CO2data  
*GNP and CO2 Data Set*

**Description**

This data set gives the gross national product (GNP) per capita in 1996 for various countries as well as their estimated carbon dioxide (CO2) emission per capita for the same year.

**Usage**

CO2data

**Format**

This data frame consists of 28 countries and the following columns:

- **GNP**  The gross national product per capita in 1996.
- **CO2**  The estimated carbon dioxide emission per capita in 1996.
- **country**  An abbreviation pertaining to the country measured (e.g., "GRC" = Greece and "CH" = China).

**References**


---

**compCDF**  
*Plot the Component CDF*

**Description**

Plot the components’ CDF via the posterior probabilities.

**Usage**

compCDF(x, weights)

**Arguments**

- **x**  A matrix containing the raw data. Rows are subjects and columns are repeated measurements.
- **weights**  The weights to compute the empirical CDF; however, most of time they are the posterior probabilities.
Value

compCDF returns an object which is a list with components:

result The component means and standard deviations for a k-component mixture.

plot The plotted component CDF.

References


See Also

makemultdata, multmixmodel.sel, multmixEM.

Examples

## The sulfur content of the coal seams in Texas

A<-c(1.51, 1.92, 1.08, 2.04, 2.14, 1.76, 1.17)
B<-c(1.69, 0.64, .9, 1.41, 1.01, .84, 1.28, 1.59)
C<-c(1.56, 1.22, 1.32, 1.39, 1.54, 1.04, 2.25, 1.49)
D<-c(1.3, .75, 1.26, .69, .62, .9, 1.2, .32)
E<-c(.73, .8, .9, 1.24, .82, .72, .57, 1.18, .54, 1.3)

dis.coal<-makemultdata(A, B, C, D, E, 
cuts = median(c(A, B, C, D, E)))
temp<-multmixEM(dis.coal$y, lambda = dis.coal$lambda, 
theta = dis.coal$theta)

## Now plot the components' CDF via the posterior probabilities

compCDF(dis.coal$x, temp$posterior)

---

depth

Elliptical and Spherical Depth

Description

Computation of spherical or elliptical depth.

Usage

depth(pts, x, Cx = var(x))
Arguments

pts  A kxd matrix containing the k points that one wants to compute the depth. Each row is a point.

x    A nxd matrix containing the reference data. Each row is an observation.

Cx   A dxd scatter matrix for the data x where the default is \( \text{var}(x) \). When \( Cx = I(d) \), it returns the spherical depth.

Value

\text{depth} returns a k-vector where each entry is the elliptical depth of a point in \text{pts}.

Note

\text{depth} is used in \text{regrc}.

References


See Also

\text{regrc}

Examples

\begin{verbatim}
x<-matrix(rnorm(200),nc = 2)
depth(x[1:3, ], x)
\end{verbatim}

| ellipse | Draw Two-Dimensional Ellipse Based on Mean and Covariance |

Description

Draw a two-dimensional ellipse that traces a bivariate normal density contour for a given mean vector, covariance matrix, and probability content.

Usage

\begin{verbatim}
ellipse(mu, sigma, alpha = .05, npoints = 250, newplot = FALSE,
draw = TRUE, ...)
\end{verbatim}
Arguments

mu     A 2-vector giving the mean.
sigma  A 2x2 matrix giving the covariance matrix.
alpha  Probability to be excluded from the ellipse. The default value is alpha = .05, which results in a 95% ellipse.
npoints Number of points comprising the border of the ellipse.
newplot If newplot = TRUE and draw = TRUE, plot the ellipse on a new plot. If newplot = FALSE and draw = TRUE, add the ellipse to an existing plot.
draw   If TRUE, draw the ellipse.
...    Graphical parameters passed to lines or plot command.

Value

ellipse returns an npointsx2 matrix of the points forming the border of the ellipse.

References


See Also

regcr

Examples

## Produce a 95% ellipse with the specified mean and covariance structure.

mu<-c(1, 3)
sigma<-matrix(c(1, .3, .3, 1.5), 2, 2)

ellipse(mu, sigma, npoints = 200, newplot = TRUE)

mixtools initializations

Initializations for Various EM Algorithms in 'mixtools'

Description

Internal initialization functions for EM algorithms in the package mixtools.
logisregmix.init(y, x, N, lambda = NULL, beta = NULL, k = 2)
multmix.init(y, lambda = NULL, theta = NULL, k = 2)
mvnormalmix.init(x, lambda = NULL, mu = NULL, sigma = NULL,
    k = 2, arbmean = TRUE, arbvar = TRUE)
normalmix.init(x, lambda = NULL, mu = NULL, s = NULL, k = 2,
    arbmean = TRUE, arbvar = TRUE)
poisregmix.init(y, x, lambda = NULL, beta = NULL, k = 2)
regmix.init(y, x, lambda = NULL, beta = NULL, s = NULL, k = 2,
    addintercept = TRUE, arbmean = TRUE, arbvar = TRUE)
regmix.mixed.init(y, x, w = NULL, sigma = NULL,
    arb.sigma = TRUE, alpha = NULL, lambda = NULL,
    mu = NULL, R = NULL, arb.R = TRUE, k = 2,
    mixed = FALSE, addintercept.fixed = FALSE,
    addintercept.random = TRUE)
repnormmix.init(x, lambda = NULL, mu = NULL, s = NULL, k = 2,
    arbmean = TRUE, arbvar = TRUE)

Details

These are usually not to be called by the user. Definitions of the arguments appear in the respective EM algorithms.

ldmult

Log-Density for Multinomial Distribution

Description

Return the logarithm of the multinomial density function.

Usage

ldmult(y, theta)

Arguments

y A vector of multinomial counts.
theta A vector of multinomial probabilities. May have same number of components as or one fewer component than y. In the latter case, an extra component is appended so that theta sums to one.

Details

This function is called by multmixEM.
Value

ldmult returns the logarithm of the multinomial density with parameter \( \theta \), evaluated at \( y \).

See Also

multmixEM

Examples

```r
y <- c(2, 2, 10)
theta <- c(.2, .3, .5)
ldmult(y, theta)
```

---

**logisregmixEM**

*EM Algorithm for Mixtures of (Binary) Logistic or Binomial Regressions*

Description

Returns EM algorithm output for mixtures of logistic or binomial regressions with arbitrarily many components.

Usage

```r
logisregmixEM(y, x, N = NULL, lambda = NULL, beta = NULL, k = 2,
               addintercept = TRUE, epsilon = 1e-08,
               maxit = 10000, verb = FALSE)
```

Arguments

- `y`: An n-vector of successes out of N trials.
- `x`: An nxp matrix of predictors. See `addintercept` below.
- `N`: An n-vector of number of trials for binomial regression. If NULL, then \( N \) is an n-vector of 1s for logistic regression.
- `lambda`: Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then `lambda` is random from uniform Dirichlet and number of components is determined by `beta`.
- `beta`: Initial value of `beta` parameters. Should be a pxk matrix, where \( p \) is the number of columns of \( x \) and \( k \) is number of components. If NULL, then `beta` is generated by binning the data into \( k \) bins and using `glm` on the values in each of the bins. If both `lambda` and `beta` are NULL, then number of components is determined by \( k \).
- `k`: Number of components. Ignored unless `lambda` and `beta` are both NULL.
addintercept  If TRUE, a column of ones is appended to the x matrix before the value of p is calculated.

epsilon    The convergence criterion.

maxit      The maximum number of iterations.

verb       If TRUE, then various updates are printed during each iteration of the algorithm.

Value

logisregmixEM returns a list of class mixEM with items:

x         The predictor values.
y         The response values.
lambda    The final mixing proportions.
beta      The final logistic regression coefficients.
loglik    The final log-likelihood.
posterior An nxk matrix of posterior probabilities for observations.
all.loglik A vector of each iteration’s log-likelihood.
ft         A character vector giving the name of the function.

References


See Also

poisregmixEM

Examples

## EM output for data generated from a 2-component binomial regression model.

beta<-matrix(c(1, .5, 2, -.8), 2, 2)
x<-runif(50, 0, 10)
x1<-cbind(1, x)
xbeta<-x1%*%beta
N<-ceiling(runif(50, 50, 75))
w<-rbinom(50, 1, .3)
y<-w*rbinom(50, size = N, prob = (1/(1+exp(-xbeta[, 1])))+
    (1-w)*rbinom(50, size = N, prob =
    (1/(1+exp(-xbeta[, 2]))))
out.1<-logisregmixEM(y, x, N, verb = TRUE)
out.1

## EM output for data generated from a 2-component logistic regression model.

beta<-matrix(c(-10, .1, 20, -.1), 2, 2)
x<-runif(500, 50, 250)
x1<-cbind(1, x)
xbeta<-x1%*%beta
w<-rbinom(500, 1, .3)
y<-w*rbinom(500, size = 1, prob = (1/(1+exp(-xbeta[,]0))))+
   (1-w)*rbinom(500, size = 1, prob =
   (1/(1+exp(-xbeta[,]1))))
out.2<-logisregmixEM(y, x, beta = beta, lambda = c(.3, .7),
   verb = TRUE, epsilon = 1e-5)
out.2

makemultdata

Produce Cutpoint Multinomial Data

Description
Change data into a matrix of multinomial counts using the cutpoint method and generate EM algorithm starting values for a k-component mixture of multinomials.

Usage
makemultdata(..., cuts)

Arguments
...
Either vectors (possibly of different lengths) of raw data or an nxm matrix of data. If ... are vectors of varying length, then makemultdata will create a matrix of size nxm where n is the sample size and m is the length of the vector with maximum length. Those vectors with length less than m will have NAs to make the corresponding row in the matrix of length m. If ... is a matrix, then the rows must correspond to the sample and the columns the repeated measures.

cuts
A vector of cutpoints. This vector is sorted by the algorithm.

Details
The (i, j)th entry of the matrix y (for j < p) is equal to the number of entries in the ith column of x that are less than or equal to cuts[j]. The (i, p)th entry is equal to the number of entries greater than cuts[j].

Value
makemultdata returns an object which is a list with components:

x
An nxm matrix of the raw data.

y
An nxp matrix of the discretized data where p is one more than the number of cutpoints. Each row is a multinomial vector of counts. In particular, each row should sum to the number of repeated measures for that sample.
References


See Also

compCDF, multmixmodel.sel, multmixEM

Examples

```r
## Randomly generated data.
y<-matrix(rpois(70, 6), 10, 7)
cuts<-c(2, 5, 7)
out1<-makemultdata(y, cuts = cuts)
out1

## The sulfur content of the coal seams in Texas.
A<-c(1.51, 1.92, 1.08, 2.04, 2.14, 1.76, 1.17)
B<-c(1.69, 0.64, .9, 1.41, 1.01, .84, 1.28, 1.59)
C<-c(1.56, 1.22, 1.32, 1.39, 1.33, 1.54, 1.04, 2.25, 1.49)
D<-c(1.3, .75, 1.26, .69, .62, .9, 1.2, .32)
E<-c(.73, .8, .9, 1.24, .82, .72, .57, 1.18, .54, 1.3)
out2<-makemultdata(A, B, C, D, E,
cuts = median(c(A, B, C, D, E)))
out2
```

**multmixEM**  
*EM Algorithm for Mixtures of Multinomials*

Description

Return EM algorithm output for mixtures of multinomial distributions.

Usage

```r
multmixEM(y, lambda = NULL, theta = NULL, k = 2, 
maxit = 10000, epsilon = 1e-08, verb = FALSE)
```

Arguments

- **y**  
  An nxp matrix of data (multinomial counts), where n is the sample size and p is the number of multinomial bins.

- **lambda**  
  Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then lambda is random from uniform Dirichlet and number of components is determined by theta.
theta  Initial value of theta parameters. Should be a kxp matrix, where p is
the number of columns of y and k is number of components. Each row
of theta should sum to 1. If NULL, then each row is random from
uniform Dirichlet. If both lambda and theta are NULL, then number of
components is determined by k.

k  Number of components. Ignored unless lambda and theta are NULL.

epsilon  The convergence criterion.

maxit  The maximum number of iterations.

verb  If TRUE, then various updates are printed during each iteration of the
algorithm.

Value

multmixEM returns a list of class mixEM with items:

y  The raw data.

lambda  The final mixing proportions.

theta  The final multinomial parameters.

loglik  The final log-likelihood.

posterior  An nxk matrix of posterior probabilities for observations.

all.loglik  A vector of each iteration’s log-likelihood.

ft  A character vector giving the name of the function.

References


See Also

cmpCDF, makemultdata, multmixmodel.sel

Examples

```r
## The sulfur content of the coal seams in Texas

A <- c(1.51, 1.92, 1.08, 2.04, 2.14, 1.76, 1.17)
B <- c(1.69, 0.64, .9, 1.41, 1.01, .84, 1.28, 1.59)
C <- c(1.56, 1.22, 1.32, 1.39, 1.33, 1.54, 1.04, 2.25, 1.49)
D <- c(1.3, .75, 1.26, .69, .62, .9, 1.2, .32)
E <- c(.73, .8, .9, 1.24, .82, .72, .57, 1.18, .54, 1.3)

dis.coal <- makemultdata(A, B, C, D, E,
                        cuts = median(c(A, B, C, D, E)))
em.out <- multmixEM(dis.coal$y, lambda = dis.coal$lambda,
         theta = dis.coal$theta)
em.out[1:4]
```

10
multimixmodel.sel  Model Selection Mixtures of Multinomials

Description
Assess the number of components in a mixture of multinomials model using the Akaike’s information criterion (AIC), Schwartz’s Bayesian information criterion (BIC), Bozdogan’s consistent AIC (CAIC), and Integrated Completed Likelihood (ICL).

Usage
multimixmodel.sel(y, comps = NULL, ...)

Arguments
y  A matrix of multinomial counts. An nxp matrix, where n is the sample size and p is the number of bins.
comps  Vector containing the numbers of components to consider. If NULL, this is set to be 1:(max possible), where (max possible) is floor((m+1)/2) and m is the minimum row sum of y.
...  Additional arguments passed to multimixEM.

Value
multimixmodel.sel returns a table summarizing the AIC, BIC, CAIC, ICL, and log-likelihood values along with the winner (the number with the lowest aforementioned values).

See Also
compCDF, makemultdata, multimixEM

Examples
##Data generated using the multinomial cutpoint method.
x<-matrix(rpois(70, 6), 10, 7)
x.new<-makemultdata(x, cuts = 5)
multimixmodel.sel(x.new$y, comps = c(1,2))
Description

Return EM algorithm output for mixtures of multivariate normal distributions.

Usage

`mvnormalmixEM(x, lambda = NULL, mu = NULL, sigma = NULL, k = 2,
               arbmean = TRUE, arbvar = TRUE, epsilon = 1e-08,
               maxit = 10000, verb = FALSE)`

Arguments

- **x**: A matrix of size nxp consisting of the data.
- **lambda**: Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then `lambda` is random from uniform Dirichlet and number of components is determined by `mu`.
- **mu**: A list of size k consisting of initial values for the p-vector mean parameters. If NULL, then the vectors are generated from a normal distribution with mean and standard deviation according to a binning method done on the data. If both `lambda` and `mu` are NULL, then number of components is determined by `sigma`.
- **sigma**: A list of size k consisting of initial values for the pxp variance-covariance matrices. If NULL, then `sigma` is generated using the data. If `lambda`, `mu`, and `sigma` are NULL, then number of components is determined by `k`.
- **k**: Number of components. Ignored unless `lambda`, `mu`, and `sigma` are all NULL.
- **arbmean**: If TRUE, then the component densities are allowed to have different `mu`s. If FALSE, then a scale mixture will be fit.
- **arbvar**: If TRUE, then the component densities are allowed to have different `sigma`es. If FALSE, then a location mixture will be fit.
- **epsilon**: The convergence criterion.
- **maxit**: The maximum number of iterations.
- **verb**: If TRUE, then various updates are printed during each iteration of the algorithm.

Value

`mvnormalmixEM` returns a list of class `mixEM` with items:

- **x**: The raw data.
- **lambda**: The final mixing proportions.
mu         A list of with the final mean vectors.
sigma      A list with the final variance-covariance matrices.
loglik     The final log-likelihood.
posterior  An nxk matrix of posterior probabilities for observations.
all.loglik  A vector of each iteration’s log-likelihood.
ft          A character vector giving the name of the function.

References


See Also

*normalmixEM*

Examples

```r
##Fitting randomly generated data with a 2-component location mixture of bivariate normals.
x.1<-rmvnorm(40, c(0, 0))
x.2<-rmvnorm(60, c(3, 4))
X.1<-rbind(x.1, x.2)
mu<-list(c(0, 0), c(1, 1))
out.1<-mvnormalmixEM(X.1, arbvar = FALSE, mu = mu)
out.1[2:5]

##Fitting randomly generated data with a 2-component scale mixture of bivariate normals.
x.3<-rmvnorm(40, c(0, 0), sigma = matrix(c(200, 1, 1, 150), 2, 2))
x.4<-rmvnorm(60, c(0, 0))
X.2<-rbind(x.3, x.4)
out.2<-mvnormalmixEM(X.2, arbmean = FALSE)
out.2[2:5]
```

NOdata      *Ethanol Fuel Data Set*

Description

This data set gives the equivalence ratios and peak nitrogen oxide emissions in a study using pure ethanol as a spark-ignition engine fuel.

Usage

NOdata
Format

This data frame consists of:

NO  The peak nitrogen oxide emission levels.

Equivalence  The equivalence ratios for the engine at compression ratios from 7.5 to 18.

Source


References


normalmixEM  

*EM Algorithm for Mixtures of Univariate Normals*

Description

Return EM algorithm output for mixtures of normal distributions.

Usage

normalmixEM(x, lambda = NULL, mu = NULL, sigma = NULL, k = 2, 
arbmean = TRUE, arbvar = TRUE, epsilon = 1e-08, 
maxit = 10000, verb = FALSE)

Arguments

x  
A vector of length n consisting of the data.

lambda  
Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then lambda is random from uniform Dirichlet and number of components is determined by mu.

mu  
A k-vector of initial values for the mean parameters. If NULL, then the vector is generated from a normal distribution according to a binning method done on the data. If both lambda and mu are NULL, then number of components is determined by sigma.

sigma  
A k-vector of initial values for the standard deviation parameters. If NULL, then 1/sigma^2 has random exponential entries according to a binning method done on the data. If lambda, mu, and sigma are NULL, then number of components is determined by k.

k  
Number of components. Ignored unless lambda, mu, and sigma are all NULL.
arbmean  If TRUE, then the component densities are allowed to have different \textit{mu}s. If FALSE, then a scale mixture will be fit.

arbvar  If TRUE, then the component densities are allowed to have different \textit{sigma}s. If FALSE, then a location mixture will be fit.

epsilon  The convergence criterion.

maxit  The maximum number of iterations.

verb  If TRUE, then various updates are printed during each iteration of the algorithm.

Value

\texttt{normalmixEM} returns a list of class \texttt{mixEM} with items:

- \texttt{x}  The raw data.
- \texttt{lambda}  The final mixing proportions.
- \texttt{mu}  The final mean parameters.
- \texttt{sigma}  The final standard deviations. If \texttt{arbmean} = FALSE, then only the smallest standard deviation is returned. See \texttt{scale} below.
- \texttt{scale}  If \texttt{arbmean} = FALSE, then the scale factor for the component standard deviations is returned. Otherwise, this is omitted from the output.
- \texttt{loglik}  The final log-likelihood.
- \texttt{posterior}  An nxk matrix of posterior probabilities for observations.
- \texttt{all.loglik}  A vector of each iteration’s log-likelihood.
- \texttt{ft}  A character vector giving the name of the function.

References


See Also

\texttt{mvnormalmixEM}

Examples

```r
##Analyzing the Old Faithful geyser data with a 2-component mixture of normals.

data(faithful)
attach(faithful)
out<-normalmixEM(waiting, arbvar = FALSE)
out
```
normmix.sim  

Simulate from Mixtures of Normals

Description
Simulate from a mixture of univariate normal distributions.

Usage

`normmix.sim(n, lambda, mu, sigma, m = 1)`

Arguments

- `n`: Number of cases to simulate.
- `lambda`: Vector of mixture probabilities, with length equal to desired number of components. This is assumed to sum to 1; if not, it is normalized.
- `mu`: Vector of means.
- `sigma`: Vector of standard deviations.
- `m`: Number of repeated measurements per case.

Value

`normmix.sim` returns an nxm matrix in which each row is an i.i.d. sample from one of the components of a mixture of univariate normals. Every entry of the matrix has a marginal distribution equal to a mixture of normals, though there is dependence among observations in the same row due to the fact that the component is held fixed in each row.

See Also

`makemultdata`

Examples

```r
##Generate data from a 2-component mixture of normals.

n<-500
lambda<-rep(1, 2)/2
mu<-c(0, 5)
sigma<-rep(1, 2)
mixnorm.data<-normmix.sim(n, lambda, mu, sigma)

##A histogram of the simulated data.
hist(mixnorm.data)
```
plot.mixMCMC  Various Plots Pertaining to Mixture Model Output Using MCMC Methods

Description

Takes an object of class mixMCMC and returns various graphical output for select mixture models.

Usage

## S3 method for class 'mixMCMC':
plot(x, trace.plots = TRUE,
     summary.plots = FALSE, burnin = 2000,
     ...)  

Arguments

x  An object of class mixMCMC.

trace.plots  If TRUE, trace plots of the various parameters estimated by the MCMC methods is given.

summary.plots  Graphics pertaining to certain mixture models. The details are given below.

burnin  The values 1 to burnin are dropped when producing the plots in summary.plots.

...  Graphical parameters passed to regcr function.

Value

plot.mixMCMC returns trace plots of the various parameters estimated by the MCMC methods for all objects of class mixMCMC. In addition, other plots may be produced for the following k-component mixture model functions:

regmixMH  Credible bands for the regression lines in a mixture of linear regressions. See regcr for more details.

See Also

regcr

Examples

## M-H algorithm for NOdata with acceptance rate about 40%.

data(NOdata)
attach(NOdata)
beta<-matrix(c(1.3, -0.1, 0.6, 0.1), 2, 2)
sigma<-c(.02, .05)
MH.out<-regmixMH(Equivalence, NO, beta = beta, s = sigma,
sampsize = 10000, omega = .0013
plot(MH.out, summary.plots = TRUE, burnin = 9950, alpha = 0.01)

### plot.mixEM

#### Various Plots Pertaining to Mixture Models

**Description**

Takes an object of class mixEM and returns various graphical output for select mixture models.

**Usage**

```r
## S3 method for class 'mixEM':
plot(x, loglik = TRUE, density = FALSE, 
     w = 1.0, alpha = 0.05, marginal = FALSE, 
     ...)
```

**Arguments**

- **x**: An object of class mixEM.
- **loglik**: If TRUE, a plot of the log-likelihood versus the EM iterations is given.
- **density**: Graphics pertaining to certain mixture models. The details are given below.
- **w**: A graphical parameter to control the height of the y-axis on the histogram when the univariate mixture of normal component’s density curves are overlaid.
- **alpha**: A vector of significance levels when constructing confidence ellipses and confidence bands for the mixture of multivariate normals and mixture of regressions cases, respectively. The default is 0.05.
- **marginal**: For the mixture of bivariate normals, should optional marginal histograms be included?
- **...**: Graphical parameters passed to `plot` command.

**Value**

`plot.mixEM` returns a plot of the log-likelihood versus the EM iterations by default for all objects of class mixEM. In addition, other plots may be produced for the following k-component mixture model functions:

- **normalmixEM**: A histogram of the raw data is produced along with k density curves determined by `normalmixEM`.
- **repnormmixEM**: A histogram of the raw data produced in a similar manner as for `normalmixEM`. 
mvnormalmixEM  A 2-dimensional plot with each point color-coded to denote its most probable component membership. In addition, the estimated component means are plotted along with (1 - alpha)% bivariate normal density contours. These ellipses are constructed by assigning each value to their component of most probable membership and then using normal theory. Optional marginal histograms may also be produced.

regmixEM  A plot of the response versus the predictor with each point color-coded to denote its most probable component membership. In addition, the estimated component regression lines are plotted along with (1 - alpha)% Working-Hotelling confidence bands. These bands are constructed by assigning each value to their component of most probable membership and then performing least squares estimation.

logisregmixEM  A plot of the binary response versus the predictor with each point color-coded to denote its most probable component membership. In addition, the estimate component logistic regression lines are plotted.

regmixEM.mixed  Provides a 2x2 matrix of plots summarizing the posterior slope and posterior intercept terms from a mixture of random effects regression. See post.beta for a more detailed description.

See Also

post.beta

Examples

##Analyzing the Old Faithful geyser data with a 2-component mixture of normals.

data(faithful)
attach(faithful)
out<-normalmixEM(waiting, arbvar = FALSE, verb = TRUE)
plot(out, density = TRUE, w = 1.1)

##Fitting randomly generated data with a 2-component location mixture of bivariate normals.

x.1<-rmvnorm(40, c(0, 0))
x.2<-rmvnorm(60, c(3, 4))
X.1<-rbind(x.1, x.2)

out.1<-mvnormalmixEM(X.1, arbvar = FALSE, verb = TRUE)
plot(out.1, density = TRUE, alpha = c(0.01, 0.05, 0.10),
     marginal = TRUE)

poisregmixEM  EM Algorithm for Mixtures of Poisson Regressions
Description

Returns EM algorithm output for mixtures of Poisson regressions with arbitrarily many components.

Usage

poisregmixEM(y, x, lambda = NULL, beta = NULL, k = 2,
            addintercept = TRUE, epsilon = 1e-08,
            maxit = 10000, verb = FALSE)

Arguments

y An n-vector of response values.
x An nxp matrix of predictors. See addintercept below.
lambda Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then lambda is random from uniform Dirichlet and number of components is determined by beta.
beta Initial value of beta parameters. Should be a pxk matrix, where p is the number of columns of x and k is number of components. If NULL, then beta is generated by binning the data into k bins and using glm on the values in each of the bins. If both lambda and beta are NULL, then number of components is determined by k.
k Number of components. Ignored unless lambda and beta are both NULL.
addintercept If TRUE, a column of ones is appended to the x matrix before the value of p is calculated.
epsilon The convergence criterion.
maxit The maximum number of iterations.
verb If TRUE, then various updates are printed during each iteration of the algorithm.

Value

poisregmixEM returns a list of class mixEM with items:
x The predictor values.
y The response values.
lambda The final mixing proportions.
beta The final Poisson regression coefficients.
loglik The final log-likelihood.
posterior An nxk matrix of posterior probabilities for observations.
all.loglik A vector of each iteration’s log-likelihood.
ft A character vector giving the name of the function.
References

See Also
logisregmixEM

Examples

```r
## EM output for data generated from a 2-component model.

beta <- matrix(c(1, .5, .7, -.8), 2, 2)
x <- runif(50, 0, 10)
xbeta <- cbind(1, x) %*% beta
w <- rbinom(50, 1, .5)
y <- w * rpois(50, exp(xbeta[, 1])) + (1 - w) * rpois(50, exp(xbeta[, 2]))
out <- poisregmixEM(y, x, verb = TRUE)
out
```

<table>
<thead>
<tr>
<th>post.beta</th>
<th>Summary of Posterior Regression Coefficients in Mixtures of Random Effects Regressions</th>
</tr>
</thead>
</table>

Description

Returns a 2x2 matrix of plots summarizing the posterior intercept and slope terms in a mixture of random effects regression with arbitrarily many components.

Usage

```
post.beta(y, x, p.beta, p.z)
```

Arguments

- **y**: A list of N response trajectories with (possibly) varying dimensions of length \( n_i \).
- **x**: A list of N predictor values of dimension \( n_i \). Each trajectory in \( y \) has its own design vector.
- **p.beta**: A list of N 2xk matrices giving the posterior intercept and slope values from the output of an EM algorithm.
- **p.z**: An Nxk matrix of posterior membership probabilities from the output of an EM algorithm.

Details

This is primarily used for within `plot.mixEM`. 

21
Value

post.beta returns a 2x2 matrix of plots giving:

(1, 1) The data plotted on the x-y axes with all posterior regression lines.
(1, 2) The data plotted on the x-y axes with most probable posterior regression lines.
(2, 1) A beta-space plot of all posterior regression coefficients.
(1, 1) A beta-space plot of most probable posterior regression coefficients.

See Also
regmixEM.mixed, plot.mixEM

Examples

```r
## EM output for simulated data from 2-component mixture of random effects.
library(MASS)
data(RanEffdata)
x<-lapply(1:length(RanEffdata), function(i) matrix(RanEffdata[[i]][, 2:3], ncol = 2))
y<-lapply(1:length(RanEffdata), function(i) matrix(RanEffdata[[i]][, 1], ncol = 1))
em.out<-regmixEM.mixed(y, x, addintercept.random = FALSE)

## Obtaining the 2x2 matrix of plots.
x.ran<-lapply(1:length(RanEffdata), function(i) RanEffdata[[i]][,3])
p.beta<-em.out$posterior.beta
p.z<-em.out$posterior.z
post.beta(y, x.ran, p.beta = p.beta, p.z = p.z)
```

<table>
<thead>
<tr>
<th>RanEffdata</th>
<th>Simulated Data from 2-Component Mixture of Regressions with Random Effects</th>
</tr>
</thead>
</table>

Description

This data set was generated from a 2-component mixture of regressions with random effects.

Usage

RanEffdata

Format

This data set consists of a list with 100 25x3 matrices. The first column is the response variable, the second column is a column of 1’s and the last column is the predictor variable.
See Also

regmixEM.mixed

---

**regcr**  
Add a Bayesian Credible Region for Regression Lines to a Scatterplot

---

Description

Produce a credible region for regression lines based on a sample of beta parameters (e.g., a sample from the posterior distribution). The beta parameters are the intercept and slope from a simple linear regression.

Usage

```r
regcr(beta, x, alpha = .05, nonparametric = FALSE, plot = FALSE, xyaxes = TRUE, ...)
```

Arguments

- **beta**: An nx2 matrix of regression parameters. The first column gives the intercepts and the second column gives the slopes.
- **x**: An n-vector of the predictor variable which is necessary when nonparametric = TRUE.
- **alpha**: The proportion of the beta sample to remove. In other words, 1-alpha is the level of the credible region.
- **nonparametric**: If nonparametric = TRUE, then the region is based on the convex hull of the remaining beta after trimming, which is accomplished using a data depth technique. If nonparametric = FALSE, then the region is based on the asymptotic normal approximation.
- **plot**: If plot = TRUE, lines are added to the existing plot. The type of plot created depends on the value of xyaxes.
- **xyaxes**: If xyaxes = TRUE and plot = TRUE, then a credible region for the regression lines is plotted on the x-y axes, presumably overlaid on a scatterplot of the data. If xyaxes = FALSE and plot = TRUE, the (convex) credible region for the regression line is plotted on the beta, or intercept-slope, axes, presumably overlaid on a scatterplot of beta.
- **...**: Graphical parameters passed to `lines` or `plot` command.

Value

**regcr** returns a list containing the following items:

- **boundary**: A matrix of points in beta, or intercept-slope, space arrayed along the boundary of the credible region.
upper  A matrix of points in x-y space arrayed along the upper credible limit for the regression line.

lower  A matrix of points in x-y space arrayed along the lower credible limit for the regression line.

See Also

regmixEM, regmixMH

Examples

## Nonparametric credible regions for mixtures of regressions fit to NOdata.

data(NOdata)
attach(NOdata)
beta<-matrix(c(1.3, -0.1, 0.6, 0.1), 2, 2)
sigma<-c(.02, .05)
MH.out<-regmixMH(Equivalence, NO, beta = beta, s = sigma, sampsize = 10000, omega = .0013)
attach(data.frame(MH.out$theta))
beta.c1<-cbind(beta0.1[9900:9999], beta1.1[9900:9999])
beta.c2<-cbind(beta0.2[9900:9999], beta1.2[9900:9999])
plot(NO, Equivalence)
regcr(beta.c1, x = NO, nonparametric = TRUE, plot = TRUE, col = 2)
regcr(beta.c2, x = NO, nonparametric = TRUE, plot = TRUE, col = 3)

regmixEM.mixed  EM Algorithm for Mixtures of Regressions with Random Effects

Description

Returns EM algorithm output for mixtures of multiple regressions with random effects and an option to incorporate mixed effects.

Usage

regmixEM.mixed(y, x, w = NULL, sigma = NULL, arb.sigma = TRUE, alpha = NULL, lambda = NULL, mu = NULL, R = NULL, arb.R = TRUE, k = 2, mixed = FALSE, addintercept.fixed = FALSE, addintercept.random = TRUE, epsilon = 1e-08, maxit = 10000, verb = FALSE)
Arguments

**y**  
A list of N response trajectories with (possibly) varying dimensions of length \( n_i \).

**x**  
A list of N design matrices of dimensions \((n_i) \times p\). Each trajectory in \( y \) has it's own design matrix.

**w**  
A list of N known explanatory variables having dimensions \((n_i) \times q\). If \( \text{mixed} = \text{FALSE} \), then \( w \) is replaced by a list of N zeros.

**sigma**  
A vector of standard deviations. If NULL, then \(1/\sigma^2\) has random standard exponential entries according to a binning method done on the data.

**arb.sigma**  
If TRUE, then \( \sigma \) is k-dimensional. Else a common standard deviation is assumed.

**alpha**  
A q-vector of unknown regression parameters for the fixed effects. If NULL and \( \text{mixed} = \text{TRUE} \), then \( \alpha \) is random from a normal distribution with mean and variance according to a binning method done on the data. If \( \text{mixed} = \text{FALSE} \), then \( \alpha = 0 \).

**lambda**  
Initial value of mixing proportions for the assumed mixture structure on the regression coefficients. Entries should sum to 1. This determines number of components. If NULL, then \( \lambda \) is random from uniform Dirichlet and the number of components is determined by \( \mu \).

**mu**  
A \( p \times k \) matrix of the mean for the mixture components of the random regression coefficients. If NULL, then the columns of \( \mu \) are random from a multivariate normal distribution with mean and variance determined by a binning method done on the data.

**R**  
A list of N \( p \times p \) covariance matrices for the mixture components of the random regression coefficients. If NULL, then each matrix is random from a standard Wishart distribution according to a binning method done on the data.

**arb.R**  
If TRUE, then \( R \) is a list of N \( p \times p \) covariance matrices. Else, one common covariance matrix is assumed.

**k**  
Number of components. Ignored unless \( \lambda \) is NULL.

**mixed**  
If TRUE, then fixed effects are incorporated. If FALSE, then only random effect are incorporated.

**addintercept.fixed**  
If TRUE, a column of ones is appended to the matrices in \( w \).

**addintercept.random**  
If TRUE, a column of ones is appended to the matrices in \( x \) before \( p \) is calculated.

**epsilon**  
The convergence criterion.

**maxit**  
The maximum number of iterations.

**verb**  
If TRUE, then various updates are printed during each iteration of the algorithm.
Value

regmixEM returns a list of class mixEM with items:

- **x**: The predictor values.
- **y**: The response values.
- **lambda**: The final mixing proportions.
- **mu**: The final mean vectors.
- **R**: The final covariance matrices.
- **sigma**: The final component error variances.
- **alpha**: The final regression coefficients for the fixed effects.
- **loglik**: The final log-likelihood.
- **posterior.z**: An Nxk matrix of posterior membership probabilities.
- **posterior.beta**: A list of N pxk matrices giving the posterior regression coefficient values.
- **all.loglik**: A vector of each iteration's log-likelihood.
- **ft**: A character vector giving the name of the function.

References


See Also

regmixEM, post.beta

Examples

```r
## EM output for simulated data from 2-component mixture of random effects.

data(RanEffdata)
x<-lapply(1:length(RanEffdata), function(i)
    matrix(RanEffdata[[i]][, 2:3], ncol = 2))
x<-x[1:20]
y<-lapply(1:length(RanEffdata), function(i)
    matrix(RanEffdata[[i]][, 1], ncol = 1))
y<-y[1:20]
em.out<-regmixEM.mixed(y, x, addintercept.random = FALSE,
    epsilon = 1e-02, verb = TRUE)
em.out[3:8]
```
Description

Returns EM algorithm output for mixtures of multiple regressions with arbitrarily many components.

Usage

```r
regmixEM(y, x, lambda = NULL, beta = NULL, sigma = NULL, k = 2,
          addintercept = TRUE, arbmean = TRUE, arbvar = TRUE,
          epsilon = 1e-08, maxit = 10000, verb = FALSE)
```

Arguments

- `y` An n-vector of response values.
- `x` An nxp matrix of predictors. See `addintercept` below.
- `lambda` Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then `lambda` is random from uniform Dirichlet and number of components is determined by `beta`.
- `beta` Initial value of `beta` parameters. Should be a pxk matrix, where p is the number of columns of `x` and k is number of components. If NULL, then `beta` has standard normal entries according to a binning method done on the data. If both `lambda` and `beta` are NULL, then number of components is determined by `sigma`.
- `sigma` A vector of standard deviations. If NULL, then 1/sigma^2 has random standard exponential entries according to a binning method done on the data. If `lambda`, `beta`, and `sigma` are NULL, then number of components is determined by k.
- `k` Number of components. Ignored unless all of `lambda`, `beta`, and `sigma` are NULL.
- `addintercept` If TRUE, a column of ones is appended to the `x` matrix before the value of p is calculated.
- `arbmean` If TRUE, each mixture component is assumed to have a different set of regression coefficients (i.e., the `betas`).
- `arbvar` If TRUE, each mixture component is assumed to have a different `sigma`.
- `epsilon` The convergence criterion.
- `maxit` The maximum number of iterations.
- `verb` If TRUE, then various updates are printed during each iteration of the algorithm.
Value

`regmixEM` returns a list of class `mixEM` with items:

- **x**  The set of predictors (which includes a column of 1’s if `addintercept = TRUE`.
- **y**  The response values.
- **lambda**  The final mixing proportions.
- **beta**  The final regression coefficients.
- **sigma**  The final standard deviations. If `arbmean = FALSE`, then only the smallest standard deviation is returned. See `scale` below.
- **scale**  If `arbmean = FALSE`, then the scale factor for the component standard deviations is returned. Otherwise, this is omitted from the output.
- **loglik**  The final log-likelihood.
- **posterior**  An nxk matrix of posterior probabilities for observations.
- **all.loglik**  A vector of each iteration’s log-likelihood.
- **ft**  A character vector giving the name of the function.

References


See Also

`regcr, regmixMH`

Examples

```r
## EM output for NOdata.
data(NOdata)
attach(NOdata)
em.out<-regmixEM(Equivalence, NO, verb = TRUE)
em.out[3:6]
```

---

**regmixMH**  *Metropolis-Hastings Algorithm for Mixtures of Regressions*

Description

Return Metropolis-Hastings (M-H) algorithm output for mixtures of multiple regressions with arbitrarily many components.
Usage

```r
gmixMH(y, x, lambda = NULL, beta = NULL, s = NULL, k = 2,
    addintercept = TRUE, mu = NULL, sig = NULL,
    sampsize = 1000, omega = .01, thin = 1)
```

Arguments

- **y**: An n-vector of response values.
- **x**: An nxp matrix of predictors. See `addintercept` below.
- **lambda**: Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then `lambda` is random from uniform Dirichlet and number of components is determined by `beta`.
- **beta**: Initial value of `beta` parameters. Should be a pxk matrix, where p is the number of columns of x and k is number of components. If NULL, then `beta` has uniform standard normal entries. If both `lambda` and `beta` are NULL, then number of components is determined by `s`.
- **s**: k-vector of standard deviations. If NULL, then 1/s^2 has random standard exponential entries. If `lambda`, `beta`, and `s` are NULL, then number of components determined by `k`.
- **k**: Number of components. Ignored unless all of `lambda`, `beta`, and `s` are NULL.
- **addintercept**: If TRUE, a column of ones is appended to the x matrix before the value of p is calculated.
- **mu**: The prior hyperparameter of same size as `beta`; the means of `beta` components. If NULL, these are set to zero.
- **sig**: The prior hyperparameter of same size as `beta`; the standard deviations of `beta` components. If NULL, these are all set to five times the overall standard deviation of y.
- **sampsize**: Size of posterior sample returned.
- **omega**: Multiplier of step size to control M-H acceptance rate. Values closer to zero result in higher acceptance rates, generally.
- **thin**: Lag between parameter vectors that will be kept.

Value

`gmixMH` returns a list of class `mixMCMC` with items:

- **x**: A nxp matrix of the predictors.
- **y**: A vector of the responses.
- **theta**: A `(sampsize/thin) x q` matrix of MCMC-sampled q-vectors, where q is the total number of parameters in `beta`, `s`, and `lambda`.
- **k**: The number of components.
References


See Also

regcr

Examples

```r
## M-H algorithm for NOdata with acceptance rate about 40%.

data(NOdata)
attach(NOdata)
beta<-matrix(c(1.3, -0.1, 0.6, 0.1), 2, 2)
sigma<-c(.02, .05)
MH.out<-regmixMH(Equivalence, NO, beta = beta, s = sigma,
sampsize = 10000, omega = .0013)
MH.out$theta[9990:9999,]
```

---

**regmixmodel.sel**  
*Model Selection in Mixtures of Regressions*

Description

Assess the number of components in a mixture of regressions model using the Akaike’s information criterion (AIC), Schwartz’s Bayesian information criterion (BIC), Bozdogan’s consistent AIC (CAIC), and Integrated Completed Likelihood (ICL).

Usage

```r
regmixmodel.sel(x, y, k = 2, type = c("fixed", "random",
  "mixed"), ...)
```

Arguments

- `x` An nxp matrix of predictors. If an intercept is required, then x must NOT include a column of 1’s! Requiring an intercept may be controlled through arguments specified in ....
- `y` An n-vector of response values.
- `k` The maximum number of components to assess.
- `type` The type of regression mixture to use. If "fixed", then a mixture of regressions with fixed effects will be used. If "random", then a mixture of regressions where the random effects regression coefficients are assumed to come from a mixture will be used. If "mixed", the mixture structure used is the same as "random", except a coefficient of fixed effects is also assumed.
- `...` Additional arguments passed to the EM algorithm used for calculating the type of regression mixture specified in `type`.

---

30
Value

`regmixmodel.sel` returns a matrix of the AIC, BIC, CAIC, and ICL values along with the winner (i.e., the highest value given by the model selection criterion) for various types of regression mixtures.

References


See Also

`regmixEM`, `repnormmixEM.mixed`

Examples

```r
### Assessing the number of components for NOdata.
data(NOdata)
attach(NOdata)
regmixmodel.sel(x = NO, y = Equivalence, k = 3, type = "fixed")
```

---

`repnornmixEM` *EM Algorithm for Mixtures of Normals with Repeated Measurements*

Description

Returns EM algorithm output for mixtures of normals with repeated measurements and arbitrarily many components.

Usage

```r
repnornmixEM(x, lambda = NULL, mu = NULL, sigma = NULL, k = 2,
              arbmean = TRUE, arbvar = TRUE, epsilon = 1e-08,
              maxit = 10000, verb = FALSE)
```

Arguments

- `x`  
  An mxn matrix of data. The columns correspond to the subjects and the rows correspond to the repeated measurements.

- `lambda`  
  Initial value of mixing proportions. Entries should sum to 1. This determines number of components. If NULL, then `lambda` is random from uniform Dirichlet and number of components is determined by `mu`.  

---

31
mu  A k-vector of component means. If NULL, then mu is determined by a normal distribution according to a binning method done on the data. If both lambda and mu are NULL, then number of components is determined by sigma.

sigma  A vector of standard deviations. If NULL, then 1/sigma^2 has random standard exponential entries according to a binning method done on the data. If lambda, mu, and sigma are NULL, then number of components is determined by k.

k  Number of components. Ignored unless all of lambda, beta, and sigma are NULL.

arbmean  If TRUE, then the component densities are allowed to have different mus. If FALSE, then a scale mixture will be fit.

arbvar  If TRUE, then the component densities are allowed to have different sigmas. If FALSE, then a location mixture will be fit.

epsilon  The convergence criterion.

maxit  The maximum number of iterations.

verb  If TRUE, then various updates are printed during each iteration of the algorithm.

Value

repnormmixEM returns a list of class mixEM with items:

x  The raw data.

lambda  The final mixing proportions.

mu  The final mean parameters.

sigma  The final standard deviations. If arbmean = FALSE, then only the smallest standard deviation is returned. See scale below.

scale  If arbmean = FALSE, then the scale factor for the component standard deviations is returned. Otherwise, this is omitted from the output.

loglik  The final log-likelihood.

posterior  An nxk matrix of posterior probabilities for observations.

all.loglik  A vector of each iteration’s log-likelihood.

ft  A character vector giving the name of the function.

References


See Also

normalmixEM
Examples

```r
## EM output for the water-level task data set.

data(Waterdata)
water<-t(as.matrix(Waterdata))
em.out<-repnormmixEM(water, k = 2, verb = TRUE)
em.out
```

**repnormmixmodel.sel**  
*Model Selection in Mixtures of Normals with Repeated Measures*

**Description**
Assess the number of components in a mixture model with normal components and repeated measures using the Akaike's information criterion (AIC), Schwartz's Bayesian information criterion (BIC), Bozdogan's consistent AIC (CAIC), and Integrated Completed Likelihood (ICL).

**Usage**

```r
repnormmixmodel.sel(x, k = 2)
```

**Arguments**

- **x**: An mxn matrix of observations. The rows correspond to the repeated measures and the columns correspond to the subject.
- **k**: The maximum number of components to assess.

**Value**

`repnormmixmodel.sel` returns a matrix of the AIC, BIC, CAIC, and ICL values along with the winner (i.e., the highest value given by the model selection criterion) for a mixture of normals with repeated measures.

**References**


**See Also**

`repnormmixEM`
Examples

```r
## Assessing the number of components for the water-level task data set.

data(Waterdata)
water<-t(as.matrix(Waterdata))
out<-repnormmixmodel.sel(water, k = 2)
out
```

### Waterdata

<table>
<thead>
<tr>
<th>Waterdata</th>
<th>Water-Level Task Data Set</th>
</tr>
</thead>
</table>

**Description**

This data set arises from the water-level task proposed by the Swiss psychologist Jean Piaget to assess children’s understanding of the physical world. This involves presenting a child with a rectangular shaped, two-dimensional vessel on a sheet of paper tilted in specified clock-hour orientations. The child is then required to draw a line representing the still liquid in the tilted vessel. Then, the degree of deviation about the horizontal is recorded.

**Usage**

- **Waterdata**

**Format**

This data frame consists of 405 children (the rows) and the degree of deviation about the horizontal (the columns) for 8 specified clock-hour orientations (1, 2, 4, 5, 7, 8, 10, and 11 o’clock).

**Source**