Package ‘bayesMCClust’

February 19, 2015

Type Package

Title Mixtures-of-Experts Markov Chain Clustering and Dirichlet Multinomial Clustering

Version 1.0

Date 2012-01-26

Author Christoph Pamminger <christoph.pamminger@gmail.com>

Maintainer Christoph Pamminger <christoph.pamminger@gmail.com>

Description This package provides various Markov Chain Monte Carlo (MCMC) sampler for model-based clustering of discrete-valued time series obtained by observing a categorical variable with several states (in a Bayesian approach). In order to analyze group membership, we provide also an extension to the approaches by formulating a probabilistic model for the latent group indicators within the Bayesian classification rule using a multinomial logit model.

Depends R (>= 2.14.1), gplots, xtable, grDevices, mnormt, MASS, bayesm, boa, e1071, gtools

Suggests nnet

License GPL-2

LazyLoad yes

Repository CRAN

Date/Publication 2012-01-31 10:57:02

NeedsCompilation no

R topics documented:

`bayesMCClust-package` *Mixtures-of-Experts Markov Chain Clustering and Dirichlet Multinomial Clustering*

Description

This package provides various Markov Chain Monte Carlo (MCMC) samplers for model-based clustering of discrete-valued time series obtained by observing a categorical variable with several states (in a Bayesian approach). These methods are based on finite mixtures of first-order time-homogeneous Markov chain (models) with unknown transition matrices. In the Markov chain clustering approach the individual transition probabilities are fixed to a group-specific transition matrix. In the second approach called Dirichlet multinomial clustering it is assumed that within each group unobserved heterogeneity is still existent and is captured by allowing the individual transition matrices to deviate from the group means by describing this variation for each row through a Dirichlet distribution with unknown hyperparameters. Further, in order to analyze group membership, we provide also an extension to these approaches by formulating a probabilistic model for the latent group indicators within the Bayesian classification rule using a multinomial logit model. In other words, unobserved group membership is modeled as a multinomial logit model which allows for dependence on individual-specific and other characteristics. Additionally, functions to process the results are provided.

Details

Package:	bayesMCClust
Type:	Package
Version:	1.0

Date:	2012-01-26
License:	GPL-2
LazyLoad:	yes

The main functions are mcClust for Markov Chain Clustering and dmClust for Dirichlet Multinomial Clustering as well as mcClustExtended and dmClustExtended which also include the mixtures-of-experts extension. These functions use a special structure of the data (see Njk.i in the **Examples** therein and/or e.g. MCCExampleData and MCCExtExampleData). Therefore dataListToNjki and dataFrameToNjki are provided to help preparing the data (see examples therein). Additionally, a function MNLAuxMix is provided for multinomial logit regression using the auxiliary mixture approach (see **References**). Note that also prior information may be incorporated as these methods are "Bayesian" approaches. Thus, to estimate the parameters such as transition probabilities, regression coefficients or mixing proportions, MCMC algorithms are used. For more details about the models and estimation procedures see **References**. The results are returned in lists and also saved to output files. To process the results some more functions are provided to analyse and visualise the results; so for example the (group-specific) transition probabilities can be visualised with plotTransProbs. Finally, also some well-known model selection criteria can be calculated with calcMSCrit.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

Maintainer: Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

Sylvia Fruehwirth-Schnatter and Rudolf Fruehwirth, (2010), "Data augmentation and MCMC for binary and multinomial logit models". In T. Kneib and G. Tutz (eds): *Statistical Modelling and Regression Structures: Festschrift in Honour of Ludwig Fahrmeir*. Physica Verlag, Heidelberg, pp. 111-132. DOI: 10.1007/978-3-7908-2413-1_7 http://www.springerlink.com/content/t4h810017645wh68/. See also: IFAS Research Paper Series 2010-48 (http://www.jku.at/ifas/content/e108280/e108491/e108471/e109880/ifas_rp48.pdf).

See Also

mcClust, dmClust, mcClustExtended, dmClustExtended, MNLAuxMix, calcAllocations

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended, MNLAuxMix
```

---

calcAllocations — *Computes Group Sizes, Group Membership and Individual Posterior Classification Probabilities*

Description

Computes (estimates) group sizes, group membership and individual posterior classification probabilities based on the outcome of a specificed MCMC run of either mcClust, mcClustExtended, dmClust or dmClustExtended as well as MNLAuxMix.

Usage

```
calcAllocationsMCC(outList, thin = 1, maxi = 50,
                    M0 = outList$Mcmc$M0, plotPathsForEta = TRUE)
calcAllocationsMCCExt(outList, thin = 1, maxi = 50,
                    M0 = outList$Mcmc$M0)
calcAllocationsDMC(outList, thin = 1, maxi = 50,
                    M0 = outList$Mcmc$M0, plotPathsForEta = TRUE)
calcAllocationsDMCExt(outList, thin = 1, maxi = 50,
                    M0 = outList$Mcmc$M0)
calcAllocationsMNL(outList, thin = 1, maxi = 50,
                    M0 = outList$Mcmc$M0)
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of mcClust, dmClust, mcClustExtended, dmClustExtended or MNLAuxMix.
`thin`	An integer specifying the thinning parameter (default is 1).
`maxi`	specifies the number of draws to be actually taken (after thinning) from the MCMC draws beginning from the end of the chain (default is 50), except for mixing proportions/weights η where all `thin`-th draws beginning at `M0` are used.
`M0`	specifies the number of the first MCMC draw after burn-in (default is `outList$Mcmc$M0`).
`plotPathsForEta`	If `TRUE` (default) paths of the MCMC draws of the mixing proportions/weights η (corresponding to group sizes) are drawn.

Details

The last `maxi` MCMC draws of each `thin`-th draw are taken for calculations, except for mixing proportions η (which are part of MCC and DMC *without* MNL extension) where *all* `thin`-th draws beginning at `M0` are used.

Value

A list containing:

- `estGroupSize` — A vector of dimension H containing the posterior mean of group sizes. For MCC and DMC *without* MNL extension `estGroupSize` contains the mixing proportions/weights $\hat{\eta}$. In these cases each `thin`-th MCMC draw beginning at `M0` (after burn-in) is used for calculation. For MCC and DMC *with* MNL extension and `MNLAuxMix` the group sizes are calculated based on the individual posterior classification probabilities which are calculated using the last `maxi` draws of each `thin`-th MCMC draw.
- `class` — A vector of length N containing the group membership, which is determined for each individual according to the *maximum* individual posterior classification probability.
- `classProbs` — A matrix with dimension $N \times H$ containing the individual posterior classification probabilities which are calculated using the last `maxi` draws of each `thin`-th MCMC draw.

Note

The last `maxi` MCMC draws of each `thin`-th draw are taken for calculations, except for mixing proportions η (which are part of MCC and DMC *without* MNL extension) where all `thin`-th draws beginning at `M0` are used.

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 `http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract`

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 `http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf`

See Also

`mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended`, `MNLAuxMix`

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended, MNLAuxMix
```

---

calcEntropy *Calculates the Entropy of a Given Classification*

Description

Calculates the entropy of a given classification based on the outcome of a specificed MCMC run of either `mcClust`, `mcClustExtended`, `dmClust` or `dmClustExtended` as well as `MNLAuxMix`.

Usage

```
calcEntropy(outList, classProbs, class,
            grLabels = paste("Group", 1:outList$Prior$H),
            printXtable = TRUE)
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended` or `MNLAuxMix`.
`classProbs`	A matrix with dimension $N \times H$ containing the individual posterior classification probabilities returned by `calcAllocations`.
`class`	A vector of length N containing the group membership returned by `calcAllocations`.
`grLabels`	A character vector giving user-specified names for the clusters/groups.
`printXtable`	If `TRUE` (default) a LaTeX-style table of the entropy is generated.

Value

A matrix of dimension $(H + 1) \times 3$, where H is the number of clusters/groups, containing the contribution of each cluster/group to the (total) entropy – absolute and relative to group size (number of group members). The calculation of the entropy is based on the individual posterior classification probabilities.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

`calcAllocations`, `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended`, `MNLAuxMix`

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended, MNLAuxMix
```

---

calcEquiDist   *Calculates (And Plots) the Stationary Distribution (Steady State)*

Description

Calculates (and plots) the posterior expectations of the cluster-specific stationary distributions (also equilibrium distributions or steady states) of the Markov chains (outcome variable) based on the transition matrices for each cluster/group.

Usage

```
calcEquiDist(outList, thin = 1, maxi = 50, M0 = outList$Mcmc$M0,
             grLabels = paste("Group", 1:outList$Prior$H),
             printEquiDist = TRUE, plotEquiDist = TRUE)
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended` or `dmClustExtended`.
`thin`	An integer specifying the thinning parameter (default is 1).
`maxi`	specifies the number of draws to be actually taken (after thinning) from the MCMC draws beginning from the end of the chain (default is 50).
`M0`	specifies the number of the first MCMC draw after burn-in (default is `outList$Mcmc$M0`).
`grLabels`	A character vector giving user-specified names for the clusters/groups.
`printEquiDist`	If `TRUE` (default) a LaTeX-style table containing the stationary distributions is generated.
`plotEquiDist`	If `TRUE` (default) a barplot of the stationary distributions is drawn.

Details

The last `maxi` MCMC draws of each `thin`-th draw are taken for calculations.

Value

A matrix of dimension $(K + 1) \times H$ containing the stationary distributions (steady states) of the Markov chains (outcome variable) based on the transition matrices in the various clusters/groups. Note, H is the number of clusters/groups and $K+1$ the number of states of the categorical outcome variable.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 `http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract`

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 `http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf`

See Also

`mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended`, `barplot2`

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended
```

---

calcLongRunDist — *Calculates And Plots the Long-Run Distribution Over the Categories of the Outcome Variable After Certain Periods.*

Description

Calculates and plots the posterior expectation of the cluster-specific 'long-run' distribution over the categories of the outcome variable after a period of certain time units t in the various clusters starting at a specified initial state vector (corresponding to $t = 0$). The calculation is based on the transition matrices for each cluster/group. It includes also the stationary distribution ($t = \infty$).

Usage

```
calcLongRunDist(outList, initialStateData, class, equiDist,
                thin = 1, maxi = 50, M0 = outList$Mcmc$M0,
                printLongRunDist = TRUE,
                grLabels = paste("Group", 1:outList$Prior$H) )
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended` or `dmClustExtended`.
`initialStateData`	A vector of length N containing the initial states where to start from.
`class`	A vector of length N containing the group membership returned by `calcAllocations`.
`equiDist`	A matrix of dimension $(K + 1) \times H$ containing the stationary distributions (steady states) of the Markov chains (outcome variable) in the various clusters returned by `calcEquiDist`.
`thin`	An integer specifying the thinning parameter (default is 1).
`maxi`	specifies the number of draws to be actually taken (after thinning) from the MCMC draws beginning from the end of the chain (default is 50).
`M0`	specifies the number of the first MCMC draw after burn-in (default is `outList$Mcmc$M0`).
`printLongRunDist`	If `TRUE` (default) a LaTeX-style table containing the long-run distribution for each cluster/group is generated.
`grLabels`	A character vector giving user-specified names for the clusters/groups.

Details

A barplot of the long-run distributions is drawn for each cluster/group, including also the stationary distribution (steady state).

The last `maxi` MCMC draws of each `thin`-th draw are taken for calculations.

Value

A list containing the long-run distributions for each cluster/group.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \dots, K$ (instead of $1, \dots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

calcAllocations, calcEquiDist, mcClust, dmClust, mcClustExtended, dmClustExtended, barplot2

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended
```

---

calcMSCrit *Calculates Model Selection Criteria For Several (Independent) MCMC Runs And Various Numbers H of Clusters*

Description

Calculates and plots a set of model selection criteria (depending on the underlying model: e.g. BIC, adjusted BIC, DIC – Deviance Information Criterion, AWE – Approximate Weight of Evidence, CLC – Classification Likelihood Criteria, ICL – Integrated Classification Likelihood, ICL-BIC) for all estimated models produced by one and the same cluster method (for the sake of comparability) and for various numbers H of clusters/groups and several independent MCMC runs saved in output files located in the specified directory. Therefore several maximisation methods are available. For more information about the criteria see **Details**, **References** and references therein.

Usage

```
calcMSCritMCC(workDir, myLabel = "model choice for ...", H0 = 3,
          whatToDoList = c("approxMCL", "approxML", "postMode"))
calcMSCritMCCExt(workDir, NN, myLabel = "model choice for ...",
          ISdraws = 3, H0 = 3,
          whatToDoList = c("approxMCL", "approxML", "postMode"))
calcMSCritDMC(workDir, myLabel = "model choice for ...",
          myN0 = "N0 = ...",
          whatToDoList = c("approxMCL", "approxML", "postMode"))
calcMSCritDMCExt(workDir, myLabel = "model choice for ...",
          myN0 = "N0 = ...",
          whatToDoList = c("approxMCL", "approxML", "postMode"))
```

Arguments

- `workDir` A character giving the name (or full path) of the directory containing the output files of the estimated models produced by one and the same cluster method (for the sake of comparability) for which model selection criteria have to be calculated.
- `NN` Number of individuals N (just for argument/parameter checks).
- `myLabel` Specifies (part of) labeling of the plots.
- `myN0` A character documenting the value of `Prior$N0` (has to be equal for all processed models for the sake of comparability!) – just for labeling.
- `H0` Number of 'expected' clusters/groups by user. Necessary for the calculation of the model prior *adjusted BIC*. See **Details**.
- `ISdraws` Number of draws for the importance sampling step to approximate the logICL.
- `whatToDoList` A character vector containing a subset of `c("approxMCL", "approxML", "postMode")`. Depending on the entries in this list (`whatToDoList`) the calculation of (all) the criteria is based on the MCMC draws (iteration) corresponding to the maximum of the log classification likelihood (`"approxMCL"`), log likelihood (`"approxML"`) and/or log posterior density (`"postMode"`).

Details

For each maximisation method in `whatToDoList` all (available) model selection criteria are calculated (in an iterative manner). Depending on the entries in this list (`whatToDoList`) the calculation of (all) these criteria is based on the MCMC draws (iteration) corresponding to the maximum of the log classification likelihood (`"approxMCL"`), log likelihood (`"approxML"`) and/or (for the sake of completeness) log posterior density (`"postMode"`).

Note, that the user has to decide which criteria are admissible.

Which criteria needs which maximisation method? The AWE and the logICL are based on the maximum of the (log) classification likelihood, all the others on the maximum of the (log) likelihood (see **References**).

By the way, it internally calculates the log-likelihood and related values such as `LK` (observed log-likelihood), `CLK` (classification or complete log-likelihood), `CK` (classification-type log-likelihood), `EK` (entropy term) as well as d_h (number of parameters) which are essential parts of the model selection criteria.

We calculate the model prior *adjusted BIC* using $adjBIC = BIC - 2H\log(H_0) + 2log\Gamma(H + 1) + 2H_0$.

According to the used model type the following criteria are calculated: Bic, adjusted Bic, Aic, Awe, IclBic, Clc, Dic2, Dic4 and logICL (see **References**). Furthermore, plots and tables of selected critera are generated (and plots are also saved in directory `workDir`).

To document the iteration progress, some information is recorded for each output file (containing an MCMC run) – depending on maximisation method – like: a running number, maximisation method, number of cluster/groups, BIC, adjusted BIC, AIC, AWE, CLC, IclBic, DIC2, DIC4a, ICL and additionally adj Rand (which compares the starting with the final allocation).

For each entry in `whatToDo` a matrix `MSCritTable` is produced. Each row represents a processed output file (containing an MCMC run) and the colums contain:

`H` number of clusters/groups

`mMax` number/position of the MCMC draw/iteration leading to the maximum value of the (log-)posterior density or (classification) log-likelihood (depending on `whatToDo`) which is calculated for each MCMC draw

`maxLPD` the maximum value of the (log-)posterior density itself, only if `whatToDo` includes `"postMode"` – corresponding to the posterior mode

`maxLL` the maximum value of the log-likelihood itself, only if `whatToDo` includes `"approxML"` – corresponding to the 'approximate maximum likelihood'

`maxLCL` the maximum value of the classification log-likelihood itself, only if `whatToDo` includes `"approxMCL"` – corresponding to the 'approximate maximum classification likelihood'

`BIC` Bayesian Information Criterion (Schwarz Criterion)

`adjBIC` adjusted BIC – Note: not available/implemented for DMC[Ext]!

`AIC` Akaike Information Criterion

`AWE` Approximate Weight of Evidence, see Banfield and Raftery (1993)

`CLC` Classification Likelihood Criterion

`IclBic` Integrated Classification Likelihood-BIC

`DIC2` Deviance Information Criterion (DIC2), see Fruehwirth-Schnatter and Pyne (2010) and Fruehwirth-Schnatter et al. (2011) – Note: not available/implemented for DMC!

`DIC4a` Deviance Information Criterion (DIC4a), see Fruehwirth-Schnatter and Pyne (2010) and Fruehwirth-Schnatter et al. (2011) – Note: not available/implemented for DMC!

`logICL` log Integrated Classification Likelihood – Note: not available/implemented for DMC[Ext]!

`adjRand` adjusted Rand-Index for (estimated) group membership VS starting values `Initial$S.i.start` (only if not `NULL`)

For each entry in `whatToDo` the corresponding `MSCritTable` is printed together with the current working directory and the content of the current `whatToDo`. Further, plots of the model selection criteria are produced and saved (with type `eps` and `pdf`).

If *MCCExt* is considered also the number of importance sampling draws `ISdraws` (necessary for logICL) is printed.

Additionally, after each iteration the workspace containing the model selection criteria and other stuff is saved to a .RData-file via `save.image` within directory `workDir`.

Finally, a list containing the names of the processed output files (each containing an MCMC run) is printed.

Value

A list containing:

`postMode` the corresponding `MSCritTable` (see **Details**), only if `whatToDo` includes `"postMode"`

`approxML` the corresponding `MSCritTable` (see **Details**), only if `whatToDo` includes `"approxML"`

`approxMCL` the corresponding `MSCritTable` (see **Details**), only if `whatToDo` includes `"approxMCL"`

`ISdraws` the number of importance sampling draws for approximating logICL (only for *MCCExt*)

`outFileNames` a list (character vector) containing the names of the processed output files (each containing an MCMC run)

Note

Note, that the user has to decide which criteria are admissible.

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Jeffrey D. Banfield and Adrian E. Raftery, (1993), "Model-Based Gaussian and Non-Gaussian Clustering". *Biometrics*, Vol. 49, No. 3, pp. 803-821. http://www.jstor.org/stable/2532201

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Sylvia Fruehwirth-Schnatter and Saumyadipta Pyne, (2010), "Bayesian inference for finite mixtures of univariate and multivariate skew-normal and skew-t distributions". *Biostatistics*, Vol. 11, No. 2, pp. 317-336. DOI: 10.1093/biostatistics/kxp062 http://biostatistics.oxfordjournals.org/content/11/2/317.full.pdf+html

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

classAgreement, savePlot, mcClust, dmClust, mcClustExtended, dmClustExtended

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended
```

---

calcNumEff — *Calculates Inefficiency Factors of the MCMC Draws Obtained for the Cluster-Specific Parameters*

Description

Calculates the inefficiency factors of the MCMC draws using numEff from the R package **bayesm** (see **References**).

Usage

```
calcNumEff(outList, thin = 1, printXi = TRUE, printE = TRUE,
           printBeta = TRUE,
           grLabels = paste("Group", 1:outList$Prior$H))
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended` or `MNLAuxMix`.
`thin`	An integer specifying the thinning parameter (default is 1).
`printXi`	If `TRUE` (default) a LaTeX-style table containing the inefficiency factors of the cluster-specific transition matrices is generated and also printed.
`printE`	If `TRUE` (default) a LaTeX-style table containing the inefficiency factors of the cluster-specific parameter matrices is generated and also printed.
`printBeta`	If `TRUE` (default) a LaTeX-style table containing the inefficiency factors of the MNL regression coefficients is generated and also printed.
`grLabels`	A character vector giving user-specified names for the clusters/groups.

Value

A list containing tables of inefficiency factors:

`numEffXi[h]m`	Inefficiency factors of the MCMC draws obtained for each row $j = 1, \dots, K+1$ of the cluster-specific transition matrices $\boldsymbol{\xi}_{h,j\cdot}$ for each cluster/group.
`numEffEhm`	Inefficiency factors of the MCMC draws obtained for each row $j = 1, ..., K+1$ of the cluster-specific parameter matrices (only for DMC[Ext]) $\mathbf{e}_{h,j\cdot}$ for each cluster/group.
`numEffBeta`	Inefficiency factors of the MCMC draws obtained for the MNL regression coefficients for each cluster.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \dots, K$ (instead of $1, \dots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

Peter E. Rossi, Greg M. Allenby and Rob McCulloch, (2005), *Bayesian Statistics and Marketing*, Chichester: Wiley. http://www.perossi.org/home/bsm-1

See Also

numEff, mcClust, dmClust, mcClustExtended, dmClustExtended, MNLAuxMix

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended
```

---

calcParMatDMC — *Calculates the Posterior Expectation of the Cluster-Specific Parameter Matrices (only for DMC[Ext])*

Description

Calculates the posterior expectation of the cluster-specific parameter matrices $\mathbf{e}_h$ (only for DMC[Ext]).

Usage

```
calcParMatDMC(outList, thin = 1, M0 = outList$Mcmc$M0,
              grLabels = paste("Group", 1:outList$Prior$H),
              printPar = TRUE)
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of dmClust or dmClustExtended.
`thin`	An integer specifying the thinning parameter (default is 1).
`M0`	specifies the number of the first MCMC draw after burn-in (default is `outList$Mcmc$M0`).
`grLabels`	A character vector giving user-specified names for the clusters/groups.
`printPar`	If `TRUE` (default) a LaTeX-style table containing the posterior expectation of the cluster-specific parameter matrices $\mathbf{e}_h$ is also printed.

Value

A 3-dim array containing the posterior expectation of the cluster-specific parameter matrices $\mathbf{e}_h$.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

dmClust, dmClustExtended

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended
```

---

calcRegCoeffs — *Calculates Posterior Expectations, Standard Deviations and (Optionally) HPD Intervals for the MNL Regression Coefficients*

Description

Calculates posterior expectations, standard deviations and (optional) highest probability density (HPD) intervals for the multinomial logit (MNL) regression coefficients (using boa.hpd from package **boa**) and also offers some other analyses like plotting paths and autocorrelation functions (ACFs) for the corresponding MCMC draws.

Usage

```
calcRegCoeffs(outList, hBase = 1, thin = 1, M0 = outList$Mcmc$M0,
              grLabels = paste("Group", 1:outList$Prior$H),
              printHPD = TRUE, plotPaths = TRUE, plotACFs = TRUE)
```

Arguments

outList	specifies a list containing the outcome (return value) of an MCMC run of mcClustExtended, dmClustExtended or MNLAuxMix.
hBase	specifies the cluster/group which should serve as *baseline* cluster/group.
thin	An integer specifying the thinning parameter (default is 1).
M0	specifies the number of the first MCMC draw after burn-in (default is outList$Mcmc$M0).

`grLabels` A character vector giving user-specified names for the clusters/groups.

`printHPD` If `TRUE` (default) a LaTeX-style table containing the highest probability density (HPD) intervals for each MNL regression coefficient is calculated (using `boa.hpd` from package **boa**) and also printed.

`plotPaths` If `TRUE` (default) the paths of the MCMC draws of the MNL regression coefficients are drawn for each cluster/group (without thinning).

`plotACFs` If `TRUE` (default) the autocorrelation function (ACF) for the MCMC draws of the regression coefficients are drawn for each cluster/group (with thinning and burn-in discarded).

Value

A list containing:

`[[h]], h=1,..,H`
A matrix containing posterior expectation (`"Post Exp"`), standard deviation (`"Post Sd"`) and HPD interval (`"HPD Lower B", "HPD Upper B"`) for the MNL regression coefficients in cluster/group h except for the baseline cluster/group.

`regCoeffsAll` A matrix containing posterior expectation (`"Post Exp"`) and (in parenthesis) standard deviation (`"Post Sd"`) for the MNL regression coefficients for all clusters/groups.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

`boa.hpd`, `acf`, `mcClustExtended`, `dmClustExtended`, `MNLAuxMix`

Examples

```
# please run the examples in mcClustExtended, dmClustExtended and
# MNLAuxMix
```

calcSegmentationPower *Calculates the 'Segmentation Power' of the Specified Classification*

Description

Calculates the 'segmentation power' and optionally the 'sharpness' of the specified classification. The 'segmentation power' corresponds to the *maximum* individual posterior classification probability. The closer the *maximum* individual posterior classification probability is to 1, the higher is the segmentation power for individual i. Note that one minus these numbers corresponds to the *misclassification risk* in each group; hence the closer to one, the smaller is the misclassification risk.

The 'sharpness' on the other hand considers the difference between highest (maximum) and second highest individual posterior classification probabilities, which gives some hints about the 'sharpness' of the classification.

Usage

```
calcSegmentationPower(outList, classProbs, class,
            printXtable = TRUE, calcSharp = TRUE,
            printSharpXtable = TRUE,
            grLabels = paste("Group", 1:outList$Prior$H))
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended` or `MNLAuxMix`.
`classProbs`	A matrix with dimension $N \times H$ containing the individual posterior classification probabilities returned by `calcAllocations`.
`class`	A vector of length N containing the group membership returned by `calcAllocations`.
`printXtable`	If `TRUE` (default) a LaTeX-style table of the segmentation power is generated/printed.
`calcSharp`	If `TRUE` (default) also the 'sharpness' is calculated.
`printSharpXtable`	If `TRUE` (default) the 'sharpness' is also printed (provided that `calcSharp=TRUE`).
`grLabels`	A character vector giving user-specified names for the clusters/groups.

Details

Reported are summary statistics including the quartiles and the median of the distributions of the segmentation power and the 'sharpness' for all individuals within a certain cluster/group as well as for all individuals.

Value

A list containing:

segPowTab
: A matrix containing the segmentation power: reported are summary statistics of the distribution of the maximum individual posterior classification probabilities for all individuals within a certain cluster as well as for all individuals.

sharpTab
: A matrix containing the 'sharpness': reported are summary statistics of the difference between highest and second highest individual posterior classification probabilities within groups and overall.

maxProbs
: A vector containing the *maximum* individual posterior classification probabilities.

sharp
: A vector containing the differences of the individual maximum and the second highest posterior classification probabilities.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

calcAllocations, mcClust, dmClust, mcClustExtended, dmClustExtended, MNLAuxMix

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended, MNLAuxMix
```

calcTransProbs *Calculates the Posterior Expectation and Standard Deviations of the Average Cluster-Specific Transition Matrices*

Description

Calculates the posterior expectation and standard deviations of the average cluster-specific transition matrices and also offers some other analyses like plotting paths of MCMC draws.

Usage

```
calcTransProbs(outList, estGroupSize, thin = 1, M0 = outList$Mcmc$M0,
               grLabels = paste("Group", 1:outList$Prior$H),
               printXtable = FALSE, printSd = FALSE,
               printTogether = TRUE, plotPaths = TRUE,
               plotPathsForE = TRUE)
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended` or `dmClustExtended`.
`estGroupSize`	A vector of dimension H containing the (estimated) group sizes returned by `calcAllocations`.
`thin`	An integer specifying the thinning parameter (default is 1).
`M0`	specifies the number of the first MCMC draw after burn-in (default is `outList$Mcmc$M0`).
`grLabels`	A character vector giving user-specified names for the clusters/groups.
`printXtable`	If `TRUE` a LaTeX-style table containing the posterior expectation of the average cluster-specific transition matrices of each cluster/group is generated/printed.
`printSd`	If `TRUE` a LaTeX-style table containing the posterior standard deviations (multiplied by 100) of the average cluster-specific transition matrices of each cluster/group is generated/printed.
`printTogether`	If `TRUE` (default) a LaTeX-style table containing the posterior expectation and standard deviations (multiplied by 100) of the average cluster-specific transition matrices of each cluster/group is generated/printed.
`plotPaths`	If `TRUE` (default) the paths of the MCMC draws of the transition probabilities $\xi_{h,j,k}$ are drawn for each cluster/group.
`plotPathsForE`	If `TRUE` (default) the paths of the MCMC draws of the transition parameters $e_{h,j,k}$ are drawn for each cluster/group (only DMC[Ext]).

Value

A list containing:

`estTransProb`	A 3-dim array containing the posterior expectation of the average transition matrices of all clusters/groups using each `thin`-th draw from `M0` to `M`.

`estTransProbSd`
: A 3-dim array containing the posterior standard deviations of the average transition matrices for each cluster/group.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

`calcAllocations`, `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended`

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended
```

`calcVariationDMC` *Analyses How Much Unobserved Heterogeneity Is Present in the Various Clusters by Computing the Within-Group Variability of the Cluster-Specific Transition Parameters of DMC*

Description

Calculates the posterior expectation of the variance of the individual transition probabilities as well as posterior expectation and standard deviation of the row-specific unobserved heterogeneity measure in each group to analyse how much *unobserved heterogeneity* is present in the various clusters (see Pamminger and Fruehwirth-Schnatter (2010) in **References**).

Usage

```
calcVariationDMC(outList, thin = 1, maxi = 50, M0 = outList$Mcmc$M0,
                 grLabels = paste("Group", 1:outList$Prior$H),
                 printVarE = FALSE, printUnobsHet = FALSE,
                 printUnobsHetSd = FALSE, printUnobsHetAll = FALSE,
                 printAllTogether = TRUE)
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `dmClust` or `dmClustExtended`.
`thin`	An integer specifying the thinning parameter (default is 1).
`maxi`	specifies the number of draws to be actually taken (after thinning) from the MCMC draws beginning from the end of the chain (default is 50).
`M0`	specifies the number of the first MCMC draw after burn-in (default is `outList$Mcmc$M0`).
`grLabels`	A character vector giving user-specified names for the clusters/groups.
`printVarE`	If `TRUE` a LaTeX-style table of the posterior expectation of the variance of the individual transition probabilities (in percent) in each cluster/group is generated/printed.
`printUnobsHet`	If `TRUE` a LaTeX-style table of the posterior expectation of the row-specific unobserved heterogeneity measure in each group multiplied by 100 is generated/printed.
`printUnobsHetSd`	If `TRUE` a LaTeX-style table of the posterior standard deviation of the row-specific unobserved heterogeneity measure in each group multiplied by 100 is generated/printed.
`printUnobsHetAll`	If `TRUE` a LaTeX-style table of the posterior expectation and, in parenthesis, posterior standard deviation of the row-specific unobserved heterogeneity measure in each group multiplied by 100 is generated/printed.
`printAllTogether`	If `TRUE` (default) a LaTeX-style table of the posterior expectation of the variance of the individual transition probabilities (in percent) in each cluster/group as well as the posterior expectation and, in parenthesis, posterior standard deviation of the row-specific unobserved heterogeneity measure in each group multiplied by 100 is generated/printed.

Details

The last `maxi` MCMC draws of each `thin`-th draw are taken for calculations.

Value

A list containing:

`var_e`	A 3-dim array containing the posterior expectation of the variance of the individual transition probabilities in each group.

`het` A matrix containing the posterior expectation of the row-specific unobserved heterogeneity measure in each group.

`hetsd` A matrix containing the posterior standard deviation of the row-specific unobserved heterogeneity measure in each group.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 `http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract`

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 `http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf`

See Also

`dmClust`, `dmClustExtended`

Examples

```
# please run the examples in dmClust, dmClustExtended
```

---

dmClustering *Dirichlet Multinomial Clustering With And Without Mixtures-of-Experts Extension*

Description

This function provides Dirichlet Multinomial Clustering with or without multinomial logit model (mixtures-of-experts) extension (see **References**). That is an MCMC sampler for the mixtures-of-experts extension of Dirichlet Multinomial clustering. It requires four mandatory arguments: `Data`, `Prior`, `Initial` and `Mcmc`; each representing a list of (mandatory) arguments: `Data` contains data information, `Prior` contains prior information, `Initial` contains information about starting conditions (initial values) and `Mcmc` contains the setup for the MCMC sampler.

Usage

```
dmClust(
    Data = list(
        dataFile =
            stop("'dataFile' must be specified: filename or data"),
        storeDir = "try01", mccFile = "mcc.RData"),
    Prior = list( H = 4, alpha0 = 4, a0 = 1, alpha = 1, N0 = 10,
        isPriorNegBin = FALSE, mccAsPrior = FALSE,
        xiPooled = TRUE, persPrior = 7/10),
    Initial = list( mccUse = FALSE, pers = 1/6, S.i.start = NULL),
    Mcmc = list( kNo = 2, M = 50, M0 = 20, mOut = 5, mSave = 10,
        showAcc = TRUE, monitor = FALSE, seed = 12345))

dmClustExtended(
    Data = list(
        dataFile =
            stop("'dataFile' must be specified: filename or data"),
        storeDir = "try01", mccFile = "mcc.RData",
        X = stop("X (matrix of covariates) must be specified")),
    Prior = list( H = 4, a0 = 1, alpha = 1, N0 = 10,
        isPriorNegBin = FALSE, mccAsPrior = FALSE,
        xiPooled = TRUE, persPrior = 7/10,
        betaPrior = "informative", betaPriorMean = 0,
        betaPriorVar = 1),
    Initial = list( mccUse = FALSE, pers = 1/6,
        S.i.start = rep(1:H, N), Beta.start = NULL),
    Mcmc = list( kNo = 2, M = 50, M0 = 20, mOut = 5, mSave = 10,
        showAcc = TRUE, monitor = FALSE, seed = 12345))
```

Arguments

`Data`	a list consisting of: `dataFile, storeDir, mccFile, X`. See **Details**.
`Prior`	a list consisting of: `H, alpha0, a0, alpha, N0, isPriorNegBin, mccAsPrior, xiP` See **Details**.
`Initial`	a list consisting of: `mccUse, pers, S.i.start, Beta.start`. See **Details**.
`Mcmc`	a list consisting of: `kNo, M, M0, mOut, mSave, showAcc, monitor, seed`. See **Details**.

Details

Note that the values of the arguments indicated here have nothing to do with *default values*! For a call of these functions this lists-of-arguments structure requires a complete specification of all arguments!

The following arguments which are lists have to be completely provided (note that there are no such things as default values within lists!):

`Data` contains:

`dataFile` A 3-dim array having the transition counts/frequencies structure (like `Njk.i` in the example data sets) already loaded into the current environment/workspace. Or a character with the name of or the path to an .RData-*file* which contains such a data set, in which case it must have the name "`Njk.i`".

It is required that this data have to be a 3-dimensional array of dimension $(K+1)\times(K+1)\times N$ containing the transition counts/frequencies, where $K+1$ is the number of categories $k = 0,\ldots,K$ and N the number of objects/units/individuals. The number of transitions (equal to time series length minus one) may be individual.

`storeDir` A character indicating the name of the directory (will be created if not already existing) where the log file and the results are to be stored.

`mccFile` If not `NULL` the prior data (must have same format as `mccXiPrior` in `LMEntryPaperData` – at least the H-th entry in the list has to be provided) or a character with the name of or the path to a file containing such data, which in this case must be named "`mcc`". The prior data contain prior information (in terms of probabilities) about transition probabilities (possibly from another estimation procedure). For further information see Section **Prior Data** and `mccXiPrior` in `LMEntryPaperData`.

`X` The matrix of covariates (with N rows) including the unit vector for the intercept to be included in the multinomial logit model extension.

`Prior` contains (see also Section **Prior Data**):

`H` An integer ≥ 1 indicating the number of clusters/groups.

`alpha0` A numerical value determining the value of the prior parameter of the Dirichlet-prior for the group sizes η_h (`alpha0` $= \alpha_1 = \ldots = \alpha_H$, thus equal for all h).

`a0` A numerical value determining a parameter of the negative multinomial prior (see references for more details).

`alpha` A numerical value determining a parameter of the negative multinomial prior (see references for more details).

`N0` A numerical value determining a parameter of the negative multinomial prior (see references for more details).

`isPriorNegBin` If `TRUE`, the product of negative binomial distributions is used instead of the negative multinomial distribution (see references for more details).

`mccAsPrior` If `mccAsPrior=TRUE`, prior information for the transition probabilities as provided by `mccFile` are used as prior parameters for the estimation process. In this case there are two further options depending on the value of `xiPooled`: If `xiPooled=TRUE`, equal apriori transition probabilities are used for all groups (using `mcc[[1]]$xi`) and if `xiPooled=FALSE` group-specific apriori transition probabilities are used (using `mcc[[H]]$xi`).

If `mccAsPrior=FALSE`, a priori transition probabilities are determined depending on `persPrior`. In this case the diagonal elements are set to `persPrior` and the off-diagonal elements to (1 - `persPrior`)/K, equal for all groups.

`xiPooled` Only used if `mccAsPrior=TRUE` (see above): if `xiPooled=TRUE` equal apriori transition probabilities are used for all groups (using `mcc[[1]]$xi`) and if `xiPooled=FALSE` group-specific apriori transition probabilities are used (using
`mcc[[H]]$xi`).

persPrior Only used if mccAsPrior=FALSE: a numerical value (between 0 and 1) indicates the persistence probability (equal for all diagonal elements) for the a priori transition probabilities. 1/(K + 1) corresponds to uniform distribution in each row.

betaPrior A character. If "uninformative" (improper) prior parameters are used for the regression coefficients (i.e. betaPriorVar = ∞). Otherwise mean and variance of the normal prior distribution for the regression coefficients have to be specified.

betaPriorMean, betaPriorVar Numerical values specifying the parameters of the normal prior distribution for the regression coefficients, only if betaPrior!="uninformative".

Initial contains:

mccUse If TRUE, prior information for the group sizes and the transition probabilities as provided with mccFile are used for the estimation process as initial values. If FALSE, initial values for group sizes are set to $1/H$ and for transition probabilities determined by use of pers for the diagonal elements and (1 - pers)/K for the off-diagonal elements.

pers Only used if mccUse=FALSE: A numerical value (between 0 and 1) which indicates the persistence probabilities (equal for all diagonal elements). Note, that $1/(K+1)$ corresponds to the uniform distribution in each row.

S.i.start A vector of length N giving an initial allocation (mandatory for dmClustExtended).

Beta.start A matrix of dimension ncol(X) x H giving start values for the regression coefficients including the zero vector in the first column representing the baseline group.

Mcmc contains:

kNo A numerical value between 1 and $K+1$ indicating the number of row elements to be updated in each iteration. Note that eventually notation l is used in the literature.

M An integer indicating the overall number of iterations.

M0 An integer indicating the number of the first iteration *after* the burn-in phase.

mOut An integer indicating that after each mOut-th iteration a report line is written to the output window/screen.

mSave An integer indicating that after each mSave-th iteration an intermediate storage of the workspace is carried out.

showAcc If TRUE, additionally the current acceptance rate of the recent mOut draws of the M-H-steps is shown in the log-file and on the screen. Rule of thumb for the acceptance probability: should be around 0.25, at least between 0.15 and 0.4.

monitor If TRUE, the paths of the draws of $\mathbf{e}_h$ and $\boldsymbol{\xi}_h$ starting at the beginning ($m = 1$) up to the current draws are shown and currently updated in a diagram.

seed An integer indicating a random seed.

Value

A list containing (/the output file contains):

workspaceFile A character indicating the name of and the path (based on the currend working directory) to the output file, wherein all the results are saved. The name of the output file starts with “DMC_” or “DMC_Logit_newAux_” respectively followed by the number of groups `H`, the number of iterations `M` and the particular point in time when the function was called, with format: `yyyymmdd_hhmmss`. E.g. `DMC_H4_M10000_20110218_045254.RData` or `DMC_Logit_newAux_H4_M10000_20111121_165723.RData`.

`accept` A 3-dimensional array with dimension $M \times H * (K+1) \times 2$. This array contains the (calculated) acceptance probabilities (`accProb`) of the M-H-algorithm and whether the draw(s) were accepted or not (`accYesNo`) for each row j in each group h in the m-th iteration. The first dimension indicates the m-th iteration, the second dim row $1, \ldots, K+1$ in group 1, then row $1, \ldots, K+1$ in group 2 and so on. The third dim indicates `accProb` and `accYesNo`.

`Beta.m` A 3-dimensional array of dimension `ncol(X)` $\times H \times M$ containing the draws for the regression coefficients β_h in each m-th iteration step.

`bk0` The prior parameters for the mean vectors of the normal (prior) distributions of the regression coefficients.

`Bk0inv` The prior parameters for the inverse variance-covariance matrices of the normal (prior) distributions of the regression coefficients.

`Data` The argument `Data`.

`e_h_0` A 3-dimensional array with dimension $K+1 \times K+1 \times H$ containing the (calculated) initial values for $\boldsymbol{e}_h$.

`e_h_m` A 4-dimensional array with dimension $K+1 \times K+1 \times H \times M$ containing the draws for $\boldsymbol{e}_h$ in the m-th iteration step.

`eta_m` A matrix of dimension $M \times H$ containing the draws for η_h in each m-th iteration step.

`fileName` A character value indicating the name of the output file (see also `workspaceFile`).

`Initial` The argument `Initial`.

`K` An integer indicating the number of categories minus one (!). See **Note**.

`logFileName` A character value indicating the name of the log file and the corresponding directory.

`mcc` The prior data (see Section **Prior Data**) provided with `mccFile`, `NULL` otherwise.

`Mcmc` The argument `Mcmc`.

`N` An integer indicating N, the number of individuals/units/objects.

`Njk.i` The data (see **Details**) provided with `dataFile`.

`Prior` The argument `Prior`.

`S_i_freq` A $H \times N$-matrix containing the frequencies how often individual i was allocated to a certain group during the iterations from `M0+1` to codeM.

`xi_h_m` A 4-dimensional array of dimension $(K+1) \times (K+1) \times H \times M$ containing the draws for $\boldsymbol{\xi}_h$ in each m-th iteration step.

`xi_prior` A 3-dimensional array of dimension $(K+1) \times (K+1) \times H$ that contains the finally used a priori parameter values for $\boldsymbol{\xi}_h$.

`bkN` The posterior parameters (in the last iteration step) for the mean vectors of the normal (posterior) distributions from which the regression coefficients were drawn.

BkN	The posterior parameters (in the last iteration step) for the variance-covariance matrices of the normal (posterior) distributions from which the regression coefficients were drawn.
logLike	A vector containing the values of the log-likelihood calculated in each iteration step.
logBetaPrior	A vector containing the values of the prior distribution for the regression coefficients calculated in each iteration step.
logEPrior	A vector containing the values of the prior distribution for e calculated in each iteration step.
logPostDens	A vector containing the values of the posterior density calculated in each iteration step.
mMax	An integer giving the position (number of iteration) of the maximum value in the posterior density `logPostDens`.
logClassLike	A vector containing the values of the log classification likelihood calculated in each iteration step.
entropy	A vector containing the values of the entropy calculated in each iteration step.
logEtaPrior	A vector containing the values of the prior distribution for the mixing proportions (group sizes) calculated in each iteration step.

Prior Data

The prior data (called `mcc` in the following) – to be passed via `mccFile` in argument-list `Data` – has to be a list of lists, indexed by $1, \ldots, H, H+1, \ldots$. Note that, depending on parameter H (the number of groups – to be passed via `H` in argument-list `Prior`), there have to be at least H entries (each a list). See `mccXiPrior` in `LMEntryPaperData` for example. Within a call to `dmClustering` or `mcClustering`, at least `mcc[[H]]` has to be provided as a list containing `eta` and `xi`. `eta` is a vector of length H containing prior information about the relative group sizes of group $h = 1, \ldots, H$. `xi` is a 3-dimensional array of dimension $(K+1) \times (K+1) \times H$, containing prior information in terms of probabilities about the transition probabilities of group $h = 1, \ldots, H$ (see examples).

Log File

The log file keeps record of the progress of the estimation procedure (which is also shown on the screen). At first some prior parameters and the MCMC-settings and the name of the output file are documented. Then for each `mOut`-th iteration step (at least for $m = 1, \ldots, 5, 10, 20, 50, 100, 200, 500$) information about the elapsed time and the expected time to the end and optionally the current acceptance rate (`showAcc=TRUE`) is indicated. Finally the total time is shown.

For example:

```
Data loaded!
Data Information: Datafile = no file name , N = 9809 , K = 5
Manual Settings: No of groups H = 4 , kNo = 2
MCMC Parameters: M = 10000 , M0 = 5000 , mOut = 200 , mSave = 5000 , seed = 123456 , showAcc = TRUE
Prior Parameters for e_h (Neg Multinom): a0 = 1 , alpha = 1 , N0 = 10 , xi_prior (see below)
Information on xi_prior (for Neg Bin/Neg Multinom Prior):  with persPrior =  0.7  created xi_prior (equ
Prior information and parameters set!
```

```
Inital Values Information: mccUse = FALSE , pers = 0.7
Initial values set!
Initialisations done!
MCMC Iteration...
m = 1 ; Acc Rate of first draws = 0.54
m = 2 ; duration of iter proc so far: 8.17 sec. ,  exp time to end: 1361.53  min. ; Acc Rate of last 2 dr
m = 3 ; duration of iter proc so far: 16.45 sec. ,  exp time to end: 1370.56  min. ; Acc Rate of last 3 d
m = 4 ; duration of iter proc so far: 24.62 sec. ,  exp time to end: 1367.37  min. ; Acc Rate of last 4 d
m = 5 ; duration of iter proc so far: 32.84 sec. ,  exp time to end: 1367.79  min. ; Acc Rate of last 5 d
m = 10 ; duration of iter proc so far: 73.97 sec. ,  exp time to end: 1368.58  min. ; Acc Rate of last 10
m = 20 ; duration of iter proc so far: 156.61 sec. ,  exp time to end: 1371.16  min. ; Acc Rate of last 2
m = 50 ; duration of iter proc so far: 404.42 sec. ,  exp time to end: 1368.84  min. ; Acc Rate of last 5
m = 100 ; duration of iter proc so far: 815.86 sec. ,  exp time to end: 1359.9  min. ; Acc Rate of last 1
m = 200 ; duration of iter proc so far: 1635.61 sec. ,  exp time to end: 1342.6  min. ; Acc Rate of last
m = 400 ; duration of iter proc so far: 3270.83 sec. ,  exp time to end: 1311.75  min. ; Acc Rate of las
m = 500 ; duration of iter proc so far: 4087.97 sec. ,  exp time to end: 1297.25  min. ; Acc Rate of las
m = 1000 ; duration of iter proc so far: 8165.91 sec. ,  exp time to end: 1226.25  min. ; Acc Rate of la
...
m = 10000 ; duration of iter proc so far: 81362.58 sec. ,  exp time to end: 0.14  min. ; Acc Rate of las
Total time:  22 hours 36 min
```

Warning

Note that there are no such things as *default* values (see Section **Arguments**)!

Note

Note that the required data files have to be provided in the current working directory and that the results (see Section **Value**) are to be saved in the directory provided by `storeDir` within the current working directory. Make sure that the current working directory is set appropriately before the function is called.

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 `http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract`

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 `http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf`

Sylvia Fruehwirth-Schnatter and Rudolf Fruehwirth, (2010), "Data augmentation and MCMC for binary and multinomial logit models". In T. Kneib and G. Tutz (eds): *Statistical Modelling and Regression Structures: Festschrift in Honour of Ludwig Fahrmeir*. Physica Verlag, Heidelberg, pp. 111-132. DOI: 10.1007/978-3-7908-2413-1_7 http://www.springerlink.com/content/t4h810017645wh68/. See also: IFAS Research Paper Series 2010-48 (http://www.jku.at/ifas/content/e108280/e108491/e108471/e109880/ifas_rp48.pdf).

See Also

mcClust, mcClustExtended, MNLAuxMix, MCCExampleData, MCCExtExampleData

Examples

```
#rm(list=ls(all=TRUE))

# ==================================================================================
if ( FALSE ) {
# ==================================================================================

# set working directory
oldDir <- getwd()
curDir <- tempdir()
setwd(curDir)

if ( !file.exists("bayesMCClust-wd") ) dir.create("bayesMCClust-wd")
setwd("bayesMCClust-wd")
myOutfilesDir <- "dmClust-Example-Outfiles"

# load data
data(MCCExampleData)

# function call
system.time(
  outList <- dmClust(   # parameter lists (every four) must be complete!
     Data = list( dataFile=MCCExampleData$Njk.i,
                  storeDir=myOutfilesDir,
                  mccFile=MCCExampleData$somePrior),
     Prior   = list( H=2, # sample(2:5, 1), # 3
                  alpha0=4,
                  a0=1,
                  alpha=1,
                  N0=10,
                  isPriorNegBin=FALSE,
                  mccAsPrior=TRUE,
                  xiPooled=FALSE,
                  persPrior=0.7),
     Initial = list( mccUse=FALSE,
                  pers=1/3 ),
     Mcmc    = list( kNo=2,
                  M=100,
                  M0=20,
```

```
                        mOut=5,
                        mSave=50,
                        showAcc=TRUE,
                        monitor=FALSE,
                        seed=sample(1:100000, 1) # 12345
        )
    )
)

str(outList)

#outFileName
#results <- load(outFileName)
#results

if (outList$Prior$H > 1) {
    apply(outList$xi_h_m[,,,seq(outList$Mcmc$M0, outList$Mcmc$M, 1)], c(1,2,3), mean)
    } else {
    apply(outList$xi_h_m[,,,seq(outList$Mcmc$M0,outList$Mcmc$M,1)], c(1, 2), mean)
}

allocList <- calcAllocationsDMC(outList, thin=1, maxi=50) # , plotPathsForEta=TRUE
str(allocList)

myTransProbs <- calcTransProbs(outList, estGroupSize=allocList$estGroupSize, thin=1,
    printXtable=FALSE, printSd=FALSE, printTogether=TRUE )
    # grLabels=paste("Group", 1:Prior$H), plotPaths=TRUE
str(myTransProbs)

myTransList <- plotTransProbs(outList, estTransProb=myTransProbs$estTransProb,
    estGroupSize=allocList$estGroupSize, class=allocList$class, plotPooled=TRUE,
    plotContTable=TRUE, printContTable=TRUE, plotContPooled=TRUE)
    # , grLabels=paste("Group", 1:Prior$H)
str(myTransList)

(equiDist <- calcEquiDist(outList, thin=1, maxi=50))
# , printEquiDist=TRUE, plotEquiDist=TRUE, grLabels=paste("Group", 1:Prior$H)

myVariation <- calcVariationDMC(outList, thin=1, maxi=50)
# , printVarE=TRUE, printUnobsHet=TRUE, printUnobsHetSd=TRUE,
# printUnobsHetAll=TRUE, printAllTogether=TRUE, grLabels=paste("Group", 1:Prior$H)
str(myVariation)

myPars <- calcParMatDMC(outList, thin=1)
# , grLabels=paste("Group", 1:Prior$H), printPar=TRUE
str(myPars)

myLongRunDistList <- calcLongRunDist(outList,
    initialStateData=MCCExampleData$initialState,
    class=allocList$class, equiDist=equiDist, thin=1, maxi=5)
    # , printLongRunDist=TRUE, , grLabels=paste("Group", 1:Prior$H)
str(myLongRunDistList)
```

```
myTypicalMembs <- plotTypicalMembers(outList, moreTypMemb=c(10,13,17,20,23,27,30),
    myObsList=MCCExampleData$obsList, classProbs=allocList$classProbs)
    # , noTypMemb=7, moreTypMemb=c(10,25,50,100,200,500,1000)
str(myTypicalMembs)

plotScatter(outList, thin=1, xi11=c(1,1), xi12=c(2,2), xi21=c(2,2),
    xi22=c(3,3), xi31=c(1,1), xi32=c(3,3) )

mySegPower <- calcSegmentationPower(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE, calcSharp=TRUE, printSharpXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
str(mySegPower)

myEntropy <- calcEntropy(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
myEntropy

plotLikeliPaths(outList, from=10, by=1 )

myNumEffTables <- calcNumEff( outList, thin=1, printXi=TRUE, printE=TRUE,
    printBeta=TRUE, grLabels=paste("Group", 1:outList$Prior$H) )
str(myNumEffTables)

myMSCrits <- calcMSCritDMC(workDir=myOutfilesDir, myLabel="dmClust-Example",
    myN0=paste("N0 =",outList$Prior$N0),
    whatToDoList=c("postMode", "approxML", "approxMCL") )
str(myMSCrits)

setwd(oldDir)

} # end if

# ==================================================================================
# ==================================================================================
# ==================================================================================

# ==================================================================================
if ( FALSE ) {
# ==================================================================================

rm(list=ls(all=TRUE))

# set working directory
oldDir <- getwd()
curDir <- tempdir()
setwd(curDir)

if ( !file.exists("bayesMCClust-wd") ) dir.create("bayesMCClust-wd")
setwd("bayesMCClust-wd")
myOutfilesDir <- "dmClustExtended-Example-Outfiles"

# load data
```

```
data(MCCExtExampleData)
if (!is.element("MCCExtExampleData$covariates", search())) {
    attach(MCCExtExampleData$covariates)
}

# ====================================================================================

groupNr <- 2 # sample(2:6, 1) # 3

# ====================================================================================

results <- kmeans( log( MCCExtExampleData$NjkiMat + 0.5 ) , groupNr, nstart=2)

# ====================================================================================

require(nnet, quietly = TRUE)
H <- groupNr
X = cbind( intercept=1, alrateBezNew, unskilled, skilled, angStart )

N <- dim(X)[1]
mX <- data.frame( cbind(group=as.factor( results$cluster ), X[,-1],
    matrix(sample(1:H,H*N,replace=TRUE),N,H)) )

colnames(mX)[6:(6+groupNr-1)] <-
    c( "as.1", "as.2", "as.3", "as.4", "as.5", "as.6" )[1:groupNr]

tempMNom <- multinom(group ~ alrateBezNew+ unskilled+ skilled+ angStart,
    data=as.data.frame(mX))

toStartBeta <- t(rbind(0,coef( tempMNom )))

outList <- dmClustExtended(
     Data = list( dataFile=MCCExtExampleData$Njk.i,
                  storeDir=myOutfilesDir,
                  mccFile=NULL,
                  X = cbind(intercept=1, alrateBezNew, unskilled, skilled, angStart )),
     Prior   = list( H=groupNr,
                     a0=1,
                     alpha=1,
                     N0=10,
                     isPriorNegBin=FALSE,
                     mccAsPrior=FALSE,
                     xiPooled=FALSE,
                     persPrior=0.7,
                     betaPrior = "informative", # N(0,1)
                     betaPriorMean = 0,
                     betaPriorVar = 1),
     Initial = list( mccUse=FALSE,
                     pers=1/3,
                     S.i.start = results$cluster,
                     Beta.start = toStartBeta ),
     Mcmc    = list( kNo=2,
                     M=100,
```

```
                     M0=50,
                     mOut=10,
                     mSave=50,
                     showAcc=TRUE,
                     monitor=FALSE,
                     seed=sample(1:100000, 1) # 564847
                   )
)

str(outList)

#outFileName <- outList$workspaceFile
#outFileName
#results <- load(outFileName)
#results

if (outList$Prior$H > 1) {
    apply( outList$xi_h_m[,,,seq(outList$Mcmc$M0, outList$Mcmc$M, 1)], c(1,2,3), mean)
    } else {
    apply(outList$xi_h_m[,,,seq(outList$Mcmc$M0,outList$Mcmc$M,1)], c(1, 2), mean)
}

allocList <- calcAllocationsDMCExt(outList, thin=1, maxi=50)
str(allocList)

myTransProbs <- calcTransProbs(outList, estGroupSize=allocList$estGroupSize, thin=1,
    printXtable=FALSE, printSd=FALSE, printTogether=TRUE )
    # grLabels=paste("Group", 1:Prior$H), plotPaths=TRUE
str(myTransProbs)

myTransList <- plotTransProbs(outList, estTransProb=myTransProbs$estTransProb,
    estGroupSize=allocList$estGroupSize, class=allocList$class, plotPooled=TRUE,
    plotContTable=TRUE, printContTable=TRUE, plotContPooled=TRUE)
    # , grLabels=paste("Group", 1:Prior$H)
str(myTransList)

(equiDist <- calcEquiDist(outList, thin=1, maxi=50))
# , printEquiDist=TRUE, plotEquiDist=TRUE, grLabels=paste("Group", 1:Prior$H)

myVariation <- calcVariationDMC(outList, thin=1, maxi=50)
# , printVarE=TRUE, printUnobsHet=TRUE, printUnobsHetSd=TRUE,
# printUnobsHetAll=TRUE, printAllTogether=TRUE, grLabels=paste("Group", 1:Prior$H)
str(myVariation)

myPars <- calcParMatDMC(outList, thin=1)
# , grLabels=paste("Group", 1:Prior$H), printPar=TRUE
str(myPars)

myRegCoeffs <- calcRegCoeffs(outList, hBase=2, thin=1)
#, M0=Mcmc$M0, grLabels=paste("Group", 1:Prior$H), printHPD=TRUE,
# plotPaths=TRUE, plotACFs=TRUE
str(myRegCoeffs)
```

```
myLongRunDistList <- calcLongRunDist(outList, initialStateData=initialState,
    class=allocList$class, equiDist=equiDist, maxi=2)
    # , printLongRunDist=TRUE
str(myLongRunDistList)

myTypicalMembs <- plotTypicalMembers(outList, myObsList=MCCExtExampleData$obsList,
    classProbs=allocList$classProbs)
    # , noTypMemb=7, moreTypMemb=c(10,25,50,100,200,500,1000)
str(myTypicalMembs)

plotScatter(outList, thin=1, xi11=c(1,1), xi12=c(2,2), xi21=c(2,2),
    xi22=c(3,3), xi31=c(1,1), xi32=c(3,3) )

mySegPower <- calcSegmentationPower(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE, calcSharp=TRUE,
    printSharpXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
str(mySegPower)

myEntropy <- calcEntropy(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
myEntropy

plotLikeliPaths(outList, from=10, by=1 )

myNumEffTables <- calcNumEff( outList, thin=1, printXi=TRUE, printE=TRUE,
    printBeta=TRUE, grLabels=paste("Group", 1:outList$Prior$H) )
str(myNumEffTables)

myMSCrits <- calcMSCritDMCExt(workDir=myOutfilesDir, myLabel="dmClustExtended-Example",
    myN0=paste("N0 =",outList$Prior$N0),
    whatToDoList=c("postMode", "approxML", "approxMCL") )
str(myMSCrits)

setwd(oldDir)

# ==================================================================================

if (is.element("MCCExtExampleData$covariates", search())) {
    detach(MCCExtExampleData$covariates)
}

# ==================================================================================
} # end if
# ==================================================================================

# ==================================================================================
```

LMEntryPaperData *Data From Fruehwirth-Schnatter et al. (2011): "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering"*

Description

The empirical analysis in Fruehwirth-Schnatter et al. (2011) is based on data from the Austrian Social Security Database (ASSD), which combines detailed longitudinal information on employment and earnings of all private sector workers in Austria since 1972 (see **References**). The IEW Working Paper Zweimueller et al. (2009) (see **Source**) gives an overview and a description of the main characteristics of the Austrian Social Security Database.

The ASSD was made available for the Austrian Center of Labor Economics and the Analysis of the Welfare State (http://www.labornrn.at/). The considered sample consists of $N = 49279$ male Austrian workers, who enter the labor market for the first time in the years 1975 to 1985 and are less than 25 years old at entry. The cohort analysis is based on an observation period from 1975 to 2005.

Usage

```
data(LMEntryPaperData)
```

Format

The format is:

```
List of 6
 $ InitValBetas: num [1:25, 1:4] 0 0 0 0 0 0 0 0 0 0 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:25] "intercept" "unEmplRDist" "unskilled" "skilled" ...
  .. ..$ : chr [1:4] "h1" "h2" "h3" "h4"
 $ InitValClass: int [1:49279] 2 3 1 4 3 2 3 2 4 1 ...
 $ covariates  :'data.frame':   49279 obs. of  25 variables:
  ..$ intercept    : num [1:49279] 1 1 1 1 1 1 1 1 1 1 ...
  ..$ unEmplRDist  : num [1:49279] 0.91 0.697 0.905 0.91 1.051 ...
  ..$ unskilled    : num [1:49279] 0 0 0 0 0 0 0 1 0 0 ...
  ..$ skilled      : num [1:49279] 0 1 1 1 0 0 0 0 1 0 ...
  ..$ whiteColl    : num [1:49279] 0 0 1 0 1 0 0 1 1 1 ...
  ..$ wageCat1Dummy: num [1:49279] 1 1 1 0 0 1 1 1 0 0 ...
  ..$ wageCat2Dummy: num [1:49279] 0 0 0 0 1 0 0 0 0 1 ...
  ..$ wageCat3Dummy: num [1:49279] 0 0 0 1 0 0 0 0 1 0 ...
  ..$ wageCat4Dummy: num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ wageCat5Dummy: num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear76  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear77  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear78  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear79  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
```

```
  ..$ entryYear80  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear81  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear82  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear83  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear84  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ entryYear85  : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ ia.ueRD.wc1D : num [1:49279] 0.91 0.697 0.905 0 0 ...
  ..$ ia.ueRD.wc2D : num [1:49279] 0 0 0 0 1.05 ...
  ..$ ia.ueRD.wc3D : num [1:49279] 0 0 0 0.91 0 ...
  ..$ ia.ueRD.wc4D : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ ia.ueRD.wc5D : num [1:49279] 0 0 0 0 0 0 0 0 0 0 ...
 $ mccXiPrior  :List of 1
  ..$ :List of 1
  .. ..$ xi: num [1:6, 1:6] 0.7 0.15 0.0333 0.0333 0.0333 ...
 $ NjkiMat     : num [1:49279, 1:36] 0 0 0 2 7 0 4 0 0 1 ...
 $ Njk.i       : num [1:6, 1:6, 1:49279] 0 0 0 0 0 0 0 1 1 0 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. ..$ : NULL
```

Details

LMEntryPaperData is a list containing the following objects:

InitValBetas contains a matrix with the initial values (used in our paper) for the logit regression coefficients.

InitValClass contains a vector with some initial values (used in our paper) for the classification variable (group membership for 4 groups).

covariates contains the data.frame with the covariates used in the logit regression model. It contains the following variables:

unEmplRDist	unemployment rate in the district
unskilled	dummy for unskilled workers
skilled	dummy for skilled workers
whiteColl	dummy for white collar workers
wageCat1Dummy,..., wageCat5Dummy	dummies for starting in the corresponding wage category
entryYear76,..., entryYear85	dummies for starting in the corresponding year
ia.ueRD.wc1D,..., ia.ueRD.wc5D	interaction variable for unemployment rate in the district and the dummies for starting in the corresponding wage category

mccXiPrior contains the prior-parameters (used in the paper) for the transition matrices.

NjkiMat contains the Njk.i-data in matrix format of dimension 49279×36 (each row corresponds to the columns of the matrices in `Njk.i`).

`Njk.i` contains the transition frequencies in a 3-dim array of dimension $6 \times 6 \times 49279$ containing the transition frequencies (6×6-matrices) of 49279 individuals. These represent the counts of transitions between wage categories from year to year with varying observation periods. Categories 1 to 5 correspond to the wage quintiles and 0 to no income.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Source

The following IEW Working Paper gives an overview and a description of the main characteristics of the Austrian Social Security Database:

Zweimueller, Josef, Winter-Ebmer, Rudolf, Lalive, Rafael, Kuhn, Andreas, Wuellrich, Jean-Philippe, Ruf, Oliver and Buechi, Simon, Austrian Social Security Database (May 4, 2009). Available at SSRN: http://ssrn.com/abstract=1399350 or at http://www.labornrn.at/wp/wp0903.pdf.

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Link to Journal of Applied Econometrics Data Archive: http://econ.queensu.ca/jae/forthcoming/fruehwirth-schnatter-et-al/

See Also

mcClustExtended

Examples

```
data(LMEntryPaperData)
str(LMEntryPaperData)

# =====================   LMEntry Paper Data   =================================
#rm(list=ls(all=TRUE))

# set working directory
curDir <- getwd()

if ( !file.exists("bayesMCClust-wd") ) dir.create("bayesMCClust-wd")
setwd("bayesMCClust-wd")
myOutfilesDir <- "LMEntry-Paper-Data-Outfiles"
# ==============================================================================
if (!is.element("LMEntryPaperData$covariates", search())) {
```

```
    attach(LMEntryPaperData$covariates)
}
# ================================================================================
groupNr <- 4
# ================================================================================
if ( FALSE ) {
  try(mcClustExtended(      # parameter lists (all four) must be complete!!!
     Data=list(dataFile=LMEntryPaperData$Njk.i,
               storeDir=myOutfilesDir,
               priorFile= LMEntryPaperData$mccXiPrior,
               X = cbind( intercept=1, unEmplRDist, unskilled, skilled, whiteColl,
                                    wageCat1Dummy, wageCat2Dummy, wageCat3Dummy,
                                    wageCat4Dummy, wageCat5Dummy,
                                    entryYear76, entryYear77, entryYear78,
                                    entryYear79, entryYear80, entryYear81,
                                    entryYear82, entryYear83, entryYear84,
                                    entryYear85,
                                    ia.ueRD.wc1D, ia.ueRD.wc2D, ia.ueRD.wc3D,
                                    ia.ueRD.wc4D, ia.ueRD.wc5D
                        ) ),
     Prior=list(H=groupNr,
                c=1,
                cOff=1,
                usePriorFile=TRUE,
                xiPooled=TRUE,
                N0=10,
                betaPrior = "informative", # N(0,1)
                betaPriorMean = 0,
                betaPriorVar = 1),
     Initial=list(xi.start.ind=3,
                  pers=0.7,
                  S.i.start = LMEntryPaperData$InitValClass,
                  Beta.start = LMEntryPaperData$InitValBetas ),
     Mcmc=list(M=15000,
               M0=10000,
               mOut=500,
               mSave=5000,
               seed=3546541)
  ))
}

setwd(curDir)

if (is.element("LMEntryPaperData$covariates", search())) {
    detach(LMEntryPaperData$covariates)
}
# ================================================================================
```

MCCExampleData *A Small MCC/DMC Example Data Set*

Description

A small MCC/DMC example data set – a small data set for demonstration purposes...

This small data set is from data from the Austrian Social Security Database (ASSD), which combines detailed longitudinal information on employment and earnings of all private sector workers in Austria since 1972. The IEW Working Paper Zweimueller et al. (2009) (see **Source**) gives an overview and a description of the main characteristics of the Austrian Social Security Database.

The ASSD was made available for the Austrian Center of Labor Economics and the Analysis of the Welfare State (http://www.labornrn.at/). This small sample consists of $N = 1000$ male Austrian workers, who enter the labor market for the first time in the years 1975 to 1985 and are less than 25 years old at entry. The cohort analysis is based on an observation period from 1975 to 2005.

Usage

```
data(MCCExampleData)
```

Format

The format is:

```
List of 4
 $ Njk.i       : num [1:6, 1:6, 1:1000] 0 0 0 0 0 0 0 0 0 0 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. ..$ : NULL
 $ initialState: num [1:1000] 4 1 4 3 0 1 2 1 4 2 ...
 $ obsList     :List of 1000
  ..$ SVNR1680347701: int [1:26] 4 4 5 5 5 5 5 5 5 5 ...
  ..$ SVNR1681207417: int [1:26] 1 1 0 0 0 0 0 2 0 0 ...
  ..$ SVNR1681671288: int [1:26] 4 0 0 1 0 5 5 5 5 5 ...
  .. [list output truncated]
 $ somePrior   :List of 5
  ..$ :List of 2
  .. ..$ xi : num [1:6, 1:6] 0.7303 0.1521 0.0901 0.0589 0.0435 ...
  .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. .. .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. ..$ eta: num 1
  ..$ :List of 2
  .. ..$ eta: num [1:2] 0.632 0.368
  .. ..$ xi : num [1:6, 1:6, 1:2] 0.2163 0.1072 0.0576 0.0373 0.0286 ...
  ..$ :List of 2
  .. ..$ eta: num [1:3] 0.243 0.258 0.5
  .. ..$ xi : num [1:6, 1:6, 1:3] 0.5075 0.2408 0.1595 0.1048 0.0744 ...
  ..$ :List of 2
  .. ..$ eta: num [1:4] 0.193 0.221 0.238 0.348
  .. ..$ xi : num [1:6, 1:6, 1:4] 0.556 0.245 0.196 0.136 0.1 ...
```

```
..$ :List of 2
.. ..$ eta: num [1:5] 0.246 0.232 0.156 0.143 0.223
.. ..$ xi : num [1:6, 1:6, 1:5] 0.2104 0.1581 0.0665 0.0414 0.0388 ...
```

Details

`MCCExampleData` is a list containing the following objects:

`Njk.i` A 3-dimensional array of dimension $6 \times 6 \times 1000$ containing the transition frequencies (6×6-matrices) of 1000 individuals. These represent the counts of transitions between wage categories from year to year with varying observation periods. Categories 1 to 5 correspond to the wage quintiles and 0 to no income.

`initialState` A vector giving the initial wage category for 1000 individuals.

`obsList` A list of 1000 numeric vectors (of integers with variable lengths) representing wage categories. Wage mobility time series with variable lengths describing (transitions between) wage categories (from year to year) of 1000 individuals where categories 1 to 5 correspond to the wage quintiles (in the income distribution of the corresponding year) and 0 to no income. Each positive number represents the position in the income distribution in terms of quintiles of a particular year.

`somePrior` A list of lists each containing prior-parameters for the group sizes and transition probabilities where the (index) number of the list corresponds to the number of clusters/groups.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Source

The following IEW Working Paper gives an overview and a description of the main characteristics of the Austrian Social Security Database:

Zweimueller, Josef, Winter-Ebmer, Rudolf, Lalive, Rafael, Kuhn, Andreas, Wuellrich, Jean-Philippe, Ruf, Oliver and Buechi, Simon, Austrian Social Security Database (May 4, 2009). Available at SSRN: `http://ssrn.com/abstract=1399350` or at `http://www.labornrn.at/wp/wp0903.pdf`.

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 `http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract`

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 `http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf`

See Also

LMEntryPaperData, MCCExtExampleData, mcClust, dmClust

Examples

```
data(MCCExampleData)
str(MCCExampleData)

# see example(s) in mcClust and dmClust
```

MCCExtExampleData — *An Extended MCC/DMC Example Data Set Including Covariates*

Description

An extended MCC/DMC example data set including covariates and response variables – a data set for demonstration purposes...

This small data set is from data from the Austrian Social Security Database (ASSD), which combines detailed longitudinal information on employment and earnings of all private sector workers in Austria since 1972. The IEW Working Paper Zweimueller et al. (2009) (see **Source**) gives an overview and a description of the main characteristics of the Austrian Social Security Database.

The ASSD was made available for the Austrian Center of Labor Economics and the Analysis of the Welfare State (http://www.labornrn.at/). This small sample consists of $N = 9402$ male Austrian workers, who enter the labor market for the first time in the years 1975 to 1985 and are less than 25 years old at entry. The cohort analysis is based on an observation period from 1975 to 2005.

Usage

```
data(MCCExtExampleData)
```

Format

The format is:

```
List of 4
 $ Njk.i     : num [1:6, 1:6, 1:9402] 0 0 0 0 0 0 0 0 0 0 ...
  ..- attr(*, "dimnames")=List of 3
  .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. ..$ : chr [1:6] "0" "1" "2" "3" ...
  .. ..$ : NULL
 $ covariates:'data.frame': 9402 obs. of  4 variables:
  ..$ alrateBezNew        : num [1:9402] 5.97 2.1 2.47 4.26 5.05 ...
  ..$ angStart            : num [1:9402] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ skilled             : int [1:9402] 0 0 0 0 0 0 0 0 0 0 ...
  ..$ unskilled           : int [1:9402] 0 0 0 0 0 0 0 0 0 0 ...
 $ NjkiMat   : num [1:9402, 1:36] 0 0 3 0 1 0 0 1 0 2 ...
```

```
 $ obsList   :List of 9402
  ..$ SVNR2166110217: int [1:9] 0 2 2 2 2 2 2 2 2
  ..$ SVNR1924158211: int [1:10] 1 0 3 2 3 2 3 4 4 2
  ..$ SVNR1982609045: int [1:10] 1 0 2 3 0 0 0 4 0 0
  .. [list output truncated]
 $ MNLresponse2gr: int [1:9402] 2 2 2 2 1 2 2 2 2 1 ...
 $ MNLresponse3gr: int [1:9402] 3 2 2 3 1 3 3 2 3 1 ...
 $ MNLresponse4gr: int [1:9402] 2 4 3 4 1 2 4 4 4 4 ...
```

Details

MCCExtExampleData is a list containing the following objects:

Njk.i A 3-dimensional array of dimension $6 \times 6 \times 9402$ containing the transition frequencies (6×6-matrices) of 9402 individuals. These represent the counts of transitions between wage categories from year to year with varying observation periods. Categories 1 to 5 correspond to the wage quintiles and 0 to no income.

covariates contains the data.frame with the covariates used in the logit regression model. It contains the following variables:

alrateBezNew	unemployment rate in the district
angStart	dummy for white collar workers
skilled	dummy for skilled workers
unskilled	dummy for unskilled workers

NjkiMat contains the Njk.i-data in matrix format of dimension 9402×36 (each row corresponds to the columns of the matrices in Njk.i).

obsList A list of 9402 numeric vectors (of integers with variable lengths) representing wage categories. Wage mobility time series with variable lengths describing (transitions between) wage categories (from year to year) of 9402 individuals where categories 1 to 5 correspond to the wage quintiles (in the income distribution of the corresponding year) and 0 to no income. Each positive number represents the position in the income distribution in terms of quintiles of a particular year.

MNLresponse2gr,...,MNLresponse4gr vectors containing the response variable for $h = 2, 3, 4$ clusters/groups, (necessary) for use in MNLAuxMix (for demonstration purposes).

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Source

The following IEW Working Paper gives an overview and a description of the main characteristics of the Austrian Social Security Database:

Zweimueller, Josef, Winter-Ebmer, Rudolf, Lalive, Rafael, Kuhn, Andreas, Wuellrich, Jean-Philippe, Ruf, Oliver and Buechi, Simon, Austrian Social Security Database (May 4, 2009). Available at SSRN: http://ssrn.com/abstract=1399350 or at http://www.labornrn.at/wp/wp0903.pdf.

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

LMEntryPaperData, MCCExampleData, mcClustExtended, dmClustExtended, MNLAuxMix

Examples

```
data(MCCExtExampleData)
str(MCCExtExampleData)

# see example(s) in mcClustExtended, dmClustExtended, MNLAuxMix or LMEntryPaperData
```

---

mcClustering — *Markov Chain Clustering With And Without Mixtures-of-Experts Extension*

Description

This function provides Markov chain clustering with or without multinomial logit model (mixtures-of-experts) extension (see **References**). That is an MCMC sampler for the mixtures-of-experts extension of Markov chain clustering. It requires four mandatory arguments: `Data`, `Prior`, `Initial` and `Mcmc`; each representing a list of (mandatory) arguments: `Data` contains data information, `Prior` contains prior information, `Initial` contains information about starting conditions (initial values) and `Mcmc` contains the setup for the MCMC sampler.

Usage

```
mcClust(
    Data = list(
        dataFile = stop(
 "'dataFile' (=> Njk.i) must be specified: either 'filename' (path) or data”),
        storeDir = "try01", priorFile = NULL),
    Prior = list( H = 4, e0 = 4, c = 1, cOff = 1, usePriorFile = FALSE,
        xiPooled = FALSE, N0 = 5),
```

```
    Initial = list( xi.start.ind = 3, pers = 0.7, S.i.start = NULL),
    Mcmc = list( M = 50, M0 = 20, mOut = 5, mSave = 10, seed = 12345))


mcClustExtended(
    Data = list(
        dataFile = stop(
 "'dataFile' (=> Njk.i) must be specified: either 'filename' (path) or data"),
        storeDir = "try01", priorFile = NULL,
        X = stop("X (matrix of covariates) must be specified")),
    Prior = list( H = 4, c = 1, cOff = 1, usePriorFile = FALSE,
        xiPooled = FALSE, N0 = 5, betaPrior = "informative",
        betaPriorMean = 0, betaPriorVar = 1),
    Initial = list( xi.start.ind = 3, pers = 0.7,
        S.i.start = rep(1:H, N), Beta.start = NULL),
    Mcmc = list( M = 50, M0 = 20, mOut = 5, mSave = 10,
                 seed = 12345))
```

Arguments

`Data`	a list consisting of: `dataFile, storeDir, priorFile, X`. See **Details**.
`Prior`	a list consisting of: `H, e0, c, cOff, usePriorFile, xiPooled, N0, betaPrior, betaPriorMean,` See **Details**.
`Initial`	a list consisting of: `xi.start.ind, pers, S.i.start, Beta.start`. See **Details**.
`Mcmc`	a list consisting of: `M, M0, mOut, mSave, seed`. See **Details**.

Details

Note that the values of the arguments indicated here have nothing to do with default values! For a call of these functions this lists-of-arguments structure requires a complete specification of all arguments!

The following arguments which are lists have to be completely provided (note that there are no such things as default values within lists!):

`Data` contains:

`dataFile` A 3-dim array having the transition counts/frequencies structure (like `Njk.i` in the example data sets) already loaded into the current environment/workspace. Or a character with the name of or the path to an .RData-*file* which contains such a data set, in which case it must have the name “`Njk.i`”.

It is required that this data have to be a 3-dimensional array of dimension $(K{+}1)\times(K{+}1)\times N$ containing the transition counts/frequencies, where $K + 1$ is the number of categories $k = 0, \ldots, K$ and N the number of objects/units/individuals. The number of transitions (equal to time series length minus one) may be individual.

`storeDir` A character indicating the name of the directory (will be created if not already existing) where the results are to be stored.

`priorFile` If not `NULL` the prior data (must have same format as `mccXiPrior` in `LMEntryPaperData` – at least the H-th entry in the list has to be provided) or a character with the name of or the path to a file containing such data, which in this case must be named "`mcc`". The prior data contain prior information (in terms of probabilities) about transition probabilities (possibly from another estimation procedure). For further information see Section **Prior Data** and `mccXiPrior` in `LMEntryPaperData`.

`X` The matrix of covariates (with N rows) including the unit vector for the intercept to be included in the multinomial logit model extension.

`Prior` contains (see also Section **Prior Data**):

`H` An integer ≥ 1 indicating the number of clusters/groups.

`e0` A numerical value determining the value of the prior parameter of the Dirichlet-prior for the group sizes η_h (`e0` $= \alpha_1 = \ldots = \alpha_H$, thus equal for all h).

`c,cOff` are necessary to calculate the prior parameter matrix for ξ (equal for all groups): `diag(c) + cOff`. Only used when `usePriorFile=FALSE` – see below.

`usePriorFile` If `usePriorFile=TRUE`, prior information for the transition probabilities as provided by `priorFile` are used as prior parameters for the estimation process. In this case there are two further options depending on the value of `xiPooled`: If `xiPooled=TRUE`, equal apriori transition probabilities are used for all groups (using `ceiling(Prior$N0*mcc[[1]]$xi)`) and if `xiPooled=FALSE` group-specific apriori transition probabilities are used (using `ceiling(Prior$N0*mcc[[H]]$xi)`).

If `usePriorFile=FALSE`, a priori transition probabilities are determined depending on `c` and `cOff`. In this case the diagonal elements are set to `c + cOff` and the off-diagonal elements to `cOff`, equal for all groups.

`xiPooled` Only used if `usePriorFile=TRUE` (see above): if `xiPooled=TRUE` equal apriori transition probabilities are used for all groups (using `ceiling(Prior$N0*mcc[[1]]$xi)`) and if `xiPooled=FALSE` group-specific apriori transition probabilities are used (using `ceiling(Prior$N0*mcc[[H]]$xi)`).

`N0` A numerical value determining a parameter for use in calculating the prior parameter matrix for ξ (see `usePriorFile`).

`betaPrior` A character. If "uninformative" (improper) prior parameters are used for the regression coefficients (i.e. betaPriorVar = ∞). Otherwise mean and variance of the normal prior distribution for the regression coefficients have to be specified.

`betaPriorMean, betaPriorVar` Numerical values specifying the parameters of the normal prior distribution for the regression coefficients, only if `betaPrior!="uninformative"`.

`Initial` contains:

`xi.start.ind` An integer taking a value out of 1, 2, 3 or 4 to determine how to define the start values for ξ: If `xi.start.ind = 1`: the uniform distribution is used, meaning that all elements are equal to $1/(K+1)$ in all groups. If `xi.start.ind = 2`: the empirical distribution/transition matrix (classical ML estimate of the transition matrix) is used (equal for all groups). If `xi.start.ind = 3`: a 'persistence' distribution is used, meaning that the diagonal elements are equal to `pers` whereas all off-diagonal elements are equal to (1-`pers`)/K (equal for all groups). If `xi.start.ind = 4`: entry in prior file `mcc[[H]]$xi` is used directly for initial values.

pers Only used if xi.start.ind = 3: A numerical value (between 0 and 1) which indicates the persistence probabilities (equal for all diagonal elements). Note, that $1/(K+1)$ corresponds to the uniform distribution in each row.

S.i.start A vector of length N giving an initial allocation (mandatory for mcClustExtended).

Beta.start A matrix of dimension ncol(X) x H giving start values for the regression coefficients including the zero vector in the first column representing the baseline group.

Mcmc contains:

M An integer indicating the overall number of iterations.

M0 An integer indicating the number of the first iteration *after* the burn-in phase.

mOut An integer indicating that after each mOut-th iteration a report line is written to the output window/screen.

mSave An integer indicating that after each mSave-th iteration an intermediate storage of the workspace is carried out.

seed An integer indicating a random seed.

Value

A list containing (/the output file contains):

workspaceFile	A character indicating the name of and the path (based on the currend working directory) to the output file, wherein all the results are saved. The name of the output file starts with "MCC_" or "MCC_Logit_newAux_" respectively followed by the number of groups H, the number of iterations M and the particular point in time when the function was called, with format: yyyymmdd_hhmmss. E.g. MCC_H4_M10000_20110218_045254.RData or MCC_Logit_newAux_H4_M10000_20111121_165723.RData.
Data	The argument Data.
Prior	The argument Prior.
Initial	The argument Initial.
Mcmc	The argument Mcmc.
Beta.m	A 3-dimensional array of dimension ncol(X) $\times H \times M$ containing the draws for the regression coefficients β_h in each m-th iteration step.
bk0	The prior parameters for the mean vectors of the normal (prior) distributions of the regression coefficients.
Bk0inv	The prior parameters for the inverse variance-covariance matrices of the normal (prior) distributions of the regression coefficients.
c0	A 3-dimensional array with dimension $(K+1) \times (K+1) \times H$ that contains the finally used a priori parameter values for $\boldsymbol{\xi}_h$.
estTransProb	A 3-dimensional array with dimension $(K+1) \times (K+1) \times H$ that contains the ergodic average of $\boldsymbol{\xi}_h$ for all groups (using draws from M0 to M without thinning parameter).
fileName	A character value indicating the name of the output file (see also workspaceFile).

`freq`	matrix-matching (pattern recognition): a numerical vector containing the frequencies of different (!) transition matrices. (in ascending order)
`indizes`	matrix-matching (pattern recognition): a numerical vector containing the indices of different (!) transition matrices.
`K`	An integer indicating the number of categories minus one (!). See **Note**.
`mcc`	The prior data (see Section **Prior Data**) provided with `priorFile`, `NULL` otherwise.
`N`	An integer indicating N, the number of individuals/units/objects.
`Njk.i`	The data (see **Details**) provided with `dataFile`.
`Njk.i.ind`	matrix-matching (pattern recognition): the resulting `Njk.i` after matrix-matching.
`R`	matrix-matching (pattern recognition): number of different (!) transition matrices.
`S.i.counts`	A $N \times H$-matrix containing the frequencies how often individual i was allocated to a certain group during the iterations from `M0+1` to codeM.
`totalTime`	A numeric value indicating the total time (in secs) used for the function call.
`xi.hat`	A matrix with dimension $(K+1) \times (K+1)$ containing the empirical transition probabilities (overall relative transition freqs).
`xi.m`	A 4-dimensional array of dimension $M \times (K+1) \times (K+1) \times H$ containing the draws for $\boldsymbol{\xi}_h$ in each m-th iteration step.
`xi.start`	A matrix of dimension $(K+1) \times (K+1)$ that contains the starting values for $\boldsymbol{\xi}_h$ (only if `xi.start.ind = 3`).
`xi.start.ind`	An integer indicating the used method to calculate/determine the starting values for $\boldsymbol{\xi}_h$.
`bkN`	The posterior parameters (in the last iteration step) for the mean vectors of the normal (posterior) distributions from which the regression coefficients were drawn.
`BkN`	The posterior parameters (in the last iteration step) for the variance-covariance matrices of the normal (posterior) distributions from which the regression coefficients were drawn.
`logLike`	A vector containing the values of the log-likelihood calculated in each iteration step.
`logBetaPrior`	A vector containing the values of the prior distribution for the regression coefficients calculated in each iteration step.
`logXiPrior`	A vector containing the values of the prior distribution for the transition matrices calculated in each iteration step.
`logPostDens`	A vector containing the values of the posterior density calculated in each iteration step.
`mMax`	An integer giving the position (number of iteration) of the maximum value in the posterior density `logPostDens`.
`logClassLike`	A vector containing the values of the log classification likelihood calculated in each iteration step.
`entropy`	A vector containing the values of the entropy calculated in each iteration step.

`eta.start` Either a numeric value equal to 1/H or if `xi.start.ind = 4` the corresponding data (vector) from the prior file.

`estGroupSize` A numerical vector containing the ergodic average of η_h for all groups (using draws from `M0+1` to codeM without thinning parameter).

`eta.m` A matrix of dimension $H \times M$ containing the draws for η_h in each m-th iteration step.

`logEtaPrior` A vector containing the values of the prior distribution for the mixing proportions (group sizes) calculated in each iteration step.

Prior Data

The prior data (called `mcc` in the following) – to be passed via `priorFile` in argument-list `Data` – has to be a list of lists, indexed by $1, \ldots, H, H + 1, \ldots$. Note that, depending on parameter H (the number of groups – to be passed via `H` in argument-list `Prior`), there have to be at least H entries (each a list). See `mccXiPrior` in `LMEntryPaperData` for example. Within a call to `dmClustering` or `mcClustering`, at least `mcc[[H]]` has to be provided as a list containing `eta` and `xi`. `eta` is a vector of length H containing prior information about the relative group sizes of group $h = 1, \ldots, H$. `xi` is a 3-dimensional array of dimension $(K + 1) \times (K + 1) \times H$, containing prior information in terms of probabilities about the transition probabilities of group $h = 1, \ldots, H$ (see examples).

Reporting Progress (Log Protocol)

The log protocol keeps record of the progress of the estimation procedure and is shown on the screen. At first the name of the workspace file is documented. Then for each `mOut`-th iteration step (at least for $m = 1, \ldots, 5, 10, 20, 50, 100, 200, 500$) information about the elapsed time and the expected time to the end is reported. Finally the total time is shown.

For example:

```
workspaceFile:  tryN50000-sample02-01\MCC_Logit_newAux_H4_M10000_20111124_155650.RData   (within curr
m = 1 ; duration of iter proc so far:  13.75 sec.
m = 2 ; duration of iter proc so far: 21.59 sec.,  exp time to end: 3597.97  min.
m = 3 ; duration of iter proc so far: 29.48 sec.,  exp time to end: 2456.18  min.
m = 4 ; duration of iter proc so far: 37.36 sec.,  exp time to end: 2074.93  min.
m = 5 ; duration of iter proc so far: 45.25 sec.,  exp time to end: 1884.66  min.
m = 10 ; duration of iter proc so far: 84.94 sec.,  exp time to end: 1571.55  min.
m = 20 ; duration of iter proc so far: 164.5 sec.,  exp time to end: 1440.24  min.
m = 50 ; duration of iter proc so far: 403.08 sec.,  exp time to end: 1364.3  min.
m = 100 ; duration of iter proc so far: 801.15 sec.,  exp time to end: 1335.38  min.
m = 200 ; duration of iter proc so far: 1530.5 sec.,  exp time to end: 1256.32  min.
m = 400 ; duration of iter proc so far: 3074.03 sec.,  exp time to end: 1232.82  min.
m = 500 ; duration of iter proc so far: 3804.67 sec.,  exp time to end: 1207.35  min.
m = 600 ; duration of iter proc so far: 4532.04 sec.,  exp time to end: 1185.47  min.
m = 800 ; duration of iter proc so far: 6075.54 sec.,  exp time to end: 1166.06  min.
m = 1000 ; duration of iter proc so far: 7715.48 sec.,  exp time to end: 1158.61  min.
...
```

Warning

Note that there are no such things as *default* values (see Section **Arguments**)!

Note

Note that the required data files have to be provided in the current working directory and that the results (see Section **Value**) are to be saved in the directory provided by `storeDir` within the current working directory. Make sure that the current working directory is set appropriately before the function is called.

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

Sylvia Fruehwirth-Schnatter and Rudolf Fruehwirth, (2010), "Data augmentation and MCMC for binary and multinomial logit models". In T. Kneib and G. Tutz (eds): *Statistical Modelling and Regression Structures: Festschrift in Honour of Ludwig Fahrmeir*. Physica Verlag, Heidelberg, pp. 111-132. DOI: 10.1007/978-3-7908-2413-1_7 http://www.springerlink.com/content/t4h810017645wh68/. See also: IFAS Research Paper Series 2010-48 (http://www.jku.at/ifas/content/e108280/e108491/e108471/e109880/ifas_rp48.pdf).

See Also

`dmClust`, `dmClustExtended`, `MNLAuxMix`, `LMEntryPaperData`, `MCCExampleData`, `MCCExtExampleData`

Examples

```
#rm(list=ls(all=TRUE))

# ==================================================================================
if ( TRUE ) {
# ==================================================================================

# set working directory
oldDir <- getwd()
curDir <- tempdir()
```

```
setwd(curDir)

if ( !file.exists("bayesMCClust-wd") ) dir.create("bayesMCClust-wd")
setwd("bayesMCClust-wd")
myOutfilesDir <- "mcClust-Example-Outfiles"

# load data
data(MCCExampleData)

# ==================================================================================

# function call
system.time(
  outList <- mcClust(    # parameter lists (every four) must be complete!
     Data=list(dataFile=MCCExampleData$Njk.i,
               storeDir=myOutfilesDir,
               priorFile= NULL),
     Prior=list(H=2, # sample(2:6, 1), # 4
                e0=4,
                c=1,
                cOff=1,
                usePriorFile=FALSE,
                xiPooled=FALSE,
                N0=5),
     Initial=list(xi.start.ind=3,
                  pers=0.7),
     Mcmc=list(M=100,
               M0=20,
               mOut=5,
               mSave=50,
               seed=sample(1:100000, 1) # 123
     )
  )
)

str(outList)

#outFileName
#results <- load(outFileName)
#results
#estTransProb

allocList <- calcAllocationsMCC(outList, thin=1, maxi=50) # , plotPathsForEta=TRUE
str(allocList)

myTransProbs <- calcTransProbs(outList, estGroupSize=allocList$estGroupSize, thin=1,
    printXtable=FALSE, printSd=FALSE, printTogether=TRUE )
    # , plotPaths=TRUE, grLabels=paste("Group", 1:Prior$H)
str(myTransProbs)

myTransList <- plotTransProbs(outList, estTransProb=myTransProbs$estTransProb,
    estGroupSize=allocList$estGroupSize, class=allocList$class, plotPooled=TRUE,
    plotContTable=TRUE, printContTable=TRUE, plotContPooled=TRUE)
```

```
    # , grLabels=paste("Group", 1:Prior$H)
str(myTransList)

(equiDist <- calcEquiDist(outList, thin=1, maxi=50))
#, printEquiDist=TRUE, plotEquiDist=TRUE , grLabels=paste("Group", 1:Prior$H)

myLongRunDistList <- calcLongRunDist(outList,
    initialStateData=MCCExampleData$initialState,
    class=allocList$class, equiDist=equiDist, maxi=50)
    # , printLongRunDist=TRUE, grLabels=paste("Group", 1:Prior$H)
str(myLongRunDistList)

myTypicalMembs <- plotTypicalMembers(outList, moreTypMemb=c(10,25,40,55,70,85,100),
    myObsList=MCCExampleData$obsList, classProbs=allocList$classProbs) # noTypMemb=7
str(myTypicalMembs)

plotScatter(outList, thin=1, xi11=c(1,1), xi12=c(2,2), xi21=c(2,2), xi22=c(3,3),
    xi31=c(1,1), xi32=c(3,3) )

mySegPower <- calcSegmentationPower(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE, calcSharp=TRUE, printSharpXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
str(mySegPower)

myEntropy <- calcEntropy(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
myEntropy

plotLikeliPaths(outList, from=10, by=1 )

myNumEffTables <- calcNumEff( outList, thin=1, printXi=TRUE, printE=TRUE,
    printBeta=TRUE, grLabels=paste("Group", 1:outList$Prior$H) )
str(myNumEffTables)

myMSCrits <- calcMSCritMCC(workDir=myOutfilesDir, myLabel="mcClust-Example", H0=4,
    whatToDoList=c("approxML", "approxMCL", "postMode") )
str(myMSCrits)

setwd(oldDir)

} # end if

# ==================================================================================
# ==================================================================================
# ==================================================================================

# ==================================================================================
if ( FALSE ) {
# ==================================================================================

rm(list=ls(all=TRUE))
```

```
# set working directory
oldDir <- getwd()
curDir <- tempdir()
setwd(curDir)

if ( !file.exists("bayesMCClust-wd") ) dir.create("bayesMCClust-wd")
setwd("bayesMCClust-wd")
myOutfilesDir <- "mcClustExtended-Example-Outfiles"

# load data
data(MCCExtExampleData)
if (!is.element("MCCExtExampleData$covariates", search())) {
    attach(MCCExtExampleData$covariates)
}

# ====================================================================================

groupNr <- 2 # sample(2:6, 1) # 3

# ====================================================================================

results <- kmeans( log( MCCExtExampleData$NjkiMat + 0.5 ) , groupNr, nstart=2)

# ====================================================================================

require(nnet, quietly = TRUE)
H <- groupNr
X = cbind( intercept=1, alrateBezNew, unskilled, skilled, angStart )

N <- dim(X)[1]
mX <- data.frame( cbind(group=as.factor( results$cluster ), X[,-1],
    matrix(sample(1:H,H*N,replace=TRUE),N,H)) )

colnames(mX)[6:(6+groupNr-1)] <-
    c( "as.1", "as.2", "as.3", "as.4", "as.5", "as.6" )[1:groupNr]

tempMNom <- multinom(group ~ alrateBezNew+ unskilled+ skilled+ angStart,
    data=as.data.frame(mX))

toStartBeta <- t(rbind(0,coef( tempMNom )))

# ====================================================================================
# function call
outList <- mcClustExtended(
     Data=list(dataFile=MCCExtExampleData$Njk.i, # parameter lists must be complete!!!
               storeDir=myOutfilesDir,
               priorFile= NULL,
               X = cbind( intercept=1, alrateBezNew, unskilled, skilled, angStart ) ),
     Prior=list(H=groupNr,
                c=1,
                cOff=1,
                usePriorFile=FALSE,
                xiPooled=FALSE,
```

```
                N0=5,
                betaPrior = "informative", # N(0,1)
                betaPriorMean = 0,
                betaPriorVar = 1),
     Initial=list(xi.start.ind=3,
                  pers=0.7,
                  S.i.start = results$cluster,
                  Beta.start = toStartBeta ),
     Mcmc=list(M=100,
               M0=50,
               mOut=10,
               mSave=50,
               seed=sample(1:100000, 1) # 69814651
              )
     )

str(outList)

#outFileName <- outList$workspaceFile
#results <- load(outFileName)
#results
#estTransProb

allocList <- calcAllocationsMCCExt(outList, thin=1, maxi=50)
str(allocList)

myTransProbs <- calcTransProbs(outList, estGroupSize=allocList$estGroupSize, thin=1,
    printXtable=FALSE, printSd=FALSE, printTogether=TRUE )
    # plotPaths=TRUE, grLabels=paste("Group", 1:Prior$H)
str(myTransProbs)

myTransList <- plotTransProbs(outList, estTransProb=myTransProbs$estTransProb,
    estGroupSize=allocList$estGroupSize, class=allocList$class, plotPooled=TRUE,
    plotContTable=TRUE, printContTable=TRUE, plotContPooled=TRUE)
    # , grLabels=paste("Group", 1:Prior$H)
str(myTransList)

(equiDist <- calcEquiDist(outList, thin=1, maxi=50))
# , printEquiDist=TRUE, plotEquiDist=TRUE, grLabels=paste("Group", 1:Prior$H)

myRegCoeffs <- calcRegCoeffs(outList, hBase=2, thin=1)
#, M0=Mcmc$M0, grLabels=paste("Group", 1:Prior$H),
# printHPD=TRUE, plotPaths=TRUE, plotACFs=TRUE
str(myRegCoeffs)

myLongRunDistList <- calcLongRunDist(outList, initialStateData=initialState,
    class=allocList$class, equiDist=equiDist, maxi=50)
    # , printLongRunDist=TRUE
str(myLongRunDistList)

myTypicalMembs <- plotTypicalMembers(outList, myObsList=MCCExtExampleData$obsList,
    classProbs=allocList$classProbs)
    # , noTypMemb=7, moreTypMemb=c(10,25,50,100,200,500,1000)
```

```
str(myTypicalMembs)

plotScatter(outList, thin=1, xi11=c(1,1), xi12=c(2,2), xi21=c(2,2), xi22=c(3,3),
    xi31=c(1,1), xi32=c(3,3) )

mySegPower <- calcSegmentationPower(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE, calcSharp=TRUE, printSharpXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
str(mySegPower)

myEntropy <- calcEntropy(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
myEntropy

plotLikeliPaths(outList, from=10, by=1 )

myNumEffTables <- calcNumEff( outList, thin=1, printXi=TRUE, printE=TRUE,
    printBeta=TRUE, grLabels=paste("Group", 1:outList$Prior$H) )
str(myNumEffTables)

myMSCrits <- calcMSCritMCCExt(workDir=myOutfilesDir, NN=outList$N,
    myLabel="mcClustExtended-Example", ISdraws=100, H0=3,
    whatToDoList=c("approxML", "approxMCL", "postMode" ) )
str(myMSCrits)

setwd(oldDir)

# ====================================================================================

if (is.element("MCCExtExampleData$covariates", search())) {
    detach(MCCExtExampleData$covariates)
}

# ====================================================================================
} # end if
# ====================================================================================

# ====================================================================================
# ====================================================================================
```

`MNLAuxMix` *Bayesian Multinomial Logit Regression Using Auxiliary Mixture Sampling*

Description

This function provides Bayesian multinomial logit regression using auxiliary mixture sampling. See Fruehwirth-Schnatter and Fruehwirth (2010). That is an MCMC sampler that is also used for the mixtures-of-experts extension of Dirichlet Multinomial (`dmClustExtended`) and Markov chain clustering (`mcClustExtended`). It requires four mandatory arguments: `Data`, `Prior`, `Initial` and `Mcmc`; each representing a list of (mandatory) arguments: `Data` contains data information, `Prior` contains prior information, `Initial` contains information about starting conditions (initial values) and `Mcmc` contains the setup for the MCMC sampler.

Usage

```
MNLAuxMix(
    Data = list( storeDir = "try01",
                 X = stop("X (matrix of covariates) must be specified")),
    Prior = list( H = 4, betaPrior = "informative",
                  betaPriorMean = 0, betaPriorVar = 1),
    Initial = list( S.i.start = rep(1:H, N), Beta.start = NULL),
    Mcmc = list( M = 50, M0 = 20, mOut = 5, mSave = 10, seed = 12345))
```

Arguments

`Data`	a list consisting of: `storeDir, X`. See **Details**.
`Prior`	a list consisting of: `H, betaPrior, betaPriorMean, betaPriorVar`. See **Details**.
`Initial`	a list consisting of: `S.i.start, Beta.start`. See **Details**.
`Mcmc`	a list consisting of: `M, M0, mOut, mSave, seed`. See **Details**.

Details

Note that the values of the arguments indicated here have nothing to do with default values! For a call of these functions this lists-of-arguments structure requires a complete specification of all arguments!

The following arguments which are lists have to be completely provided (note that there are no such things as default values within lists!):

`Data` contains:

`storeDir` A character indicating the name of the directory (will be created if not already existing) where the results are to be stored.

`X` The matrix of covariates (with N rows) including the unit vector for the intercept to be included in the multinomial logit model.

`Prior` contains (see also Section **Prior Data**):

`H` An integer ≥ 1 indicating the number of response categories.

`betaPrior` A character. If "uninformative" (improper) prior parameters are used for the regression coefficients (i.e. betaPriorVar = ∞). Otherwise mean and variance of the normal prior distribution for the regression coefficients have to be specified.

`betaPriorMean, betaPriorVar` Numerical values specifying the parameters of the normal prior distribution for the regression coefficients, only if `betaPrior!="uninformative"`.

`Initial` contains:

S.i.start A vector of length N giving initial response categories.

Beta.start A matrix of dimension ncol(X) x H giving start values for the regression coefficients including the zero vector in the first column representing the baseline response category.

Mcmc contains:

M An integer indicating the overall number of iterations.

M0 An integer indicating the number of the first iteration *after* the burn-in phase.

mOut An integer indicating that after each mOut-th iteration a report line is written to the output window/screen.

mSave An integer indicating that after each mSave-th iteration an intermediate storage of the workspace is carried out.

seed An integer indicating a random seed.

Value

A list containing (/the output file contains):

workspaceFile	A character indicating the name of and the path (based on the currend working directory) to the output file, wherein all the results are saved. The name of the output file starts with "mnLogit_newAux_" respectively followed by the number of groups H, the number of iterations M and the particular point in time when the function was called, with format: yyyymmdd_hhmmss. E.g. mnLogit_newAux_H4_M10000_20110218_045
Data	The argument Data.
Prior	The argument Prior.
Initial	The argument Initial.
Mcmc	The argument Mcmc.
Beta.m	A 3-dimensional array of dimension ncol(X) $\times H \times M$ containing the draws for the regression coefficients β_h in each m-th iteration step.
bk0	The prior parameters for the mean vectors of the normal (prior) distributions of the regression coefficients.
Bk0inv	The prior parameters for the inverse variance-covariance matrices of the normal (prior) distributions of the regression coefficients.
fileName	A character value indicating the name of the output file (see also workspaceFile).
N	An integer indicating N, the number of individuals/units/objects.
totalTime	A numeric value indicating the total time (in secs) used for the function call.
bkN	The posterior parameters (in the last iteration step) for the mean vectors of the normal (posterior) distributions from which the regression coefficients were drawn.
BkN	The posterior parameters (in the last iteration step) for the variance-covariance matrices of the normal (posterior) distributions from which the regression coefficients were drawn.
logLike	A vector containing the values of the log-likelihood calculated in each iteration step.

Reporting Progress (Log Protocol)

The log protocol keeps record of the progress of the estimation procedure and is shown on the screen. At first the name of the workspace file is documented. Then for each mOut-th iteration step (at least for $m = 1, \ldots, 5, 10, 20, 50, 100, 200, 500$) information about the elapsed time and the expected time to the end is reported. Finally the total time is shown.

For example:

```
workspaceFile:  MNLAuxMix-Example-Outfiles\mnLogit_newAux_H2_M100_20111129_083023.RData  (within cur
m = 1 ; duration of iter proc so far:  0.25 sec.
m = 2 ; duration of iter proc so far: 0.33 sec.,  exp time to end: 0.54  min.
m = 3 ; duration of iter proc so far: 0.44 sec.,  exp time to end: 0.36  min.
m = 4 ; duration of iter proc so far: 0.52 sec.,  exp time to end: 0.28  min.
m = 5 ; duration of iter proc so far: 0.6 sec.,  exp time to end: 0.24  min.
m = 10 ; duration of iter proc so far: 1.04 sec.,  exp time to end: 0.18  min.
m = 20 ; duration of iter proc so far: 1.93 sec.,  exp time to end: 0.14  min.
m = 30 ; duration of iter proc so far: 2.8 sec.,  exp time to end: 0.11  min.
m = 40 ; duration of iter proc so far: 3.79 sec.,  exp time to end: 0.1  min.
m = 50 ; duration of iter proc so far: 4.79 sec.,  exp time to end: 0.08  min.
m = 60 ; duration of iter proc so far: 5.89 sec.,  exp time to end: 0.07  min.
m = 70 ; duration of iter proc so far: 6.8 sec.,  exp time to end: 0.05  min.
m = 80 ; duration of iter proc so far: 7.68 sec.,  exp time to end: 0.03  min.
m = 90 ; duration of iter proc so far: 8.63 sec.,  exp time to end: 0.02  min.
m = 100 ; duration of iter proc so far: 9.52 sec.,  exp time to end: 0  min.
Total time: 0 hours 0 min
```

Warning

Note that there are no such things as *default* values (see Section **Arguments**)!

Note

Note that the required data files have to be provided in the current working directory and that the results (see Section **Value**) are to be saved in the directory provided by storeDir within the current working directory. Make sure that the current working directory is set appropriately before the function is called.

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

Sylvia Fruehwirth-Schnatter and Rudolf Fruehwirth, (2010), "Data augmentation and MCMC for binary and multinomial logit models". In T. Kneib and G. Tutz (eds): *Statistical Modelling and Regression Structures: Festschrift in Honour of Ludwig Fahrmeir*. Physica Verlag, Heidelberg, pp. 111-132. DOI: 10.1007/978-3-7908-2413-1_7 http://www.springerlink.com/content/t4h810017645wh68/. See also: IFAS Research Paper Series 2010-48 (http://www.jku.at/ifas/content/e108280/e108491/e108471/e109880/ifas_rp48.pdf).

See Also

mcClustExtended, dmClustExtended, MCCExtExampleData, calcAllocationsMNL, calcRegCoeffs, calcSegmentationPower, calcEntropy, plotLikeliPaths, calcNumEff

Examples

```
#rm(list=ls(all=TRUE))

# ==================================================================================
if ( FALSE ) {
# ==================================================================================

# set working directory
oldDir <- getwd()
curDir <- tempdir()
setwd(curDir)

if ( !file.exists("bayesMCClust-wd") ) dir.create("bayesMCClust-wd")
setwd("bayesMCClust-wd")
myOutfilesDir <- "MNLAuxMix-Example-Outfiles"

data(MCCExtExampleData)

if (!is.element("MCCExtExampleData$covariates", search())) {
    attach(MCCExtExampleData$covariates)
}

# ==================================================================================

response <- MCCExtExampleData[[ sample(5:7, 1) ]] # MCCExtExampleData$MNLresponse2gr
# MCCExtExampleData$MNLresponse3gr # MCCExtExampleData$MNLresponse4gr #

groupNr <- max(response) # 3

# ==================================================================================
# ==================================================================================
```

```
require(nnet, quietly = TRUE)
H <- groupNr
X = cbind( intercept=1, alrateBezNew, unskilled, skilled, angStart )

N <- dim(X)[1]
mX <- data.frame( cbind(group=as.factor( response ), X[,-1],
                  matrix(sample(1:H,H*N,replace=TRUE),N,H)) )

colnames(mX)[6:(6+groupNr-1)] <- c(  "as.1", "as.2", "as.3", "as.4" )[1:groupNr]

tempMNom <- multinom(group ~ alrateBezNew+ unskilled+ skilled+ angStart,
                     data=as.data.frame(mX))

toStartBeta <- t(rbind(0,coef( tempMNom )))

# ====================================================================================
system.time(
  outList <- MNLAuxMix(
    Data = list( storeDir = myOutfilesDir,
                 # will be created if not existing (in current working directory!)
                 X = cbind( intercept=1, alrateBezNew, unskilled, skilled, angStart ) ),
    Prior = list( H = groupNr, # number of alternatives 1,...,H
                  betaPrior = "informative",
                 # 'uninformative' (improper) prior pars for beta (betaPriorVar = infty)
                  betaPriorMean = 0,
                  betaPriorVar = 1), # 'informative' prior pars for beta -> N(0,1)
    Initial = list( S.i.start = response, #  vector of multinomial outcomes / choice made
                    Beta.start = toStartBeta ),
    Mcmc = list( M = 100,
                 M0 = 50,
                 mOut = 10,
                 mSave = 50,
                 seed = sample(1:100000, 1) # 6984684
    )
  )
)

str(outList)

#outFileName <- outList$workspaceFile
#outFileName
#results <- load(outFileName)
#results

allocList <- calcAllocationsMNL(outList, thin=1, maxi=50)
str(allocList)

myRegCoeffs <- calcRegCoeffs(outList, hBase=2, thin=1)
#, M0=Mcmc$M0, grLabels=paste("Group", 1:Prior$H),
# printHPD=TRUE, plotPaths=TRUE, plotACFs=TRUE
str(myRegCoeffs)

mySegPower <- calcSegmentationPower(outList, classProbs=allocList$classProbs,
```

```
    class=allocList$class, printXtable=TRUE, calcSharp=TRUE, printSharpXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
str(mySegPower)

myEntropy <- calcEntropy(outList, classProbs=allocList$classProbs,
    class=allocList$class, printXtable=TRUE )
    # , grLabels=paste("Group", 1:Prior$H)
myEntropy

plotLikeliPaths(outList, from=10, by=1 )

myNumEffTables <- calcNumEff( outList, thin=1, printXi=TRUE, printE=TRUE,
    printBeta=TRUE, grLabels=paste("Group", 1:outList$Prior$H) )
str(myNumEffTables)

setwd(oldDir)

# ==================================================================================

if ( is.element("MCCExtExampleData$covariates", search())) {
    detach(MCCExtExampleData$covariates)
}

# ==================================================================================
} # end if
# ==================================================================================

# ==================================================================================
```

---

plotLikeliPaths *Plots Paths of Likelihoods And (Prior) Densities*

Description

Plots *paths* of all sorts of likelihood and (prior) densities, like the log-likelihood, log posterior density, log classification likelihood and the entropy all including markings for the position of the maximum value, and further log prior densities for η, β, ξ and e (depending on availability/model type).

Usage

```
plotLikeliPaths(outList, from = 10, by = 1)
```

Arguments

outList	specifies a list containing the outcome (return value) of an MCMC run of mcClust, dmClust, mcClustExtended, dmClustExtended or MNLAuxMix.
from	specifies number of MCMC draw where to start plotting from.
by	specifies with which 'step size' plotting should be done.

Details

All these likelihoods and (prior) densities ware already calculated (for each MCMC draw) by `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended` and `MNLAuxMix` and saved in `outList`.

Value

No value returned.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

`mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended`, `MNLAuxMix`

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended, MNLAuxMix
```

`plotScatter` *Produces Scatter Plots of MCMC Draws*

Description

Produces three scatter plots of MCMC draws of selected transition probabilities over all clusters/groups.

Usage

```
plotScatter(outList, thin = 1, xi11 = c(1, 1), xi12 = c(2, 2),
            xi21 = c(2, 2), xi22 = c(3, 3),
            xi31 = c(1, 1), xi32 = c(3, 3))
```

Arguments

outList	specifies a list containing the outcome (return value) of an MCMC run of mcClust, dmClust, mcClustExtended or dmClustExtended.
thin	An integer specifying the thinning parameter (default is 1).
xi11	A vector with 2 (valid) integers specifying j and k of $\xi_{\cdot,j,k}$, the transition probability to use on the x-axis of the first plot.
xi12	A vector with 2 (valid) integers specifying j and k of $\xi_{\cdot,j,k}$, the transition probability to use on the y-axis of the first plot.
xi21	A vector with 2 (valid) integers specifying j and k of $\xi_{\cdot,j,k}$, the transition probability to use on the x-axis of the second plot.
xi22	A vector with 2 (valid) integers specifying j and k of $\xi_{\cdot,j,k}$, the transition probability to use on the y-axis of the second plot.
xi31	A vector with 2 (valid) integers specifying j and k of $\xi_{\cdot,j,k}$, the transition probability to use on the x-axis of the third plot.
xi32	A vector with 2 (valid) integers specifying j and k of $\xi_{\cdot,j,k}$, the transition probability to use on the y-axis of the third plot.

Value

No value returned.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

mcClust, dmClust, mcClustExtended, dmClustExtended

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended
# and/or dmClustExtended
```

---

plotTransProbs *Produces Balloon Plots and LaTeX-Style Tables of the Transition Matrices*

Description

Produces balloon plots and LaTeX-style tables of the transition matrices and cluster-specific contingency tables (transition frequency matrices).

Usage

```
plotTransProbs(outList, estTransProb, estGroupSize, class,
               grLabels = paste("Group", 1:outList$Prior$H),
               plotPooled = TRUE,
               plotContTable = TRUE, printContTable = TRUE,
               plotContPooled = TRUE)
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended` or `dmClustExtended`.
`estTransProb`	A 3-dim array containing the posterior expectation of the average transition matrices of all clusters/groups as returned by `calcTransProbs`.
`estGroupSize`	A vector of dimension H containing the (estimated) group sizes returned by `calcAllocations`.
`class`	A vector of length N containing the group membership returned by `calcAllocations`.
`grLabels`	A character vector giving user-specified names for the clusters/groups.
`plotPooled`	If `TRUE` (default) a balloon plot of the pooled transition matrix (ML estimate for all individuals) is produced. See **Value**: `relNjk`.
`plotContTable`	If `TRUE` (default) balloon plots of the cluster-specific contingency tables (transition frequency matrices) are produced. See **Details** and **Value**: `relTransFreq`.
`printContTable`	If `TRUE` (default) a LaTeX-style table containing the absolute and relative row sums of the cluster-specific contingency tables (transition frequency matrices) is generated/printed (iff `plotContTable` is `TRUE`). See **Value**: `contTable`.
`plotContPooled`	If `TRUE` (default) a balloon plot of the pooled contingency table (transition frequency matrix) is produced (iff `plotContTable` is `TRUE`). See **Value**: `relNjkMat`.

Details

This function visualizes the posterior expectation of the group-specific transition matrices (`estTransProb`) using "balloon plots" (function `balloonplot` from package **gplots**). The circular areas are proportional to the size of the corresponding entry in the transition matrix. The corresponding group sizes (`estGroupSize`) are indicated in parentheses.

Furthermore, the "balloons" are appropriately scaled (automatically) to be comparable within and *between* groups.

The (cluster-specific) contingency tables report for each cluster in cell (j, k) the probability of observing the categories (j, k) in consecutive time points/periods for an individual in this cluster. The entries to this table/figure sum to one (see **Value**: `relTransFreq`).

Value

A list containing:

- `relNjk` A matrix containing the ML estimate of the transition matrix for all individuals (pooled). That is the matrix containing the total sum of all observed transitions where each row is scaled to 1.
- `contTable` A matrix containing the row sums of the group-specific contingency tables (absolute transition frequencies).
- `relTransFreq` A 3-dim array containing the cluster-specific contingency tables.
- `relNjkMat` A matrix containing the sum of all observed transitions where the whole matrix is scaled to 1.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

`calcTransProbs`, `calcAllocations`, `balloonplot`, `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended`

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended,
# dmClustExtended
```

`plotTypicalMembers` *Plots Time Series of 'Typical' Group Members*

Description

Plots time series of the most 'typical' group members showing the highest classification probabilities.

Usage

```
plotTypicalMembers(outList, myObsList, classProbs, noTypMemb = 7,
                   moreTypMemb = c(10, 25, 50, 100, 200, 500, 1000),
                   grLabels = paste("Group", 1:outList$Prior$H))
```

Arguments

`outList`	specifies a list containing the outcome (return value) of an MCMC run of `mcClust`, `dmClust`, `mcClustExtended` or `dmClustExtended`.
`myObsList`	A list containing N numeric vectors (of integers with possibly variable lengths) corresponding to the individual time series.
`classProbs`	A matrix with dimension $N \times H$ containing the individual posterior classification probabilities returned by `calcAllocations`.
`noTypMemb`	An integer indicating the number of most typical group members to be drawn from each cluster/group.
`moreTypMemb`	A vector with length `noTypMemb` containing the positions (ranks) in the individual posterior classification probability ranking of further (typical) group members.
`grLabels`	A character vector giving user-specified names for the clusters/groups.

Value

A list containing:

`typicalMemb`	The index numbers of the individuals being the first `noTypMemb` most typical group members according to their positions (ranks) in the individual posterior classification probability ranking.
`typicalMemb2`	The index numbers of the individuals being the `moreTypMemb`-th most typical group members. according to their positions (ranks) in the individual posterior classification probability ranking.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 `http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract`

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 `http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf`

See Also

`calcAllocations`, `mcClust`, `dmClust`, `mcClustExtended`, `dmClustExtended`

Examples

```
# please run the examples in mcClust, dmClust, mcClustExtended
# and/or dmClustExtended
```

`transformDataToNjki` *Transform Markov Chain (Time Series) Data Into Transition Frequency Structure*

Description

Transform time series (Markov chain) data with several states/categories into the required Njk.i-structure containing the transition frequencies between these states/categories.

The functions `dataFrameToNjki` and `dataListToNjki` transform time series data representing Markov chains with several states/categories in a format ready for use in `mcClustering` and `dmClustering` and their versions without extension.

The resulting data format is a 3-dim array which contains the absolute transition frequencies stored in a matrix for each individual (see section **Value**).

With `dataFrameToNjki` a `data.frame` or `matrix` where the *rows* contain the time series (implying equal lengths T) can be transformed.

Note that by using a special (different) 'number' (end-of-line) to indicate the (earlier) end (and/or remainder) of a time series (and with which the vector may be filled afterwards), it is also possible to use this procedure when later deleting the corresponding row and column in the transition frequency matrices.

With `dataListToNjki` a `list` of vectors representing the time series (which may have individual lengths T_i) can be transformed.

Usage

```
dataFrameToNjki(dataFrame)
dataListToNjki(dataList)
```

Arguments

`dataFrame`	`data.frame` or `matrix` of dimension $N \times T$ where the i-th row contains the time series of the i-th individual. N is the number of individuals/units/objects and T is the number of columns not necessarily equal to the length of the time series. The time series itself may be of different lengths and the end and/or remainder of the rows are indicated or filled up with a different (special) number (end-of-line; e.g. zero). In such a case it is necessary to delete the corresponding row and column in the resulting transition frequency matrices.
`dataList`	A `list` of N vectors where the i-th entry corresponds to the time series (with possibly individual length T_i) of the i-th individual.

Details

Note that for a single individual the number of *transitions* is always equal to one minus length of time series; that is $T - 1$ or $T_i - 1$, respectively.

The categories/states of the Markov chain and optionally the end-of-line number should have consecutive numbering. By default, either functions DO NOT transform the (original) indexing of the categories/states into $0, \ldots, K$ (e.g. if the original numbering started with 1). The ORIGINAL numbering IS used for the indexing of the (resulting) transition matrices. Note that the number of different categories here is $K + 1$ (see remark in **Note**).

In other words, the (consecutive) numbering of the categories is NOT transformed into $0, \ldots, K$. If an end-of-line or end-of-time-series symbol/number appears (in `dataFrame`) the corresponding rows/columns in the returned 3-dim array (see **Value**) can be deleted afterwards.

Value

A three-dimensional array of format $(K+1) \times (K+1) \times N$ where each i-th matrix represents the transition frequencies of individual i. $(K + 1)$ is equal to the number of different categories/states.

Note

Note, that in contrast to the literature (see **References**), the numbering (labelling) of the states of the categorical outcome variable (time series) in this package is sometimes $0, \ldots, K$ (instead of $1, \ldots, K$), however, there are $K + 1$ categories (states)!

Author(s)

Christoph Pamminger <christoph.pamminger@gmail.com>

References

Sylvia Fruehwirth-Schnatter, Christoph Pamminger, Andrea Weber and Rudolf Winter-Ebmer, (2011), "Labor market entry and earnings dynamics: Bayesian inference using mixtures-of-experts Markov chain clustering". *Journal of Applied Econometrics*. DOI: 10.1002/jae.1249 http://onlinelibrary.wiley.com/doi/10.1002/jae.1249/abstract

Christoph Pamminger and Sylvia Fruehwirth-Schnatter, (2010), "Model-based Clustering of Categorical Time Series". *Bayesian Analysis*, Vol. 5, No. 2, pp. 345-368. DOI: 10.1214/10-BA606 http://ba.stat.cmu.edu/journal/2010/vol05/issue02/pamminger.pdf

See Also

mcClust, dmClust, mcClustExtended, dmClustExtended

Examples

```
# rm(list=ls(all=TRUE))

# set working directory
getwd()
if ( !file.exists("bayesMCClust-wd") ) dir.create("bayesMCClust-wd")
setwd("bayesMCClust-wd")

# define data
data(MCCExampleData)

myObsList <- MCCExampleData$obsList
class(myObsList)
length(myObsList)
myObsList[1:5]  # no end-of-line here!
table( unlist(myObsList) ) # categories consecutively numbered?

njki <- dataListToNjki(myObsList) # generate array for N transition matrices
dim(njki)
njki[,,1:5]  # for verification
apply(njki, c(1, 2), sum) # sum up all transitions of all individuals

tsLength <- sapply(myObsList, length) # calculate time series lengths
table(tsLength) # at least 2? -- corresponds to at least 1 transition

Njk.i <- njki # store Njk.i
# save( Njk.i, file = "Njk_i.RData" )      # save Njk.i in "Njk_i.RData"
```

Index