Package ‘dsdp’

February 11, 2023

Title  Density Estimation with Semidefinite Programming
Version  0.1.1
Description  The models of probability density functions are Gaussian or
exponential distributions with polynomial correction terms.
Using a maximum likelihood method, 'dsdp' computes parameters of Gaussian
or exponential distributions together with degrees of polynomials by
a grid search, and coefficient of polynomials by a variant of semidefinite
programming. It adopts Akaike Information Criterion for model selection.
See a vignette for a tutorial and more on our 'Github' repository
<https://github.com/tsuchiya-lab/dsdp/>.

Encoding  UTF-8
RoxygenNote  7.2.1
Depends  R (>= 2.10)
LazyData  true
Suggests  rmarkdown, knitr
VignetteBuilder  knitr, rmarkdown
Imports  ggplot2, rlang, stats
License  MIT + file LICENSE

URL  https://tsuchiya-lab.github.io/dsdp/
BugReports  https://github.com/tsuchiya-lab/dsdp/issues
NeedsCompilation  yes
Author  Satoshi Kakihara [aut, cre],
        Takashi Tsuchiya [aut]
Maintainer  Satoshi Kakihara <skakahara@gmail.com>
Repository  CRAN
Date/Publication  2023-02-11 10:10:20 UTC
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdf_expmodel</td>
<td>3</td>
</tr>
<tr>
<td>cdf_gaussmodel</td>
<td>4</td>
</tr>
<tr>
<td>databinning</td>
<td>5</td>
</tr>
<tr>
<td>datastats</td>
<td>5</td>
</tr>
<tr>
<td>dsdp</td>
<td>6</td>
</tr>
<tr>
<td>estimate</td>
<td>6</td>
</tr>
<tr>
<td>estimate.expmodel</td>
<td>7</td>
</tr>
<tr>
<td>estimate.gaussmodel</td>
<td>8</td>
</tr>
<tr>
<td>eval_poly</td>
<td>9</td>
</tr>
<tr>
<td>expmodel</td>
<td>10</td>
</tr>
<tr>
<td>exp_est</td>
<td>11</td>
</tr>
<tr>
<td>func</td>
<td>12</td>
</tr>
<tr>
<td>func.expmodel</td>
<td>12</td>
</tr>
<tr>
<td>func.gaussmodel</td>
<td>13</td>
</tr>
<tr>
<td>gaussmodel</td>
<td>14</td>
</tr>
<tr>
<td>gauss_est</td>
<td>15</td>
</tr>
<tr>
<td>histmean</td>
<td>16</td>
</tr>
<tr>
<td>igamma</td>
<td>17</td>
</tr>
<tr>
<td>igammac</td>
<td>17</td>
</tr>
<tr>
<td>mix2gauss</td>
<td>18</td>
</tr>
<tr>
<td>mix2gaussHist</td>
<td>19</td>
</tr>
<tr>
<td>mix2gauss_fun</td>
<td>20</td>
</tr>
<tr>
<td>mix2gauss_gen</td>
<td>20</td>
</tr>
<tr>
<td>mix3gauss</td>
<td>21</td>
</tr>
<tr>
<td>mix3gauss_fun</td>
<td>22</td>
</tr>
<tr>
<td>mix3gauss_gen</td>
<td>22</td>
</tr>
<tr>
<td>mixexpgamma</td>
<td>23</td>
</tr>
<tr>
<td>mixExpGammaHist</td>
<td>23</td>
</tr>
<tr>
<td>mixexpgamma_fun</td>
<td>24</td>
</tr>
<tr>
<td>mixexpgamma_gen</td>
<td>25</td>
</tr>
<tr>
<td>pdf_expmodel</td>
<td>25</td>
</tr>
<tr>
<td>pdf_gaussmodel</td>
<td>26</td>
</tr>
<tr>
<td>plot.expmodel</td>
<td>27</td>
</tr>
<tr>
<td>plot.gaussmodel</td>
<td>28</td>
</tr>
<tr>
<td>polyaxb</td>
<td>29</td>
</tr>
<tr>
<td>printf</td>
<td>30</td>
</tr>
<tr>
<td>summary.expmodel</td>
<td>31</td>
</tr>
<tr>
<td>summary.gaussmodel</td>
<td>32</td>
</tr>
</tbody>
</table>

**Index**: 33
**cdf_expmodel**

Cumulative distribution function of Exponential-based model

**Description**

A cumulative distribution function (CDF) of Exponential-based model. To access parameters and coefficients in an object `emodel` of a class `expmodel`, use `emodel$result[k, "lmd1"], emodel$coeffs[[k]]` for some index `k`. This index appears in the leftmost column of estimation table generated by `summary(emodel)`.

**Usage**

cdf_expmodel(coeff, lmd, x)

**Arguments**

- **coeff**: A coefficient vector in increasing order of degrees; the first element is 0th degree, ..., and last element is the largest degree of coefficients.
- **lmd**: A rate parameter, which is positive.
- **x**: A numeric vector of input.

**Value**

A numeric vector of CDF of an exponential-based model.

**See Also**

- `expmodel()`
- `summary.expmodel()`
- `estimate.expmodel()`
- `func.expmodel()`
- `cdf_expmodel()`

**Examples**

```r
## Create an object of `expmodel`
emodel <- expmodel(mixexpgamma$n200)
## Estimate with degree 4 and rate parameter 2.0
emodel <- estimate(emodel, 4, 2.0)
## Input vector
x <- seq(0, 12, 0.1)
## Output of PDF in above estimation
yv <- cdf_expmodel(emodel$coeffs[[1]], emodel$result[1, "lmd1"], x)
```
cdf_gaussmodel  

Cumulative distribution function of Gaussian-based model

Description

A cumulative distribution function (CDF) of Gaussian-based model. To access parameters and coefficients in an object `gmodel` of a class `gaussmodel`, use `gmodel$result[k, "mu1"], gmodel$result[k, "sig1"], gmodel$coeffs[[k]]` for some index k. This index appears in the leftmost column of estimation table generated by `summary(gmodel)`.

Usage

cdf_gaussmodel(coeff, mu, sig, x)

Arguments

- `coeff`: A coefficient vector in increasing order of degrees; the first element is 0th degree, ..., and last element is the largest degree of coefficients.
- `mu`: A mean of Gaussian distribution.
- `sig`: A standard deviation of Gaussian distribution, which is positive.
- `x`: A numeric input vector.

Value

A numeric vector of CDF of Gaussian-based model.

See Also

gaussmodel(), summary.gaussmodel(), estimate.gaussmodel(), func.gaussmodel(), pdf_gaussmodel()

Examples

```r
## Create an object of `gaussmodel`
gmodel <- gaussmodel(mix2gauss$n200)
## Estimate with a degree 6, a mean 0, and standard deviations 0.5
gmodel <- estimate(gmodel, 6, 0, 0.5)
## Input vector
x <- seq(-3, 3, 0.1)
## Output of PDF in above estimation
yv <- cdf_gaussmodel(gmodel$coeffs[[1]], gmodel$result[1, "mu1"],
gmodel$result[1, "sig1"], x)
```
### databinning

Reduce a data set to representatives of bins and their frequencies

**Description**

Reduce a data set to a named list of 'values' and 'freq', which are representatives of bins and their frequencies, respectively.

**Usage**

```r
databinning(data, bins = 40)
```

**Arguments**

- `data`: A numeric vector of a data set.
- `bins`: A positive integer to represent the number of bins.

**Value**

A named list of `values` and `freq` whose length is `bins`.

**See Also**

`base::rle()`

**Examples**

```r
rlst <- databinning(mix2gauss$n200)
```

### datastats

Compute the mean and the standard deviation of a data set

**Description**

Compute the mean and the standard deviation of a data set represented by the pair of the numeric vectors `data` and optionally its frequency vector `freq`.

**Usage**

```r
datastats(data, freq = NULL)
```

**Arguments**

- `data`: A numeric vector of a data set.
- `freq`: A frequency vector corresponding to the `data` vector. The default value is `NULL`, which means all frequencies are one.
Value

The mean and the standard deviation of a data set.

See Also

histmean()

Examples

## Without a frequency data
datastats(mix2gauss$n200)
## With a frequency data
datastats(mix2gaussHist$n200p, mix2gaussHist$n200f)

ddsdp

ddsp: Density Estimation using Semidefinite Programming

Description

Density estimation with Semidefinite Programming. The models of probability density functions are Gaussian or exponential distributions with polynomial correction terms. Using a maximum likelihood method, it computes parameters of Gaussian or exponential distributions together with degrees of polynomials by a grid search, and coefficients of polynomials by a variant of semidefinite programming. It adopts Akaike Information Criterion for model selection. See vignettes for tutorials and more information.

estimate

Generic Method for estimation

Description

This is a generic S3 method for estimation.

Usage

estimate(model, ...)

Arguments

model An instance of a class model to be estimated.
...
  additional arguments affecting the estimate produced.

Value

An instance of a model with estimated data.
estimate.expmodel

Description

Estimates Exponential-based model expmodel among parameter vectors, deglist, lmdlist. Then it sorts the results by AIC.

Usage

```r
## S3 method for class 'expmodel'
estimate(  
  model,  
  deglist = deglist,  
  lmdlist = lmdlist,  
  recompute = FALSE,  
  stepsize = NULL,  
  verbose = FALSE,  
  ...  
)
```

Arguments

- `model`: An object of expmodel class.
- `deglist`: A vector of degrees of polynomials. The element should be positive integers.
- `lmdlist`: A vector of rate parameters of Exponential-based models. The element should be larger than 0.
- `recompute`: If TRUE, recomputes the results for better estimation and accuracy. Parameters whose accuracies had been already attained sufficiently, namely around 1.0e-6, are not included in candidates for recomputing.
- `stepsize`: A vector in descending order whose values are between 0 and 1. If a small step size is supplied, it can attain successful estimates, but it might take more iterations.
- `verbose`: If TRUE, it shows the detailed message of SDP solver.
- `...`: Arguments to be passed to or from other methods.

Value

A expmodel object including the estimates. Those estimates are stored in `model$result` with data.frame format and `model$coeffs` in list format.

See Also

`expmodel()` `summary.expmodel()` `plot.expmodel()`
**Examples**

```r
## Create an expmodel object
emodel <- expmodel(mixexpgamma$n200)
## Estimate a model with parameters
emodel <- estimate(emodel, deglist=c(4,5), lmdlist=c(0.5, 1, 2))
```

**Description**

Estimates Gaussian-based model `gaussmodel` among parameter vectors, `deglist`, `mulist`, `sdlist`. Then it sorts the results by AIC.

**Usage**

```r
## S3 method for class 'gaussmodel'
estimate(
  model,
  deglist = deglist,
  mulist = mulist,
  sdlist = sdlist,
  scaling = FALSE,
  recompute = FALSE,
  stepsize = NULL,
  verbose = FALSE,
  ...
)
```

**Arguments**

- `model` An object of a `gaussmodel` class.
- `deglist` A vector of degrees of polynomials. The element should be positive even numbers.
- `mulist` A vector of means for Gaussian-based models.
- `sdlist` A vector of standard deviations for Gaussian-based models. The element should be larger than 0.
- `scaling` A logical scalar, which indicates whether or not it scales means and standard deviations in `mulist` and `sdlist`. The default value is `FALSE`.
- `recompute` If `TRUE`, recomputes the results for better estimation and accuracy. Parameters whose accuracies had been already attained sufficiently, namely around `1.0e-6`, are not included in candidates for recomputing.
- `stepsize` A vector in descending order whose values are between 0 and 1. If a small step size is supplied, it can attain successful estimates, but it might take more iterations.
- `verbose` If `TRUE`, it shows the detailed message of SDP solver.
- `...` Arguments to be passed to or from other methods.
**eval_poly**

**Value**

A `gaussmodel` object including the estimates. Those estimates are stored in `model$result` with `data.frame` format and `model$coeffs` in `list` format.

**See Also**

`gaussmodel()` `summary.gaussmodel()` `plot.gaussmodel()`

**Examples**

```r
## Create a `gaussmodel` object
gmodel <- gaussmodel(mix2gauss$n200)
## Estimate a model with parameters
gmodel <- estimate(gmodel, deglist=c(2, 4), mulist=c(0.0, 0.2),
                   sdlist=c(0.75, 1.0))
```

---

**Description**

Evaluate the polynomial whose coefficients are represented in `coeff` vector. The order of the coefficient is an increasing order, i.e., `coeff[1]` is a constant term, and `coeff[2]` is a coefficient of 1st degree term, etc. Evaluation is done using Horner’s method.

**Usage**

```r
eval_poly(coeff, x)
```

**Arguments**

- `coeff`: A coefficient vector in increasing order of degrees; the first element is 0th degree, ..., and the last element is the largest degree of coefficients.
- `x`: A numeric input vector.

**Value**

A vector of values of a polynomial whose coefficient is `coeff`.

**See Also**

`pdf_gaussmodel()` `pdf_expmodel()` `cdf_gaussmodel()` `cdf_expmodel()`

**Examples**

```r
## Evaluate a polynomial x^2 - 2x + 2 with x = 1, 2, 3.
## 0th, 1st, 2nd degree of coefficients
coeff <- c(2, -2, 1)
x <- c(1, 2, 3)
eval_poly(coeff, x)
```
Description

This function is a constructor for S3 class `expmodel`, which represents Exponential-based model. It usually takes `data` and optionally `freq` as arguments and also optionally `stepsize`. Members of interest in practice are `result` and `coeffs`, which maintain the information of estimates and coefficients of polynomials, respectively.

Usage

```r
expmodel(data = data, freq = NULL, stepsize = c(0.5, 0.3))
```

Arguments

data
A nonnegative numeric vector of a data set to be estimated.

freq
A frequency vector corresponding to the `data` vector. The default value is `NULL`, which means all frequencies are one. If supplied, the length of a vector should be same as `data` and each element should be a nonnegative integer.

stepsize
A numeric vector whose element is larger than 0 and smaller than 1, and decreasing order. The default value is `c(0.5, 0.3)`. If you encounter numerical difficulties, decreasing its values, for example, to `c(0.4, 0.2)`, might help to estimate a model.

Value

An object of Exponential-based model `expmodel`.

See Also

`summary.expmodel()` `plot.expmodel()` `estimate.expmodel()`

Examples

```r
## Create `expmodel` object from a data set `mixexpgamma$n200`.
emodel <- expmodel(mixexpgamma$n200)
## Create `expmodel` object from a data set `mixExpGammaHist$n800p` and
## its frequencies `mixExpGammaHist$n800f`.
emodel <- expmodel(mixExpGammaHist$n800p, mixExpGammaHist$n800f)
```
exp_est

Estimate coefficients of a polynomial in Exponential-based Model

Description

Estimate coefficients of a polynomial in Exponential-based model:

\[ \text{poly}(x; \alpha) \text{Exp}(x; \lambda) \]

where \( \alpha \) is a coefficient vector, \( \lambda \) is a rate parameter of an exponential distribution:

\[ \text{Exp}(x; \lambda) := \lambda e^{-\lambda x} \]

Using data and optionally its frequencies \( \text{freq} \), and a degree of a polynomial, a rate parameter \( \lambda \) of an exponential distribution, it computes the coefficients of polynomial, along with Akaike Information Criterion (AIC) and an accuracy information from underlying SDP solver. In general, the smaller the AIC is, the better the model is. An accuracy around \( 1e^{-7} \) is a good indication for a computational result of coefficients estimation.

Usage

\[
\text{exp_est}(\text{deg}, \lambda, \text{data}, \text{freq}, \text{verbose}, \text{stepvec})
\]

Arguments

- \( \text{deg} \) A degree of polynomial, which is positive even integer.
- \( \lambda \) A rate parameter of an exponential distribution, which is positive.
- \( \text{data} \) A numeric vector of a data set to be estimated.
- \( \text{freq} \) A numeric vector of frequencies for a data set \( \text{data} \). The default value is \( \text{NULL} \), which indicates that all frequencies are equally one. If \( \text{freq} \) is not \( \text{NULL} \), then it should be the same length as \( \text{data} \), and all values should be positive integers.
- \( \text{verbose} \) If \( \text{TRUE} \), it shows a detail information about SDP solver.
- \( \text{stepvec} \) It designates the stepsize for SDP solver. If the problem is easy, i.e., the number of a data set are small and a degree of a polynomial is small, then, for example, \( 0.9 \) might be ok. If it looks difficult, then \( c(0.5, 0.3) \) might work.

Value

A list of \( \text{deg} \), \( \lambda \), \( \text{aic} \), \( \text{accuracy} \), coefficient vector

See Also

\[ \text{estimate.expmodel()} \]

Examples

\[
\text{rlst} \leftarrow \text{exp_est}(3, 1.0, \text{mixexpgamma$n200}, \text{NULL}, \text{FALSE}, c(0.7, 0.4))
\]
**func**

*Generic Method for evaluate the estimate*

---

**Description**

This is a generic S3 method for estimate.

**Usage**

```r
func(model, x, ...)
```

**Arguments**

- `model`: An instance of a class model to be evaluated.
- `x`: A numeric vector for input.
- `...`: additional arguments affecting the `func` produced.

**Value**

An evaluation of `x` with `model`.

---

**func.expmodel**

*Return the evaluation of a vector with Exponential-based model*

---

**Description**

Evaluate an input vector `x` with Exponential-based model and return its vector. By default, it evaluate with the best model and its density, but it can designate the model by index and also can evaluate with a cumulative distribution.

**Usage**

```r
## S3 method for class 'expmodel'
func(model, x, cdf = FALSE, n = 1, ...)
```

**Arguments**

- `model`: expmodel object.
- `x`: A numeric vector to be evaluated with a distribution.
- `cdf`: A logical scalar whether the evaluation is done with a cumulative distribution or not. A default value is `FALSE`, which means that the evaluation is done with a density.
- `n`: The index indicates the estimates. 1, by default, is the best estimate, and 2 is the 2nd best, etc.
- `...`: Arguments to be passed to or from other methods.
Value

A numeric vector of the evaluation of input vector \(x\) with a model.

See Also

`expmodel()` `summary.expmodel()` `plot.expmodel()` `estimate.expmodel()` `pdf_expmodel()` `cdf_expmodel()`

Examples

```r
## Create an `expmodel` object
emodel <- expmodel(mixexpgamma$n200)
## Estimate an model with parameters
emodel <- estimate(emodel, deglist=5, lmdlist=3.75)
## A vector for input
x <- seq(0, 14, by=0.1)
## Density function
y <- func(emodel, x)
## Cumulative distribution
y <- func(emodel, x, cdf=TRUE)
```

func.gaussmodel

Return the evaluation of a vector with Gaussian-based model

Description

Evaluate an input vector \(x\) with Gaussian-based model and return its vector. By default, it evaluate with the best model and its density, but it can designate the model by index and also can evaluate with a cumulative distribution.

Usage

```r
## S3 method for class 'gaussmodel'
func(model, x, cdf = FALSE, n = 1, scaling = FALSE, ...)
```

Arguments

- `model` gaussmodel object.
- `x` A numeric vector to be evaluated with a distribution.
- `cdf` A logical scalar whether the evaluation is done with a cumulative distribution or not. A default value is FALSE, which means that the evaluation is done with a density.
- `n` The index indicates the estimates. 1, by default, is the best estimate, and 2 is the 2nd best, etc.
- `scaling` A logical scalar, which indicates whether or not it scales means and standard deviations in `mulist` and `sdlst`. The default value is FALSE.
- `...` Arguments to be passed to or from other methods.
Value

A numeric vector of the evaluation of input vector \( x \) with a model.

See Also

gaussmodel(), summary.gaussmodel(), plot.gaussmodel(), estimate.gaussmodel(), pdf_gaussmodel(),
cdf_gaussmodel()

Examples

```r
## Create an `gaussmodel` object
gmodel <- gaussmodel(mix2gauss$n200)
## Estimate an model with parameters
gmodel <- estimate(gmodel, deglist=4, mulist=0.15, sdlist=0.73)
## A vector for input
x <- seq(-4, 4, by=0.1)
## Density function
y <- func(gmodel, x)
## Cumulative distribution
y <- func(gmodel, x, cdf=TRUE)
```

---

**gaussmodel**  
*Constructor for S3 class gaussmodel*

Description

This function is a constructor for S3 class gaussmodel, which represents Gaussian-based model. It usually takes `data` and optionally `freq` as arguments and also optionally `stepsize`. Members of interest in practice are `result` and `coeffs`, which maintain the information of estimates and coefficients of polynomials, respectively.

Usage

gaussmodel(data = data, freq = NULL, stepsize = c(0.5, 0.3))

Arguments

data
A numeric vector of a data set to be estimated.

freq
A frequency vector corresponding to the `data` vector. The default value is `NULL`, which means all frequencies are one. If supplied, the length of a vector should be same as `data` and each element should be a nonnegative integer.

stepsize
A numeric vector whose element is larger than 0 and smaller than 1, and decreasing order. The default value is `c(0.5, 0.3)`. If you encounter numerical difficulties, decreasing its values, for example, to `c(0.4, 0.2)`, might help to estimate a model.
gauss_est

Value
An object of Gaussian-based model gaussmodel.

See Also
summary.gaussmodel() plot.gaussmodel() estimate.gaussmodel()

Examples
## Create 'gaussmodel' object from a data set 'mix2gauss$n200'.
gmodel <- gaussmodel(mix2gauss$n200)
## Create 'gaussmodel' object from a data set 'mix2gaussHist$n200p' and
## its frequencies 'mix2gaussHist$n200f'.
gmodel <- gaussmodel(mix2gaussHist$n200p, mix2gaussHist$n200f)

Description
Estimate coefficients of a polynomial in Gaussian-based model:

\[
poly(x, \alpha)N(x; \mu, \sigma^2)
\]

where \( \alpha \) is a coefficient vector, \( \mu \) and \( \sigma \) are a mean and a standard deviation of Gaussian distribution:

\[
N(x; \mu, \sigma^2) := \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)
\]

Using data and optionally its frequencies \( freq \), and a degree of a polynomial, a mean \( mu \) and a standard deviation \( sig \) of Gaussian distribution, it computes the coefficients of a polynomial, along with Akaike Information Criterion (AIC) and an accuracy information from an underlying SDP solver. In general, the smaller the AIC is, the better the model is. An accuracy around \( 1e-7 \) is a good indication for a computational result of coefficients estimation.

Usage
gauss_est(deg, mu, sig, data, freq, verbose, stepsize)

Arguments
deg A degree of polynomial, which is positive even integer.
mu A mean of Gaussian distribution.
sig A standard deviation of Gaussian distribution, which is positive.
data A numeric vector of a data set to be estimated.
freq A numeric vector of frequencies for a data set data. The default value is NULL, which indicates that all frequencies are equally one. If freq is not NULL, then it should be the same length as data, and all values should be positive integers.
histmean

value 
If TRUE, it shows a detail information about SDP solver.

stepsize 
It designates the stepsize for SDP solver. If the problem is easy, i.e., the number of a data set are small and a degree of a polynomial is small, then, for example, 0.9 might be ok. If it looks difficult, then c(0.5, 0.3) might work.

Value
A list of deg, mu, sig, aic, accuracy, coefficient vector.

See Also
estimate.gaussmodel()

Examples
rlst <- gauss_est(4, 0, 1, mix2gauss$n200, NULL, FALSE, c(0.7, 0.4))

histmean
Compute the mean of a data set

Description
Compute the mean of a data set represented by the pair of the numeric vectors data and optionally its frequency vector freq.

Usage
histmean(data, freq = NULL)

Arguments
data A numeric vector of a data set.
freq A frequency vector corresponding to the data vector. The default value is NULL, which means all frequencies are one.

Value
The mean of a data set.

See Also
datastats()

Examples
## Without a frequency data
histmean(mix2gauss$n200)
## With a frequency data
histmean(mix2gaussHist$n200p, mix2gaussHist$n200f)
igamma

Incomplete Gamma Function

Description
Evaluate an incomplete gamma function:
\[
\gamma(a, x) = \int_0^x t^{a-1}e^{-t} dt,
\]
using SLATEC dgami in https://netlib.org/slatec/. When \((x > 0 \text{ and } a \geq 0)\) or \((x \geq 0 \text{ and } a > 0)\), compute the result, otherwise the value is NaN.

Usage
\[
\text{igamma}(a, x)
\]

Arguments
\begin{itemize}
  \item \texttt{a} \quad \text{A positive numeric vector.}
  \item \texttt{x} \quad \text{A nonnegative numeric vector with same length as \texttt{a}.}
\end{itemize}

Value
A vector of values of an incomplete gamma function.

See Also
\texttt{igammac()}

Examples
\[
\text{igamma}(1, 1)
\]

---

igammac

Complementary Incomplete Gamma Function

Description
Evaluate a complementary incomplete gamma function:
\[
\gamma^*(a, x) = \int_x^\infty t^{a-1}e^{-t} dt,
\]
using SLATEC dgamic in https://netlib.org/slatec/. When \((x > 0 \text{ and } a \geq 0)\) or \((x \geq 0 \text{ and } a > 0)\), compute the result, otherwise the value is NaN.
Usage

\text{igammac}(a, x)

Arguments

\text{a} \quad \text{A numeric vector.}
\text{x} \quad \text{A nonnegative numeric vector with same length as a.}

Value

A vector of values of a complementary incomplete gamma function.

See Also

\text{igamma()}

Examples

\text{igammac}(1, 1)

\begin{verbatim}
mix2gauss

Datasets of Mixture of 2 Gaussian Distributions

Description

Dataset generated by mixture of 2 Gaussian Distributions whose density is:

\[ \frac{0.3}{\sqrt{2\pi0.5^2}} \exp\left(\frac{(x + 1)^2}{2 \cdot 0.5^2}\right) + \frac{0.7}{\sqrt{2\pi0.5^2}} \exp\left(\frac{(x - 1)^2}{2 \cdot 0.5^2}\right) \]

Usage

\text{mix2gauss}

Format

A list of numeric vectors of Bimodal Gaussian Mixture Model.

- \text{n200} A numeric vector with 200 elements.
- \text{n400} A numeric vector with 400 elements.
- \text{n600} A numeric vector with 600 elements.
- \text{n800} A numeric vector with 800 elements.
- \text{n1000} A numeric vector with 1000 elements.
- \text{n1200} A numeric vector with 1200 elements.

Source

\text{mix2gauss_gen()}

Dataset of Mixture of 2 Gaussian Distributions: Histogram version

Description

Dataset of Mixture of 2 Gaussian Distributions, histogram version, whose density is:

\[
0.3 \sqrt{2\pi0.5^2} \exp \left( \frac{(x + 1)^2}{2 \cdot 0.5^2} \right) + 0.7 \sqrt{2\pi0.5^2} \exp \left( \frac{(x - 1)^2}{2 \cdot 0.5^2} \right)
\]

Usage

mix2gaussHist

Format

A list of numeric vectors of Bimodal Gaussian Mixture Model.

- **n200** A numeric vector with 200 elements.
- **n200p** Histogram sample data with 25 bins.
- **n200f** Histogram frequency data with 25 bins.
- **n400** A numeric vector with 400 elements.
- **n400p** Histogram sample data with 50 bins.
- **n400f** Histogram frequency data with 50 bins.
- **n800** A numeric vector with 800 elements.
- **n800p** Histogram sample data with 100 bins.
- **n800f** Histogram frequency data with 100 bins.

Source

mix2gauss_gen() databinning()
mix2gauss_fun  
*A density function of mixed Gaussian distributions*

**Description**

A density function of mixed Gaussian distributions whose density is:

\[
\frac{0.3}{\sqrt{2\pi 0.5^2}} \exp \left( \frac{(x + 1)^2}{2 \cdot 0.5^2} \right) + \frac{0.7}{\sqrt{2\pi 0.5^2}} \exp \left( \frac{(x - 1)^2}{2 \cdot 0.5^2} \right)
\]

**Usage**

mix2gauss_fun(x)

**Arguments**

- **x**: A numeric vector for arguments of a density function.

**Value**

A numeric vector of probabilities for a given argument x.

**See Also**

mix2gauss_gen()

---

mix2gauss_gen  
*Generate mixed Gaussian random numbers*

**Description**

Generate mix gaussian random numbers whose density is:

\[
\frac{0.3}{\sqrt{2\pi 0.5^2}} \exp \left( \frac{(x + 1)^2}{2 \cdot 0.5^2} \right) + \frac{0.7}{\sqrt{2\pi 0.5^2}} \exp \left( \frac{(x - 1)^2}{2 \cdot 0.5^2} \right)
\]

**Usage**

mix2gauss_gen(n = 100, seed = NULL)

**Arguments**

- **n**: The number of random numbers.
- **seed**: A seed for random number generator.
mix3gauss

Value
A numeric vector of random numbers whose a density is described in Description.

See Also
mix2gauss_fun()

mix3gauss Datasets of Mixture of 3 Gaussian Distributions

Description
Datasets generated by Mixture of 3 Gaussian Distributions whose density is proportional to:
\[
\exp\left(\frac{x^2}{2}\right) + 5 \exp\left(\frac{(x - 1)^2}{0.2}\right) + 3 \exp\left(\frac{(x - 1)^2}{0.5}\right).
\]

Usage
mix3gauss

Format
A list of numeric vectors of Unimodal Gaussian Asymmetric Mixture Model.

- **n200** A numeric vector with 200 elements.
- **n400** A numeric vector with 400 elements.
- **n600** A numeric vector with 600 elements.
- **n800** A numeric vector with 800 elements.
- **n1000** A numeric vector with 1000 elements.
- **n1200** A numeric vector with 1200 elements.

Source
mix3gauss_gen()
### mix3gauss_fun

**A density function of mixed gaussian distribution**

**Description**

A density function proportional to:

$$\exp\left(\frac{x^2}{2}\right) + 5\exp\left(\frac{(x - 1)^2}{0.2}\right) + 3\exp\left(\frac{(x - 1)^2}{0.5}\right).$$

**Usage**

mix3gauss_fun(x)

**Arguments**

- **x**  
  A numeric vector for arguments of a density function.

**Value**

A numeric vector of probabilities for a given argument x.

**See Also**

mix3gauss_gen()

---

### mix3gauss_gen

**Generate Mixed Gaussian Random Numbers**

**Description**

A random number generator whose density function is proportional to:

$$\exp\left(\frac{x^2}{2}\right) + 5\exp\left(\frac{(x - 1)^2}{0.2}\right) + 3\exp\left(\frac{(x - 1)^2}{0.5}\right).$$

**Usage**

mix3gauss_gen(n = 100, seed = NULL)

**Arguments**

- **n**  
  The number of random numbers.

- **seed**  
  A seed for random number generator.

**Value**

A numeric vector of probabilities for a given argument x.
**mixexpgamma**

See Also

- `mix3gauss_fun()`

---

**mixexpgamma**

**Dataset of Mixture of Exponential Distribution and Gamma Distribution**

**Description**

Dataset of mixture of exponential distribution and gamma distribution whose density is:

\[
0.2(2e^{-2x}) + 0.8 \frac{x^3}{3!} e^{-x}.
\]

**Usage**

`mixexpgamma`

**Format**

A list of numeric vectors of Mixture of Exponential and Gamma distribution Model.

- `n200` A numeric vector with 200 elements.
- `n400` A numeric vector with 400 elements.
- `n600` A numeric vector with 600 elements.
- `n800` A numeric vector with 800 elements.
- `n1000` A numeric vector with 1000 elements.
- `n1200` A numeric vector with 1200 elements.

**Source**

`mixexpgamma_gen()`

---

**mixExpGammaHist**

**Dataset of Mixture of Exponential Distribution and Gamma Distribution: Histogram Version**

**Description**

Dataset of mixture of exponential distribution and gamma distribution, histogram version, whose density is:

\[
0.2(2e^{-2x}) + 0.8 \frac{x^3}{3!} e^{-x}.
\]
Usage

mixExpGammaHist

Format

A list of numeric vectors of Mixture of Exponential and Gamma distribution Model.

n200  A numeric vector with 200 elements.
n200p  Histogram sample data with 25 bins.
n200f  Histogram frequency data with 25 bins.
n400  A numeric vector with 400 elements.
n400p  Histogram sample data with 50 bins.
n400f  Histogram frequency data with 50 bins.
n800  A numeric vector with 800 elements.
n800p  Histogram sample data with 100 bins.
n800f  Histogram frequency data with 100 bins.

Source

mix2gauss_gen() databinning()

Description

A density function of

\[ 0.2(2e^{-2x}) + 0.8 \frac{x^3}{3!} e^{-x}. \]

Usage

mixexpgamma_fun(x)

Arguments

x  A numeric vector for arguments of a density function.

Value

A numeric vector of probabilities for a given argument x.

See Also

mixexpgamma_gen()
mixexpgamma_gen

Generate random numbers of Mixed Exponential and Gamma Distributions

Description

Generate random numbers whose density function:

\[0.2(2e^{-2x}) + 0.8\frac{x^3}{3!}e^{-x}.

Usage

mixexpgamma_gen(n = 100, seed = NULL)

Arguments

- n: The number of random numbers.
- seed: A seed for random number generator.

Value

A numeric vector of probabilities for a given argument \(x\).

See Also

mixexpgamma_fun()

pdf_expmodel

Probability density function of Exponential-based model

Description

A probability density function (PDF) of Exponential-based model. It is an underlying routine for plot.expmodel to compute the values of PDF. To access parameters and coefficients in an object emodel of a class expmodel, use emodel$result[k, "lmd1"], emodel$coeffs[[k]] for some index \(k\). This index appears in the leftmost column of estimation table generated by summary(emodel).

Usage

pdf_expmodel(coef, lmd, x)

Arguments

- coef: A coefficient vector in increasing order of degrees; the first element is 0th degree, ..., and last element is the largest degree of coefficients.
- lmd: A rate parameter of an exponential distribution, which is positive.
- x: A numeric input vector.
Value

A numeric vector of PDF of an exponential-based model.

See Also

expmodel() summary.expmodel() estimate.expmodel() func.expmodel() plot.expmodel()
cdf_expmodel()

Examples

## Create an object of `expmodel`
emodel <- expmodel(mixexpgamma$n200)
## Estimate with degree 4 and rate parameter 2.0
emodel <- estimate(emodel, 4, 2.0)
## Input vector
x <- seq(0, 12, 0.1)
## Output of PDF in above estimation
yv <- pdf_expmodel(emodel$coeffs[[1]], emodel$result[1, "lmd1"], x)

pdf_gaussmodel

Probability density function of Gaussian-based model

Description

A probability density function (PDF) of a Gaussian model. It is an underlying routine for plot.gaussmodel to compute the values of PDF. To access parameters and coefficients in an object gmodel of a class gaussmodel, use gmodel$result[k, "mu1"], gmodel$result[k, "sig1"], gmodel$coeffs[[k]] for some index k. This index appears in the leftmost column of estimation table generated by summary(gmodel).

Usage

pdf_gaussmodel(coef, mu, sig, x)

Arguments

coeff A coefficient vector in increasing order of degrees; the first element is 0th degree, ..., and last element is the largest degree of coefficients.

mu A mean of Gaussian distribution.

sig A standard deviation of Gaussian distribution, which is positive.

x A numeric input vector.

Value

A numeric vector of PDF of Gaussian-based distribution.
See Also

`gaussmodel() summary.gaussmodel() estimate.gaussmodel() func.gaussmodel() plot.gaussmodel()
cdf.gaussmodel()`

Examples

```r
## Create an object of `gaussmodel`
gmodel <- gaussmodel(mix2gauss$n200)
## Estimate with a degree 6, a mean 0, and standard deviations 0.5
gmodel <- estimate(gmodel, 6, 0, 0.5)
## Input vector
x <- seq(-3, 3, 0.1)
## Output of PDF in above estimation
yv <- pdf_gaussmodel(gmodel$coeffs[[1]], gmodel$result[1, "mu1"],
gmodel$result[1, "sig1"], x)
```

**Description**

Plot the histogram and, if available, estimated densities or cumulative distributions of expmodel object.

**Usage**

```r
## S3 method for class 'expmodel'
plot(
x,  
cum = FALSE,
nmax = 4,
graphs = NULL,
bins = 40,
hist = TRUE,
linesize = 1,
...)
```

**Arguments**

- `x` expmodel object.
- `cum` A logical scalar, whether or not it plots cumulative histogram/distributions instead of plain histogram/densities. Default value is `FALSE`.
- `nmax` A maximum number of estimates to be plotted in the graph. The default value is 4.
graphs
A vector of indices to be displayed in the graph. These indices appear in the leftmost column of the table in \texttt{summary.expmodel}. The default value is \texttt{NULL}, and if it is not \texttt{NULL}, only the estimated densities designated by graphs option appear, and \texttt{nmax} is ignored.

bins
A number of bins of the histogram.

hist
A logical scalar. If \texttt{TRUE}, display a histogram, otherwise not. The default value is \texttt{TRUE}.

linesize
A positive numeric scalar, which indicates the thickness of lines. The default value is 1.

... Arguments to be passed to or from other methods.

Value
A \texttt{ggplot2} object.

See Also
\texttt{expmodel()} \texttt{summary.expmodel()} \texttt{func.expmodel()} \texttt{pdf.expmodel()} \texttt{cdf.expmodel()}

Examples
```r
## Create \code{expmodel} object from a data set \code{mixexpgamma$n200}
emodel <- expmodel(mixexpgamma$n200)
## Plot it (histogram only)
plot(emodel)
```

---

**plot.gaussmodel**

Plot a histogram and estimated densities/distributions of Gaussian-based model object

Description
Plot the histogram and, if available, estimated densities or cumulative distributions of \texttt{gaussmodel} object.

Usage
```r
## S3 method for class 'gaussmodel'
plot(  
  x,  
  cum = FALSE,  
  nmax = 4,  
  graphs = NULL,  
  bins = 40,  
  hist = TRUE,  
  scaling = FALSE,  
  linesize = 1,  
  ...  
)
```
Arguments

x  gaussmodel object.
cum  A logical scalar, whether or not it plots cumulative histogram/distributions instead of plain histogram/densities. Default value is FALSE.
nmax  A maximum number of estimates to be plotted in the graph. The default value is 4.
graphs  A vector of indices to be displayed in the graph. These indices appear in the leftmost column of the table in estimate.gaussmodel. The default value is NULL, and if it is not NULL, only the estimated densities designated by graphs option appear, and nmax is ignored.
bins  A number of bins of the histogram.
hist  A logical scalar. If TRUE, display a histogram, otherwise not. The default value is TRUE.
scaling  A logical scalar, which indicates whether or not it scales means and standard deviations in mulist and sdlist. The default value is FALSE.
linesize  A positive numeric scalar, which indicates the thickness of lines. The default value is 1.
...  Arguments to be passed to or from other methods.

Value

A ggplot2 object.

See Also

 gaussmodel() summary.gaussmodel() func.gaussmodel() pdf_gaussmodel() cdf_gaussmodel()

Examples

## Create `gaussmodel` object from a data set mix2gauss$n200
gmodel <- gaussmodel(mix2gauss$n200)
## Plot it (histogram only)
plot(gmodel)

polyaxb  Substitute a coefficient of polynomial

Description

Substitute a coefficient of a polynomial with $a \cdot x + b$. For a polynomial with a coefficient vector \(\text{poly}(x; \text{coeff})\), compute the coefficient vector of

\[ \text{poly}(a \cdot x + b; \text{coeff}) .\]
Usage

polyaxb(coeff, c, a, b)

Arguments

coeff  A coefficient vector in increasing order of degrees; the first element is 0th degree, ..., and the last element is the largest degree of coefficients.
c    A multiple factor of constant to be applied to all coefficients.
a    A coefficient of 1st degree of \(ax + b\).
b    A coefficient of 0th degree of \(ax + b\).

Value

A substituted coefficient.

See Also

eval_poly()

Examples

coeff <- c(2, -2, 1)
a <- 1.1
b <- 1.2
coeff1 <- c(b, a)
coeff2 <- polyaxb(coeff, 1, a, b)
xv <- c(1, 2, 3)
## a*x + b
yv1 <- eval_poly(coeff1, xv)
## polynomial(a*x + b, coeff)
yv2 <- eval_poly(coeff, yv1)
## polynomial(x, coeff2)
yv <- eval_poly(coeff2, xv)
## This value is nearly 0 in the presence of rounding errors
yv - yv2

Description

printf

Usage

printf(...)

printf
summary.expmodel

Arguments

... Any number of arguments to be printed.

Value

None.

Summary of Exponential-based expmodel object.

Description

Summary of expmodel object, including a mean and quantiles. If some estimation has done, also print out estimates, up to nmax number of them.

Usage

## S3 method for class 'expmodel'
summary(object, nmax = 10, estonly = FALSE, ...)

Arguments

object expmodel object.
nmax A number of estimates to show in the summary. The default is 10.
estonly Show only the results of estimates. The default value is FALSE.
... Arguments to be passed to or from other methods.

Value

None.

See Also

expmodel() plot.expmodel() estimate.expmodel()

Examples

## Create expmodel object from a data set mixexpgamma$n200
emodel <- expmodel(mixexpgamma$n200)
## Print a summary of an object
summary(emodel)
Summary of Gaussian-based model

**Description**

Summary of `gaussmodel` object, including a mean and a standard deviation and quantiles. If some estimation has done, also print out estimates, up to `nmax` number of them.

**Usage**

```r
## S3 method for class 'gaussmodel'
summary(object, nmax = 10, estonly = FALSE, ...)
```

**Arguments**

- `object` `gaussmodel` object.
- `nmax` A number of estimates to show in the summary. The default is 10.
- `estonly` Show only the results of estimates. The default value is `FALSE`.
- `...` Arguments to be passed to or from other methods.

**Value**

None.

**See Also**

- `gaussmodel()`
- `plot.gaussmodel()`
- `estimate.gaussmodel()`

**Examples**

```r
## Create gaussmodel object from a data set mix2gauss$n200
gmodel <- gaussmodel(mix2gauss$n200)
## Print a summary of an object
summary(gmodel)
```
Index

* datasets
  mix2gauss, 18
  mix2gaussHist, 19
  mix3gauss, 21
  mixexpgamma, 23
  mixExpGammaHist, 23

  histmean, 16
  histmean(), 6
  igamma, 17
  igamma(), 18
  igammac, 17
  igammac(), 17
  mix2gauss, 18
  mix2gauss_fun, 20
  mix2gauss_fun(), 21
  mix2gauss_gen, 20
  mix2gauss_gen(), 18–20, 24
  mix2gaussHist, 19
  mix3gauss, 21
  mix3gauss_fun, 22
  mix3gauss_fun(), 23
  mix3gauss_gen, 22
  mix3gauss_gen(), 21, 22
  mixexpgamma, 23
  mixexpgamma_fun, 24
  mixexpgamma_fun(), 25
  mixexpgamma_gen, 25
  mixexpgamma_gen(), 23, 24
  mixExpGammaHist, 23

  pdf_expmodel, 25
  pdf_expmodel(), 9, 13, 28
  pdf_gaussmodel, 26
  pdf_gaussmodel(), 4, 9, 14, 29
  plot.expmodel, 27
  plot.expmodel(), 7, 10, 13, 26, 31
  plot.gaussmodel, 28
  plot.gaussmodel(), 9, 14, 15, 27, 32
  polyaxb, 29
  printf, 30

  summary.expmodel, 31
  summary.expmodel(), 3, 7, 10, 13, 26, 28
  summary.gaussmodel, 32
  summary.gaussmodel(), 4, 9, 14, 15, 27, 29

base::rle(), 5

cdf_expmodel, 3
  cdf_expmodel(), 3, 9, 13, 26, 28
  cdf_gaussmodel, 4
  cdf_gaussmodel(), 9, 14, 27, 29

datafitting, 5
  datafitting(), 19, 24

  datatats, 5
  datatats(), 16
  dsdp, 6

  estimate, 6
  estimate.expmodel, 7
  estimate.expmodel(), 3, 10, 11, 13, 26, 31
  estimate.gaussmodel, 8
  estimate.gaussmodel(), 4,14–16, 27, 32
  eval_poly, 9
  eval_poly(), 30
  exp_est, 11
  expmodel, 10
  expmodel(), 3,7,13,26,28,31

  func, 12
  func.expmodel, 12
  func.expmodel(), 3, 26, 28
  func.gaussmodel, 13
  func.gaussmodel(), 4, 27, 29

  gauss_est, 15
  gaussmodel, 14
  gaussmodel(), 4, 9, 14, 27, 29, 32