# Package ‘distrEllipse’

April 4, 2019

**Version** 2.8.0

**Date** 2019-04-02

**Title** S4 Classes for Elliptically Contoured Distributions

**Description** Distribution (S4-)classes for elliptically contoured distributions (based on package 'distr').

**Depends** R(>= 3.4), methods, graphics, mvtnorm, setRNG(>= 2006.2-1), distr(>= 2.8.0), distrEx(>= 2.8.0), distrSim(>= 2.2)

**Suggests** distrMod(>= 2.8.0), distrTEst(>= 2.2)

**Imports** startupmsg, stats

**ByteCompile** yes

**License** LGPL-3

**URL** http://distr.r-forge.r-project.org/

**Encoding** latin1

**LastChangedDate** {$LastChangedDate: 2019-04-02 21:10:55 +0200 (Di, 02 Apr 2019) $}

**LastChangedRevision** {$LastChangedRevision: 1337 $}

**VCS/SVNRevision** 1336

**NeedsCompilation** no

**Author** Peter Ruckdeschel [aut, cre, cph]

**Maintainer** Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

**Repository** CRAN

**Date/Publication** 2019-04-04 16:40:02 UTC

## R topics documented:

---

`distrEllipse-package` *distrEllipse – S4 Classes for Elliptically Contoured Distributions*

---

## Description

**distrEllipse** provides infrastructure / (S4-)classes for elliptically contoured distributions (based on package distr).

## Details

| | |
|---|---|
| Package: | distrEllipse |
| Version: | 2.8.0 |
| Date: | 2019-04-02 |
| Depends: | R(>= 3.4), methods, graphics, mvtnorm, setRNG(>= 2006.2-1), distr(>= 2.8.0),distrEx(>= 2.8.0), distrS |
| Suggests: | distrMod(>= 2.8.0), distrTEst(>= 2.2) |
| Imports: | startupmsg, stats |
| ByteCompile: | yes |
| License: | LGPL-3 |
| URL: | http://distr.r-forge.r-project.org/ |
| VCS/SVNRevision: | 1336 |

## Classes

```
####################################
Distribution Classes
```

```
####################################
[*]: there is a generating function with the same name
"Distribution" (from distr)
|>"MultivariateDistribution" (from distrEx)
|>|>"MultivarMixingDistribution" [*]
|>|>"SphericalDistribution" [*]
|>|>|>|>"EllipticalDistribution" [*]
|>|>|>|>"MVNormDistribution" [*]
"DistrList" (from distr)
|>"MultivarDistrList" [*/class union of "MVDistrList", "UnivarDistrList"]
|>|>"MVDistrList"
|>"UnivarDistrList" (from distr) [*]
```

## Methods

```
plot-methods            Methods for Function plot
                       (for SphericalDistribution)
show-methods            Methods for Function show
                       (for Simulation/Contsimulation)
```

## Functions

```
distrEllipseoptions            Functions to change the global variables of the
                        package 'distrEllipse'
```

## Slot accessors / -replacement functions

All slots are inspected / modified by corresponding accessors / -replacement functions.

## Start-up-Banner

You may suppress the start-up banner/message completely by setting `options("StartupBanner"="off")` somewhere before loading this package by `library` or `require` in your R-code / R-session. If option `"StartupBanner"` is not defined (default) or setting `options("StartupBanner"=NULL)` or `options("StartupBanner"="complete")` the complete start-up banner is displayed. For any other value of option `"StartupBanner"` (i.e., not in `c(NULL,"off","complete")`) only the version information is displayed. The same can be achieved by wrapping the `library` or `require` call into either `suppressStartupMessages()` or `onlytypeStartupMessages(.,atypes="version")`.

## Package versions

Note: The first two numbers of package versions do not necessarily reflect package-individual development, but rather are chosen for the distrXXX family as a whole in order to ease updating "depends" information.

## Start-up-Banner

You may suppress the start-up banner/message completely by setting `options("StartupBanner"="off")` somewhere before loading this package by `library` or `require` in your R-code / R-session. If option `"StartupBanner"` is not defined (default) or setting `options("StartupBanner"=NULL)`

or `options("StartupBanner"="complete")` the complete start-up banner is displayed. For any other value of option `"StartupBanner"` (i.e., not in `c(NULL,"off","complete")`) only the version information is displayed. As for general `packageStartupMessage`'s, you may also suppress all the start-up banner by wrapping the `library` or `require` call into `suppressPackageStartupMessages()` from **startupmsg**-version 0.5 on.

## Note

Global options controlling the plots and summaries of Dataclass and Simulation/Contsimulation objects may be inspected / set by `distrEllipseoptions()` and `getdistrEllipseOption()`.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>,
*Maintainer:* Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## References

P. Ruckdeschel, M. Kohl, T. Stabla, F. Camphausen (2006): S4 Classes for Distributions, *R News*, *6*(2), 2-6. https://CRAN.R-project.org/doc/Rnews/Rnews_2006-2.pdf A vignette for packages **distr**, **distrSim**, **distrTEst**, **distrEx**, **distrTeach**, **distrMod**, and **distrEllipse** is included into the mere documentation package **distrDoc** and may be called by `require("distrDoc");vignette("distr")`.

A homepage to this package is available under
http://distr.r-forge.r-project.org/.

---

`distrEllipse-defunct` *Defunct Functions in Package* **distrEllipse**

---

## Description

Functions which are no longer provided in **distrEllipse** due to clashes with S3-method inheritance.

## Methods

From version 2.7 on, former versions of S4-methods `rRd`, `dRd`, `pRd`, `qRd`, and `plotRd` of style <name>.rd are defunct due to clashes with S3-method inheritance. More specifically, this concerns the following methods:

**r.rd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `r` of slot `radDistr`.

**d.rd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `d` of slot `radDistr`.

**p.rd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `p` of slot `radDistr`.

**q.rd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `q` of slot `radDistr`.

**plot.rd** `signature(x = "SphericalDistribution")`: utility; calls `plot` for slot `radDistr`.

### See Also

Defunct

---

`distrEllipseMASK` *Masking of/by other functions in package "distrEllipse"*

---

### Description

Provides information on the (intended) masking of and (non-intended) masking by other other functions in package **distrEllipse**

### Usage

```
distrEllipseMASK(library = NULL)
```

### Arguments

| | |
|---|---|
| `library` | a character vector with path names of R libraries, or `NULL`. The default value of `NULL` corresponds to all libraries currently known. If the default is used, the loaded packages are searched before the libraries |

### Value

no value is returned

### Author(s)

Peter Ruckdeschel `<peter.ruckdeschel@uni-oldenburg.de>`

### Examples

```
distrEllipseMASK()
```

---

`distrEllipseoptions` *functions to change the global variables of the package 'distrEllipse'*

---

### Description

With `distrEllipseoptions` and `getdistrEllipseOption` you may inspect and change the global variables used by package **distrEllipse**.

### Usage

```
distrEllipseoptions(...)
getdistrEllipseOption(x)
```

### Arguments

| | |
|---|---|
| `...` | any options can be defined, using name = value or by passing a list of such tagged values. |
| `x` | a character string holding an option name. |

### Details

Invoking `distrEllipseoptions()` with no arguments returns a list with the current values of the options. To access the value of a single option, one should use `getdistrEllipseOption("WarningSim")`, e.g., rather than `distrEllipseoptions("WarningSim")` which is a *list* of length one.

### Value

`distrEllipseoptions()` returns a list of the global options of **distrEllipse**.
`distrEllipseoptions("Nsim")` returns the global option `Nsim` as a list of length 1.
`distrEllipseoptions("Nsim" = 3000)` sets the value of the global option `Nsim` to 3000. `getdistrEllipseOption("Nsim`
the current value set for option `Nsim`.

### Currently available options

`Nsim` for plotting: number of (simulated) points to be plotted.

`withED` for plotting: logical; shall principal axes of the contour ellipsoid be plot in (for each panel)?

`lwd.Ed` for plotting: line width of principal axes (for each panel).

`col.Ed` for plotting: color of principal axes (for each panel).

`withMean` for plotting: logical; shall mean be plot in (for each panel)?

`cex.mean` for plotting: size of the mean symbol (for each panel).

`pch.mean` for plotting: mean symbol (for each panel).

`col.mean` for plotting: color of the mean symbol (for each panel).

### Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

### See Also

`options`, `getOption`

### Examples

```
distrEllipseoptions("Nsim") # returns the value of Nsim, by default = 5
currentDistrOptions <- distrEllipseoptions()
distrEllipseoptions(Nsim = 6000)
distrEllipseoptions("Nsim")
getdistrEllipseOption("Nsim")
distrEllipseoptions(c("Nsim","withED"))
```

`EllipticalDistribution`

*Generating function for EllipticalDistribution-class*

### Description

Generates an object of class "`EllipticalDistribution`".

### Usage

```
EllipticalDistribution(radDistr = sqrt(Chisq(df = length(loc))),
                       loc = c(0,0), scale = diag(length(loc)), p = NULL, q = NULL)
```

### Arguments

| | |
|---|---|
| `radDistr` | an object of class `UnivariateDistribution` with positive support, i.e. `p(radDistr)(0)==0`; the radial distribution. |
| `loc` | real number: location / center of the elliptical distribution. |
| `scale` | a square matrix (with `nrow(scale)==ncol(scale)==length(loc)`) of full rank: the / a scale matrix of the elliptical distribution — unique only upto `scale%*%t(scale)`, i.e. if `A1` and `A2` are two square matrices of full rank such that `A1%*%t(A1)==A2%*%t(A2)`, then we obtain the same elliptical distribution for `scale = A1` and for `scale = A2`. |
| `p` | optional: p-slot of the corresponding distribution; |
| `q` | optional: q-slot of the corresponding distribution; |

### Value

Object of class "`EllipticalDistribution`"

### Author(s)

Peter Ruckdeschel <`peter.ruckdeschel@uni-oldenburg.de`>

### See Also

`EllipticalDistribution-class`

### Examples

```
E0 <- EllipticalDistribution()
plot(E0)
E1 <- diag(1,2)%*%E0+c(1,2)
plot(E1)
E(E1)
var(E1)
```

EllipticalDistribution-class
*Elliptical distribution class*

## Description

Class `EllipticalDistribution` implements general elliptically symmetric distributions, i.e. starting from a spherically distribution realized as an object `S` of class `SphericalDistribution`, this is the distribution of an affine linear transformation `AS+b`.

## Objects from the Class

Objects could in principle be created by calls to `new`, but more frequently you would create them via the generating function `EllipticalDistribution`.

## Slots

`img` Object of class `"Reals"`.

`param` Object of class `"EllipticalParameter"`.

`r` function with argument `n`; random number generator

`d` optional function; in case it exists: the density of the distribution

`p` optional function; in case it is non-null: the cdf of the distribution evaluated on rectangles, i.e. if a random variable `X` is distributed according to an `object` of class `"EllipticalDistribution"`, for `q` a matrix of dimension $d \times n$ `p(object)(q)` returns, for each of the `n` columns $P(X_i \leq q_i,\; i = 1, \ldots, d)$.

`q` optional function; in case it is non-null: the quantile of the distribution evaluated on rectangles, i.e. if a random variable `X` is distributed according to an `object` of class `"EllipticalDistribution"`, for `p` a vector of length $n$, returns, for each of the `n` components the infinimal number $q_j$ such that $P(X_i \leq q_j,\; i = 1, \ldots, d) \geq p_j$.

`radDistr` an object of class `UnivariateDistribution` with positive support, i.e. `p(radDistr)(0)==0`; the radial distribution.

`.withArith` logical: used internally to issue warnings as to interpretation of arithmetics

`.withSim` logical: used internally to issue warnings as to accuracy

`.logExact` logical: used internally to flag the case where there are explicit formulae for the log version of density, cdf, and quantile function

`.lowerExact` logical: used internally to flag the case where there are explicit formulae for the lower tail version of cdf and quantile function

`Symmetry` object of class `"EllipticalSymmetry"` about center `loc`; used internally to avoid unnecessary calculations.

## Extends

Class `"SphericalDistribution"`, directly.
Class `"MultivariateDistribution"`, by class `"SphericalDistribution"`. Class `"Distribution"`, by class `"MultivariateDistribution"`.

## Methods

**location** `signature(object = "EllipticalDistribution")`: wrapped access method for slot `location` of slot `param`.

**scale** `signature(x = "EllipticalDistribution")`: wrapped access method for slot `scale` of slot `param`.

**location<-** `signature(object = "EllipticalDistribution")`: wrapped replace method for slot `location` of slot `param`.

**scale<-** `signature(x = "EllipticalDistribution")`: wrapped replace method for slot `scale` of slot `param`.

**E** `signature(object = "EllipticalDistribution", fun = "missing", cond = "missing")`: expectation of an elliptically symmetric distribution; exact.

**E** `signature(object = "EllipticalDistribution", fun = "function", cond = "missing")`: expectation of an elliptically symmetric distribution; by simulation.

**var** `signature(x = "EllipticalDistribution")`: expectation of an elliptically symmetric distribution; exact.

`+` `signature(e1 = "EllipticalDistribution", e2 = "numeric")`: affine linear transformation; exact.

`-` `signature(e1 = "EllipticalDistribution", e2 = "numeric")`: affine linear transformation; exact.

`*` `signature(e1 = "EllipticalDistribution", e2 = "numeric")`: affine linear transformation; exact.

`%*%` `signature(e1 = "numeric", e2 = "EllipticalDistribution")`: affine linear transformation; exact.

**coerce** `signature(from = "EllipticalDistribution", to = "UnivariateDistribution")`: create a `UnivariateDistribution` object from a (one-dimensional) elliptically symmetric distribution.

**coerce** `signature(from = "UnivariateDistribution", to = "EllipticalDistribution")`: create a `EllipticalDistribution` object from a (symmetric) univariate distribution.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## Examples

```
new("EllipticalDistribution") ## better use EllipticalDistribution()
```

EllipticalParameter-class

*Paramter of an Elliptical distributions*

## Description

The class of the parameter of Elliptical distributions.

## Objects from the Class

Objects can be created by calls of the form `new("EllipticalParameter", ...)`.

## Slots

`loc` numeric; center / location of the distribution.

`scale` matrix; the scale matrix; the number of rows of this matrix must be the same as the length of `location`.

`name` default name is "parameter of a Elliptical distribution".

## Extends

Class `"Parameter"`, directly.
Class `"OptionalParameter"`, by class `"Parameter"`.

## Methods

**location** `signature(object = "EllipticalParameter")`: access method for slot `location`.

**scale** `signature(x = "EllipticalParameter")`: access method for slot `scale`.

**location<-** `signature(object = "EllipticalParameter")`: replace method for slot `location`.

**scale<-** `signature(object = "EllipticalParameter")`: replace method for slot `scale`.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## See Also

EllipticalDistribution-class, Parameter-class

## Examples

```
new("EllipticalParameter")
```

---

`MultivarDistrList` *Generating function for MultivarDistrList-class*

---

## Description

Generates an object of class "`MultivarDistrList`".

## Usage

```
MultivarDistrList(..., Dlist)
```

## Arguments

| | |
|---|---|
| `...` | Objects of class "`MultivariateDistribution`" (or subclasses) |
| `Dlist` | an optional list or object of class "`MultivarDistrList`"; if not missing it is appended to argument `...`; this way `MultivarMixingDistribution` may also be called with a list (or "`MultivarDistrList`"-object) as argument as suggested in an e-mail by Krunoslav Sever (thank you!) |

## Value

Object of class "`MVDistrList`" or of class "`UnivarDistrList`", hence of class union "`MultivarDistrList`"

## Author(s)

Peter Ruckdeschel <`peter.ruckdeschel@uni-oldenburg.de`>

## See Also

`DistrList-class`, `MultivarDistrList-class`, `MultivarDistrList`

## Examples

```
(DL1 <- MultivarDistrList(Norm(), Exp(), Pois()))
(DL2 <- MultivarDistrList(MVNorm(),
                          EllipticalDistribution(radDistr=Exp(), loc=c(1,2),
                          scale=diag(c(3,1))),MVt()))
```

MultivarDistrList-class *List of multivariate distributions*

## Description

Create a list of multivariate distributions

## Objects from the Class

Objects can be created by calls of the form `new("MVDistrList", ...)`. More frequently they are created via the generating function `MultivarDistrList`.

## Slots

`.Data`**:** Object of class `"list"`. A list of multivariate distributions of the same dimension.

## Extends

Class `"DistrList"`, directly.
Class `"list"`, by class `"DistrList"`.
Class `"vector"`, by class `"DistrList"`.

## Methods

**coerce** `signature(from = "MultivariateDistribution", to = "MultivarDistrList")`: create a `MultivarDistrList` object from a univariate distribution

**dimension** dim of the range space.

**dim** synonym to dimension.

## Details

In fact, class `"MultivarDistrList"` is an inbetween class between class `"DistrList"` and class `"UnivarDistrList"`, which is a case for `setIs`, but we would have to modify the metadata information in package **distr** to realize this. So we introduce a new (sister) class `"MVDistrList"` which implements strictly lists of multivariate distributions, and which together with `"UnivarDistrList"` is a subclass of the common class union class `"MultivarDistrList"`.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## See Also

`MultivarDistrList`, `DistrList-class`, `MultivariateDistribution-class`

## Examples

```
(DL1 <- MultivarDistrList(Norm(), Exp(), Pois()))
(DL2 <- MultivarDistrList(MVNorm(),
                          EllipticalDistribution(radDistr=Exp(), loc=c(1,2),
                          scale=diag(c(3,1))),MVt()))
```

---

MultivarMixingDistribution
*Generating function for Class "MultivarMixingDistribution"*

---

## Description

Generates an object of class "MultivarMixingDistribution".

## Usage

```
MultivarMixingDistribution(..., Dlist, mixCoeff

                    )
```

## Arguments

| | |
|---|---|
| ... | Objects of class "MultivariateDistribution" (or subclasses) |
| Dlist | an optional list or object of class "MultivarDistrList"; if not missing it is appended to argument ...; this way MultivarMixingDistribution may also be called with a list (or "MultivarDistrList"-object) as argument as suggested in an e-mail by Krunoslav Sever (thank you!) |
| mixCoeff | Objects of class "numeric" : a vector of probabilities for the mixing components (must be of same length as arguments in ...). |

## Details

If mixCoeff is missing, all elements in ... are equally weighted.

## Value

Object of class "MultivarMixingDistribution", or if argument withSimplify is TRUE and the resulting object would have one mixing component with probability (almost) 1, MultivarMixingDistribution will return this component.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## See Also

MultivarMixingDistribution-class

## Examples

```
mylist <- MultivarMixingDistribution(Binom(3,.3), Dirac(2), Norm(),
          mixCoeff=c(1/4,1/5,11/20))
```

---

MultivarMixingDistribution-class
*Class "MultivarMixingDistribution"*

---

## Description

`MultivarMixingDistribution`-class is a class to formalize multivariate mixing distributions; it is a subclass to class `MultivariateDistribution`.

## Objects from the Class

Objects can be created by calls of the form `new("MultivarMixingDistribution", ...)`. More frequently they are created via the generating function `MultivarMixingDistribution`.

## Slots

- `mixCoeff` Object of class `"numeric"`: a vector of probabilities for the mixing components.
- `mixDistr` Object of class `"MultivarDistrList"`: a list of multivariate distributions containing the mixing components; must be of same length as `mixCoeff`.
- `img` Object of class `"Reals"`: the space of the image of this distribution which has dimension 1 and the name "Real Space"
- `param` Object of class `"Parameter"`: the parameter of this distribution, having only the slot name "Parameter of a discrete distribution"
- `r` Object of class `"function"`: generates random numbers
- `d` fixed to `NULL`
- `p` Object of class `"OptionalFunction"`: if non-null cumulative distribution function
- `q` Object of class `"OptionalFunction"`: if non-null quantile function
- `.withArith` logical: used internally to issue warnings as to interpretation of arithmetics
- `.withSim` logical: used internally to issue warnings as to accuracy
- `.logExact` logical: used internally to flag the case where there are explicit formulae for the log version of density, cdf, and quantile function
- `.lowerExact` logical: used internally to flag the case where there are explicit formulae for the lower tail version of cdf and quantile function
- `Symmetry` object of class `"DistributionSymmetry"`; used internally to avoid unnecessary calculations.

## Extends

Class `"MultivariateDistribution"` class `"Distribution"` by class `"MultivariateDistribution"`.

## Methods

**show** `signature(object = "MultivarMixingDistribution")` prints the object

**mixCoeff<-** `signature(object = "MultivarMixingDistribution")` replaces the corresponding slot

**mixCoeff** `signature(object = "MultivarMixingDistribution")` returns the corresponding slot

**mixDistr<-** `signature(object = "MultivarMixingDistribution")` replaces the corresponding slot

**mixDistr** `signature(object = "MultivarMixingDistribution")` returns the corresponding slot

**support** `signature(object = "MultivarMixingDistribution")` returns the corresponding slot

**gaps** `signature(object = "MultivarMixingDistribution")` returns the corresponding slot

**.logExact** `signature(object = "Distribution")`: returns slot `.logExact` if existing; else tries to convert the object to a newer version of its class by `conv2NewVersion` and returns the corresponding slot of the converted object.

**.lowerExact** `signature(object = "Distribution")`: returns slot `.lowerExact` if existing; else tries to convert the object to a newer version of its class by `conv2NewVersion` and returns the corresponding slot of the converted object.

**Symmetry** returns slot `Symmetry` if existing; else tries to convert the object to a newer version of its class by `conv2NewVersion` and returns the corresponding slot of the converted object.

**plot** `signature(x = "MultivarMixingDistribution", y = "missing")`: plot for an spherically symmetric distribution; see `plot-methods`.

**E** corresponding expectation — see `E`.

**dimension** dim of the range space.

**dim** synonym to dimension.

**show** `signature(object = "MultivarMixingDistribution")`: show method for spherically symmetric distributions.

**showobj** `signature(object = "MultivarMixingDistribution")`: showobj method for spherically symmetric distributions.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## See Also

`Parameter-class`, `MultivariateDistribution-class`, `LatticeDistribution-class`, `AbscontDistribution-class`, `simplifyD-methods`, `flat.mix`

## Examples

```
mylist <- MultivarMixingDistribution(Binom(3,.3), Dirac(2), Norm(),
          mixCoeff=c(1/4,1/5,11/20))
mylist2 <- MultivarMixingDistribution(Binom(3,.3), mylist,
          mixCoeff=c(.3,.7))
mylist2
p(mylist)(0.3)
mixDistr(mylist2)
E(mylist)
var(mylist)

##multivariate
E1 <- diag(1,2)%*%EllipticalDistribution(radDistr=Gammad())+c(1,2)
mylistD <- MultivarMixingDistribution(MVNorm(), E1, MVt(),
          mixCoeff=c(1/4,1/5,11/20))
mylistD2 <- MultivarMixingDistribution(E1+c(-2,2), mylistD,
          mixCoeff=c(.3,.7))
mylistD2
p(mylistD)
mixDistr(mylistD2)
E(mylistD2)
var(mylistD2)
```

---

`MVNormDistribution` *Generating function for MVNormDistribution-class*

---

## Description

Generates an object of class "MVNormDistribution".

## Usage

```
MVNorm(loc=c(0,0), scale = diag(length(loc)))
```

## Arguments

| | |
|---|---|
| loc | real number: location / center of the elliptical distribution. |
| scale | a square matrix (with nrow(scale)==ncol(scale)==length(loc)) of full rank: the / a scale matrix of the elliptical distribution — unique only upto scale , i.e. if A1 and A2 are two square matrices of full rank such that A1%*%t(A1)==A2%*%t(A2) , then we obtain the same elliptical distribution for scale = A1 and for scale = A2. |

## Value

Object of class "MVNormDistribution"

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## See Also

MVNormDistribution-class

## Examples

```
E0 <- MVNorm()
plot(E0)
E1 <- diag(1,2)%*%E0+c(1,2)
plot(E1)
E(E1)
var(E1)
```

---

MVNormDistribution-class

*MVNorm distribution class*

---

## Description

Class MVNormDistribution implements a general multivariate distribution using code from package **mvtnorm**. For details to this implementation confer to the references given in this package.

## Objects from the Class

Objects could in principle be created by calls to new, but more frequently you would create them via the generating function MVNormDistribution.

## Slots

**img:** Object of class "Reals".

**param:** Object of class "MVtParameter".

**r:** function with argument n; random number generator

**d:** the density of this distribution, pmvnorm

**p:** the (vectorized) function pmvnorm.

**q:** the (vectorized) function qmvnorm.

**radDistr:** the distribution sqrt(Chisq(df=dim0))

**.withArith:** FALSE

**.withSim:** FALSE

**.logExact:** TRUE

**.lowerExact:** TRUE

**Symmetry:** object of class "EllipticalSymmetry" about center loc; used internally to avoid unnecessary calculations.

### Extends

Class "EllipticalDistribution", directly.
Class "SphericalDistribution", by class "EllipticalDistribution".
Class "MultivariateDistribution", by class "SphericalDistribution". Class "Distribution", by class "MultivariateDistribution".

### Methods

**sigma** signature(object = "MVNormDistribution"): wrapped access method for slot sigma of slot param.

**mean** signature(object = "MVNormDistribution"): wrapped access method for slot location of slot param.

### Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

### See Also

Package **mvtnorm**

### Examples

```
new("MVNormDistribution") ## better use generating function MVNormDistribution()
```

---

## MVNormParameter-class *Paramter of a multivariate normal distribution*

---

### Description

The class of the parameter of MVNorm distributions.

### Objects from the Class

Objects can be created by calls of the form new("MVNormParameter", ...).

### Slots

loc**:** numeric; center / location of the distribution.

scale**:** matrix; the scale matrix; the number of rows of this matrix must be the same as the length of location.

name**:** default name is “parameter of a Elliptical distribution”.

### Extends

Class "EllipticalParameter", directly.
Class "Parameter", by class "EllipticalParameter".
Class "OptionalParameter", by class "Parameter".

## Methods

**mean** `signature(object = "MVNormParameter")`: access method for slot `location`.

**sigma** `signature(x = "MVNormParameter")`: utility function; returns `S%*%t(S)` for `S=scale(x)`.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## See Also

MVNormDistribution-class, Parameter-class

## Examples

```
new("MVNormParameter")
```

---

MVtDistribution *Generating function for MvtDistribution-class*

---

## Description

Generates an object of class "MvtDistribution".

## Usage

```
MVt(loc = c(0,0), scale = diag(length(loc)), df = 1, ncp = 0)
```

## Arguments

| | |
|---|---|
| loc | real number: location / center of the elliptical distribution. |
| scale | a square matrix (with `nrow(scale)==ncol(scale)==length(loc)`) of full rank: the / a scale matrix of the elliptical distribution — unique only upto `scale%*%t(scale)` , i.e. if `A1` and `A2` are two square matrices of full rank such that `A1%*%t(A1)==A2%*%t(A2)` , then we obtain the same elliptical distribution for `scale = A1` and for `scale = A2`. |
| df | integer; degrees of freedom |
| ncp | positive real number; non-centrality parameter |

## Value

Object of class "MvtDistribution"

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

### See Also

MVtDistribution-class

### Examples

```
E0 <- MVt()
plot(E0)
E1 <- diag(1,2)%*%E0+c(1,2)
plot(E1)
E(E1)
var(E1)
```

---

## MVtDistribution-class *MVt distribution class*

---

### Description

Class MVtDistribution implements multivariate t distributions using code from package **mvtnorm**. For details to this implementation confer to the references given in this package.

### Objects from the Class

Objects could in principle be created by calls to new, but more frequently you would create them via the generating function MVtDistribution.

### Slots

img: Object of class "Reals".

param: Object of class "MVtParameter".

r: function with argument n; random number generator

d: the density of this distribution, dmvt

p: the (vectorized) function pmvt.

q: the (vectorized) function qmvt.

radDistr: an object of class AbscontDistribution with density

$$\mathrm{dim}\binom{(\mathrm{dim}+\mathrm{df}-1)/2}{\mathrm{df}/2-1}x^{\mathrm{dim}-1}\mathrm{df}^{-\mathrm{dim}/2}/(1+x^2/\mathrm{df})^{(\mathrm{dim}+\mathrm{df})/2}$$

.withArith: FALSE

.withSim: FALSE

.logExact: TRUE

.lowerExact: TRUE

Symmetry: object of class "EllipticalSymmetry" about center loc; used internally to avoid unnecessary calculations.

## Extends

Class "EllipticalDistribution", directly.
Class "SphericalDistribution", by class "EllipticalDistribution".
Class "MultivariateDistribution", by class "SphericalDistribution". Class "Distribution", by class "MultivariateDistribution".

## Methods

**sigma** `signature(object = "MVtDistribution")`: wrapped access method for slot `sigma` of slot `param`.

**ncp** `signature(object = "MVtDistribution")`: wrapped access method for slot `ncp` of slot `param`.

**df** `signature(x = "MVtDistribution")`: wrapped access method for slot `scale` of slot `param`.

## Author(s)

Peter Ruckdeschel `<peter.ruckdeschel@uni-oldenburg.de>`

## See Also

Package **mvtnorm**

## Examples

```
new("MVtDistribution") ## better use generating function MVtDistribution()
```

---

`MVtParameter-class` *Paramter of a multivariate t distribution*

---

## Description

The class of the parameter of MVt distributions.

## Objects from the Class

Objects can be created by calls of the form `new("MVtParameter", ...)`.

## Slots

`loc`**:** numeric; center / location of the distribution.

`scale`**:** matrix; the scale matrix; the number of rows of this matrix must be the same as the length of `location`.

`df`**:** integer; the degrees of freedom.

`ncp`**:** positive real; the non-centrality parameter.

`name`**:** default name is “parameter of a Elliptical distribution”.

## Extends

Class "Parameter", directly.
Class "OptionalParameter", by class "Parameter".

## Methods

**mean** signature(object = "MVnormParameter"): access method for slot location.

**sigma** signature(x = "MVnormParameter"): utility function; returns S%*%t(S) for S=scale(x).

**ncp** signature(object = "MVnormParameter"): access method for slot ncp.

**df** signature(x = "MVnormParameter"): access method for slot df.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## See Also

MVtDistribution-class, Parameter-class

## Examples

```
new("MVtParameter")
```

---

plot-methods *Methods for Function plot in Package 'distrEllipse'*

---

## Description

plot-methods

## Usage

```
plot(x, y, ...)
## S4 method for signature 'SphericalDistribution,missing'
plot(x, Nsim = getdistrEllipseOption("Nsim"), ...,
               withED = getdistrEllipseOption("withED"),
               lwd.Ed = getdistrEllipseOption("lwd.Ed"),
               col.Ed = getdistrEllipseOption("col.Ed"),
               withMean = getdistrEllipseOption("withMean"),
               cex.mean = getdistrEllipseOption("cex.mean"),
               pch.mean = getdistrEllipseOption("pch.mean"),
               col.mean = getdistrEllipseOption("col.mean"))
## S4 method for signature 'MultivarMixingDistribution,missing'
plot(x, Nsim = getdistrEllipseOption("Nsim"), ...,
               withED = getdistrEllipseOption("withED"),
```

```
lwd.Ed = getdistrEllipseOption("lwd.Ed"),
col.Ed = getdistrEllipseOption("col.Ed"),
withMean = getdistrEllipseOption("withMean"),
cex.mean = getdistrEllipseOption("cex.mean"),
pch.mean = getdistrEllipseOption("pch.mean"),
col.mean = getdistrEllipseOption("col.mean"))
```

## Arguments

| | |
|---|---|
| x | object of class "SphericalDistribution" distribution to be plotted |
| y | missing |
| Nsim | number of (simulated) points to be plotted. |
| withED | logical; shall principal axes of the contour ellipsoid be plot in (for each panel)? |
| lwd.Ed | line width of principal axes (for each panel). |
| col.Ed | color of principal axes (for each panel). |
| withMean | logical; shall mean be plot in (for each panel)? |
| cex.mean | size of the mean symbol (for each panel). |
| pch.mean | mean symbol (for each panel). |
| col.mean | color of the mean symbol (for each panel). |
| ... | addtional arguments for plot — see plot, plot.default, plot.stepfun |

## Details

Using pairs, plots all pairs of coordinates of the object, using simulated values. Any parameters of pairs may be passed on to this particular plot method.

## See Also

pairs, plot plot.default, plot.stepfun, par

## Examples

```
E0 <- matrix(c(2,1,1,4),2,2)%*%EllipticalDistribution()+c(2,1)
E1 <- matrix(c(3,2,2,4),2,2)%*%EllipticalDistribution(radDistr = exp(Binom(10,.8)))
plot(E0)
plot(E1, withED=FALSE, Nsim=5000)
mylist <- MultivarMixingDistribution(E0,E1, mixCoeff=c(1/4,3/4))
plot(mylist)
```

## SphericalDistribution *Generating function for SphericalDistribution-class*

### Description

Generates an object of class "SphericalDistribution".

### Usage

```
SphericalDistribution(radDistr = sqrt(Chisq(df=dim)), dim = 2,
                                p = NULL, q = NULL)
```

### Arguments

| | |
|---|---|
| radDistr | an object of class UnivariateDistribution with positive support, i.e. p(radDistr)(0)==0; the radial distribution. |
| dim | positive integer: dimension of the distribution. |
| p | optional: p-slot of the corresponding distribution; |
| q | optional: q-slot of the corresponding distribution; |

### Value

Object of class "SphericalDistribution"

### Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

### See Also

SphericalDistribution-class

### Examples

```
E0 <- SphericalDistribution()
plot(E0)
E1 <- diag(1,2)%*%E0+c(1,2)
plot(E1)
E(E1)
var(E1)
```

SphericalDistribution-class
*Spherical distribution class*

## Description

Class SphericalDistribution implements general spherically symmetric distributions, i.e. starting from a random variable L distributed according to a univariate distribution radDistr with positive support serving as radial distribution, and an independent random variable U distributed uniformly on the dim dimensional sphere, this is the distribution of LU.

## Objects from the Class

Objects could in principle be created by calls to new, but more frequently you would create them via the generating function SphericalDistribution.

## Slots

img Object of class "Reals".

param Object of class "SphericalParameter".

r function with argument n; random number generator

d optional function; in case it exists: the density of the distribution

p optional function; in case it is non-null: the cdf of the distribution evaluated on rectangles, i.e. if a random variable X is distributed according to an object of class "SphericalDistribution", for q a matrix of dimension $d \times n$ p(object)(q) returns, for each of the n columns $P(X_i \leq q_i,\; i = 1, \ldots, d)$.

q optional function; in case it is non-null: the quantile of the distribution evaluated on rectangles, i.e. if a random variable X is distributed according to an object of class "SphericalDistribution", for p a vector of length $n$, returns, for each of the n components the infinimal number $q_j$ such that $P(X_i \leq q_j,\; i = 1, \ldots, d) \geq p_j$.

radDistr an object of class UnivariateDistribution with positive support, i.e. p(radDistr)(0)==0; the radial distribution.

.withArith logical: used internally to issue warnings as to interpretation of arithmetics

.withSim logical: used internally to issue warnings as to accuracy

.logExact logical: used internally to flag the case where there are explicit formulae for the log version of density, cdf, and quantile function

.lowerExact logical: used internally to flag the case where there are explicit formulae for the lower tail version of cdf and quantile function

Symmetry object of class "SphericalSymmetry" about center loc; used internally to avoid unnecessary calculations.

## Extends

Class "MultivariateDistribution", directly.
Class "Distribution", by class "MultivariateDistribution".

## Methods

**dimension** `signature(object = "SphericalDistribution")`: returns the dimension of the distribution.

**dim** `signature(object = "SphericalDistribution")`: synonym to `dimension`.

**location** `signature(object = "SphericalDistribution")`: helper function to have the same interface as class `"EllipticalDistribution"`; always returns `0` (in the respective dimension).

**scale** `signature(object = "SphericalDistribution")`: helper function to have the same interface as class `"EllipticalDistribution"`; always returns the unit matrix (in the respective dimension).

**radDistr** `signature(object = "SphericalDistribution")`: access method for slot `radDistr`.

**rRd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `r` of slot `radDistr`. From version 2.7 on, replaces defunct `r.Rd` to avoid clashes with S3-method inheritance.

**dRd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `d` of slot `radDistr`. From version 2.7 on, replaces defunct `d.Rd` to avoid clashes with S3-method inheritance.

**pRd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `p` of slot `radDistr`. From version 2.7 on, replaces defunct `p.Rd` to avoid clashes with S3-method inheritance.

**qRd** `signature(object = "SphericalDistribution")`: wrapped access method for slot `q` of slot `radDistr`. From version 2.7 on, replaces defunct `q.Rd` to avoid clashes with S3-method inheritance.

**plotRd** `signature(x = "SphericalDistribution")`: utility; calls `plot` for slot `radDistr`. From version 2.6 on, replaces deprecated `plot.Rd` to avoid clashes with S3-method inheritance.

**plot** `signature(x = "SphericalDistribution", y = "missing")`: plot for an spherically symmetric distribution; see `plot-methods`.

**show** `signature(object = "SphericalDistribution")`: `show` method for spherically symmetric distributions.

**showobj** `signature(object = "SphericalDistribution")`: `showobj` method for spherically symmetric distributions.

**E** `signature(object = "SphericalDistribution", fun = "missing", cond = "missing")`: expectation of an elliptically symmetric distribution; exact.

**var** `signature(x = "SphericalDistribution")`: expectation of an elliptically symmetric distribution; exact.

**coerce** `signature(from = "SphericalDistribution", to = "EllipticalDistribution")`: create a `EllipticalDistribution` object from a spherically symmetric distribution.

`+` `signature(e1 = "SphericalDistribution", e2 = "numeric")`: affine linear transformation; exact.

`-` `signature(e1 = "SphericalDistribution", e2 = "numeric")`: affine linear transformation; exact.

- `-` `signature(e1 = "SphericalDistribution", e2 = "missing")`: affine linear transformation; exact.
- `*` `signature(e1 = "SphericalDistribution", e2 = "numeric")`: affine linear transformation; exact.
- `+` `signature(e1 = "numeric", e2 = "SphericalDistribution")`: affine linear transformation; exact.
- `-` `signature(e1 = "numeric", e2 = "SphericalDistribution")`: affine linear transformation; exact.
- `*` `signature(e1 = "numeric", e2 = "SphericalDistribution")`: affine linear transformation; exact.
- `%*%` `signature(e1 = "numeric", e2 = "SphericalDistribution")`: affine linear transformation; exact.

## Author(s)

Peter Ruckdeschel <peter.ruckdeschel@uni-oldenburg.de>

## Examples

```
new("SphericalDistribution") ## better use SphericalDistribution()
```

# Index