

Package ‘MixtureFitting’

May 18, 2026

Version 0.8.0

Date 2026-05-18

Title Fitting of Univariate Mixture Distributions to Data using Various Approaches

Depends R (>= 2.0.1), sn

Description Methods for fitting mixture distributions to univariate data using expectation maximization, HWHM and other methods. Supports Gaussian, Cauchy, Student's t, skew-normal and von Mises mixtures. For more details see Merkys (2018) <https://www.lvb.lt/permalink/370LABT_NETWORK/1m6ui06/alma9910036312108451>.

License GPL-2

NeedsCompilation yes

Author Andrius Merkys [aut, cre]

Maintainer Andrius Merkys <andrius.merkys@gmail.com>

Repository CRAN

Date/Publication 2026-05-18 13:50:09 UTC

Contents

abs_convergence	3
aic	3
bhattacharyya_dist	4
bic	5
cmm_fit_em	5
cmm_fit_hwhm_spline_deriv	6
cmm_init_vector	7
cmm_init_vector_kmeans	8
cmm_intersections	9
dggmm	9
dcmm	10
dgmm	11
digamma_approx	11
ds	12

dsmm	13
dsnmm	13
dvmm	14
gmm_fit_em	15
gmm_fit_hwhm	16
gmm_fit_hwhm_spline_deriv	16
gmm_fit_kmeans	17
gmm_init_vector	18
gmm_init_vector_kmeans	18
gmm_init_vector_quantile	19
gmm_intersections	20
gmm_merge_components	20
gmm_size_probability	21
gmm_size_probability_nls	22
gradient_descent	22
kldiv	23
kmeans_circular	24
llcmm	24
llgmm	25
llgmm_conservative	26
llgmm_opposite	26
llsmm	27
llsnmm	28
llvmm	28
llvmm_opposite	29
mk_fit_images	30
plot_circular_hist	30
plot_density	31
polyroot_NR	32
pssd	32
pssd_gradient	33
ratio_convergence	34
rcmm	34
rgmm	35
rsimplex_start	36
rvmm	36
simplex	37
smm_fit_em	38
smm_fit_em_APK10	39
smm_fit_em_CWL04	40
smm_fit_em_GNL08	41
smm_init_vector	42
smm_init_vector_kmeans	43
smm_split_component	43
snmm_fit_em	44
snmm_init_vector	45
ssd	46
ssd_gradient	47

<code>abs_convergence</code>	3
<code>s_fit_primitive</code>	47
<code>vmm_fit_em</code>	48
<code>vmm_fit_em_by_diff</code>	49
<code>vmm_fit_em_by_ll</code>	50
<code>vmm_init_vector</code>	51
<code>wmedian</code>	51
Index	53

<code>abs_convergence</code>	<i>Absolute Convergence Check.</i>
------------------------------	------------------------------------

Description

Compare two values to tell whether an optimization process has converged.

Usage

```
abs_convergence( p_now, p_prev, epsilon = 1e-6 )
```

Arguments

<code>p_now</code>	function value of <i>i</i> -th iteration.
<code>p_prev</code>	function value of <i>i-1</i> -th iteration.
<code>epsilon</code>	convergence criterion

Value

TRUE if deemed to have converged, FALSE otherwise

Author(s)

Andrius Merkys

<code>aic</code>	<i>Akaike Information Criterion (AIC)</i>
------------------	---

Description

Calculates Bayesian Information Criterion (AIC) for any type of mixture model. Log-likelihood function has to be provided.

Usage

```
aic( x, p, llf )
```

Arguments

x	data vector
p	vector of mixture model parameters
llf	function calculating log-likelihood, called as llf(x, p)

Value

Akaike Information Criterion value.

Author(s)

Andrius Merkys

bhattacharyya_dist *Bhattacharyya distance for univariate Gaussian distributions.*

Description

Measures Bhattacharyya distance for two univariate Gaussian distributions.

Usage

```
bhattacharyya_dist( mu1, mu2, sigma1, sigma2 )
```

Arguments

mu1	mean of the first Gaussian distribution.
mu2	mean of the second Gaussian distribution.
sigma1	standard deviation of the first Gaussian distribution.
sigma2	standard deviation of the second Gaussian distribution.

Value

Bhattacharyya distance as double.

Author(s)

Andrius Merkys

bic	<i>Bayesian Information Criterion (BIC)</i>
-----	---

Description

Calculates Bayesian Information Criterion (BIC) for any type of mixture model. Log-likelihood function has to be provided.

Usage

```
bic( x, p, llf )
```

Arguments

x	data vector
p	vector of mixture model parameters
llf	function calculating log-likelihood, called as llf(x, p)

Value

Bayesian Information Criterion value.

Author(s)

Andrius Merkys

cmm_fit_em	<i>Estimate Cauchy Mixture parameters using Expectation Maximization.</i>
------------	---

Description

Estimates parameters for Cauchy mixture using Expectation Maximization algorithm.

Usage

```
cmm_fit_em( x, p, epsilon = c( 0.000001, 0.000001, 0.000001 ),  
            iter.cauchy = 20, debug = FALSE, implementation = "C" )
```

Arguments

<code>x</code>	data vector
<code>p</code>	initialization vector of $3*n$ parameters, where n is number of mixture components. Structure of <code>p</code> vector is <code>p = c(A1, A2, ..., An, mu1, mu2, ..., mun, gamma1, gamma2, ..., gamman)</code> , where A_i is the proportion of i -th component, μ_i is the center of i -th component and γ_i is the Cauchy scale of i -th component.
<code>epsilon</code>	tolerance threshold for convergence. Structure of <code>epsilon</code> is <code>epsilon = c(epsilon_A, epsilon_mu, epsilon_gamma)</code> , where <code>epsilon_A</code> is threshold for component proportions, <code>epsilon_mu</code> is threshold for component centers and <code>epsilon_gamma</code> is threshold for component Cauchy scales.
<code>iter.cauchy</code>	number of iterations to fit a single Cauchy component.
<code>debug</code>	flag to turn the debug prints on/off.
<code>implementation</code>	flag to switch between C (default) and R implementations.

Value

Vector of mixture parameters, whose structure is the same as of input parameter's `p`.

Author(s)

Andrius Merkys

References

Ferenc Nahy. Parameter Estimation of the Cauchy Distribution in Information Theory Approach (2006). Journal of Universal Computer Science

`cmm_fit_hwhm_spline_deriv`

Estimate Cauchy Mixture Parameters Using Derivatives and Half-Width at Half-Maximum Method.

Description

Estimate Cauchy mixture parameters using derivatives and half-width at half-maximum (HWHM) method. The method smooths the histogram before attempting to locate the modes. Then it describes them using HWHM.

Usage

```
cmm_fit_hwhm_spline_deriv( x, y )
```

Arguments

<code>x</code>	data vector
<code>y</code>	response vector for <code>x</code>

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \gamma1, \gamma2, \dots, \gamma n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and γ_i is the Cauchy scale of i -th component.

Author(s)

Andrius Merkys

cmm_init_vector	<i>Estimate Cauchy Mixture parameters using Expectation Maximization.</i>
-----------------	---

Description

Estimate an initialization vector for Cauchy mixture fitting via Expectation Maximization. Proportions are set to equal, centers are equispaced through the whole domain of input sample, and scales are set to 1.

Usage

```
cmm_init_vector( x, m, implementation = "C" )
```

Arguments

`x` data vector
`m` number of mixture components
`implementation` flag to switch between C (default) and R implementations.

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \gamma1, \gamma2, \dots, \gamma n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and γ_i is the Cauchy scale of i -th component.

Author(s)

Andrius Merkys

`cmm_init_vector_kmeans`

Estimate Cauchy Mixture parameters using Expectation Maximization.

Description

Estimate an initialization vector for Cauchy mixture fitting using k-means. R implementation of k-means in `kmeans()` is used to find data point assignment to clusters. Then several iterations of Cauchy mixture fitting (per Nahy 2006) is used to derive mixture parameters.

Usage

```
cmm_init_vector_kmeans( x, m, iter.cauchy = 20 )
```

Arguments

<code>x</code>	data vector
<code>m</code>	number of mixture components
<code>iter.cauchy</code>	number of iterations to fit a single Cauchy component.

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of `p` vector is `p = c(A1, A2, ..., An, mu1, mu2, ..., mun, gamma1, gamma2, ..., gamman)`, where A_i is the proportion of i -th component, μ_i is the center of i -th component and γ_i is the Cauchy scale of i -th component.

Author(s)

Andrius Merkys

References

Ferenc Nahy. Parameter Estimation of the Cauchy Distribution in Information Theory Approach (2006). Journal of Universal Computer Science

cmm_intersections *Intersections of Two Cauchy Distributions*

Description

Finds intersections of two Cauchy distributions by finding roots of a quadratic equation.

Usage

```
cmm_intersections( p )
```

Arguments

p parameter vector of 6 parameters. Structure of **p** vector is $p = c(A1, A2, \mu1, \mu2, \gamma1, \gamma2)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, γ_i is the Cauchy scale of i -th component.

Value

A vector of x values of intersections (zero, one or two). Returns NaN if both distributions are identical.

Author(s)

Andrius Merkys

dcgmm *Density of The Cauchy-Gaussian Distribution*

Description

Density function for the Cauchy-Gaussian distribution, according to Eqn. 2 of Swami (2000).

Usage

```
dcgmm( x, p )
```

Arguments

x data vector

p parameter vector of $5*n$ parameters, where n is number of mixture components. Structure of **p** vector is $p = c(A1, A2, \dots, A_n, e1, e2, \dots, e_n, \mu1, \mu2, \dots, \mu_n, \gamma1, \gamma2, \dots, \gamma_n, \sigma1, \sigma2, \dots, \sigma_n)$, where A_i is the proportion of i -th component, e_i is the proportion of Cauchy subcomponent of i -th component, μ_i is the center of i -th component, γ_i is the Cauchy concentration of i -th component and σ_i is the Gaussian standard deviation of i -th component.

Value

A vector.

Author(s)

Andrius Merkys

References

Swami, A. Non-Gaussian mixture models for detection and estimation in heavy-tailed noise 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100), 2000, 6, 3802-3805

 dcmm

Density of The Cauchy Mixture Distribution

Description

Density function for the Cauchy mixture distribution.

Usage

```
dcmm( x, p, implementation = "C" )
```

Arguments

x data vector

p parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \gamma1, \gamma2, \dots, \gamma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, γ_i is the Cauchy scale of i -th component.

implementation flag to switch between C (default) and R implementations.

Value

A vector.

Author(s)

Andrius Merkys

 dgmm

The Gaussian Mixture Distribution

Description

Density function for the Gaussian mixture distribution.

Usage

```
dgmm( x, p, normalise_proportions = FALSE, restrict_sigmas = FALSE,
      implementation = "C" )
```

Arguments

x data vector

p parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

normalise_proportions if TRUE, make component proportions sum up to 1 by dividing each one of them by their sum (R implementation only).

restrict_sigmas if TRUE, skip components with scales less or equal to zero (R implementation only).

implementation flag to switch between C (default) and R implementations.

Value

A vector.

Author(s)

Andrius Merkys

 digamma_approx

Calculate Approximate Value of The Digamma Function.

Description

Calculates approximate value of the digamma function using first eight non-zero members of asymptotic expression for $\text{digamma}(x)$. Implemented according to Wikipedia.

Usage

```
digamma_approx( x )
```

Arguments

x data vector

Value

Digamma function value.

Author(s)

Andrius Merkys

References

Users of Wikipedia. Digamma function. https://en.wikipedia.org/w/index.php?title=Digamma_function&oldid=708779689

ds

Density of The Student's t Model

Description

Density function for the Student's t Model. Wrapper around R's `dt()`, supporting center and concentration parameters.

Usage

```
ds( x, c, s, ni )
```

Arguments

x data vector
c center
s concentration
ni degrees of freedom

Value

A vector.

Author(s)

Andrius Merkys

dsmm

*Density of The Student's t Mixture Model***Description**

Density function for the Student's t Mixture Model.

Usage

```
dsmm( x, p )
```

Arguments

x	data vector
p	parameter vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and ni_i is the degrees of freedom of i -th component.

Value

A vector.

Author(s)

Andrius Merkys

dsnmm

*Density of The Skew-Normal Mixture Model***Description**

Density function for the skew-normal mixture model.

Usage

```
dsnmm( x, p )
```

Arguments

x	data vector
p	parameter vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(\omega1, \omega2, \dots, \omega n, \zeta1, \zeta2, \dots, \zeta n, \sigma1, \sigma2, \dots, \sigma n, \lambda1, \lambda2, \dots, \lambda n)$, where ω_i is the proportion of i -th component, ζ_i is the location of i -th component, σ_i is the scale of i -th component and λ_i is the shape of i -th component.

Value

A vector.

Author(s)

Andrius Merkys

dvmm

Density of The von Mises Mixture Model.

Description

Density function for the von Mises Mixture Model.

Usage

```
dvmm( x, p, implementation = "C" )
```

Arguments

<code>x</code>	data vector
<code>p</code>	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of <code>p</code> vector is $p = c(A_1, A_2, \dots, A_n, \mu_1, \mu_2, \dots, \mu_n, k_1, k_2, \dots, k_n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.
<code>implementation</code>	flag to switch between C (default) and R implementations.

Value

A vector.

Author(s)

Andrius Merkys

gmm_fit_em	<i>Estimate Gaussian Mixture parameters using Expectation Maximization.</i>
------------	---

Description

Estimates parameters for Gaussian mixture using Expectation Maximization algorithm.

Usage

```
gmm_fit_em( x, p, w = numeric(), epsilon = c( 0.000001, 0.000001, 0.000001 ),
            max_steps = 0, debug = FALSE, implementation = "C", ... )
```

Arguments

x	data vector
p	initialization vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and σ_i is the scale of i -th component.
w	weights of data points, must have the same length as the data vector; if not given or has different length, equal weights are assumed.
epsilon	tolerance threshold for convergence. Structure of epsilon is $\epsilon = c(\epsilon_A, \epsilon_\mu, \epsilon_\sigma)$, where ϵ_A is threshold for component proportions, ϵ_μ is threshold for component centers and ϵ_σ is threshold for component scales.
max_steps	maximum number of steps (0 = no limit).
debug	flag to turn the debug prints on/off.
implementation	flag to switch between C (default) and R implementations.
...	additional arguments passed to <code>gmm_fit_em_R()</code> when R implementation is used.

Value

Vector of mixture parameters, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

gmm_fit_hwhm	<i>Estimate Gaussian Mixture Parameters Using Half-Width at Half-Maximum Method.</i>
--------------	--

Description

Estimate Gaussian mixture parameters using half-width at half-maximum (HWHM) method. Given a histogram, the method attempts to locate most prominent modes and describe them using HWHM.

Usage

```
gmm_fit_hwhm( x, y, n )
```

Arguments

x	data vector
y	response vector for x
n	number of mixture components

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Author(s)

Andrius Merkys

gmm_fit_hwhm_spline_deriv	<i>Estimate Gaussian Mixture Parameters Using Derivatives and Half-Width at Half-Maximum Method.</i>
---------------------------	--

Description

Estimate Gaussian mixture parameters using derivatives and half-width at half-maximum (HWHM) method. The method smooths the histogram before attempting to locate the modes. Then it describes them using HWHM.

Usage

```
gmm_fit_hwhm_spline_deriv( x, y )
```

Arguments

x data vector
 y response vector for x

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Author(s)

Andrius Merkys

gmm_fit_kmeans	<i>Estimate Gaussian Mixture parameters from kmeans.</i>
----------------	--

Description

Estimates parameters for Gaussian mixture using kmeans.

Usage

```
gmm_fit_kmeans( x, n )
```

Arguments

x data vector
 n number of mixture components

Value

Vector of $3*n$ mixture parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Author(s)

Andrius Merkys

gmm_init_vector	<i>Estimate Gaussian Mixture parameters using Expectation Maximization.</i>
-----------------	---

Description

Estimate an initialization vector for Gaussian mixture fitting via Expectation Maximization. Proportions and scales are set to equal, centers are equispaced through the whole domain of input sample.

Usage

```
gmm_init_vector( x, n, implementation = "C" )
```

Arguments

x	data vector
n	number of mixture components
implementation	flag to switch between C (default) and R implementations.

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Author(s)

Andrius Merkys

gmm_init_vector_kmeans	<i>Estimate Gaussian Mixture parameters using Expectation Maximization.</i>
------------------------	---

Description

Estimate an initialization vector for Gaussian mixture fitting using k-means. R implementation of k-means in `kmeans()` is used to find data point assignment to clusters.

Usage

```
gmm_init_vector_kmeans( x, m )
```

Arguments

x	data vector
m	number of mixture components

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Author(s)

Andrius Merkys

gmm_init_vector_quantile

Estimate Gaussian Mixture parameters using Expectation Maximization.

Description

Estimate an initialization vector for Gaussian mixture fitting using (weighted) quantiles. Proportions and scales are set to equal, centers are placed at equispaced quantiles.

Usage

```
gmm_init_vector_quantile( x, m, w = numeric() )
```

Arguments

x	data vector
m	number of mixture components
w	weight vector

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Author(s)

Andrius Merkys

`gmm_intersections` *Intersections of Two Gaussian Distributions*

Description

Finds intersections of two Gaussian distributions by finding roots of a quadratic equation.

Usage

```
gmm_intersections( p )
```

Arguments

`p` parameter vector of 6 parameters. Structure of `p` vector is $p = c(A1, A2, \mu1, \mu2, \sigma1, \sigma2)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Value

A vector of x values of intersections (zero, one or two). Returns NaN if both distributions are identical.

Author(s)

Andrius Merkys

`gmm_merge_components` *Merge two Gaussian components into one.*

Description

Merges i th and j th components of Gaussian mixture model. Implemented in the same venue as in [mergeparameters](#) of `fpc`.

Usage

```
gmm_merge_components( x, p, i, j )
```

Arguments

x	data vector
p	vector of Gaussian mixture parameters. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn)$, where n is number of mixture components, A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component.
i	index of the first component to be merged. Component with this index will be replaced by a merged one in the output.
j	index of the second component to be merged. Component with this index will be removed in the output.

Value

Vector of mixture parameters, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

References

Hennig, C. Methods for merging Gaussian mixture components *Advances in Data Analysis and Classification*, Springer Nature, 2010, 4, 3-34

`gmm_size_probability` *The Gaussian Mixture Distribution*

Description

Calculates the posterior probability of a Gaussian mixture with n components. Internally, it attempts to maximize log-likelihood of data by calling `optim()` and returns the list as received from `optim()`.

Usage

```
gmm_size_probability( x, n, method = "SANN" )
```

Arguments

x	data vector
n	number of mixture components
method	optimization method passed to <code>optim()</code>

Value

List representing the converged `optim()` run.

Author(s)

Andrius Merkys

`gmm_size_probability_nls`*The Gaussian Mixture Distribution*

Description

Calculates the posterior probability of a Gaussian mixture with n components. Internally, it bins the data vector and calls `nls()` to optimize the mixture fit. Returns the list of the same form as received from `optim()`.

Usage

```
gmm_size_probability_nls( x, n, bins = 100, trace = FALSE )
```

Arguments

<code>x</code>	data vector
<code>n</code>	number of mixture components
<code>bins</code>	number of bins
<code>trace</code>	should debug trace be printed?

Value

List of the same form as received from `optim()`.

Author(s)

Andrius Merkys

`gradient_descent`*Gradient Descent*

Description

Simple implementation of gradient descent method. Given a derivative function, it follows its decrease until convergence criterion is met.

Usage

```
gradient_descent( gradfn, start, gamma = 0.1, ..., epsilon = 0.01 )
```

Arguments

gradfn	derivative function
start	starting value
gamma	learning rate
...	additional arguments passed to derivative function
epsilon	convergence threshold for absolute squared difference

Value

log-likelihood

Author(s)

Andrius Merkys

kldiv	<i>Kullback–Leibler Divergence of ith Student's t Mixture component.</i>
-------	--

Description

Measures Kullback–Leibler divergence of i th Student's t Mixture component using Dirac's delta function. Implemented according to Chen et al. (2004).

Usage

```
kldiv( x, p, k )
```

Arguments

x	data vector
p	vector of Student's t mixture parameters. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where n is number of mixture components, A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and n_{ii} is the degrees of freedom of i -th component.
k	number of the component.

Value

Kullback–Leibler divergence as double.

Author(s)

Andrius Merkys

References

Chen, S.; Wang, H. & Luo, B. Greedy EM Algorithm for Robust T-Mixture Modeling Third International Conference on Image and Graphics (ICIG'04), Institute of Electrical & Electronics Engineers (IEEE), 2004, 548–551

kmeans_circular	<i>K-Means Clustering for Points on Circle</i>
-----------------	--

Description

Perform k-means clustering on angular data (in degrees).

Usage

```
kmeans_circular( x, centers, iter.max = 10 )
```

Arguments

x	data vector
centers	vector of initial centers (in degrees)
iter.max	maximum number of iterations

Value

Vector of the same length as *centers* defining cluster centers (in degrees).

Author(s)

Andrius Merkys

llcmm	<i>Log-likelihood for Cauchy Mixture</i>
-------	--

Description

Calculates log-likelihood for a given data vector using a Cauchy mixture distribution.

Usage

```
llcmm( x, p, implementation = "C" )
```

Arguments

- `x` data vector
- `p` parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of `p` vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \gamma1, \gamma2, \dots, \gamma n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and γ_i is the Cauchy scale of i -th component.
- `implementation` flag to switch between C (default) and R implementations.

Value

log-likelihood

Author(s)

Andrius Merkys

llgmm

Log-likelihood for Gaussian Mixture

Description

Calculates log-likelihood for a given data vector using a Gaussian mixture distribution.

Usage

```
llgmm( x, p, implementation = "C" )
```

Arguments

- `x` data vector
- `p` parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of `p` vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and σ_i is the scale of i -th component.
- `implementation` flag to switch between C (default) and R implementations.

Value

log-likelihood

Author(s)

Andrius Merkys

llgmm_conservative *Log-likelihood for Gaussian Mixture*

Description

Calculates log-likelihood for a given data vector using a Gaussian mixture distribution. This is a straightforward implementation, different from llgmm() in that that it does not detect and shortcut edge cases.

Usage

```
llgmm_conservative( x, p )
```

Arguments

x data vector

p parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and σ_i is the scale of i -th component.

Value

log-likelihood

Author(s)

Andrius Merkys

llgmm_opposite *Opposite Log-likelihood for Gaussian Mixture*

Description

Calculates opposite log-likelihood for a given data vector using a Gaussian mixture distribution.

Usage

```
llgmm_opposite( x, p )
```

Arguments

x data vector

p parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and σ_i is the scale of i -th component.

Value

opposite log-likelihood (negated log-likelihood value)

Author(s)

Andrius Merkys

llsmm

Log-likelihood for Student's t Mixture

Description

Calculates log-likelihood for a given data vector using a Student's t mixture distribution.

Usage

```
llsmm( x, p )
```

Arguments

x	data vector
p	parameter vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and ni_i is the degrees of freedom of i -th component.

Value

log-likelihood

Author(s)

Andrius Merkys

llsnmm *Log-likelihood for Skew-Normal Mixture*

Description

Calculates log-likelihood for a given data vector using a skew-normal mixture distribution.

Usage

```
llsnmm( x, p )
```

Arguments

x	data vector
p	parameter vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(\omega_1, \omega_2, \dots, \omega_n, \zeta_1, \zeta_2, \dots, \zeta_n, \sigma_1, \sigma_2, \dots, \sigma_n, \lambda_1, \lambda_2, \dots, \lambda_n)$, where ω_i is the proportion of i -th component, ζ_i is the location of i -th component, σ_i is the scale of i -th component and λ_i is the shape of i -th component.

Value

log-likelihood

Author(s)

Andrius Merkys

llvmm *Log-likelihood for von Mises Mixture*

Description

Calculates log-likelihood for a given data vector using a von Mises mixture distribution.

Usage

```
llvmm( x, p, implementation = "C" )
```

Arguments

x	data vector
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A_1, A_2, \dots, A_n, \mu_1, \mu_2, \dots, \mu_n, k_1, k_2, \dots, k_n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.
implementation	flag to switch between C (default) and R implementations.

Value

log-likelihood

Author(s)

Andrius Merkys

llvmm_opposite	<i>Opposite Log-likelihood for von Mises Mixture</i>
----------------	--

Description

Calculates opposite log-likelihood for a given data vector using a von Mises mixture distribution.

Usage

```
llvmm_opposite( x, p )
```

Arguments

x	data vector
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.

Value

opposite log-likelihood (negated log-likelihood value)

Author(s)

Andrius Merkys

mk_fit_images *Mixture Distribution Modeling*

Description

Draw a PNG histogram with a mixture density on top of it for each iteration of mixture optimization process.

Usage

```
mk_fit_images( h, l, prefix = "img_" )
```

Arguments

h	histogram object, as returned from hist()
l	list containing model vectors
prefix	prefix of file name to write

Author(s)

Andrius Merkys

plot_circular_hist *Mixture Distribution Modeling*

Description

Plot a circular histogram.

Usage

```
plot_circular_hist( x, breaks = 72, ball = 0.5, ... )
```

Arguments

x	data vector
breaks	number of breaks in histogram
ball	radius of the drawn circle
...	parameters passed to plot()

Author(s)

Andrius Merkys

plot_density	<i>Mixture Distribution Modeling</i>
--------------	--------------------------------------

Description

Draw a PNG histogram with a mixture density on top of it.

Usage

```
plot_density( x, model, density_f, width, height,  
             cuts = 400, main = "",  
             filename = NULL,  
             obs_good = c(), obs_bad = c(),  
             scale_density = FALSE )
```

Arguments

x	data vector
cuts	number of breaks in histogram
main	main title of the plot
model	model passed to density_f()
density_f	probability density function
filename	name of the file to write
width	image width, passed to png()
height	image height, passed to png()
obs_good	vector of values to mark with rug() in green color
obs_bad	vector of values to mark with rug() in red color
scale_density	should probability density be scaled?

Author(s)

Andrius Merkys

polyroot_NR	<i>Find one real polynomial root using Newton–Raphson method.</i>
-------------	---

Description

Finds one real polynomial root using Newton–Raphson method, implemented according to Wikipedia.

Usage

```
polyroot_NR( p, init = 0, epsilon = 1e-6, debug = FALSE, implementation = "C" )
```

Arguments

p	vector of polynomial coefficients.
init	initial value.
epsilon	tolerance threshold for convergence.
debug	flag to turn the debug prints on/off.
implementation	flag to switch between C (default) and R implementations.

Value

Real polynomial root.

Author(s)

Andrius Merkys

References

Users of Wikipedia. Newton’s method. https://en.wikipedia.org/w/index.php?title=Newton%27s_method&oldid=710342140

pssd	<i>Penalized Sum of Squared Differences Using Gaussian Mixture Distribution</i>
------	---

Description

Given two vectors of same length and a Gaussian mixture, calculate the penalized sum of squared differences (SSD) between the first vector and Gaussian mixture densities measured at points from second vector. Penalties are included for proportions and scales that are less than or equal to 0.

Usage

```
pssd( x, y, p )
```

Arguments

x	data vector
y	response vector
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Value

Penalized sum of squared differences.

Author(s)

Andrius Merkys

pssd_gradient	<i>Penalized Sum of Squared Differences Using Gaussian Mixture Distribution</i>
---------------	---

Description

Gradient (derivative) function of pssd().

Usage

```
pssd_gradient( x, y, p )
```

Arguments

x	data vector
y	response vector
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Value

Gradient values measured at x .

Author(s)

Andrius Merkys

ratio_convergence	<i>Ratio Convergence Check.</i>
-------------------	---------------------------------

Description

Compare two values to tell whether an optimization process has converged. The absolute difference between values of two iterations is divided by the value of previous iteration and compared to the epsilon value.

Usage

```
ratio_convergence( p_now, p_prev, epsilon = 1e-6 )
```

Arguments

p_now	function value of i -th iteration.
p_prev	function value of $i-1$ -th iteration.
epsilon	convergence criterion

Value

TRUE if deemed to have converged, FALSE otherwise

Author(s)

Andrius Merkys

rcmm	<i>Random Sample of The Cauchy Mixture Distribution</i>
------	---

Description

Generates a random sample of the Cauchy mixture distribution.

Usage

```
rcmm( n, p )
```

Arguments

n	sample size
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \gamma1, \gamma2, \dots, \gamma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, γ_i is the Cauchy scale of i -th component.

Value

A vector.

Author(s)

Andrius Merkys

rgmm

Random Sample of the Gaussian Mixture Distribution

Description

Generates a random sample of the Gaussian mixture distribution.

Usage

```
rgmm( n, p )
```

Arguments

n	data vector
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Value

A vector.

Author(s)

Andrius Merkys

rsimplex_start	<i>Nelder-Mead's Simplex Method for Function Minimization.</i>
----------------	--

Description

Generate initial simplices for simplex().

Usage

```
rsimplex_start( seed, n, lower, upper )
```

Arguments

seed	seed for random number generator
n	number of simplices
lower	vector with lower bounds of each dimension
upper	vector with upper bounds of each dimension

Value

A list with n simplices.

Author(s)

Andrius Merkys

rvmm	<i>Random Sample of the von Mises Mixture Model.</i>
------	--

Description

Generates a random sample of the von Mises Mixture Model.

Usage

```
rvmm( n, p )
```

Arguments

n	sample size
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A_1, A_2, \dots, A_n, \mu_1, \mu_2, \dots, \mu_n, k_1, k_2, \dots, k_n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.

Value

A vector.

Author(s)

Andrius Merkys

References

Best & Fisher. Efficient Simulation of the von Mises Distribution. Journal of the RSS, Series C, 1979, 28, 152-157.

simplex

Nelder-Mead's Simplex Method for Function Minimization.

Description

Nelder-Mead's Simplex Method for Function Minimization.

Usage

```
simplex( fn, start, ..., epsilon = 0.000001, alpha = 1,  
        gamma = 2, rho = 0.5, delta = 0.5, trace = FALSE )
```

Arguments

fn	minimized function, has to accept the argmin vector as first parameter
start	start vector
...	other parameters passed to the minimized function
epsilon	convergence criterion
alpha	reflection coefficient
gamma	expansion coefficient
rho	contraction coefficient
delta	shrink coefficient
trace	should debug trace be printed?

Value

Vector yielding the minimum value of the minimized function

Author(s)

Andrius Merkys

References

Nelder, J. A. & Mead, R. A Simplex Method For Function Minimization. The Computer Journal, 1965, 308-313.

Users of Wikipedia. Nelder-Mead method. https://en.wikipedia.org/w/index.php?title=Nelder%E2%80%93Mead_method&oldid=1287347131

smm_fit_em	<i>Estimate Student's t Mixture parameters using Expectation Maximization.</i>
------------	--

Description

Estimates parameters for Student's t mixture using Expectation Maximization algorithm. Calls smm_fit_em_APK10().

Usage

```
smm_fit_em( x, p, ... )
```

Arguments

x	data vector
p	initialization vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and ni_i is the degrees of freedom of i -th component.
...	additional arguments passed to smm_fit_em_GNL08().

Value

Vector of mixture parameters, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

smm_fit_em_APK10	<i>Estimate Student's t Mixture parameters using Expectation Maximization.</i>
------------------	--

Description

Estimates parameters for univariate Student's t mixture using Expectation Maximization algorithm, according to Fig. 2 of Aeschliman et al. (2010).

Usage

```
smm_fit_em_APK10( x, p, epsilon = c( 1e-6, 1e-6, 1e-6, 1e-6 ),
                  collect.history = FALSE, debug = FALSE )
```

Arguments

x	data vector
p	initialization vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and ni_i is the degrees of freedom of i -th component.
epsilon	tolerance threshold for convergence. Structure of epsilon is $\epsilon = c(\epsilon_A, \epsilon_\mu, \epsilon_k, \epsilon_{ni})$, where ϵ_A is threshold for component proportions, ϵ_μ is threshold for component centers, ϵ_k is threshold for component concentrations and ϵ_{ni} is threshold for component degrees of freedom.
collect.history	flag to turn accumulation of estimation history on/off.
debug	flag to turn the debug prints on/off.

Value

A list.

Author(s)

Andrius Merkys

References

Aeschliman, C.; Park, J. & Kak, A. C. A Novel Parameter Estimation Algorithm for the Multivariate t-Distribution and Its Application to Computer Vision European Conference on Computer Vision 2010, 2010 <https://engineering.purdue.edu/RVL/Publications/Aeschliman2010ANovel.pdf>

smm_fit_em_CWL04 *Greedy estimate Student's t Mixture parameters using Expectation Maximization.*

Description

Estimates (greedily) parameters for univariate Student's t mixture using Expectation Maximization algorithm, implemented according to Chen et al. (2004). The algorithm relies upon smm_fit_em_GNL08() to estimate mixture parameters iteratively.

Usage

```
smm_fit_em_CWL04( x, p, collect.history = FALSE, debug = FALSE,
  ... )
```

Arguments

x	data vector
p	initialization vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and ni_i is the degrees of freedom of i -th component.
collect.history	logical. If set to TRUE, a list of parameter values of all iterations is returned.
debug	flag to turn the debug prints on/off.
...	parameters passed to smm_fit_em_GNL08().

Value

A list.

Author(s)

Andrius Merkys

References

Chen, S.; Wang, H. & Luo, B. Greedy EM Algorithm for Robust T-Mixture Modeling Third International Conference on Image and Graphics (ICIG'04), Institute of Electrical & Electronics Engineers (IEEE), 2004, 548–551

smm_fit_em_GNL08	<i>Estimate Student's t Mixture parameters using Expectation Maximization.</i>
------------------	--

Description

Estimates parameters for univariate Student's t mixture using Expectation Maximization algorithm, according to Eqns. 12–17 of Gerogiannis et al. (2009).

Usage

```
smm_fit_em_GNL08( x, p, epsilon = c( 1e-6, 1e-6, 1e-6, 1e-6 ),
  collect.history = FALSE, debug = FALSE,
  min.sigma = 1e-256, min.ni = 1e-256,
  max.df = 1000, max.steps = Inf,
  polyroot.solution = 'jenkins_taub',
  convergence = abs_convergence,
  unif.component = FALSE )
```

Arguments

- x data vector
- p initialization vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where Ai is the proportion of i -th component, μi is the center of i -th component, ki is the concentration of i -th component and nii is the degrees of freedom of i -th component.
- epsilon tolerance threshold for convergence. Structure of epsilon is $\text{epsilon} = c(\text{epsilon}_A, \text{epsilon}_\mu, \text{epsilon}_k, \text{epsilon}_{ni})$, where epsilon_A is threshold for component proportions, epsilon_μ is threshold for component centers, epsilon_k is threshold for component concentrations and epsilon_{ni} is threshold for component degrees of freedom.
- collect.history logical. If set to TRUE, a list of parameter values of all iterations is returned.
- debug flag to turn the debug prints on/off.
- min.sigma minimum value of sigma
- min.ni minimum value of degrees of freedom
- max.df maximum value of degrees of freedom
- max.steps maximum number of steps, may be infinity
- polyroot.solution polyroot finding method used to approximate digamma function. Possible values are 'jenkins_taub' and 'newton_raphson'.
- convergence function to use for convergence checking. Must accept function values of the last two iterations and return TRUE or FALSE.
- unif.component should a uniform component for outliers be added, as suggested by Cousineau & Chartier (2010)?

Value

A list.

Author(s)

Andrius Merkys

References

Gerogiannis, D.; Nikou, C. & Likas, A. The mixtures of Student's t-distributions as a robust framework for rigid registration. *Image and Vision Computing*, Elsevier BV, 2009, 27, 1285–1294 <https://www.cs.uoi.gr/~arly/papers/imavis09.pdf>

Cousineau, D. & Chartier, S. Outliers detection and treatment: a review. *International Journal of Psychological Research*, 2010, 3, 58–67 <https://revistas.usb.edu.co/index.php/IJPR/article/view/844>

smm_init_vector	<i>Estimate Student's t Mixture parameters using Expectation Maximization.</i>
-----------------	--

Description

Estimate an initialization vector for Student's t mixture fitting via Expectation Maximization. Proportions are set to be equal, centers are equispaced through the whole domain of input sample, concentrations and degrees of freedom are set to 1.

Usage

```
smm_init_vector( x, n )
```

Arguments

x	data vector
n	number of mixture components

Value

Parameter vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and ni_i is the degrees of freedom of i -th component.

Author(s)

Andrius Merkys

smm_init_vector_kmeans

Estimate Student's t Mixture parameters using Expectation Maximization.

Description

Estimate an initialization vector for Student's t mixture fitting via Expectation Maximization. R implementation of k-means in `kmeans()` is used to find data point assignment to clusters. `s_fit_primitive()` is then used to estimate component parameters for each cluster.

Usage

```
smm_init_vector_kmeans( x, m )
```

Arguments

x	data vector
m	number of mixture components

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Author(s)

Andrius Merkys

smm_split_component *Split a component of Student's t-distribution in two.*

Description

Splits a component of Student's t-distribution mixture. Implemented according to Eqns. 30–36 of Chen et al. (2004).

Usage

```
smm_split_component( p, alpha = 0.5, beta = 0.5, u = 0.5 )
```

Arguments

p	vector of Student's t mixture parameters. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn, ni1, ni2, \dots, nin)$, where n is number of mixture components, A_i is the proportion of i -th component, μ_i is the center of i -th component, k_i is the concentration of i -th component and ni_i is the degrees of freedom of i -th component.
alpha	split proportion for component proportions
beta	split proportion for component concentrations
u	split proportion for component centers

Value

Vector of parameters for resulting two-component mixture, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

References

Chen, S.-B. & Luo, B. Robust t-mixture modelling with SMEM algorithm Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826), Institute of Electrical & Electronics Engineers (IEEE), 2004, 6, 3689–3694

snmm_fit_em	<i>Estimate Skew-Normal Mixture parameters using Expectation Maximization.</i>
-------------	--

Description

Estimates parameters for skew-normal mixture using Expectation Maximization algorithm.

Usage

```
snmm_fit_em( x, p, w = numeric(),
             epsilon = c( 0.000001, 0.000001, 0.000001, 0.000001 ) )
```

Arguments

x	data vector
p	initialization vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(\omega1, \omega2, \dots, \omega n, \zeta1, \zeta2, \dots, \zeta n, \sigma1, \sigma2, \dots, \sigma n, \lambda1, \lambda2, \dots, \lambda n)$, where ω_{i} is the proportion of i -th component, ζ_{i} is the location of i -th component, σ_{i} is the scale of i -th component and λ_{i} is the shape of i -th component.

w	weights of data points, must have the same length as the data vector; if not given or has different length, equal weights are assumed.
epsilon	tolerance threshold for convergence. Structure of epsilon is $\text{epsilon} = c(\text{epsilon_omega}, \text{epsilon_dzeta}, \text{epsilon_sigma}, \text{epsilon_lambda})$, where epsilon_omega is threshold for component proportions, epsilon_dzeta is threshold for component locations, epsilon_sigma is threshold for component scales and epsilon_lambda is threshold for component shapes.

Value

Vector of mixture parameters, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

References

Arnold et al. The nontruncated marginal of a truncated bivariate normal distribution, *Psychometrika* 58, pages 471–488 (1993)

Lin et al. Finite mixture modelling using the skew normal distribution, *Statistica Sinica* 17 (2007), 909–927 <https://www3.stat.sinica.edu.tw/statistica/oldpdf/A17n35.pdf>

snmm_init_vector	<i>Estimate Skew-Normal Mixture parameters using Expectation Maximization.</i>
------------------	--

Description

Estimate an initialization vector for skew-normal mixture fitting via Expectation Maximization. Estimation method follows Lin et al. (2007) suggestion to use k-means clustering for initial cluster assignment. Calculation of moments and dzeta and sigma parameters are done according to Equation 3 of Lin et al. (2007), but lambda is estimated from Equation 18c of Arnold et al. (1993).

Usage

```
snmm_init_vector( x, n = 1 )
```

Arguments

x	data vector
n	number of mixture components

Value

Parameter vector of $4*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(\omega_1, \omega_2, \dots, \omega_n, \zeta_1, \zeta_2, \dots, \zeta_n, \sigma_1, \sigma_2, \dots, \sigma_n, \lambda_1, \lambda_2, \dots, \lambda_n)$, where ω_i is the proportion of i -th component, ζ_i is the location of i -th component, σ_i is the scale of i -th component and λ_i is the shape of i -th component.

Author(s)

Andrius Merkys

References

Arnold et al. The nontruncated marginal of a truncated bivariate normal distribution, *Psychometrika* 58, pages 471–488 (1993)

Lin et al. Finite mixture modelling using the skew normal distribution, *Statistica Sinica* 17 (2007), 909–927 <https://www3.stat.sinica.edu.tw/statistica/oldpdf/A17n35.pdf>

 ssd

Sum of Squared Differences Using Gaussian Mixture Distribution

Description

Given two vectors of same length and a Gaussian mixture, calculate the sum of squared differences (SSD) between the first vector and Gaussian mixture densities measured at points from second vector.

Usage

```
ssd( x, y, p )
```

Arguments

x	data vector
y	response vector
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A_1, A_2, \dots, A_n, \mu_1, \mu_2, \dots, \mu_n, \sigma_1, \sigma_2, \dots, \sigma_n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Value

Sum of squared differences.

Author(s)

Andrius Merkys

ssd_gradient	<i>Sum of Squared Differences Using Gaussian Mixture Distribution</i>
--------------	---

Description

Gradient (derivative) function of ssd().

Usage

```
ssd_gradient( x, y, p )
```

Arguments

x	data vector
y	response vector
p	parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, \sigma1, \sigma2, \dots, \sigma n)$, where A_i is the proportion of i -th component, μ_i is the location of i -th component, σ_i is the scale of i -th component.

Value

Gradient values measured at x .

Author(s)

Andrius Merkys

s_fit_primitive	<i>Estimate Student's t distribution parameters using Batch Approximation Algorithm.</i>
-----------------	--

Description

Estimates parameters for univariate Student's t distribution parameters using Batch Approximation Algorithm, according to Fig. 2 of Aeschliman et al. (2010).

Usage

```
s_fit_primitive( x )
```

Arguments

x	data vector
---	-------------

Value

Vector $c(\mu, k, n_i)$, where μ is the center, k is the concentration and n_i is the degrees of freedom of the distribution.

Author(s)

Andrius Merkys

References

Aeschliman, C.; Park, J. & Kak, A. C. A Novel Parameter Estimation Algorithm for the Multivariate t-Distribution and Its Application to Computer Vision European Conference on Computer Vision 2010, 2010 <https://engineering.purdue.edu/RVL/Publications/Aeschliman2010ANovel.pdf>

vmm_fit_em	<i>Estimate von Mises Mixture parameters using Expectation Maximization.</i>
------------	--

Description

Estimates parameters for univariate von Mises mixture using Expectation Maximization algorithm.

Usage

```
vmm_fit_em( x, p, epsilon = c( 0.000001, 0.000001, 0.000001 ),
            debug = FALSE, implementation = "C" )
```

Arguments

x	data vector
p	initialization vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A_1, A_2, \dots, A_n, \mu_1, \mu_2, \dots, \mu_n, k_1, k_2, \dots, k_n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.
epsilon	tolerance threshold for convergence. Structure of epsilon is $\text{epsilon} = c(\text{epsilon}_A, \text{epsilon}_\mu, \text{epsilon}_k)$, where epsilon_A is threshold for component proportions, epsilon_μ is threshold for component centers and epsilon_k is threshold for component concentrations.
debug	flag to turn the debug prints on/off.
implementation	flag to switch between C (default) and R implementations.

Value

Vector of mixture parameters, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

References

Banerjee et al. Expectation Maximization for Clustering on Hyperspheres (2003), manuscript, accessible on: <https://web.archive.org/web/20130120061240/http://www.lans.ece.utexas.edu/~abanerjee/papers/05/banerjee05a.pdf>

vmm_fit_em_by_diff	<i>Estimate von Mises Mixture parameters using Expectation Maximization.</i>
--------------------	--

Description

Estimates parameters for univariate von Mises mixture using Expectation Maximization algorithm. In this version stopping criterion is the difference between parameters in the subsequent iterations.

Usage

```
vmm_fit_em_by_diff( x, p, epsilon = c( 0.000001, 0.000001, 0.000001 ),
                   debug = FALSE, implementation = "C" )
```

Arguments

x	data vector
p	initialization vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.
epsilon	tolerance threshold for convergence. Structure of epsilon is $\text{epsilon} = c(\text{epsilon}_A, \text{epsilon}_\mu, \text{epsilon}_k)$, where epsilon_A is threshold for component proportions, epsilon_μ is threshold for component centers and epsilon_k is threshold for component concentrations.
debug	flag to turn the debug prints on/off.
implementation	flag to switch between C (default) and R implementations.

Value

Vector of mixture parameters, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

References

Banerjee et al. Expectation Maximization for Clustering on Hyperspheres (2003), manuscript, accessible on: <https://web.archive.org/web/20130120061240/http://www.lans.ece.utexas.edu/~abanerjee/papers/05/banerjee05a.pdf>

vmm_fit_em_by_ll	<i>Estimate von Mises Mixture parameters using Expectation Maximization.</i>
------------------	--

Description

Estimates parameters for univariate von Mises mixture using Expectation Maximization algorithm. In this version stopping criterion is the difference between log-likelihood estimates of subsequent iterations.

Usage

```
vmm_fit_em_by_ll( x, p, epsilon = .Machine$double.eps,
                 debug = FALSE, implementation = "C" )
```

Arguments

x	data vector
p	initialization vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A_1, A_2, \dots, A_n, \mu_1, \mu_2, \dots, \mu_n, k_1, k_2, \dots, k_n)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.
epsilon	tolerance threshold for convergence
debug	flag to turn the debug prints on/off.
implementation	flag to switch between C (default) and R implementations.

Value

Vector of mixture parameters, whose structure is the same as of input parameter's p.

Author(s)

Andrius Merkys

References

Banerjee et al. Expectation Maximization for Clustering on Hyperspheres (2003), manuscript, accessible on: <https://web.archive.org/web/20130120061240/http://www.lans.ece.utexas.edu/~abanerjee/papers/05/banerjee05a.pdf>

vmm_init_vector	<i>Estimate von Mises Mixture parameters using Expectation Maximization.</i>
-----------------	--

Description

Estimate an initialization vector for von Mises mixture fitting via Expectation Maximization. Proportions are set to equal, centers are equispaced through the whole domain of input sample, and concentrations are set to $(m/(12*180))^2$.

Usage

```
vmm_init_vector( m, implementation = "C" )
```

Arguments

m number of mixture components
implementation flag to switch between C (default) and R implementations.

Value

Parameter vector of $3*n$ parameters, where n is number of mixture components. Structure of p vector is $p = c(A1, A2, \dots, An, \mu1, \mu2, \dots, \mu n, k1, k2, \dots, kn)$, where A_i is the proportion of i -th component, μ_i is the center of i -th component and k_i is the concentration of i -th component.

Author(s)

Andrius Merkys

wmedian	<i>Calculate Weighted Median.</i>
---------	-----------------------------------

Description

Calculated weighted median.

Usage

```
wmedian( x, w, start = 1, end = length( x ) )
```

Arguments

x sample vector
w weights vector
start start index (default: 1)
end end index (default: last index in x)

Value

Median

Author(s)

Andrius Merkys

References

Users of Wikipedia. Weighted median. [https://en.wikipedia.org/w/index.php?title=Weighted_
median&oldid=690896947](https://en.wikipedia.org/w/index.php?title=Weighted_median&oldid=690896947)

Index

abs_convergence, 3
aic, 3

bhattacharyya_dist, 4
bic, 5

cmm_fit_em, 5
cmm_fit_hwhm_spline_deriv, 6
cmm_init_vector, 7
cmm_init_vector_kmeans, 8
cmm_intersections, 9

dcgmm, 9
dcmm, 10
dgmm, 11
digamma_approx, 11
ds, 12
dsmm, 13
dsnmm, 13
dvmm, 14

gmm_fit_em, 15
gmm_fit_hwhm, 16
gmm_fit_hwhm_spline_deriv, 16
gmm_fit_kmeans, 17
gmm_init_vector, 18
gmm_init_vector_kmeans, 18
gmm_init_vector_quantile, 19
gmm_intersections, 20
gmm_merge_components, 20
gmm_size_probability, 21
gmm_size_probability_nls, 22
gradient_descent, 22

kldiv, 23
kmeans_circular, 24

llcmm, 24
llgmm, 25
llgmm_conservative, 26
llgmm_opposite, 26

llsmm, 27
llsnmm, 28
llvmm, 28
llvmm_opposite, 29

mergeparameters, 20
mk_fit_images, 30

plot_circular_hist, 30
plot_density, 31
polyroot_NR, 32
pssd, 32
pssd_gradient, 33

ratio_convergence, 34
rcmm, 34
rgmm, 35
rsimplex_start, 36
rvmm, 36

s_fit_primitive, 47
simplex, 37
smm_fit_em, 38
smm_fit_em_APK10, 39
smm_fit_em_CWL04, 40
smm_fit_em_GNL08, 41
smm_init_vector, 42
smm_init_vector_kmeans, 43
smm_split_component, 43
snmm_fit_em, 44
snmm_init_vector, 45
ssd, 46
ssd_gradient, 47

vmm_fit_em, 48
vmm_fit_em_by_diff, 49
vmm_fit_em_by_ll, 50
vmm_init_vector, 51

wmedian, 51