Package: dpGMM (via r-universe)

October 20, 2026

Type Package

Title Dynamic Programming Based Gaussian Mixture Modelling Tool for 1D and 2D Data

Version 1.0.0

Maintainer Kamila Szumala <kamila.szumala@polsl.pl>

Description Gaussian mixture modeling of one- and two-dimensional data, provided in original or binned form, with an option to estimate the number of model components. The method uses Gaussian Mixture Models (GMM) with initial parameters determined by a dynamic programming algorithm, leading to stable and reproducible model fitting. For more details see Zyla, J., Szumala, K., Polanski, A., Polanska, J., & Marczyk, M. (2026) <doi:10.1016/j.jocs.2026.102811>.

Depends R (>= 3.5), ggplot2, RColorBrewer, stats, pracma

Imports grDevices, ggpubr, Matrix, reshape2, graphics, methods, mvtnorm

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

NeedsCompilation no

Author Michal Marczyk [aut, ctb], Kamila Szumala [aut, cre], Joanna Zyla [aut, ctb]

Config/pak/sysreqs cmake make libicu-dev

Repository https://szumx123.r-universe.dev

Date/Publication 2026-03-02 14:40:02 UTC

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

RemoteRef HEAD

RemoteSha d5a1e8dfb686f4cdf4c6952531c8a880956e0d9a

Contents

binned . . . 2
EM_iter . . . 3
EM_iter_2D . . . 4
example . . . 5
example2D . . . 5
find_class_2D . . . 6
find_thr_by_dist . . . 6
find_thr_by_params . . . 7
gaussian_mixture_2D . . . 8
gaussian_mixture_vector . . . 9
generate_dist . . . 10
generate_dset2D . . . 11
generate_norm1D . . . 12
generate_norm2D . . . 13
GMM_1D_opts . . . 13
GMM_2D_opts . . . 15
img_to_coords . . . 16
plot_gmm_1D . . . 16
plot_gmm_2D_binned . . . 17
plot_gmm_2D_orig . . . 18
plot_QQplot . . . 19
runGMM . . . 20
runGMM2D . . . 21

Index **23**

`binned` *Data example of binned problem in mass spectrometry*

Description

This data set is part of mass spectrometry measurements. First column represent X values. Second column represent counts of X.

Usage

```
data(binned)
```

Format

A matrix of X and Y (in histogram)

EM_iter *Expectation-maximization algorithm for 1D data*

Description

The function performs the EM algorithm to find the local maximum likelihood for the estimated Gaussian mixture parameters.

Usage

```
EM_iter(X, alpha, mu, sig, Y = NULL, opts = NULL)
```

Arguments

X	Vector of 1D data for GMM decomposition.
alpha	Vector containing the weights (alpha) for each component in the statistical model.
mu	Vector containing the means (mu) for each component in the statistical model.
sig	Vector containing the standard deviation (sigma) for each component in the statistical model.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
opts	Parameters of run saved in GMM_1D_opts variable.

Value

Returns a list of GMM parameter values that correspond to the local extremes for each component.

alpha Vector of optimal alpha (weights) values.

mu Vector of optimal mu (means) values.

sigma Vector of optimal sigma (standard devations) values.

logLik Log-likelihood statistic for the estimated number of components.

crit Value of the selected information criterion in local extreme of likelihood function.

See Also

runGMM and gaussian_mixture_vector

Examples

```
data("example")
opts <- GMM_1D_opts
Y <- matrix(1, 1, length(example$Dist))
rcpt <- EM_iter(example$Dist, 1, mean(example$Dist), sd(example$Dist), Y, opts)
```

EM_iter_2D *Expectation-maximization algorithm for 2D data*

Description

The function performs the EM algorithm to find the local maximum likelihood for the estimated Gaussian mixture parameters.

Usage

```
EM_iter_2D(X, Y, init, opts = NULL)
```

Arguments

X	Matrix of 2D data to decompose by GMM.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
init	Vector of initial parameters for Gaussian components.
opts	Parameters of run stored in GMM_2D_opts variable.

Value

Function returns a list of GMM parameters for tested number of components:

alpha Weights (alpha) of each component.

center Means of decomposition.

covar Covariances of each component.

KS Estimated number of components.

logL Log-likelihood statistic for the estimated number of components.

IC The value of the selected information criterion which was used to calculate the number of components.

See Also

runGMM2D

Examples

```
data("example2D")
X <- example2D[,1:2]
Y <- matrix(1, 1, nrow(X))

opts <- GMM_2D_opts

# It is necessary to define the initial conditions. Here we use random initialization.
alpha <- matrix(1, 1, opts$KS)/opts$KS
```

```
center <- as.matrix(X[sample(nrow(X), opts$KS),])
rownames(center) <- NULL
covar <- replicate(opts$KS, diag(apply(as.matrix(X), 2, sd)/opts$KS), simplify = "array")

init <- list(alpha = alpha,
             center = center,
             covar = covar,
             KS = opts$KS)

gmm <- EM_iter_2D(X, Y, init, opts)
```

`example` *Data of 1D mixed-normal distributions*

Description

This data set was randomly drown for 6 components GMM. The parameters of distributions are as follow:
means <- c(-14.56, -14.16, -11.80, -8.77, -2.89, 2.31);
sigma <- c(2.06, 4.49, 4.42, 2.39, 3.92, 1.36);
alpha <- c(0.2012, 0.2898, 0.0334, 0.0092, 0.4278, 0.0384)

Usage

```
data(example)
```

Format

A vector containing 1500 observations

Source

Randomly generated data

`example2D` *Data of 2D mixed-normal distributions*

Description

This data set contain translated image information into X and Y coordinates and count for each pair X and Y.

Usage

```
data(example2D)
```

Format

`data.frame` of X and Y coordinates and counts

---

find_class_2D *Class assignment for 2D Gaussian Mixture Model data*

Description

Function which assign each point of 2D matrix data to a cluster by maximum probability.

Usage

```
find_class_2D(X, gmm)
```

Arguments

`X`	matrix of data to decompose by GMM.
`gmm`	Results of `gaussian_mixture_2D` decomposition.

Value

Return a vector of cluster assignment of each point of X matrix.

---

find_thr_by_dist *Thresholds estimations for 1D from GMM distribution*

Description

Function to calculate cutoffs between each component of mixture normal distributions using probability distribution function.

Usage

```
find_thr_by_dist(input, sigmas.dev = 2.5, alpha, mu, sigma)
```

Arguments

`input`	output of `generate_dist` function. It is a list with following elements: **x** Numeric vector with equaliy spread data of given precison. **dist** Matrix with PDF of each GMM component and cumulative distribution.
`sigmas.dev`	Number of sigmas to secure thresholds on the ends of distributions. Equivalent to sigma.dev in merging GMMs.
`alpha`	Vector containing the weights (alpha) for each component in the statistical model.
`mu`	Vector containing the means (mu) for each component in the statistical model.
`sigma`	Vector containing the standard deviation (sigma) for each component in the statistical model.

Value

Return a vector of thresholds.

See Also

runGMM

Examples

```
data(example)

alpha <- c(0.45, 0.5, 0.05)
mu <- c(-14, -2, 5)
sigma <- c(2, 4, 1.5)

dist.plot <- generate_dist(example$Dist, alpha = alpha, mu = mu, sigma = sigma, 1e4)
thr <- find_thr_by_dist(dist.plot, 2.5, alpha = alpha, mu = mu, sigma = sigma)
```

---

find_thr_by_params *Thresholds estimations for 1D from GMM parameters*

Description

Function to calculate cutoffs between each component of mixture normal distributions based on the component parameters.

Usage

```
find_thr_by_params(alpha, mu, sigma, input, sigmas.dev = 2.5)
```

Arguments

alpha	Vector containing the weights (alpha) for each component in the statistical model.
mu	Vector containing the means (mu) for each component in the statistical model.
sigma	Vector containing the standard deviation (sigma) for each component in the statistical model.
input	output of generate_dist function. Its necessary only if arithmetical approach fails in threshold estimation and find_thr_by_dist function is called. It is a list with following elements: **x** Numeric vector with equaliy spread data of given precison. **dist** Matrix with PDF of each GMM component and cumulative distribution.
sigmas.dev	Number of sigmas to secure thresholds on the ends of distributions. Equivalent to sigma.dev in merging GMMs.

Value

Return a vector of thresholds.

See Also

runGMM

Examples

```
data(example)

alpha <- c(0.45, 0.5, 0.05)
mu <- c(-14, -2, 5)
sigma <- c(2, 4, 1.5)

dist.plot <- generate_dist(example$Dist, alpha = alpha, mu = mu, sigma = sigma, 1e4)
thr <- find_thr_by_params(alpha = alpha, mu = mu, sigma = sigma, dist.plot)
```

---

gaussian_mixture_2D *Gaussian mixture decomposition for 2D data*

Description

Function to choose the optimal number of components of a 2D mixture normal distributions, minimizing the value of the information criterion.

Usage

```
gaussian_mixture_2D(X, Y = NULL, opts = NULL)
```

Arguments

X	Matrix of 2D data to decompose by GMM.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
opts	Parameters of run saved in GMM_2D_opts variable.

Value

Function returns a list of GMM parameters for the optimal number of components:

alpha Weights (alpha) of each component.

center Means of decomposition.

covar Covariances of each component.

KS Estimated number of components.

logL Log-likelihood statistic for the estimated number of components.

IC The value of the selected information criterion which was used to calculate the number of components.

cls Assigment of point to the clusters.

See Also

runGMM2D, GMM_2D_opts

Examples

```
data(example2D)
custom.settings <- GMM_2D_opts
exp <- gaussian_mixture_2D(example2D[,1:2], example2D[,3], opts = custom.settings)
```

---

gaussian_mixture_vector

Gaussian mixture decomposition for 1D data

Description

Function to estimate number of components of a mixture normal distributions, minimizing the value of the information criterion.

Usage

```
gaussian_mixture_vector(X, Y = NULL, opts = NULL)
```

Arguments

X	Vector of 1D data for GMM decomposition.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
opts	Parameters of run saved in GMM_1D_opts variable.

Value

Function returns a list of GMM parameters for the estimated number of components:

model A list of model component parameters - mean values (mu), standard deviations (sigma) and weights (alpha) for each component.

IC The value of the selected information criterion which was used to calculate the number of components.

logLik Log-likelihood statistic for the estimated number of components.

KS Estimaged number of model components.

See Also

runGMM and generate_norm1D

Examples

```
data <- generate_norm1D(1000, alpha = c(0.2,0.4,0.4), mu = c(-15,0,15), sigma = c(1,2,3))

custom.settings <- GMM_1D_opts
custom.settings$IC <- "AIC"
custom.settings$KS <- 10

exp <- gaussian_mixture_vector(data$Dist, opts = custom.settings)
```

---

generate_dist *Generation of GMM data with high precision*

Description

Function to generate PDF of GMM distributions and its cumulative results with high lincespacing.

Usage

```
generate_dist(X, alpha, mu, sigma, precision)
```

Arguments

X	Vector of 1D data.
alpha	Vector of alphas (weights) for each distribution.
mu	Vector of means for each distribution.
sigma	Vector of sigmas for each distribution.
precision	Precision of point linespacing.

Value

List with following elements:

x Numeric vector with equaliy spread data of given precison.

dist Matrix with PDF of each GMM component and cumulative distribution.

See Also

runGMM and generate_norm1D

Examples

```
data <- generate_norm1D(1000, alpha = c(0.2, 0.4, 0.4),
                                mu = c(-15, 0, 15), sigma = c(1, 2, 3))
dist <- generate_dist(data$Dist, alpha = c(0.2, 0.4, 0.4),
                                  mu = c(-15, 0, 15),
                                  sigma = c(1, 2, 3), precision = 1000)
```

`generate_dset2D` *Generator of multiple random 2D mixed-normal distributions*

Description

Generator of multiple 2D mixed normal distribution with given model parameters ranges.

Usage

```
generate_dset2D(
  n = 1500,
  m = 1500,
  KS_range = 2:8,
  mu_range = c(-15, 15),
  cov_range = c(1, 5)
)
```

Arguments

`n`	Number of points to generate.
`m`	Number of distribution to generate.
`KS_range`	Range of possible number of components of generated distribution. Default `KS=2:8`.
`mu_range`	Range of means of components of generated distribution. Default `-15:15`.
`cov_range`	Range of means of components of generated distribution. Default `1:5`.

Value

List with 2D GMM distributions where each list contains elements of `generate_norm2D`.

See Also

`generate_norm2D`

Examples

```
dset <- generate_dset2D(n = 1500, m = 10,
                        KS_range = 2:5,
                        mu_range = c(-10, 10),
                        cov_range = c(1, 3))
```

`generate_norm1D` *Generator of 1D mixed-normal distributions*

Description

Generator of mixed-normal distribution with given model parameters for certain points number.

Usage

```
generate_norm1D(n, alpha, mu, sigma)
```

Arguments

`n`	Number of points to generate.
`alpha`	Vector of alphas (weights) for each distribution.
`mu`	Vector of means for each distribution.
`sigma`	Vector of sigmas for each distribution.

Value

List with following elements:

Dist Numeric vector with generated data

Cls Numeric vector with classification of each point to particular mixed distribution

Examples

```
data <- generate_norm1D(1000, alpha = c(0.2, 0.4, 0.4),
                               mu = c(-15, 0, 15), sigma = c(1, 2, 3))
```

generate_norm2D *Generator of 2D mixed-normal distributions*

Description

Generator of 2D mixed normal distribution with given model parameters for certain points number.

Usage

```
generate_norm2D(n, alpha, mu, cov)
```

Arguments

n	Number of points to generate.
alpha	Vector of alphas (weights) for each distribution.
mu	Matrix of means for each distribution.
cov	Vector of covariances for each distribution.

Value

List with following elements:

Dist Numeric marix with generated data.

Cls Numeric vector with classification of each point to particular distribution.

Examples

```
data <- generate_norm2D(1500, alpha = c(0.2, 0.4, 0.4),
                            mu = matrix(c(1, 2, 1, 3, 2, 2), nrow = 2),
                            cov = c(0.01, 0.02, 0.03))
```

GMM_1D_opts *Default configuration for 1D Gaussian Mixture decomposition*

Description

A list with parameters customizing a GMM for 1D and binned data. Each component of the list is an effective argument for runGMM.

Usage

```
GMM_1D_opts
```

Format

A list with the following components:

KS Maximum number of components of the model.

eps_change Criterion for early stopping of EM (1e-7, by default) given by the following formula:

$$\sum(|\alpha - \alpha_{old})| + \frac{\sum(\frac{|\sigma^2 - \sigma^2_{old}|}{\sigma^2})}{length(\alpha)}$$

max_iter Maximum number of iterations of EM algorithm. By default it is `max_iter = 10 000`.

SW Parameter for calculating minimum variance of each Gaussian component (0.25, by default) using the following formula:

$$(\frac{SW * range(x)}{no.of.components})^2$$

. Lower value means smaller component variance allowed.

IC Information criterion used to select the number of model components. Possible methods are "AIC","AICc", "BIC" (default), "ICL-BIC" or "LR".

sigmas.dev Parameter used to define close GMM components that needs to be merged. For each component, standard deviation is multiplied by `sigmas.dev` to estimate the distance from component mean. All other components within this distance are merged. By default it is `sigmas.dev = 1`. When `sigmas.dev = 0` no components are merged.

quick_stop Logical value. Determines if stop searching of the number of components earlier based on the Likelihood Ratio Test. Used to speed up the function (TRUE, by default).

signi Significance level set for Likelihood Ratio Test (0.05, by default).

fixed Logical value. Fit GMM for selected number of components given by KS (FALSE, by default).

plot Logical value. If TRUE (default), the figure visualizing GMM decomposition will be displayed.

col.pal Name of the RColorBrewer palette used in the figure. By default `"Blues"`.

Examples

```
# display all default settings
GMM_1D_opts

# create a new settings object
custom.settings <- GMM_1D_opts
custom.settings$IC <- "AIC"
custom.settings
```

GMM_2D_opts *Default configuration for 2D Gaussian Mixture decomposition*

Description

A list with parameters customizing a GMM_2D. Each component of the list is an effective argument for runGMM2D.

Usage

```
GMM_2D_opts
```

Format

A list with the following components:

eps_change Criterion for early stopping of EM (1e-7, by default).

max_iter Maximum number of iterations of EM algorithm. By default it is `max_iter = 50 000`.

SW Regularizing coefficient for covariance.

max_var_ratio Maximum dissimilarity between horizontal and vertical dispersion. By default it is `max_var_ratio = 5`.

IC Information criterion used to select the number of model components. Possible methods are "AIC","AICc", "BIC" (default), "ICL-BIC" or "LR".

cov_type Type of covariance defined for each model component. Possible "sphere","diag" or "full" (default).

init_nb Number of random initial conditions. By default it is `init_nb = 10`.

KS Maximum number of components of the model. By default it is `KS = 5`.

quick_stop Logical value. Determines if stop searching of the number of components earlier based on the Likelihood Ratio Test. Used to speed up the function (TRUE, by default).

signi Significance level set for Likelihood Ratio Test (0.05, by default).

init_con Type of initial conditions. Could be "rand" (default),"DP" or "diag".

fixed Logical value. Fit GMM for selected number of components given by KS (FALSE, by default).

plot Logical value. If TRUE, the GMM decomposition figure will be displayed (FALSE, by default).

Examples

```
# display all default settings
GMM_2D_opts

# create a new settings object
custom.settings <- GMM_2D_opts
custom.settings$IC <- "AIC"
custom.settings
```

img_to_coords *2D plot support*

Description

Transform image into coordinates data

Usage

```
img_to_coords(img)
```

Arguments

img	image in 2D array.

plot_gmm_1D *Plot of GMM decomposition for 1D data*

Description

Function plot the decomposed distribution together with histogram of data. Moreover the cut-off are marked. This plot is also return as regular output of runGMM.

Usage

```
plot_gmm_1D(X, dist, Y = NULL, threshold = NA, pal = "Blues")
```

Arguments

X	Vector of 1D data for GMM decomposition.
dist	Output of generate_dist function.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
threshold	Vector with GMM cutoffs.
pal	Name of the RColorBrewer palette used in the figure. By default "Blues".

Value

A ggplot object showing the histogram or density of the input data together with the Gaussian mixture model decomposition. Individual mixture components and the overall fitted density are displayed as line plots, and optional cut-off thresholds are marked as vertical dashed lines.

See Also

runGMM

Examples

```
data(example)

alpha <- c(0.45, 0.5, 0.05)
mu <- c(-14, -2, 5)
sigma <- c(2, 4, 1.5)

dist.plot <- generate_dist(example$Dist, alpha, mu, sigma, 1e4)
thr <- find_thr_by_params(alpha, mu, sigma, dist.plot)
plot_gmm_1D(example$Dist, dist.plot, Y = NULL, threshold = thr, pal="Dark2")
```

---

plot_gmm_2D_binned *Plot of GMM decomposition for 2D binned data*

Description

Function plot the heatmap of binned data with marked GMM decomposition. This plot is also return as regular output of runGMM2D.

Usage

```
plot_gmm_2D_binned(X, Y, gmm, opts)
```

Arguments

X	Matrix of 2D data to decompose by GMM.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
gmm	Results of gaussian_mixture_2D decomposition
opts	Parameters of run stored in GMM_2D_opts variable.

Value

A ggplot object showing the heatmap of binned two-dimensional data with an overlay of the Gaussian mixture model decomposition. Mixture component centers are indicated by points and covariance ellipses corresponding to selected probability contours are drawn around each component.

See Also

runGMM2D

Examples

```
data(example2D)
custom.settings <- GMM_2D_opts
res <- runGMM2D(example2D[,1:2], example2D[,3], opts = custom.settings)

plot_gmm_2D_binned(example2D[,1:2], example2D[,3], res$model, custom.settings)
```

`plot_gmm_2D_orig` *Plot of GMM decomposition for 2D data*

Description

Function plot the decomposed distribution together with histogram of data. This plot is also return as regular output of `runGMM`.

Usage

```
plot_gmm_2D_orig(X, gmm, opts)
```

Arguments

`X`	Matrix of 2D data to decompose by GMM.
`gmm`	Results of `gaussian_mixture_2D` decomposition
`opts`	Parameters of run stored in `GMM_2D_opts` variable.

Value

A `ggplot` object showing the scatter plot of two-dimensional data with an overlay of the Gaussian mixture model decomposition. Mixture component centers are indicated by points and covariance ellipses corresponding to selected probability contours are drawn around each component.

See Also

`runGMM2D`

Examples

```
custom.settings <- GMM_2D_opts
data <- generate_norm2D(1500, alpha = c(0.2, 0.4, 0.4),
                            mu = matrix(c(1, 2, 1, 3, 2, 2), nrow = 2),
                            cov = c(0.01, 0.02, 0.03))

res <- runGMM2D(data$Dist, opts = custom.settings)
plot_gmm_2D_orig(data$Dist, res$model, custom.settings)
```

plot_QQplot *QQplot of GMM decomposition for 1D data*

Description

Function return ggplot object with fit diagnostic Quantile-Quantile plot for one normal distribution and fitted GMM. This plot is also return as regular output of runGMM.

Usage

```
plot_QQplot(X, alpha, mu, sigma)
```

Arguments

X	Vector of 1D data for GMM decomposition.
alpha	Vector containing the weights (alpha) for each component in the statistical model.
mu	Vector containing the means (mu) for each component in the statistical model
sigma	Vector containing the standard deviation (sigma) for each component in the statistical model.

Value

An object extending ggplot that arranges two quantile-quantile plots into a single figure. One panel shows a QQ plot of the input data against a normal distribution, and the other shows a QQ plot against data simulated from the fitted Gaussian mixture model.

See Also

runGMM

Examples

```
data(example)

alpha <- c(0.45, 0.5, 0.05)
mu <- c(-14, -2, 5)
sigma <- c(2, 4, 1.5)

plot_QQplot(example$Dist, alpha, mu, sigma)
```

runGMM *Function to fit Gaussian Mixture Model (GMM) to 1D data*

Description

Function fits GMM with initial conditions found using dynamic programming-based approach by using expectation-maximization (EM) algorithm. The function works on original and binned (e.g. obtained by creating histogram on 1D data) data. Additionally, threshold values that allows to assign data to individual Gaussian components are provided. Function allows to estimate the number of GMM components using five different information criteria and merging of similar components.

Usage

```
runGMM(X, Y = NULL, opts = NULL)
```

Arguments

X	Vector of 1D data for GMM decomposition.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
opts	Parameters of run saved in GMM_1D_opts variable.

Value

Function returns a list which contains:

model A list of model component parameters - mean values (mu), standard deviations (sigma) and weights (alpha) for each component. Output of gaussian_mixture_vector.

KS Estimaged number of model components.

IC The value of the selected information criterion which was used to calculate the number of components.

logLik Log-likelihood statistic for the estimated number of components.

threshold Vector of thresholds between each component.

cluster Assignment of original X values to individual components (clusters) by thresholds.

fig ggplot object (output of the plot_gmm_1D function). It contains GMM decomposition together with a histogram of the data.

QQplot ggplot object (output of the plot_QQplot function). It presents diagnostic Quantile-Quantile plot for a single normal distribution and fitted GMM.

See Also

gaussian_mixture_vector, EM_iter

Examples

```
data(example)

custom.settings <- GMM_1D_opts
custom.settings$sigmas.dev <- 1.5
custom.settings$max_iter <- 1000
custom.settings$KS <- 10

mix_test <- runGMM(example$Dist, opts = custom.settings)
mix_test$QQplot

#example for binned data
data(binned)

custom.settings <- GMM_1D_opts
custom.settings$quick_stop <- TRUE
custom.settings$KS <- 40
custom.settings$col.pal <- "Dark2"
custom.settings$plot <- FALSE

binned_test <- runGMM(X = binned$V1, Y = binned$V2, opts = custom.settings)
binned_test$fig
```

---

runGMM2D *Function to fit Gaussian Mixture Model (GMM) to 2D data*

Description

Main function to perform GMM on 2D data. Function choose the optimal number of components of a 2D mixture normal distributions by minimizing the value of the information criterion.

Usage

```
runGMM2D(X, Y = NULL, opts = NULL)
```

Arguments

X	Matrix of 2D data to decompose by GMM.
Y	Vector of counts, with the same length as "X". Applies only to binned data (Y = NULL, by default).
opts	Parameters of run stored in GMM_2D_opts variable.

Value

Function returns a `list` of GMM parameters for the estimated number of components:

model **alpha** Weights (alpha) of each component.

center Means of decomposition.

covar Covariances of each component.

KS Estimated number of components.

logL Log-likelihood statistic for the estimated number of components.

IC The value of the selected information criterion which was used to calculate the number of components.

cls Assigment of point to the clusters.

fig Plot of decomposition.

Examples

```
data(example2D)
custom.settings <- GMM_2D_opts
custom.settings$fixed <- TRUE
custom.settings$KS <- 3
custom.settings$max_iter <- 5000
custom.settings$plot <- TRUE

res <- runGMM2D(example2D[,1:2], example2D[,3], opts = custom.settings)
```

Index

∗ **datasets**
    binned, 2
    example, 5
    example2D, 5
    GMM_1D_opts, 13
    GMM_2D_opts, 15

binned, 2

EM_iter, 3, *20*
EM_iter_2D, 4
example, 5
example2D, 5

find_class_2D, 6
find_thr_by_dist, 6, *7*
find_thr_by_params, 7

gaussian_mixture_2D, *6*, 8, *17, 18*
gaussian_mixture_vector, *3*, 9, *20*
generate_dist, *6, 7*, 10, *16*
generate_dset2D, 11
generate_norm1D, *10*, 12
generate_norm2D, *11*, 13
GMM_1D_opts, *3*, *9*, 13, *20*
GMM_2D_opts, *4*, *8, 9*, 15, *17, 18*, *21*

img_to_coords, 16

plot_gmm_1D, 16, *20*
plot_gmm_2D_binned, 17
plot_gmm_2D_orig, 18
plot_QQplot, 19, *20*

runGMM, *3*, *7, 8*, *10*, *13*, *16*, *18, 19*, 20
runGMM2D, *4*, *9*, *15*, *17, 18*, 21