Package: superstats (via r-universe)

November 3, 2026

Type Package

Title Super Statistics

Version 1.0

Date 2026-10-21

Author Michail Tsagris [aut, cre], Manos Papadakis [aut], Alzeley Omar [aut, cph]

Maintainer Michail Tsagris <mtsagris@uoc.gr>

Depends R (>= 4.0)

Imports graphics, grDevices, Rfast, Rfast2

Description Superstatistics generally applies to analyzing complex systems where traditional statistical methods are insufficient. This approach is particularly valuable in areas like physics, finance, and environmental science. It steps in when there are underlying variables or parameters in a system that fluctuate significantly over space or time, which traditional statistics might overlook. For more information see: Alzeley O. (2020) <doi:10.1002/cmm4.1085>.

License GPL (>= 2)

NeedsCompilation no

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

Date/Publication 2026-11-03 21:18:56 UTC

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

RemoteRef HEAD

RemoteSha bee53faae0b118e246cfeefc6b98ca63e1cd59fe

Contents

superstats-package 2
createalfa 3
find_cf 4
main 6

Index **11**

`superstats-package` *Super Statistics.*

Description

Superstatistics" generally applies to analyzing complex systems where traditional statistical methods are insufficient. This approach is particularly valuable in areas like physics, finance, and environmental science. It steps in when there are underlying variables or parameters in a system that fluctuate significantly over space or time, which traditional statistics might overlook.In practical terms, superstatistics could be doing the following:

1- Analyzing Turbulent Fluid Flow: In physics, especially in the study of turbulence in fluids, superstatistics helps in understanding the statistical properties of the flow under varying conditions like temperature or pressure.

2- Financial Markets Analysis: In finance, market conditions are extremely volatile and influenced by numerous factors. Superstatistics can provide insights into risk assessment and price movements by accounting for these underlying fluctuations.

3- Environmental and Climatic Studies: Climate systems are inherently complex and dynamic. Superstatistics can be used to better understand and predict climatic changes by integrating the variability of factors like temperature, pressure, and humidity over time.

4- Biological Systems Analysis: In biology, processes like the spread of diseases or ecosystem dynamics often involve layers of complexity. Superstatistics allows for more nuanced modeling of these processes.

The key feature of superstatistics is its ability to integrate the effects of fluctuating parameters into a more comprehensive statistical framework, offering deeper insights into complex systems where traditional statistical methods may fall short.

Details

Package:	nlgmcr	Type:	Package
Version:	1.0		
Date:	2026-10-21		
License:	GPL-2		

Maintainers

Michail Tsagris <mtsagris@uoc.gr>

Author(s)

Michail Tsagris <mtsagris@uoc.gr>, Manos Papadakis <papadakm95@gmail.com> and Omar Alzeley <oazeley@uqu.edu.sa>.

References

Alzeley O. (2020). On a Statistical Learning Algorithm of Superstatistics. Computational and Mathematical Methods, 2(3), e1085.

`createalfa` *Grid of all combinations of parameter values*

Description

Creates a matrix with all possible combinations of the candidate values of several parameters.

Usage

```
createalfa(alfax)
```

Arguments

`alfax`	A numeric matrix with one row per parameter. The i-th row contains the candidate values of the i-th parameter. All rows must have the same number of columns.

Details

The function creates all possible combinations (the Cartesian product, or full grid) of the values stored in the rows of `alfax`, by picking exactly one value from each row. It works recursively: if `alfax` has a single row, the transposed row is returned; otherwise, each value of the first row is repeated and bound (`cbind`) with the combinations obtained from the remaining rows.

If `alfax` has p rows and m columns, the output has m^p rows and p columns. The first column varies slowest and the last column varies fastest. Note that the number of rows grows exponentially with the number of rows of `alfax`, so large inputs can require a lot of memory.

Value

A matrix with `ncol(alfax)^nrow(alfax)` rows and `nrow(alfax)` columns. Each row of the matrix is one combination of values, with the j-th element taken from the j-th row of `alfax`.

Author(s)

Original Matlab code by Omar Alzeley.

R implementation and documentation: Manos Papadakis and Michail Tsagris.

References

Alzeley O. (2020). On a Statistical Learning Algorithm of Superstatistics. Computational and Mathematical Methods, 2(3), e1085.

See Also

main, find_cf

Examples

```
## two parameters with three candidate values each
x <- rbind( c(1, 2, 3), c(10, 20, 30) )
createalfa(x)

## three parameters with two candidate values each
y <- rbind( c(0.1, 0.5), c(1, 2), c(5, 6) )
dim( createalfa(y) )  ## 8 rows, 3 columns

## a single row is simply transposed
createalfa( rbind(1:3) )
```

---

find_cf *Characteristic function estimation with a learning algorithm*

Description

Estimates the characteristic function of multivariate data over a grid of points using a learning (neural network type) algorithm.

Usage

```
find_cf(x, outrange, N, nu, sg, g)
```

Arguments

x	A numeric matrix with the data. The rows are the observations (n) and the columns are the variables (d).
outrange	A vector with two values, the lower and upper limits of the grid at which the characteristic function is evaluated. The same range is used in every dimension.
N	A single integer value, the number of grid points (cells) in each dimension. The characteristic function is evaluated at N^d points in total.
nu	A single numeric value, the learning rate for the weights of the networks.
sg	A single positive numeric value, the standard deviation of the Gaussian kernel.
g	A single numeric value, the learning rate used when updating the estimate of the characteristic function with each observation.

Details

This function finds the characteristic function of the given data `x`. The characteristic function is evaluated on a regular grid of N^d points, with `N` equally spaced values between `outrange[1]` and `outrange[2]` in each of the d dimensions. The grid is created with `createalfa`. The parameters `nu`, `sg` and `g` are the learning parameters of the algorithm.

Two networks are used, one for the real part and one for the imaginary part of the characteristic function. Their values are initialised randomly and then updated once for each observation (row of `x`), using the cosine and sine of the inner product between the grid points and the observation. A Gaussian kernel with standard deviation `sg`, evaluated between all pairs of grid points, is used in the updating of the weights.

Since the starting values are random, the results depend on the random number generator, so use `set.seed` for reproducibility. The kernel matrix has $N^d \times N^d$ elements, hence the computational cost and memory requirements grow very fast with `N` and with the number of variables.

Value

A list including:

`Teta`	The real part of the characteristic function, evaluated at the grid points (one row for each row of `Alfa`).
`TetaI`	The imaginary part of the characteristic function, evaluated at the grid points.
`Alfa`	The domain of the output generated, i.e. a matrix with N^d rows and d columns containing the grid points at which the characteristic function was evaluated.

Author(s)

Original Matlab code by Omar Alzeley.

R implementation and documentation: Manos Papadakis and Michail Tsagris.

References

Alzeley O. (2020). On a Statistical Learning Algorithm of Superstatistics. Computational and Mathematical Methods, 2(3), e1085.

See Also

`main`, `createalfa`

Examples

```
set.seed(1)
## univariate data
x <- matrix( rnorm(100), ncol = 1 )
a <- find_cf(x, outrange = c(-2, 2), N = 11, nu = 0.1, sg = 0.5, g = 0.1)
cbind(a$Alfa, a$Teta, a$TetaI)

## bivariate data
y <- matrix( rnorm(100 * 2), ncol = 2 )
```

```
b <- find_cf(y, outrange = c(-1, 1), N = 5, nu = 0.1, sg = 0.5, g = 0.1)
dim(b$Alfa)  ## 25 rows, 2 columns
```

`main` *Characteristic function estimation with a learning algorithm: simulation example*

Description

Generates data from a Gaussian or a uniform distribution, estimates their characteristic function with a learning algorithm and compares it with the theoretical characteristic function.

Usage

```
main(N = 20, d = 2, n = 500, nu = 1.8, sg = 0.25, g = 0.00036, dist = 1)
```

Arguments

`N`	Number of grid points (resolution) in each dimension at which the characteristic function is evaluated.
`d`	Number of dimensions of the generated data. Plots are produced only for `d = 1` and `d = 2`.
`n`	Number of data to be generated.
`nu`	The value of the learning parameter (learning rate of the weights of the networks), see `find_cf`.
`sg`	The value of the learning parameter (standard deviation of the Gaussian kernel), see `find_cf`.
`g`	The value of the learning parameter (learning rate for updating the estimate of the characteristic function), see `find_cf`.
`dist`	The data distribution, 0 is for Gaussian and 1 is for uniform.

Details

If `dist = 0`, the data are generated from a `d`-dimensional Gaussian distribution with mean vector $(1, \ldots, 1)$ and covariance matrix $2I_d$, and the characteristic function is evaluated over the range $[-5, 5]$ in each dimension. Otherwise, the data are generated from the uniform distribution on $[-2, 3]$ in each dimension, and the characteristic function is evaluated over the range $[-3, 4]$. The characteristic function is then estimated with `find_cf` and compared with the theoretical characteristic function of the distribution that generated the data. The theoretical characteristic function is used to compute the error of the estimated one.

Three graphics devices are opened (via `dev.new`). They contain the generated data, the real and the imaginary part of the estimated characteristic function, and the real and imaginary part of the theoretical characteristic function together with the errors (differences between the theoretical and the estimated values).

The following is the theory behind the use of uniform distributions in superstatistics. Let $U[0,N]$ be a uniform variable on the interval $[0,N]$. Observe that

$$\mathrm{E}\,U[0,N] = \frac{1}{N}\int_0^N x\,dx = \frac{N}{2}.$$

Write

$$\begin{aligned}\mathrm{E}\,e^{-hU[0,N]} &= \frac{1}{N}\int_0^N e^{-hx}dx = e^{-hN/2}\frac{1}{N}\int_0^N e^{-h(x-N/2)}dx \\ &= e^{-hN/2}\int_0^N e^{-hN(x/N-1/2)}\frac{dx}{N}.\end{aligned}$$

Now, by considering a new variable of integration $t = \frac{x-N/2}{N}$,

$$\mathrm{E}\,e^{-hU[0,N]} = e^{-hN/2}\int_{-1/2}^{1/2} e^{-hNt}dt.$$

Assume that $c = hN$ is small. Then, by Taylor expansion, $e^{-ct} = 1 - ct + c^2t^2/2 - \cdots$ and so

$$\mathrm{E}\,e^{-hU[0,N]} = e^{-hN/2}\int_{-1/2}^{1/2}\left(1 - ct + \frac{c^2t^2}{2} - \cdots\right)dt = e^{-hN/2}\left(1 + \frac{c^2}{24} + \cdots\right).$$

Consider now the uniform distribution case, $U[a, a+b]$.

Let the random variable β be distributed uniformly in $[a, a+b]$ for some $a \geq 0$. Then, its probability density function reads as

$$f(\beta) = \begin{cases} \dfrac{1}{b}, & \text{if } a < \beta < a+b, \\ 0, & \text{otherwise.}\end{cases}$$

The mean and the variance of this distribution are therefore

$$\beta_0 = a + \frac{b}{2}, \qquad \sigma^2 = \frac{b^2}{12}. \qquad (1)$$

The superstatistics for this more general case can be found by the same argument. First, one should note that the generalised Boltzmann factor is defined by Beck and Cohen (2003) as

$$B = \int_0^\infty f(\beta)\exp(-\beta E)d\beta,$$

direct integration of which gives

$$\int_0^\infty f(\beta)\exp(-\beta E)d\beta = \int_a^{a+b}\frac{1}{b}\exp(-\beta E)d\beta = \frac{\exp(-aE)}{bE}\left[1 - \exp(-bE)\right].$$

Substituting a from (1) into the final expression of the integral, we have

$$B = \exp(-E\beta_0)\frac{\exp\left(E\frac{b}{2}\right) - \exp\left(-E\frac{b}{2}\right)}{bE}.$$

Taking into account the following series expansions for small bE:

$$\frac{\exp\left(\pm E\frac{b}{2}\right)}{bE} \approx \frac{1}{bE} \pm \frac{1}{2} + \frac{bE}{8} \pm \frac{(bE)^2}{48} + \cdots$$

we have

$$\frac{\exp\left(E\frac{b}{2}\right)-\exp\left(-E\frac{b}{2}\right)}{bE}\approx 1+\frac{(bE)^2}{24}+\frac{(bE)^4}{1920}+\cdots$$

Substituting this into the expression for B, we arrive at the expansion

$$B\approx\exp(-\beta_0 E)\left[1+\frac{(bE)^2}{24}+\frac{(bE)^4}{1920}+\cdots\right].$$

Let x be a discrete random variable with a discrete uniform distribution with parameter N, i.e. $x\in\{a+1,a+2,\ldots,a+N\}$. Then, the probability mass function is given by $f(x)=\frac{1}{N}$ and the mean is given by

$$\begin{aligned} x_0 &= \sum xf(x)=\frac{1}{N}\sum_{x=a+1}^{a+N}x=\frac{1}{N}\left((a+1)N+\sum_{j=0}^{N-1}j\right)\\ &= \frac{1}{N}\left((a+1)N+\frac{N(N-1)}{2}\right)=a+1+\frac{N-1}{2}=a+\frac{N+1}{2}. \end{aligned}$$

Let $I_{z,N}=\frac{1}{N}\sum_{j=a+1}^{a+N}z^j$. Then, taking into account that $a=x_0-\frac{N+1}{2}$, we have

$$\begin{aligned} I_{z,N} &= \frac{z^{a+1}+z^{a+2}+\cdots+z^{a+N}}{N}\\ &= \frac{z^{x_0-\frac{N+1}{2}+1}+z^{x_0-\frac{N+1}{2}+2}+\cdots+z^{x_0-\frac{N+1}{2}+N}}{N}\\ &= z^{x_0}\frac{z^{-\frac{N+1}{2}}\left(z+z^2+\cdots+z^N\right)}{N}\\ &= z^{x_0}\frac{z^{-\frac{N+1}{2}}z\left(z^N-1\right)}{N(z-1)}\\ &= \frac{z^{x_0}}{z-1}\frac{z^{\frac{1+N}{2}}-z^{\frac{1-N}{2}}}{N}. \end{aligned}$$

Note that we can also use the exponential form $z^{\frac{N}{2}}=\exp\left[\frac{N}{2}\ln z\right]$ to write

$$I_{z,N}=\frac{z^{x_0+\frac{1}{2}}}{z-1}\frac{\exp\left[\frac{N}{2}\ln z\right]-\exp\left[-\frac{N}{2}\ln z\right]}{N}.$$

Taking into account the expansions above, we obtain

$$\begin{aligned} \exp\left[\tfrac{N}{2}\ln z\right]-\exp\left[-\tfrac{N}{2}\ln z\right] &= \sum_{n=0}^{\infty}\frac{1}{n!}\left[\left(\frac{N}{2}\right)^n\ln^n z-\left(-\frac{N}{2}\right)^n\ln^n z\right]\\ &= \sum_{n=0}^{\infty}\frac{1}{n!}\left[\left(\frac{N}{2}\right)^n-(-1)^n\left(\frac{N}{2}\right)^n\right]\ln^n z. \end{aligned}$$

Evidently,

$$\left(\frac{N}{2}\right)^n-(-1)^n\left(\frac{N}{2}\right)^n=\begin{cases}2\left(\frac{N}{2}\right)^{2k+1}, & n=2k+1,\\ 0, & n=2k.\end{cases}$$

Therefore, finally, we arrive at

$$I_{z,N} = \frac{2z^{x_0+\frac{1}{2}}}{N\,(z-1)} \sum_{k=0}^{\infty} \frac{1}{(2k+1)!} \left(\frac{N}{2}\right)^{2k+1} \ln^{2k+1} z.$$

The Boltzmann factor for discrete uniform distributions reads as

$$B = \frac{1}{N} \sum_{j=a+1}^{a+N} \exp(-jE).$$

Then, taking into account that $a = x_0 - \frac{N+1}{2}$, we have

$$\begin{aligned} B &= \frac{\exp\left[\left(\frac{1}{2} - x_0\right) E\right]}{\exp(E) - 1} \frac{\exp\left(\frac{1}{2}NE\right) - \exp\left(-\frac{1}{2}NE\right)}{N} \\ &= \frac{2}{N} \frac{\exp\left[\left(\frac{1}{2} - x_0\right) E\right]}{\exp(E) - 1} \sinh\left(\frac{1}{2}NE\right). \end{aligned}$$

Taking into account that $\sinh(x) = x + \frac{x^3}{3!} + \cdots = \sum_{n=0}^{\infty} \frac{x^{2n+1}}{(2n+1)!}$, we obtain

$$\begin{aligned} B &= \frac{\exp\left[\left(\frac{1}{2} - x_0\right) E\right]}{\exp(E) - 1} \cdot \frac{2}{N} \left[\frac{1}{2}NE + \frac{(NE)^3}{2^3 \cdot 3!} + \frac{(NE)^5}{2^5 \cdot 5!} + \cdots\right] \\ &= \frac{E \exp\left[\left(\frac{1}{2} - x_0\right) E\right]}{\exp(E) - 1} \left[1 + \frac{(NE)^2}{24} + \frac{(NE)^4}{1920} + \cdots\right]. \end{aligned}$$

Value

A named numeric vector with two values: the error of the real part (`"Re error"`) and the error of the imaginary part (`"Im error"`) of the estimated characteristic function. Each error is the sum of the squared differences between the estimated and the theoretical values over the grid points.

As a side effect, three plots (graphics devices) are produced when `d` is 1 or 2, see the details.

Author(s)

Original Matlab code by Omar Alzeley.

R implementation and documentation: Manos Papadakis and Michail Tsagris.

References

Alzeley O. (2020). On a Statistical Learning Algorithm of Superstatistics. Computational and Mathematical Methods, 2(3), e1085.

Beck C. and Cohen E.G.D. (2003). Superstatistics. Physica A, 322: 267–275.

See Also

`find_cf`, `createalfa`

Examples

```
## bivariate uniform data
main(N = 10, d = 2, n = 100)

## univariate Gaussian data
main(N = 10, d = 1, n = 100, dist = 0)
```

Index

createalfa, 3, *5*, *9*

find_cf, 4, *4*, *6*, *9*

main, *4*, *5*, 6

superstats-package, 2