

Package: UMDA (via r-universe)

October 10, 2026

Type Package

Title Univariate Marginal Distribution Algorithm

Description Implements the Univariate Marginal Distribution Algorithm (UMDA), an Estimation of Distribution Algorithm (EDA) for continuous optimization problems. The method iteratively selects the best individuals from a population, estimates an independent marginal probability distribution for each decision variable, and generates new candidate solutions by sampling from the estimated distributions. This process allows the probability model to adapt toward promising regions of the search space. The implementation supports normal, triangular, histogram-based, and uniform probability distributions, together with an optional explicit exploration strategy for the initialization of the population. The implemented explicit exploration strategy in this package is described in Salinas Gutierrez and Muñoz Zavala (2023) [doi:10.1016/j.asoc.2023.110230](https://doi.org/10.1016/j.asoc.2023.110230).

Version 0.1.0

License GPL-3

Encoding UTF-8

Imports EEEA

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Rogelio Salinas Gutiérrez [aut, cre, cph] (ORCID: <https://orcid.org/0000-0002-1669-4460>), Juan Alberto Dávila del Alto [aut, cph] (ORCID: <https://orcid.org/0009-0007-2798-6251>), Jhon Daniel Aguilar Payares [aut, cph] (ORCID: <https://orcid.org/0009-0009-6509-4486>), Jonathan Michell Jauregui Carranza [aut, cph] (ORCID: <https://orcid.org/0009-0005-2688-8065>), Hiram Efraim Macias Ruelas [aut, cph] (ORCID: <https://orcid.org/0009-0009-3847-159X>), Byron Axel Morales

Gutiérrez [aut, cph] (ORCID:
<<https://orcid.org/0009-0006-3446-3830>>), María Fernanda Nieto
Guerrero [aut, cph] (ORCID:
<<https://orcid.org/0009-0000-1738-1733>>), Manuel Alonso
Segoviano Baltazar [aut, cph] (ORCID:
<<https://orcid.org/0009-0007-1494-7472>>), Pedro Abraham Montoya
Calzada [aut, cph] (ORCID:
<<https://orcid.org/0009-0002-3497-210X>>)

Maintainer Rogelio Salinas Gutiérrez <rogelio.salinas@edu.uaa.mx>
Repository <https://cran.r-universe.dev>
Date/Publication 2026-10-10 13:52:08 UTC
RemoteUrl <https://github.com/cran/UMDA>
RemoteRef HEAD
RemoteSha 193d55027b4f683295e940eeefd0396c523a7559

Contents

etrian	2
rhist	4
rtrian	5
UMDA	6
Index	10

etrian	<i>Estimation of Parameters of a Triangular Distribution</i>
--------	--

Description

Empirically estimates the parameters a (minimum), b (maximum), and c (mode) of a triangular distribution based on a vector or matrix of observed data.

Usage

etrian(X,lowconver)

Arguments

- | | |
|-----------|---|
| X | A numeric vector, matrix, or data frame containing the observed data. If a matrix or data frame is passed, the parameter estimation is performed independently for each column. |
| lowconver | Boolean flag to control convergence speed. TRUE slows down convergence by expanding the search space with wider intervals. FALSE uses standard estimation. Default is [FALSE]. |

Details

The function calculates the bounds using the empirical minimum (a) and the empirical maximum (b). To estimate the mode (c), it applies the standard formula: $c = 3 * \mu - a - b$.

The function includes several error-control safeguards:

- If a column has fewer than 2 data points or zero variance, it assigns the mean as the mode without using density.
- If extreme values are NA or infinite, the default values $a = -1$, $b = 1$, $c = 0$ are assigned.
- If the difference between the maximum and minimum is less than $1e-5$, it slightly expands the limits ($1e-4$) to ensure mathematical stability.
- It ensures that the estimated mode c always remains within the strict range $[a, b]$.

Value

Returns a list with three vector components:

a	Estimated lower bound (estimated minimum).
b	Estimated upper bound (estimated maximum).
c	Estimated mode (point of maximum density).

Note

If the data do not vary within a column, the function will fall back to the default behavior, assigning the mean as the estimated mode.

Author(s)

Rogelio Salinas Gutiérrez, Juan Alberto Dávila del Alto, Jhon Daniel Aguilar Payares, Jonathan Michell Jauregui Carranza, Hiram Efraim Macias Ruelas, Byron Axel Morales Gutiérrez, María Fernanda Nieto Guerrero, Manuel Alonso Segoviano Baltazar, Pedro Abraham Montoya Calzada.

References

Kotz, S., & van Dorp, J. R. (2004). Beyond Beta: Other Continuous Families of Distributions with Bounded Support and Applications. World Scientific.

See Also

[density](#), [apply](#), [min](#), [max](#)

Examples

```
# --- Example of use ---
set.seed(123)
example_data <- rnorm(500, mean=38, sd=2)
result <- etrian(example_data)
print(result)
```

rhist

Data Simulation from a Histogram

Description

Generates a set of simulated continuous data based on the information from a histogram object. The function calculates the probability of each interval based on its frequencies and generates uniformly distributed values within the limits of the selected interval.

Usage

```
rhist(histogram)
```

Arguments

histogram	An object of class "histogram", for example, the result of running <code>hist(x)</code> , where <code>x</code> follows any distribution with size <code>n</code> .
-----------	--

Details

The function performs a two-stage stochastic simulation process:

1. **Interval selection:** The empirical probability of each bin is calculated by dividing its frequency by the total number of observations. Then, `sample()` is used to assign each new observation to an interval proportionally to its original weight.
2. **Continuous generation:** Once the interval is assigned, a continuous random value is generated using a uniform distribution (`runif`) bounded by the limits of the corresponding bin (`min = intervals[i]`, `max = intervals[i+1]`).

This approach is useful when only the aggregated frequencies of a histogram are available and an approximation of the original continuous data needs to be recreated.

Value

Returns a vector of size `n` that follows the distribution of the histogram generated by `hist(x)`, which can be visualized by applying the `hist` function to the generated vector.

Author(s)

Rogelio Salinas Gutiérrez, Juan Alberto Dávila del Alto, Jhon Daniel Aguilar Payares, Jonathan Michell Jauregui Carranza, Hiram Efraim Macías Ruelas, Byron Axel Morales Gutiérrez, María Fernanda Nieto Guerrero, Manuel Alonso Segoviano Baltazar, Pedro Abraham Montoya Calzada.

Examples

```
data <- rnorm(200, 5, 3)
x <- hist(data)

new_data <- rhist(x)
hist(new_data) #image of the new data.
```

rtrian*Generate Random Values from a Triangular Distribution*

Description

Generates random values from a triangular probability distribution using the inverse transform sampling method.

Usage

```
rtrian(n, a, b, c)
```

Arguments

n	The number of random values to generate.
a	The lower bound of the triangular distribution.
b	The upper bound of the triangular distribution.
c	The mode of the triangular distribution, corresponding to the value where the probability density reaches its maximum.

Details

The function generates random values from a triangular distribution defined by the lower bound a , upper bound b , and mode c .

The random values are generated using the inverse transform sampling method. First, a uniform random variable is generated using `runif`. The value is then transformed according to the cumulative distribution function of the triangular distribution.

For values below the cumulative probability at the mode, the function uses the increasing part of the triangular distribution. For values above this probability, the function uses the decreasing part.

The parameters must strictly satisfy $a < c < b$. If the estimated mode falls outside this range, c is automatically adjusted to the midpoint of the $[a, b]$ interval.

Value

Returns a numeric vector containing n randomly generated values from the triangular distribution defined by a , b , and c .

Note

The function uses a uniform random number generator internally. Therefore, the generated values depend on the random seed set by `set.seed`.

The mode `c` must lie between the lower and upper bounds.

Author(s)

Rogelio Salinas Gutiérrez, Juan Alberto Dávila del Alto, Jhon Daniel Aguilar Payares, Jonathan Michell Jauregui Carranza, Hiram Efraim Macias Ruelas, Byron Axel Morales Gutiérrez, María Fernanda Nieto Guerrero, Manuel Alonso Segoviano Baltazar, Pedro Abraham Montoya Calzada.

References

Kotz, S., & van Dorp, J. R. (2004). *Beyond Beta: Other Continuous Families of Distributions with Bounded Support and Applications*. World Scientific.

Examples

```
set.seed(123)

# Generate 500 random values with min=0, max=10 and mode=5
simulated_values <- rtrian(n = 500, a = 0, b = 10, c = 5)

# See the results
head(simulated_values)

# Generate the histogram
hist(
  simulated_values,
  breaks = 20,
  col = "skyblue",
  border = "darkblue",
  main = paste("Histogram with the simulated values\n(n =", 500, ")"),
  xlab = "Values",
  ylab = "Frequency"
)
```

Description

Applies a Univariate Marginal Distribution Algorithm (UMDA) to minimize an objective function. The algorithm iteratively selects the best half of the population and generates a new population according to a specified probability distribution.

Usage

```
UMDA(fn, lower, upper, n, d, iter, dist, eee, lowconver)
```

Arguments

fn	A function representing the objective function to be minimized. The function must receive a numeric vector which represents an individual and return a numeric fitness value.
lower	Lower bounds for the variables. A single numeric value can be provided and will be replicated across all d dimensions. Alternatively, a numeric vector of length d can be used.
upper	Upper bounds for the variables. A single numeric value can be provided and will be replicated across all d dimensions. Alternatively, a numeric vector of length d can be used.
n	The population size. Must be a positive integer indicating the number of individuals generated in each iteration.
d	The dimensionality of the optimization problem, corresponding to the number of variables in each individual.
iter	The number of generations or iterations of the algorithm, it must be provided.
dist	The probability distribution used to generate new individuals. Supported values are "norm" for the normal distribution, "tri" for the triangular distribution, "hist" for a histogram-based distribution, and "unif" for the uniform distribution. The default value is "norm".
eeee	Selecte whether to use EEEA (Explicit Exploration Strategy for Evolutionary Algorithms) to generate the initial population (TRUE or FALSE)
lowconver	Boolean flag to control convergence speed. TRUE slows down convergence by expanding the search space with wider intervals in each iteration. FALSE uses standard interval estimation. Default is FALSE."

Details

The UMDA is a population-based evolutionary optimization algorithm that estimates the probability distribution of promising solutions and uses this distribution to generate new candidate solutions.

The algorithm begins by generating an initial population uniformly at random within the specified lower and upper bounds. At each iteration, the objective function is evaluated for every individual in the population. Individuals are then sorted according to their fitness, and the best half of the population is selected.

The probability distribution of each variable is estimated independently from the selected individuals. A new population is then generated from these estimated distributions.

When `dist = "norm"`, the mean and standard deviation of each variable are estimated from the selected individuals, and a new population is generated using a normal distribution.

When `dist = "tri"`, the parameters of a triangular distribution are estimated using the selected individuals. The new population is then generated using the corresponding triangular distributions.

When `dist = "hist"`, a histogram is constructed for each variable using the selected individuals. The histogram is then used to simulate the values of the corresponding variable.

When `dist = "unif"`, the parameters of a uniform distribution are estimated using the selected individuals. The new population is then generated using the corresponding uniform distributions.

The algorithm assumes that lower objective function values represent better solutions. Therefore, individuals are ordered in ascending order according to their objective function value.

Value

A list containing the best solution found during the final iteration and its corresponding objective function value.

<code>par</code>	A numeric vector containing the best individual selected in the final iteration.
<code>value</code>	A numeric value representing the objective function value of the best individual in the final iteration.
<code>Model</code>	Statistical model used in the function, if none is selected, "norm" is used by default.

Note

The function performs minimization. Therefore, the objective function must return smaller values for better solutions.

The functions `etrian`, `rtrian`, and `rhist` must be available in the R environment when using the corresponding distribution options.

The initial population is generated uniformly within the specified bounds. However, subsequent populations are generated from the estimated distributions and are not explicitly restricted to the original bounds.

Author(s)

Rogelio Salinas Gutiérrez, Juan Alberto Dávila del Alto, Jhon Daniel Aguilar Payares, Jonathan Michell Jauregui Carranza, Hiram Efraim Macias Ruelas, Byron Axel Morales Gutiérrez, María Fernanda Nieto Guerrero, Manuel Alonso Segoviano Baltazar, Pedro Abraham Montoya Calzada.

References

Larranaga, P. and Lozano, J. A. (2002). Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. Kluwer Academic Publishers.

Examples

```
# Sphere objective function
sphere <- function(x) {
  return(sum(x^2))
}

# Run UMDA using a normal distribution
set.seed(123)

result <- UMDA(
  fn = sphere,
  lower = -50,
  upper = 50,
  n = 20,
  d = 2,
  iter = 100,
  dist = "norm",
```

```
        eeea =FALSE,  
        lowconver=FALSE  
    )  
  
    # Best individual  
    result$par  
  
    # Best objective function value  
    result$value
```

Index

- * **datagen**
 - rhist, [4](#)
- * **distribution**
 - etrian, [2](#)
 - rhist, [4](#)
 - rtrian, [5](#)
 - UMDA, [6](#)
- * **evolutionary**
 - UMDA, [6](#)
- * **nonparametric**
 - rhist, [4](#)
- * **optimization**
 - UMDA, [6](#)
- * **random**
 - rtrian, [5](#)
- * **robust**
 - etrian, [2](#)
- * **simulation**
 - rtrian, [5](#)

apply, [3](#)

density, [3](#)

etrian, [2](#)

max, [3](#)

min, [3](#)

rhist, [4](#)

rtrian, [5](#)

runif, [5](#)

set.seed, [6](#)

UMDA, [6](#)