Package 'poisson.glm.mix'

February 20, 2015

Type Package

Title Fit high dimensional mixtures of Poisson GLMs

Version 1.2

Date 2014-04-17

Author Panagiotis Papastamoulis, Marie-Laure Martin-Magniette, Cathy Maugis-Rabusseau

Maintainer Panagiotis Papastamoulis <papapast@yahoo.gr>

Description High dimensional mixtures of Poisson Generalized Linear models with three different parameterizations of Poisson means are considered. Moreover, partitioning the response variables into a set of blocks is possible. The package estimates parameters via EM algorithm. For an efficient initialization, a random splitting small-EM is introduced.

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2014-04-17 23:14:54

R topics documented:

bjkmodel *EM algorithm for the β_jk (m=1) Poisson GLM mixture.*

Description

This function applies EM algorithm for estimating a K-component mixture of Poisson GLM's, using parameterization $m = 1$, that is the β_{jk} model. Initialization can be done using two different intialization schemes. The first one is a two-step small EM procedure. The second one is a random splitting small EM procedure based on results of a mixture with less components. Output of the function is the updates of the parameters at each iteration of the EM algorithm, the estimate of γ, the estimated clusters and conditional probabilities of the observations, as well as the values of the BIC, ICL and loglikelihood of the model.

Usage

```
bjkmodel(reference, response, L, m, K, nr, maxnr, m1, m2, t1, t2,
         msplit, tsplit, prev.z, prev.clust, start.type,
         prev.alpha, prev.beta)
```

Arguments

reference	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
response	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
L	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
m	positive integer denoting the maximum number of EM iterations.
K	positive integer denoting the number of mixture components.
nr	negative number denoting the tolerance for the convergence of the Newton Raphson iterations.
maxnr	positive integer denoting the maximum number of Newton Raphson iterations.
m1	positive integer denoting the number of iterations for each call of the 1st small EM iterations used by Initialization 1 (init1.1.jk.j).
m2	positive integer denoting the number of iterations for each call of the 2nd small EM iterations used by Initialization 1 (init1.2.jk.j).
t1	positive integer denoting the number of different runs of the 1st small EM used by Initialization 1 (init1.1.jk.j).
t2	positive integer denoting the number of different runs of the 2nd small EM used by Initialization 1 (init1.2.jk.j).
msplit	positive integer denoting the number of different runs for each call of the splitting small EM used by Initialization 2 (init2.jk.j).
tsplit	positive integer denoting the number of different runs for each call of the splitting small EM used by Initialization 2 (init2.jk.j).

`prev.z` numeric array of dimension $n \times (K-1)$ containing the estimates of the posterior probabilities according to the previous run of EM. This is used when Initialization 2 is adopted.

`prev.clust` numeric vector of length n containing the estimated clusters according to the MAP rule obtained by the previous run of EM. This is used when Initialization 2 is adopted.

`start.type` binary variable (1 or 2) indicating the type of initialization (1 for initialization 1 and 2 for initialization 2).

`prev.alpha` numeric array of dimension $J \times (K-1)$ containing the matrix of the ML estimates of the regression constants α_{jk}, $j = 1, \ldots, J$, $k = 1, \ldots, K-1$, based on the previous run of EM algorithm. This is used in case of Initialization 2.

`prev.beta` numeric array of dimension $J \times (K-1) \times T$ containing the matrix of the ML estimates of the regression coefficients $\beta_{jk\tau}$, $j = 1, \ldots, J$, $k = 1, \ldots, K-1$, $\tau = 1, \ldots, T$, based on the previous run of EM algorithm. This is used in case of Initialization 2.

Value

`alpha` numeric array of dimension $t_{EM} \times J \times K$, containing the updates of regression constants α_{jk}^t), $j = 1, \ldots, J$, $k = 1, \ldots, K$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

`beta` numeric array of dimension $t_{EM} \times J \times K \times T$ containing the updates of regression coefficients $\beta_{jk\tau}^t$), $j = 1, \ldots, J$, $k = 1, \ldots, K$, $\tau = 1, \ldots, T$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

`gamma` numeric array of dimension $J \times \max(L)$ containing the MLE of $\gamma_{j\ell}$, $j = 1, \ldots, J$, $\ell = 1, \ldots, L_j$.

`psim` numeric array of dimension $t_{EM} \times K$ containing the updates of mixture weights π_k^t), $k = 1, \ldots, K$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

`clust` numeric vector of length n containing the estimated cluster for each observation according to the MAP rule.

`z` numeric array of dimension $n \times K$ containing the estimated conditional probabilities τ_{ik}, $i = 1, \ldots, n$, $k =, \ldots, K$, according to the last iteration of the EM algorithm.

`bic` numeric, the value of the BIC.

`icl` numeric, the value of the ICL.

`ll` numeric, the value of the loglikelihood, computed according to the `mylogLikePoisMix` function.

Author(s)

Panagiotis Papastamoulis

See Also

`init1.1.jk.j`, `init1.2.jk.j`, `init2.jk.j`

Examples

```
##########################################################
#1.           Example with Initialization 1              #
##########################################################


## load a simulated dataset according to the b_jk model
## number of observations: 500
## design: L=(3,2,1)
data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
y <- sim.data[,-1]
## use Initialization 1 with 2 components
## the number of different 1st small runs equals t1=3,
## each one consisting of m1 = 5 iterations
## the number of different 2nd small runs equals t2=3,
## each one consisting of m2 = 5 iterations
## the maximum number of EM iterations is set to m = 1000.
nc <- 2
run <- bjkmodel(reference=x, response=y, L=c(3,2,1), m=1000, K=nc, nr=-10*log(10),
                maxnr=10, m1=5, m2=5, t1=3, t2=3, msplit, tsplit, prev.z,
                prev.clust, start.type=1, prev.alpha, prev.beta)
## retrieve the iteration that the small em converged:
tem <- length(run$psim)/nc
## print the estimate of regression constants alpha.
run$alpha[tem,,]
## print the estimate of regression coefficients beta.
beta <- run$beta[tem,,,]
## print the estimate of gamma.
run$gamma
## print the estimate of mixture weights.
run$psim[tem,]
## frequency table of the resulting clustering of the
## 500 observations among the 2 components.
table(run$clust)
## print the value of the ICL criterion
run$icl
## print the value of the BIC
run$bic
## print the value of the loglikelihood
run$ll


##########################################################
#2.           Example with Initialization 2              #
##########################################################

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Given the estimates of Example 1, estimate a 3-component mixture using   ~
# Initialization 2. The number of different runs is set to $tsplit=2$ with ~
# each one of them using $msplit=5$ em iterations.                         ~
```

```
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
run.previous<-run
## number of conditions
q <- 3
## number of covariates
tau <- 1
## number of components
nc <- 3
## estimated conditional probabilities for K=2
z <- run.previous$z
## number of iteration that the previous EM converged
ml <- length(run.previous$psim)/(nc - 1)
## estimates of alpha when K=2
alpha <- array(run.previous$alpha[ml, , ], dim = c(q, nc - 1))
## estimates of beta when K=2
beta <- array(run.previous$beta[ml, , , ], dim = c(q, nc - 1, tau))
clust <- run.previous$clust ##(estimated clusters when K=2)

run <- bjkmodel(reference=x, response=y, L=c(3,2,1), m=1000, K=3, nr=-10*log(10),
                maxnr=10, m1, m2, t1, t2, msplit=5, tsplit=2, prev.z=z,
                prev.clust=clust, start.type=2, prev.alpha=alpha, prev.beta=beta)

# retrieve the iteration that EM converged
tem <- length(run$psim)/nc
# estimates of the mixture weights
run$psim[tem,]
# estimates of the regression constants alpha_{jk}, j = 1,2,3, k=1,..,3
run$alpha[tem,,]
# estimates of the regression coefficients beta_{jk\tau}, j = 1,2,3, k=1,..,3, \tau=1
run$beta[tem,,,]



# note: useR should specify larger values for Kmax, m1, m2, t1, t2, msplit and
# tsplit for a complete analysis.
```

---

bjmodel *EM algorithm for the β_j (m=2) Poisson GLM mixture.*

Description

This function applies EM algorithm for estimating a K-component mixture of Poisson GLM's, using parameterization $m = 2$, that is the β_j model. Initialization can be done using two different intialization schemes. The first one is a two-step small EM procedure. The second one is a random splitting small EM procedure based on results of a mixture with less components. Output of the function is the updates of the parameters at each iteration of the EM algorithm, the estimate of γ,

the estimated clusters and conditional probabilities of the observations, as well as the values of the BIC, ICL and loglikelihood of the model.

Usage

```
bjmodel(reference, response, L, m, K, nr, maxnr, m1, m2, t1, t2,
        msplit, tsplit, prev.z, prev.clust, start.type,
        prev.alpha, prev.beta)
```

Arguments

`reference`	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
`response`	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
`L`	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
`m`	positive integer denoting the maximum number of EM iterations.
`K`	positive integer denoting the number of mixture components.
`nr`	negative number denoting the tolerance for the convergence of the Newton Raphson iterations.
`maxnr`	positive integer denoting the maximum number of Newton Raphson iterations.
`m1`	positive integer denoting the number of iterations for each call of the 1st small EM iterations used by Initialization 1 (`init1.1.jk.j`).
`m2`	positive integer denoting the number of iterations for each call of the 2nd small EM iterations used by Initialization 1 (`init1.2.jk.j`).
`t1`	positive integer denoting the number of different runs of the 1st small EM used by Initialization 1 (`init1.1.jk.j`).
`t2`	positive integer denoting the number of different runs of the 2nd small EM used by Initialization 1 (`init1.2.jk.j`).
`msplit`	positive integer denoting the number of different runs for each call of the splitting small EM used by Initialization 2 (`init2.jk.j`).
`tsplit`	positive integer denoting the number of different runs for each call of the splitting small EM used by Initialization 2 (`init2.jk.j`).
`prev.z`	numeric array of dimension $n \times (K-1)$ containing the estimates of the posterior probabilities according to the previous run of EM. This is used when Initialization 2 is adopted.
`prev.clust`	numeric vector of length n containing the estimated clusters according to the MAP rule obtained by the previous run of EM. This is used when Initialization 2 is adopted.
`start.type`	binary variable (1 or 2) indicating the type of initialization (1 for initialization 1 and 2 for initialization 2).
`prev.alpha`	numeric array of dimension $J \times (K-1)$ containing the matrix of the ML estimates of the regression constants α_{jk}, $j = 1, \ldots, J$, $k = 1, \ldots, K-1$, based on the previous run of EM algorithm. This is used in case of Initialization 2.

prev.beta — numeric array of dimension $J \times T$ containing the matrix of the ML estimates of the regression coefficients $\beta_{j\tau}$,$j = 1,\ldots,J$, $\tau = 1,\ldots,T$, based on the previous run of EM algorithm. This is used in case of Initialization 2.

Value

alpha — numeric array of dimension $t_{EM} \times J \times K$ containing the updates of regression constants $\alpha_{jk}^{(t)}$, $j = 1,\ldots,J$, $k = 1,\ldots,K$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

beta — numeric array of dimension $t_{EM} \times J \times T$ containing the updates of regression coefficients $\beta_{j\tau}^{(t)}$, $j = 1,\ldots,J$, $\tau = 1,\ldots,T$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

gamma — numeric array of dimension $J \times \max(L)$ containing the MLE of $\gamma_{j\ell}$, $j = 1,\ldots,J$, $\ell = 1,\ldots,L_j$.

psim — numeric array of dimension $t_{EM} \times K$ containing the updates of mixture weights $\pi_k^{(t)}$, $k = 1,\ldots,K$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

clust — numeric vector of length n containing the estimated cluster for each observation according to the MAP rule.

z — numeric array of length $n \times K$ containing the estimated conditional probabilities τ_{ik}, $i = 1,\ldots,n$, $k =,\ldots,K$, according to the last iteration of the EM algorithm.

bic — numeric, the value of the BIC.

icl — numeric, the value of the ICL.

ll — numeric, the value of the loglikelihood, computed according to the mylogLikePoisMix function.

Author(s)

Panagiotis Papastamoulis

See Also

init1.1.jk.j, init1.2.jk.j, init2.jk.j

Examples

```
############################################################
#1.            Example with Initialization 1               #
############################################################


## load a simulated dataset according to the b_jk model
## number of observations: 500
## design: L=(3,2,1)
data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
```

```
y <- sim.data[,-1]
## use Initialization 1 with 2 components
## the number of different 1st small runs equals t1=2,
## each one consisting of m1 = 5 iterations
## the number of different 2nd small runs equals t2=5,
## each one consisting of m2 = 10 iterations
## the maximum number of EM iterations is set to m = 1000.
nc <- 2
run <- bjmodel(reference=x, response=y, L=c(3,2,1), m=1000, K=nc, nr=-10*log(10),
               maxnr=10, m1=5, m2=10, t1=2, t2=5, msplit, tsplit, prev.z,
               prev.clust, start.type=1, prev.alpha, prev.beta)
## retrieve the iteration that the small em converged:
tem <- length(run$psim)/nc
## print the estimate of regression constants alpha.
run$alpha[tem,,]
## print the estimate of regression coefficients beta.
beta <- run$beta[tem,,]
## print the estimate of gamma.
run$gamma
## print the estimate of mixture weights.
run$psim[tem,]
## frequency table of the resulting clustering of the
## 500 observations among the 2 components.
table(run$clust)
## print the value of the ICL criterion
run$icl
## print the value of the BIC
run$bic
## print the value of the loglikelihood
run$ll


############################################################
#2.            Example with Initialization 2              #
############################################################

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Given the estimates of Example 1, estimate a 11-component mixture using  ~
# Initialization 2. The number of different runs is set to $tsplit=2$ with ~
# each one of them using $msplit=5$ em iterations.                         ~
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
run.previous<-run
## number of conditions
q <- 3
## number of covariates
tau <- 1
## number of components
nc <- 3
## estimated conditional probabilities for K=10
z <- run.previous$z
## number of iteration that the previous EM converged
ml <- length(run.previous$psim)/(nc - 1)
## estimates of alpha when K=2
```

```
alpha <- array(run.previous$alpha[ml, , ], dim = c(q, nc - 1))
## estimates of beta when K=2
beta <- array(run.previous$beta[ml, , ], dim = c(q, tau))
clust <- run.previous$clust ##(estimated clusters when K=2)

run <- bjmodel(reference=x, response=y, L=c(3,2,1), m=1000, K=3, nr=-10*log(10),
                maxnr=10, m1, m2, t1, t2, msplit=5, tsplit=2, prev.z=z,
                prev.clust=clust, start.type=2, prev.alpha=alpha, prev.beta=beta)

# retrieve the iteration that EM converged
tem <- length(run$psim)/nc
# estimates of the mixture weights
run$psim[tem,]
# estimates of the regression constants alpha_{jk}, j = 1,2,3, k=1,..,3
run$alpha[tem,,]
# estimates of the regression coefficients beta_{j\tau}, j = 1,2,3, \tau=1
run$beta[tem,,]


# note: useR should specify larger values for Kmax, m1, m2, t1, t2, msplit
#  and tsplit for a complete analysis.
```

---

bkmodel *EM algorithm for the β_k (m=3) Poisson GLM mixture.*

Description

This function applies EM algorithm for estimating a K-component mixture of Poisson GLM's, using parameterization $m = 3$, that is the β_k model. Initialization can be done using two different intialization schemes. The first one is a one-step small EM procedure. The second one is a random splitting small EM procedure based on results of a mixture with less components. Output of the function is the updates of the parameters at each iteration of the EM algorithm, the estimate of γ, the estimated clusters and conditional probabilities of the observations, as well as the values of the BIC, ICL and loglikelihood of the model.

Usage

```
bkmodel(reference, response, L, m, K, nr, maxnr, t2, m2,
        prev.z, prev.clust, start.type, prev.alpha, prev.beta)
```

Arguments

reference	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
response	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.

`L` numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.

`m` positive integer denoting the maximum number of EM iterations.

`K` positive integer denoting the number of mixture components.

`nr` negative number denoting the tolerance for the convergence of the Newton Raphson iterations.

`maxnr` positive integer denoting the maximum number of Newton Raphson iterations.

`t2` positive integer denoting the number of different runs of the small EM used by Initialization 1 (`init1.k`).

`m2` positive integer denoting the number of iterations for each call of the small EM iterations used by Initialization 1 (`init1.k`).

`prev.z` numeric array of dimension $n \times (K-1)$ containing the estimates of the posterior probabilities according to the previous run of EM. This is used when Initialization 2 is adopted.

`prev.clust` numeric vector of length n containing the estimated clusters according to the MAP rule obtained by the previous run of EM. This is used when Initialization 2 is adopted.

`start.type` binary variable (1 or 2) indicating the type of initialization (1 for initialization 1 and 2 for initialization 2).

`prev.alpha` numeric array of dimension $J \times (K-1)$ containing the matrix of the ML estimates of the regression constants α_{jk}, $j = 1, \ldots, J$, $k = 1, \ldots, K-1$, based on the previous run of EM algorithm. This is used in case of Initialization 2.

`prev.beta` numeric array of dimension $(K-1) \times T$ containing the matrix of the ML estimates of the regression coefficients $\beta_{k\tau}$, $k = 1, \ldots, K-1$, $\tau = 1, \ldots, T$, based on the previous run of EM algorithm. This is used in case of Initialization 2.

Value

`alpha` numeric array of dimension $t_{EM} \times J \times K$ containing the updates of regression constants $\alpha_{jk}^{(t)})$, $j = 1, \ldots, J$, $k = 1, \ldots, K$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

`beta` numeric array of dimension $t_{EM} \times K \times T$ containing the updates of regression coefficients $\beta_{k\tau}^{(t)})$, $k = 1, \ldots, K$, $\tau = 1, \ldots, T$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

`gamma` numeric array of dimension $J \times \max(L)$ containing the MLE of $\gamma_{j\ell}$, $j = 1, \ldots, J$, $\ell = 1, \ldots, L_j$.

`psim` numeric array of dimension $t_{EM} \times K$ containing the updates of mixture weights $\pi_k^{(t)})$, $k = 1, \ldots, K$, for each iteration $t = 1, 2, \ldots, t_{EM}$ of the EM algorithm.

`clust` numeric vector of length n containing the estimated cluster for each observation according to the MAP rule.

`z` numeric array of length $n \times K$ containing the estimated conditional probabilities τ_{ik}, $i = 1, \ldots, n$, $k =, \ldots, K$, according to the last iteration of the EM algorithm.

bic	numeric, the value of the BIC.
icl	numeric, the value of the ICL.
ll	numeric, the value of the loglikelihood, computed according to the mylogLikePoisMix function.

Author(s)

Panagiotis Papastamoulis

See Also

init1.k, init2.k

Examples

```
############################################################
#1.            Example with Initialization 1               #
############################################################


## load a simulated dataset according to the b_jk model
## number of observations: 500
## design: L=(3,2,1)
data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
y <- sim.data[,-1]
## use Initialization 1 with 2 components
## the number of different small runs equals t2=5,
## each one consisting of m1 = 5 iterations
## the maximum number of EM iterations is set to m = 1000.
nc <- 2
run <- bkmodel(reference=x, response=y, L=c(3,2,1), m=1000, K=nc, nr=-10*log(10),
               maxnr=10, t2=5, m2=5, prev.z, prev.clust, start.type=1,
               prev.alpha, prev.beta)
## retrieve the iteration that the small em converged:
tem <- length(run$psim)/nc
## print the estimate of regression constants alpha.
run$alpha[tem,,]
## print the estimate of regression coefficients beta.
beta <- run$beta[tem,,]
## print the estimate of gamma.
run$gamma
## print the estimate of mixture weights.
run$psim[tem,]
## frequency table of the resulting clustering of the
## 500 observations among the 2 components.
table(run$clust)
## print the value of the ICL criterion
run$icl
## print the value of the BIC
run$bic
```

```
## print the value of the loglikelihood
run$ll


############################################################
#2.           Example with Initialization 2               #
############################################################

#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# Given the estimates of Example 1, estimate a 3-component mixture using   ~
# Initialization 2. The number of different runs is set to $t2=2$ with     ~
# each one of them using $m2=5$ em iterations.                             ~
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
run.previous<-run
## number of conditions
q <- 3
## number of covariates
tau <- 1
## number of components
nc <- 3
## estimated conditional probabilities for K=10
z <- run.previous$z
## number of iteration that the previous EM converged
ml <- length(run.previous$psim)/(nc - 1)
## estimates of alpha when K=2
alpha <- array(run.previous$alpha[ml, , ], dim = c(q, nc - 1))
## estimates of beta when K=2
beta <- array(run.previous$beta[ml, , ], dim = c(nc - 1, tau))
clust <- run.previous$clust ##(estimated clusters when K=2)


run <- bkmodel(reference=x, response=y, L=c(3,2,1), m=1000, K=nc, nr=-10*log(10),
               maxnr=10, t2=2, m2=5, prev.z=z, prev.clust=clust, start.type=2,
               prev.alpha=alpha, prev.beta=beta)

# retrieve the iteration that EM converged
tem <- length(run$psim)/nc
# estimates of the mixture weights
run$psim[tem,]
# estimates of the regression constants alpha_{jk}, j = 1,2,3, k=1,..,11
run$alpha[tem,,]
# estimates of the regression coefficients beta_{k\tau}, k = 1,..,11, \tau=1
run$beta[tem,,]

# note: useR should specify larger values for Kmax, m1, m2, t1, t2
# for a complete analysis.
```

`init1.1.jk.j` *1st step of Initialization 1 for the β_jk ($m = 1$) or β_j ($m = 2$) parameterization.*

Description

This function is the first step of the two-step small initialization procedure (Initialization 1), used for the parameterizations $m = 1$ (β_{jk}) or $m = 2$ (β_j). For each condition $j = 1, \ldots, J$, a small EM is run in order to find some good starting values for the K-component mixtures: $\sum_{k=1}^{K} p_j \prod_{\ell=1}^{L_j} f(y_{ij\ell})$, independently for each $j = 1, \ldots, J$. These values are used in order to initialize the second step (`init1.2.jk.j`) of the small EM algorithm for fitting the overall mixture $\sum_{k=1}^{K} \pi_j \prod_{j=1}^{J} \prod_{\ell=1}^{L_j} f(y_{ij\ell})$.

Usage

```
init1.1.jk.j(reference, response, L, K, t1, model, m1,mnr)
```

Arguments

`reference`	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
`response`	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
`L`	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
`K`	positive integer denoting the number of mixture components.
`t1`	positive integer denoting the number of different runs.
`model`	binary variable denoting the parameterization of the model: 1 for β_{jk} and 2 for β_j parameterization.
`m1`	positive integer denoting the number of iterations for each run.
`mnr`	positive integer denoting the maximum number of Newton-Raphson iterations.

Value

`alpha`	numeric array of dimension $J \times K$ containing the selected values $\alpha_{jk}^{(0)}$, $j = 1, \ldots, J$, $k = 1, \ldots, K$ that will be used to initialize the second step of the small EM.
`beta`	numeric array of dimension $J \times K \times T$ (if `model = 1`) or $J \times T$ (if `model = 2`) containing the selected values of $\beta_{jk\tau}^{(t)}$ (or $\beta_{j\tau}^{(t)}$), $j = 1, \ldots, J$, $k = 1, \ldots, K$, $\tau = 1, \ldots, T$, that will be used to initialize the second step of the small EM.
`psim`	numeric vector of length K.
`ll`	numeric, the value of the loglikelihood, computed according to the `mylogLikePoisMix` function.

Author(s)

Panagiotis Papastamoulis

See Also

init1.2.jk.j, bjkmodel, bjmodel

Examples

```
############################################################
#1.            Example with beta_jk (m=1) model            #
############################################################
## load a simulated dataset according to the b_jk model
## number of observations: 500
## design: L=(3,2,1)
data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
y <- sim.data[,-1]
## initialize the component specific parameters
##                   for a 2 component mixture
start1 <- init1.1.jk.j(reference=x, response=y, L=c(3,2,1),
                       K=2, t1=3, model=1, m1=5,mnr = 5)
summary(start1)

############################################################
#2.            Example with beta_j (m=2) model             #
############################################################

start1 <- init1.1.jk.j(reference=x, response=y, L=c(3,2,1),
                       K=2, t1=3, model=2, m1=5,mnr = 5)
summary(start1)
```

`init1.2.jk.j` *2nd step of Initialization 1 for the β_jk ($m = 1$) or β_j ($m = 2$) parameterization.*

Description

This function is the second step of the two-step small initialization procedure (Initialization 1), used for parameterizations $m = 1$ or $m = 2$. At first, `init1.1.jk.j` is called for each condition $j = 1, \ldots, J$. The values obtained from the first step are used for initializing the second step of the small EM algorithm for fitting the overall mixture $\sum_{k=1}^{K} \pi_j \prod_{j=1}^{J} \prod_{\ell=1}^{L_j} f(y_{ij\ell})$. The selected values from the second step are the ones that initialize the EM algorithm (`bjkmodel` or `bjmodel`), when $K = K_{min}$.

Usage

```
init1.2.jk.j(reference, response, L, K, m1, m2, t1, t2, model,mnr)
```

Arguments

`reference`	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
`response`	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
`L`	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
`K`	positive integer denoting the number of mixture components.
`m1`	positive integer denoting the number of iterations for each run of `init1.1.jk.j`.
`m2`	positive integer denoting the number of iterations for each run of `init1.2.jk.j`.
`t1`	positive integer denoting the number of different runs of `init1.1.jk.j`.
`t2`	positive integer denoting the number of different runs of `init1.2.jk.j`.
`model`	binary variable denoting the parameterization of the model: 1 for β_{jk} and 2 for β_j parameterization.
`mnr`	positive integer denoting the maximum number of Newton-Raphson iterations.

Value

`alpha`	numeric array of dimension $J \times K$ containing the selected values $\alpha_{jk}^{(0)}$, $j = 1, \ldots, J$, $k = 1, \ldots, K$ that will be used to initialize main EM.
`beta`	numeric array of dimension $J \times K \times T$ (if `model = 1`) or $J \times T$ (if `model = 2`) containing the selected values of $\beta_{jk\tau}^{(0)}$ (or $\beta_{j\tau}^{(t)}$), $j = 1, \ldots, J$, $k = 1, \ldots, K$, $\tau = 1, \ldots, T$, that will be used to initialize the main EM.
`psim`	numeric vector of length K containing the weights that will initialize the main EM.
`ll`	numeric, the value of the loglikelihood, computed according to the `mylogLikePoisMix` function.

Author(s)

Panagiotis Papastamoulis

See Also

`init1.1.jk.j`, `bjkmodel`, `bjmodel`

Examples

```
############################################################
#1.            Example with beta_jk (m=1) model            #
############################################################
## load a simulated dataset according to the b_jk model
## number of observations: 500
## design: L=(3,2,1)
data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
y <- sim.data[,-1]
## initialize the parameters for a 2 component mixture
## the number of the overall small runs are t2 = 2
## each one consisting of m2 = 2 iterations of the EM.
## the number of the small runs for the first step small EM
## is t1 = 2, each one consisting of m1 = 2 iterations.
start2 <- init1.2.jk.j(reference=x, response=y, L=c(3,2,1),
                       K=2, m1=2, m2=2, t1=2, t2=2, model=1,mnr = 3)
summary(start2)

############################################################
#2.            Example with beta_j (m=2) model             #
############################################################

## initialize the parameters for a 2 component mixture
## the number of the overall small runs are t2 = 3
## each one consisting of m2 = 2 iterations of the EM.
## the number of the small runs for the first step small EM
## is t1 = 2, each one consisting of m1 = 2 iterations.
start2 <- init1.2.jk.j(reference=x, response=y, L=c(3,2,1),
                       K=2, m1=2, m2=2, t1=2, t2=3, model=2,mnr = 5)
summary(start2)
```

`init1.k` *Initialization 1 for the β_k parameterization ($m = 3$).*

Description

This function is the small initialization procedure (Initialization 1) for parameterization $m = 3$. The selected values are the ones that initialize the EM algorithm `bkmodel`.

Usage

```
init1.k(reference, response, L, K, t2, m2,mnr)
```

Arguments

reference	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
response	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
L	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
K	positive integer denoting the number of mixture components.
t2	positive integer denoting the number of different runs.
m2	positive integer denoting the number of iterations for each run.
mnr	positive integer denoting the maximum number of Newton-Raphson iterations.

Value

alpha,	numeric array of dimension $J \times K$ containing the selected values $\alpha_{jk}^{(0)}$, $j = 1, \ldots, J$, $k = 1, \ldots, K$ that will be used to initialize main EM.
beta	numeric array of dimension $K \times T$ containing the selected values of $\beta_{k\tau}^{(0)}$, $k = 1, \ldots, K$, $\tau = 1, \ldots, T$, that will be used to initialize the main EM.
psim	numeric vector of length K containing the weights that will initialize the main EM.
ll	numeric, the value of the loglikelihood, computed according to the mylogLikePoisMix function.

Author(s)

Panagiotis Papastamoulis

See Also

bkmodel, init2.k

Examples

```
## load a simulated dataset according to the b_jk model
## number of observations: 500
## design: L=(3,2,1)
data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
y <- sim.data[,-1]
## initialize the parameters for a 2 component mixture
## the number of the small runs are t2 = 3
## each one consisting of m2 = 5 iterations of the EM.
start1 <- init1.k(reference=x, response=y, L=c(3,2,1),
                      K=2, m2=5, t2=3,mnr = 5)
summary(start1)
```

`init2.jk.j` *Initialization 2 for the β_jk ($m = 1$) or β_j ($m = 2$) parameterization.*

Description

This function applies a random splitting small EM initialization scheme (Initialization 2), for parameterizations $m = 1$ or 2. It can be implemented only in case where a previous run of the EM algorithm is available (with respect to the same parameterization). The initialization scheme proposes random splits of the existing clusters, increasing the number of mixture components by one. Then an EM is ran for (`msplit`) iterations and the procedure is repeated for `tsplit` times. The best values in terms of observed loglikelihood are chosen to initialize the main EM algorithm (`bjkmodel` or `bjmodel`).

Usage

```
init2.jk.j(reference, response, L, K, tsplit, model, msplit,
           previousz, previousclust, previous.alpha, previous.beta,mnr)
```

Arguments

`reference`	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
`response`	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
`L`	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
`K`	positive integer denoting the number of mixture components.
`tsplit`	positive integer denoting the number of different runs.
`model`	binary variable denoting the parameterization of the model: 1 for β_{jk} and 2 for β_j parameterization.
`msplit`	positive integer denoting the number of iterations for each run.
`previousz`	numeric array of dimension $n \times (K-1)$ containing the estimates of the posterior probabilities according to the previous run of EM.
`previousclust`	numeric vector of length \$n\$ containing the estimated clusters according to the MAP rule obtained by the previous run of EM.
`previous.alpha`	numeric array of dimension $J \times (K-1)$ containing the matrix of the ML estimates of the regression constants α_{jk}, $j = 1, \ldots, J$, $k = 1, \ldots, K-1$, based on the previous run of EM algorithm.
`previous.beta`	numeric array of dimension $J \times (K-1) \times T$ (if `model = 1`) or $J \times T$ (if `model = 2`) containing the matrix of the ML estimates of the regression coefficients $\beta_{jk\tau}$ or $\beta_{j\tau}$, $j = 1, \ldots, J$, $k = 1, \ldots, K-1$, $\tau = 1, \ldots, T$, based on the previous run of EM algorithm.
`mnr`	positive integer denoting the maximum number of Newton-Raphson iterations.

Value

alpha	numeric array of dimension $J \times K$ containing the selected values α^0_{jk}), $j = 1, \ldots, J$, $k = 1, \ldots, K$ that will be used to initialize main EM (bjkmodel or bjmodel).
beta	numeric array of dimension $J \times K \times T$ (if model = 1) or $J \times T$ (if model = 2) containing the selected values of $\beta^0_{jk\tau}$) (or $\beta^t_{j\tau}$)), $j = 1, \ldots, J$, $k = 1, \ldots, K$, $\tau = 1, \ldots, T$, that will be used to initialize the main EM.
psim	numeric vector of length K containing the weights that will initialize the main EM.
ll	numeric, the value of the loglikelihood, computed according to the mylogLikePoisMix function.

Note

In case that an exhaustive search is desired instead of a random selection of the splitted components, use tsplit = -1.

Author(s)

Panagiotis Papastamoulis

See Also

init1.1.jk.j, init1.2.jk.j, bjkmodel, bjmodel

Examples

```
data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
y <- sim.data[,-1]

# At first a 2 component mixture is fitted using parameterization $m=1$.
run.previous<-bjkmodel(reference=x, response=y, L=c(3,2,1), m=100, K=2,
                       nr=-10*log(10), maxnr=5, m1=2, m2=2, t1=1, t2=2,
                       msplit, tsplit, prev.z, prev.clust, start.type=1,
                       prev.alpha, prev.beta)
## Then the estimated clusters and parameters are used to initialize a
##   3 component mixture using Initialization 2. The number of different
##   runs is set to $tsplit=3$ with each one of them using msplit = 2
##   em iterations.
q <- 3
tau <- 1
nc <- 3
z <- run.previous$z
ml <- length(run.previous$psim)/(nc - 1)
alpha <- array(run.previous$alpha[ml, , ], dim = c(q, nc - 1))
beta <- array(run.previous$beta[ml, , , ], dim = c(q, nc - 1, tau))
```

```
clust <- run.previous$clust
run<-init2.jk.j(reference=x, response=y, L=c(3,2,1), K=nc, tsplit=2,
                model=1, msplit=2, previousz=z, previousclust=clust,
                previous.alpha=alpha, previous.beta=beta,mnr = 5)
# note: useR should specify larger values for msplit and tsplit for a complete analysis.
```

`init2.k` *Initialization 2 for the β_k parameterization ($m = 3$).*

Description

This function applies a random splitting small EM initialization scheme (Initialization 2), for parameterization $m = 3$. It can be implemented only in case where a previous run of the EM algorithm is available (with respect to the same parameterization). The initialization scheme proposes random splits of the existing clusters, increasing the number of mixture components by one. Then EM is ran for (`m2`) iterations, and the procedure is repeated for `t2` times. The best values in terms of observed loglikelihood are chosen in order to initialize the main EM algorithm (`bkmodel`), when $K > K_{min}$.

Usage

```
init2.k(reference, response, L, K, t2, m2, previousz, previousclust,
        previous.alpha, previous.beta,mnr)
```

Arguments

`reference`	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
`response`	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
`L`	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
`K`	positive integer denoting the number of mixture components.
`t2`	positive integer denoting the number of different runs.
`m2`	positive integer denoting the number of iterations for each run.
`previousz`	numeric array of dimension $n \times (K-1)$ containing the estimates of the posterior probabilities according to the previous run of EM.
`previousclust`	numeric vector of length n containing the estimated clusters according to the MAP rule obtained by the previous run of EM.
`previous.alpha`	numeric array of dimension $J \times (K-1)$ containing the matrix of the ML estimates of the regression constants α_{jk}, $j = 1, \ldots, J$, $k = 1, \ldots, K-1$, based on the previous run of EM algorithm.
`previous.beta`	numeric array of dimension $(K-1) \times T$ containing the matrix of the ML estimates of the regression coefficients $\beta_{k\tau}$, $k = 1, \ldots, K-1$, $\tau = 1, \ldots, T$, based on the previous run of EM algorithm.
`mnr`	positive integer denoting the maximum number of Newton-Raphson iterations.

Value

alpha	numeric array of dimension $J \times K$ containing the selected values $\alpha_{jk}^{(0)}$, $j = 1, \ldots, J$, $k = 1, \ldots, K$ that will be used to initialize main EM.
beta	numeric array of dimension $K \times T$ containing the selected values of $\beta_{k\tau}^{(0)}$, $k = 1, \ldots, K$, $\tau = 1, \ldots, T$, that will be used to initialize the main EM.
psim	numeric vector of length K containing the weights that will initialize the main EM.
ll	numeric, the value of the loglikelihood, computed according to the mylogLikePoisMix function.

Note

In case that an exhaustive search is desired instead of a random selection of the splitted components, uset2 = -1.

Author(s)

Panagiotis Papastamoulis

See Also

init1.k, bkmodel

Examples

```
# this is to be used as an example with the simulated data

data("simulated_data_15_components_bjk")
x <- sim.data[,1]
x <- array(x,dim=c(length(x),1))
y <- sim.data[,-1]

# At first a 2 component mixture is fitted using parameterization $m=1$.
run.previous<-bkmodel(reference=x, response=y, L=c(3,2,1), m=100, K=2,
                      nr=-10*log(10), maxnr=5, m2=3, t2=3, prev.z,
                      prev.clust, start.type=1, prev.alpha, prev.beta)
## Then the estimated clusters and parameters are used to initialize a
##  3 component mixture using Initialization 2. The number of different
##  runs is set to tsplit=3 with each one of them using msplit = 5
##  em iterations.
q <- 3
tau <- 1
nc <- 3
z <- run.previous$z
ml <- length(run.previous$psim)/(nc - 1)
alpha <- array(run.previous$alpha[ml, , ], dim = c(q, nc - 1))
beta <- array(run.previous$beta[ml, , ], dim = c(nc - 1, tau))
clust <- run.previous$clust
run<-init2.k(reference=x, response=y, L=c(3,2,1), K=nc, t2=3, m2=5, previousz=z,
             previousclust=clust, previous.alpha=alpha, previous.beta=beta,mnr = 5)
```

```
summary(run)
# note: useR should specify larger values for m2, t2 for a complete analysis.
```

`mylogLikePoisMix` *Function to compute the loglikelihood of the mixture.*

Description

This function computes the observed loglikelihood given the means and the mixing proportions of each component. Instead of computing $L_i = \log \sum_{k=1}^{K} g_{ik}$, $i = 1, \ldots, n$, where $g_{ik} := \pi_k \prod_{j=1}^{J} \prod_{\ell=1}^{L_j} f(y_{ij\ell}|\mu_{ijlk;m})$, $h_{ik} := \log g_{ik}$ are computed for all i. Let $h_i^* = \max\{h_{ik}, k = 1, \ldots, K\}$, then $L_i = h_i^* + \log \sum_{k=1}^{K} \exp(h_{ik} - h_i^*)$.

Usage

```
mylogLikePoisMix(y, mean, pi)
```

Arguments

`y`	a numeric array of count data with dimension $n \times d$.
`mean`	a list of length K (the number of mixture components) of positive data. Each list element is a matrix with dimension $n \times d$ containing d Poisson means for each of the n observations.
`pi`	a numeric vector of length K (the number of mixture components) containing the mixture weights.

Value

`ll`	the value of the loglikelihood.

Author(s)

Panagiotis Papastamoulis

Examples

```
## This example computes the loglikelihood of a K = 10 component
##      Poisson GLM mixture. The number of response variables is
##      d = 6, while the sample size equals to n = 5000. They are
##      stored in the array sim.data[,-1]. The number of covariates
##      equals 1 (corresponding to sim.data[,1]). We will use a
##      random generation of the regression coefficients alpha and
##      beta, in order to show that the loglikelihood can be computed
##      without computational errors even in cases where the parameters
##      are quite ''bad'' for the data.

data("simulated_data_15_components_bjk_full")
K <- 10
```

```
d <- 6
n <- dim(sim.data)[1]
condmean=vector("list",length=K)
weights<-rep(1,K)/K
ar<-array(data=NA,dim=c(n,d))
for (k in 1:K){
for (i in 1:d){
ar[,i]<-runif(n)+(1+0.1*(runif(n)-1))*sim.data[,1]}
condmean[[k]]<-ar}
mylogLikePoisMix(sim.data[,-1],condmean,weights)
```

`pois.glm.mix` *Main call function of the package.*

Description

This function is the main function of the package. User has only to call it by specifying the data (x and y), the vector L, the parameterization ($m \in \{1, 2, 3\}$), the desirable range for the number of components, the type of initialization and the number of EM runs and iterations for the small-EM strategy. When $K = K_{min}$, EM algorithm is initialized according to Initialization scheme 1 (the functions `init1.1.jk.j`, `init1.2.jk.j`, `init1.k`). For consecutive run ($K > K_{min}$), EM algorithm is initialized using Initialization 2 (the functions `init2.jk.j` or `init2.k`).

Usage

```
pois.glm.mix(reference, response, L, m, max.iter, Kmin, Kmax,
             m1, m2, t1, t2, msplit, tsplit,mnr)
```

Arguments

`reference`	a numeric array of dimension $n \times V$ containing the V covariates for each of the n observations.
`response`	a numeric array of count data with dimension $n \times d$ containing the d response variables for each of the n observations.
`L`	numeric vector of positive integers containing the partition of the d response variables into $J \leq d$ blocks, with $\sum_{j=1}^{J} L_j = d$.
`m`	variable denoting the parameterization of the model: 1 for β_{jk} , 2 for β_j and 3 for β_k parameterization.
`max.iter`	positive integer denoting the maximum number of EM iterations.
`Kmin`	the minimum number of mixture components.
`Kmax`	the maximum number of mixture components.
`m1`	positive integer denoting the number of iterations for each call of the 1st small EM iterations used by Initialization 1 (`init1.1.jk.j`). Leave blank in case of parameterization $m = 3$.

`m2` positive integer denoting the number of iterations for each call of the overall small EM iterations used by Initialization 1 (`init1.2.jk.j` or `init1.k`).

`t1` positive integer denoting the number of different runs of the 1st small EM used by Initialization 1 (`init1.1.jk.j`). Leave blank in case of parameterization $m = 3$.

`t2` positive integer denoting the number of different runs of the overall small EM used by Initialization 1 (`init1.2.jk.j` or `init1.k`).

`msplit` positive integer denoting the number of different runs for each call of the splitting small EM used by Initialization 2 (`init2.jk.j` or `init2.k`).

`tsplit` positive integer denoting the number of different runs for each call of the splitting small EM used by Initialization 2 (`init2.jk.j` or `init2.k`).

`mnr` positive integer denoting the maximum number of Newton-Raphson iterations.

Details

The output of the function is a list of lists. During the run of the function `pois.glm.mix` two `R` graphic devices are opened: The first one contains the graph of the information criteria (BIC and ICL). In the second graphe, the resulting fitted clusters per condition are plotted until the ICL criterion no longer selects a better model. Notice that in this graph the L_j replicates of condition $j = 1, \ldots, J$ are summed.

The EM algorithm is run until the increase to the loglikelihood of the mixture model is less than 10^{-6}. The Newton - Raphson iterations at the Maximization step of EM algorithm are repeated until the square Euclidean norm of the gradient vector of the component specific parameters is less than 10^{-10}.

Value

`information.criteria`
numeric array of dimension $(Kmax - Kmin + 1) \times 3$ containing the values of BIC, ICL and loglikelihood for each K. The latter is computed according to the function `mylogLikePoisMix`.

`runs` A list containing the output for the estimated mixture for each K. The output is the same as in the functions `bjkmodel`, `bjmodel` and `bkmodel`.

`sel.mod.icl` The selected number of mixture components according to the ICL criterion.

`sel.mod.bic` The selected number of mixture components according to the BIC.

`est.sel.mod.icl`
The final estimates for the selected number of mixture components according to the ICL criterion. It is a list containing $\widehat{\pi}_k$, $\widehat{\alpha}_{jk}$, $\widehat{\beta}_{jkv}$, $\widehat{\gamma}_{j\ell}$, $\widehat{c}_i$, $\widehat{\tau}_{ik}$, $i = 1, \ldots, n$, $j = 1, \ldots, J$, $\ell = 1, \ldots, L_j$, $k = 1, \ldots, \widehat{K}_{icl}$, $v = 1, \ldots, V$, while $\widehat{c}_i$ denotes the estimated cluster of observation i, according to the MAP rule.

`est.sel.mod.bic`
The final estimates for the selected number of mixture components according to the BIC. It is a list containing $\widehat{\pi}_k$, $\widehat{\alpha}_{jk}$, $\widehat{\beta}_{jkv}$, $\widehat{\gamma}_{j\ell}$, $\widehat{c}_i$, $\widehat{\tau}_{ik}$, $i = 1, \ldots, n$, $j = 1, \ldots, J$, $\ell = 1, \ldots, L_j$, $k = 1, \ldots, \widehat{K}_{bic}$, $v = 1, \ldots, V$.

Note

In case that an exhaustive search is desired instead of a random selection of the splitted components, use `tsplit = -1`.

Author(s)

Panagiotis Papastamoulis

See Also

bjkmodel, bjmodel, bkmodel, init1.1.jk.j, init1.2.jk.j, init1.k, init2.jk.j, init2.k, mylogLikePoisMix

Examples

```
## load a small dataset of 500 observations
data("simulated_data_15_components_bjk")
## in this example there is V = 1 covariates (x)
##   and d = 6 response variables (y). The design is
##   L = (3,2,1).
V <- 1
x <- array(sim.data[,1],dim=c(dim(sim.data)[1],V))
y <- sim.data[,-1]

## We will run the algorithm using parameterization
##   m = 1 and the number of components in the set
##   {2,3,4}.

rr<-pois.glm.mix(reference=x, response=y, L=c(3,2,1), m=1,
                  max.iter=1000, Kmin=2, Kmax= 4,
                  m1=3, m2=3, t1=3, t2=3, msplit=3, tsplit=3,mnr = 5)

# note: useR should specify larger values for Kmax, m1, m2,
# t1, t2, msplit and tsplit for a complete analysis.


# retrieve the selected models according to BIC or ICL
rr$sel.mod.icl
rr$sel.mod.bic
# retrieve the estimates according to ICL
# alpha
rr$est.sel.mod.icl$alpha
# beta
rr$est.sel.mod.icl$beta
# gamma
rr$est.sel.mod.icl$gamma
# pi
rr$est.sel.mod.icl$pi
# frequency table with estimated clusters
table(rr$est.sel.mod.icl$clust)
# histogram of the maximum conditional probabilities
hist(apply(rr$est.sel.mod.icl$tau,1,max),30)
```

```
##(the full data of 5000 observations can be loaded using
##     data("simulated_data_15_components_bjk_full")
```

`poisson.glm.mix` *Estimation of high dimensional Poisson GLM's via EM algorithm.*

Description

This package can be used to cluster high dimensional count data under the presence of covariates. A mixture of Poisson Generalized Linear models (GLM's) is proposed. Conditionally to the covariates, Poisson multivariate distribution describing each cluster is a product of independent Poisson distributions. Different parameterizations for the slopes are proposed. Case of partioning the response variables into a set of replicates is considered. Poisson GLM mixture is estimated via Expectation Maximization (EM) algorithm with Newton-Raphson steps. An efficient initialization of EM algorithm is proposed to improve parameter estimation. It is a splitting scheme which is combined with a Small EM strategy. The user is referred to the function `pois.glm.mix` for an automatic evaluation of the proposed methodology.

Details

Package:	poisson.glm.mix
Type:	Package
Version:	1.0
Date:	2012-07-17
License:	What license is it under?

Assume that the observed data can be written as $y = (y_1, \ldots, y_n)$ where $y_i = \{y_{ij\ell}; j = 1, \ldots, J, \ell = 1, \ldots, L_j\}$, $y_i \in Z_+^d$, $i = 1, \ldots, n$, with $d = \sum_{j=1}^{J} L_j$ and $L_j \geq 1$, $j = 1, \ldots, J$. Index i denotes the observation, while the vector $L = (L_1, \ldots, L_J)$ defines a partition of the d variables into J blocks: the first block consists of the first L_1 variables, the second block consists of the next L_2 variables and so on. We will refer to j and ℓ using the terms "condition" and "replicate", respectively. In addition to y, consider that a vector of V covariates is observed, denoted by $x_i := \{x_{iv}; v = 1, \ldots, V\}$, for all $i = 1, \ldots, n$. Assume now that conditional to x_i, a model indicator m taking values in the discrete set $\{1, 2, 3\}$ and a positive integer K, the response y_i, is a realization of the corresponding random vector

$$Y_i | x_i, m \sim \sum_{k=1}^{K} \pi_k \prod_{j=1}^{J} \prod_{\ell=1}^{L_j} \mathcal{P}(\mu_{ij\ell k;m})$$

where $\mathcal{P}$ denotes the Poisson distribution. The following parameterizations for the Poisson means $\mu_{ij\ell k;m}$ are considered: If $m = 1$ (the "β_{jk}" parameterization), then

$$\mu_{ij\ell k;m} := \alpha_{jk} + \gamma_{j\ell} + \sum_{v=1}^{V} \beta_{jkv} x_i.$$

If $m = 2$ (the "β_j" parameterization), then

$$\mu_{ij\ell k;m} := \alpha_{jk} + \gamma_{j\ell} + \sum_{v=1}^{V} \beta_{jv} x_i.$$

If $m = 3$ (the "β_k" parameterization), then

$$\mu_{ij\ell k;m} := \alpha_{jk} + \gamma_{j\ell} + \sum_{v=1}^{V} \beta_{kv} x_i.$$

For identifiability purposes assume that $\sum_{\ell=1}^{L_j} \gamma_{j\ell} = 0$, $j = 1, \ldots, J$.

Author(s)

Papastamoulis Panagiotis Maintainer: Papastamoulis Panagiotis <papapast@yahoo.gr>

References

Papastamoulis, P., Martin-Magniette, M.L., and Maugis-Rabusseau, C. (2012). Efficient estimation of high dimensional mixtures of Poisson Generalized Linear Models via the EM algorithm.

Examples

```
## load a small dataset of 500 observations
data("simulated_data_15_components_bjk")
## in this example there is V = 1 covariates (x)
##   and d = 6 response variables (y). The design is
##   L = (3,2,1).
V <- 1
x <- array(sim.data[,1],dim=c(dim(sim.data)[1],V))
y <- sim.data[,-1]

## We will run the algorithm using parameterization
##   m = 1 and the number of components in the set
##   {2,3,4}.

rr<-pois.glm.mix(reference=x, response=y, L=c(3,2,1), m=1,
                 max.iter=1000, Kmin=2, Kmax= 4,
                 m1=3, m2=3, t1=3, t2=3, msplit=4, tsplit=3,mnr = 5)

# note: useR should specify larger values for Kmax, m1, m2, t1,
#  t2, msplit and tsplit for a complete analysis.

# retrieve the selected models according to BIC or ICL
rr$sel.mod.icl
rr$sel.mod.bic
# retrieve the estimates according to ICL
# alpha
rr$est.sel.mod.icl$alpha
# beta
rr$est.sel.mod.icl$beta
```

```
# gamma
rr$est.sel.mod.icl$gamma
# pi
rr$est.sel.mod.icl$pi
# frequency table with estimated clusters
table(rr$est.sel.mod.icl$clust)
# histogram of the maximum conditional probabilities
hist(apply(rr$est.sel.mod.icl$tau,1,max),30)

##(the full data of 5000 observations can be loaded using
##     data("simulated_data_15_components_bjk_full")
```

`sim.data` *Simulated data set of 500 observations*

Description

This is a small dataset of 500 observations according to the β_{jk} parameterization. The number of reference variables is 1 and correspond to the values in the first column of the array `sim.data`. The number of response variables is 6 and correspond to the values in the last 6 column of the array `sim.data`. There are $J = 3$ conditions, while the number of replicates per condition is $L = (3, 2, 1)$.

Usage

```
sim.data
```

Format

A numeric array containing 500×7 observations.

Index