRUE,
  plimit = NULL)
```

Arguments

res	The learned tensorBF model.
Y	The original input data to be plotted. If specified NULL, the function plots the data reconstruction using reconstructTensorBF (default: NULL).
k	the component number to visualize (default: 1).
modesOnAxis	which mode to plot on each axis c(Yaxis,Xaxis,lateral). Defaults to c(1,2,3).
nTopFeatures	The number of most relevant features to show for the data space visualizations in each of the modes. Defaults to c(5,15,3) for displaying top 10 features of 1^{st} mode, 20 of 2^{nd} mode and 5 of 3^{rd} mode.
margins	numeric vector of length 4 containing the margins (see par(mar= *))
cex.axis	positive numbers, used as cex.axis (default: 1)
cols	colors used for the image. Defaults to a blue-white-red color scale.
key	logical indicating whether a color-key should be drawn.
plimit	(optional) numerical number indicating the maximum absolute value to be plotted in the heatmap.

Examples

```
#Data generation
K <- 3
X <- matrix(rnorm(20*K),20,K)
W <- matrix(rnorm(30*K),30,K)
U <- matrix(rnorm(3*K),3,K)
Y = 0
for(k in 1:K) Y <- Y + outer(outer(X[,k],W[,k]),U[,k])
 Y <- Y + array(rnorm(20*30*3,0,0.25),dim=c(20,30,3))

#Run the method with default options
## Not run: res1 <- tensorBF(Y)
## Not run: plotTensorBF(res = res1,Y=Y,k=1)
```

---

predictTensorBF *Predict Missing Values using the Bayesian tensor factorization model*

Description

predictTensorBF predicts the missing values in the data Y using the learned model res.

Usage

```
predictTensorBF(Y, res)
```

Arguments

Y	is a 3-mode tensor containing missing values as NA's. See function tensorBF for details.
res	the model object returned by the function tensorBF.

Details

If the original data Y contained missing values (NA's), this function predicts them using the model. The predictions are returned in the un-normalized space if res$pre contains appropriate preprocessing information.

Value

A tensor of the same size as Y containing predicted values in place of NA's.

Examples

```
#Data generation
## Not run: K <- 2
## Not run: X <- matrix(rnorm(20*K),20,K)
## Not run: W <- matrix(rnorm(30*K),30,K)
## Not run: U <- matrix(rnorm(3*K),3,K)
```

```
## Not run: Y = 0
## Not run: for(k in 1:K) Y <- Y + outer(outer(X[,k],W[,k]),U[,k])
## Not run:  Y <- Y + array(rnorm(20*30*3,0,0.25),dim=c(20,30,3))

#insert missing values
## Not run: m.inds = sample(prod(dim(Y)),100)
## Not run: Yobs = Y[m.inds]
## Not run: Y[m.inds] = NA

#Run the method with default options and predict missing values
## Not run: res <- tensorBF(Y)
## Not run: pred = predictTensorBF(Y=Y,res=res)
## Not run: plot(Yobs,pred[m.inds],xlab="obs",ylab="pred",main=round(cor(Yobs,pred[m.inds]),2))
```

`reconstructTensorBF` *Reconstruct the data based on posterior samples*

Description

`reconstructTensorBF` returns the reconstruction of the data based on posterior samples of a given run. The function reconstructs the tensor for each posterior sample and then computes the expected value. The reconstruction is returned in the un-normalized space if `res$pre` contains appropriate preprocessing information.

Usage

```
reconstructTensorBF(res)
```

Arguments

`res`	The model object from function `tensorBF`.

Value

The reconstructed data, a tensor of the size equivalent to the data on which the model was run.

Examples

```
#Data generation
K <- 3
X <- matrix(rnorm(20*K),20,K)
W <- matrix(rnorm(30*K),30,K)
U <- matrix(rnorm(3*K),3,K)
Y = 0
for(k in 1:K) Y <- Y + outer(outer(X[,k],W[,k]),U[,k])
 Y <- Y + array(rnorm(20*30*3,0,0.25),dim=c(20,30,3))

#Run the method with default options and reconstruct the model's representation of the tensor
## Not run: res <- tensorBF(Y)
## Not run: recon = reconstructTensorBF(res)
```

```
## Not run: inds = sample(prod(dim(Y)),100)
## Not run: plot(Y[inds],recon[inds],xlab="obs",ylab="recon",main=round(cor(Y[inds],recon[inds]),2))
```

---

tensorBF *Bayesian Factorization of a Tensor*

Description

tensorBF implements the Bayesian factorization of a tensor.

Usage

```
tensorBF(Y, method = "CP", K = NULL, opts = NULL,
  fiberCentering = NULL, slabScaling = NULL, noiseProp = c(0.5, 0.5))
```

Arguments

Y	is a three-mode tensor to be factorized.
method	the factorization method. Currently only "CP" (default) is supported.
K	The number of components (i.e. latent variables or factors). Recommended to be set somewhat higher than the expected component number, so that the method can determine the model complexity by prunning excessive components (default: 20% of the sum of lower two dimensions). High values result in high CPU time. NOTE: Adjust parameter noiseProp if sufficiently large values of K do not lead to a model with pruned components.
opts	List of model options; see function getDefaultOpts for details and default.
fiberCentering	the mode for which fibers are to be centered at zero (default = NULL). Fiber is analogous to a vector in a particular mode. Fiber centering and Slab scaling are the recommended normalizations for a tensor. For details see the provided normalization functions and the references therein.
slabScaling	the mode for which slabs are to be scaled to unit variance (default = NULL). Slab is analogous to a matrix in a particular mode. Alternativly, you can preprocess the data using the provided normalization functions.
noiseProp	c(prop,conf); sets an informative noise prior for tensorBF. The model sets the noise prior such that the expected proportion of variance explained by noise is defined by this parameter. It is recommended when the standard prior from getDefaultOpts seems to overfit the model by not prunning any component with high initial K. Use NULL to switch off informative noise prior. - prop defines the proportion of total variance to be explained by noise (between 0.1 and 0.9), - conf defines the confidence in the prior (between 0.1 and 10). We suggest a default value of c(0.5,0.5) for real data sets.

Details

Bayesian Tensor Factorization performs tri-linear (CP) factorization of a tensor. The method automatically identifies the number of components, given K is initialized to a large enough value, see arguments. Missing values are supported and should be set as NA's in the data. They will not affect the model parameters, and can be predicted with function `predictTensorBF`, based on the observed values.

Value

A list containing model parameters. For key parameters, the final posterior sample ordered w.r.t. component variance is provided to aid in initial checks; all the posterior samples should be used for model analysis. The list elements are:

`K`	The number of learned components. If this value is not less then the input argument K, the model should be rerun with a larger K or use the noiseProp parameter.
`X`	a matrix of $N \times K$ dimensions, containing the last Gibbs sample of the first-mode latent variables.
`W`	a matrix of $D \times K$ dimensions, containing the last Gibbs sample of the second-mode latent variables.
`U`	a matrix of $L \times K$ dimensions, containing the last Gibbs sample of the third-mode latent variables.
`tau`	The last sample of noise precision.

and the following elements:

`posterior`	the posterior samples of model parameters (X,U,W,Z,tau).
`cost`	The likelihood of all the posterior samples.
`opts`	The options used to run the model.
`conv`	An estimate of the convergence of the model, based on reconstruction of data using the Geweke diagnostic. Values significantly above 0.05 occur when model has not converged and should therefore be rerun with a higher value of iter.burnin in `getDefaultOpts`.
`pre`	A list of centering and scaling values used to transform the data, if any. Else an empty list.

Examples

```
#Data generation
K <- 2
X <- matrix(rnorm(20*K),20,K)
W <- matrix(rnorm(25*K),25,K)
U <- matrix(rnorm(3*K),3,K)
Y = 0
for(k in 1:K) Y <- Y + outer(outer(X[,k],W[,k]),U[,k])
 Y <- Y + array(rnorm(20*25*3,0,0.25),dim=c(20,25,3))

#Run the method with default options
```

```
## Not run: res2 <- tensorBF(Y=Y)

#Run the method with K=3 and iterations=1000
## Not run: opts <- getDefaultOpts(); opts$iter.burnin = 1000
## Not run: res1 <- tensorBF(Y=Y,K=3,opts=opts)

#Vary the user defined expected proportion of noise variance
#explained. c(0.2,1) represents 0.2 as the noise proportion
#and confidence of 1
## Not run: res3 <- tensorBF(Y=Y,noiseProp=c(0.2,1))
```

`undoFiberCentering` *Postprocessing: Undo fiber Centering*

Description

`undoFiberCentering` reverts the fiber's of the o^{th} mode to undo the centering effect.

Usage

```
undoFiberCentering(Yn, pre)
```

Arguments

`Yn`	the normalized tensor data. This can be, for example, the output of `reconstructTensorBF`.
`pre`	The centering parameters used for preprocessing in the format as produced by `normFiberCentering`.

Value

The data tensor after reversing the centering operation.

References

Kolda, Tamara G., and Brett W. Bader. "Tensor decompositions and applications." SIAM review 51.3 (2009): 455-500.

Examples

```
#Given tensor Y
## Not run: Ycentered <- normFiberCentering(Y=Y,o=1)
## Not run: Yuncentered <- undoFiberCentering(Ycentered$data,Ycentered$pre)
```

undoSlabScaling *Postprocessing: Undo Slab Scaling*

Description

undoSlabScaling reverts the slabs of the o^{th} mode to undo the scaling effect.

Usage

```
undoSlabScaling(Yn, pre)
```

Arguments

Yn	the normalized tensor data. This can be, for example, the output of reconstructTensorBF.
pre	The scaling values and mode used for preprocessing in the format as produced by normSlabScaling.

Value

The data tensor after reversing the scaling operation.

References

Kolda, Tamara G., and Brett W. Bader. "Tensor decompositions and applications." SIAM review 51.3 (2009): 455-500.

Examples

```
#Given tensor Y
## Not run: Yscaled <- normSlabScaling(Y=Y,o=2)
## Not run: Yunscaled <- undoSlabScaling(Yscaled$data,Yscaled$pre)
```

Index