Package 'ROptEstOld'

April 11, 2019

Version 1.2.0
Date 2019-04-02
Title Optimally Robust Estimation - Old Version
Description Optimally robust estimation using S4 classes and methods. Old version still needed for current versions of ROptRegTS and RobRex.
Depends R(>= 3.4), methods, distr(>= 2.8.0), distrEx(>= 2.8.0), RandVar(>= 1.2.0), evd
ByteCompile yes
License LGPL-3
<pre>URL http://robast.r-forge.r-project.org/</pre>
Encoding latin1
LastChangedDate {\$LastChangedDate: 2019-04-02 21:10:23 +0200 (Di, 02. Apr 2019) \$}
LastChangedRevision {\$LastChangedRevision: 1215 \$}
VCS/SVNRevision 1214
NeedsCompilation no
Author Matthias Kohl [aut, cre, cph]
Maintainer Matthias Kohl <matthias.kohl@stamats.de></matthias.kohl@stamats.de>
Repository CRAN
Date/Publication 2019-04-11 06:35:44 UTC
R topics documented:
asBias asBias-class asCov asCov-class asGRisk-class asHampel

asMSE	10
asMSE-class	11
asRisk-class	12
asUnOvShoot	13
asUnOvShoot-class	14
BinomFamily	15
checkIC	16
checkL2deriv	17
ContIC	18
ContIC-class	19
ContNeighborhood	
ContNeighborhood-class	22
evalIC	23
EvenSymmetric	24
EvenSymmetric-class	24
ExpScaleFamily	25
fiBias	
fiBias-class	
fiCov	
fiCov-class	
fiHampel	
fiHampel-class	
fimse	
fiMSE-class	
fiRisk-class	
fiUnOvShoot	
fiUnOvShoot-class	
FixRobModel	
FixRobModel-class	
FunctionSymmetry-class	
FunSymmList	
FunSymmList-class	
GammaFamily	
generateIC	
getAsRisk	
getFiRisk	
getFixClip	
getFixRobIC	46
getIneffDiff	
getInfCent	49
getInfClip	
getInfGamma	
getInfRobIC	53
getInfStand	
getRiskIC	
Gumbel	
Gumbel-class	61 64
THIMDELL OCHIONERMITY	n4

GumbelParameter-class
IC
IC-class
InfluenceCurve
InfluenceCurve-class
infoPlot
InfRobModel
InfRobModel-class
ksEstimator
L2ParamFamily
L2ParamFamily-class
leastFavorableRadius
LnormScaleFamily
locMEstimator
lowerCaseRadius
Neighborhood-class
NonSymmetric
NonSymmetric-class
NormLocationFamily
NormLocationScaleFamily
NormScaleFamily
OddSymmetric
OddSymmetric-class
oneStepEstimator
optIC
optRisk
ParamFamily
ParamFamily-class
ParamFamParameter
ParamFamParameter-class
PoisFamily
ProbFamily-class
radiusMinimaxIC
RiskType-class
RobModel-class
ROptEstOldConstants
Total VarIC
Total VarIC-class
TotalVarNeighborhood
TotalVarNeighborhood-class
trAsCov
trAsCov-class
trFiCov
trFiCov-class
UncondNeighborhood-class

4 asBias

asBias

Generating function for asBias-class

Description

Generates an object of class "asBias".

Usage

```
asBias()
```

Value

Object of class "asBias"

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asBias-class
```

```
asBias()
## The function is currently defined as
function(){ new("asBias") }
```

asBias-class 5

asBias-class

Standardized Asymptotic Bias

Description

Class of standardized asymptotic bias; i.e., the neighborhood radius is omitted respectively, set to

Objects from the Class

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

Slots

```
type: Object of class "character": "asymptotic bias".
```

Extends

```
Class "asRisk", directly.
Class "RiskType", by class "asRisk".
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asRisk-class, asBias
```

```
new("asBias")
```

6 asCov-class

asCov

Generating function for asCov-class

Description

Generates an object of class "asCov".

Usage

```
asCov()
```

Value

Object of class "asCov"

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asCov-class
```

Examples

```
asCov()
## The function is currently defined as
function(){ new("asCov") }
```

asCov-class

Asymptotic covariance

Description

Class of asymptotic covariance.

Objects from the Class

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

asGRisk-class 7

Slots

```
type: Object of class "character": "asymptotic covariance".
```

Extends

```
Class "asRisk", directly.
Class "RiskType", by class "asRisk".
```

Methods

No methods defined with class "asCov" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asRisk-class, asCov
```

Examples

```
new("asCov")
```

asGRisk-class

Convex asymptotic risk

Description

Class of special convex asymptotic risks.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
type: Object of class "character".
```

8 asHampel

Extends

```
Class "asRisk", directly.
Class "RiskType", by class "asRisk".
```

Methods

No methods defined with class "asGRisk" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Rieder, H. (2004) Optimal Influence Curves for General Loss Functions. Statistics & Decisions (submitted).

See Also

asRisk-class

asHampel

Generating function for asHampel-class

Description

Generates an object of class "asHampel".

Usage

```
asHampel(bound = Inf)
```

Arguments

bound

positive real: bias bound

Value

Object of class asHampel

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

asHampel-class 9

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asHampel-class
```

Examples

```
asHampel()
## The function is currently defined as
function(bound = Inf){ new("asHampel", bound = bound) }
```

asHampel-class

Asymptotic Hampel risk

Description

Class of asymptotic Hampel risk which is the trace of the asymptotic covariance subject to a given bias bound (bound on gross error sensitivity).

Objects from the Class

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

Slots

```
type: Object of class "character": "trace of asymptotic covariance for given bias bound". bound: Object of class "numeric": given positive bias bound.
```

Extends

```
Class "asRisk", directly.
Class "RiskType", by class "asRisk".
```

Methods

```
bound signature(object = "asHampel"): accessor function for slot bound.
show signature(object = "asHampel")
```

10 asMSE

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asRisk-class, asHampel
```

Examples

```
new("asHampel")
```

asMSE

Generating function for asMSE-class

Description

Generates an object of class "asMSE".

Usage

asMSE()

Value

Object of class "asMSE"

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asMSE-class
```

asMSE-class 11

Examples

```
asMSE()
## The function is currently defined as
function(){ new("asMSE") }
```

asMSE-class

Asymptotic mean square error

Description

Class of asymptotic mean square error.

Objects from the Class

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

Slots

type: Object of class "character": "asymptotic mean square error".

Extends

```
Class "asGRisk", directly.
Class "asRisk", by class "asGRisk".
Class "RiskType", by class "asGRisk".
```

Methods

No methods defined with class "asMSE" in the signature.

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asGRisk-class, asMSE
```

```
new("asMSE")
```

12 asRisk-class

asRisk-class

Aymptotic risk

Description

Class of asymptotic risks.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

type: Object of class "character".

Extends

Class "RiskType", directly.

Methods

No methods defined with class "asRisk" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Ruckdeschel, P. and Rieder, H. (2004) Optimal Influence Curves for General Loss Functions. Statistics & Decisions (submitted).

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

RiskType-class

asUnOvShoot 13

asUnOvShoot

Generating function for asUnOvShoot-class

Description

Generates an object of class "asUnOvShoot".

Usage

```
asUnOvShoot(width = 1.960)
```

Arguments

width

positive real: half the width of given confidence interval.

Value

Object of class "asUnOvShoot"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asUnOvShoot-class
```

```
asUnOvShoot()
## The function is currently defined as
function(width = 1.960){ new("asUnOvShoot", width = width) }
```

14 asUnOvShoot-class

asUnOvShoot-class

Asymptotic under-/overshoot probability

Description

Class of asymptotic under-/overshoot probability.

Objects from the Class

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

Slots

```
type: Object of class "character": "asymptotic under-/overshoot probability". width: Object of class "numeric": half the width of given confidence interval.
```

Extends

```
Class "asGRisk", directly.
Class "asRisk", by class "asGRisk".
Class "RiskType", by class "asGRisk".
```

Methods

```
width signature(object = "asUnOvShoot"): accessor function for slot width.
show signature(object = "asUnOvShoot")
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

```
Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.
```

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
asGRisk-class
```

```
new("asUnOvShoot")
```

BinomFamily 15

Βi	nom	Fam	i	1v
D_{\perp}	110111	ı am	_	- y

Generating function for Binomial families

Description

Generates an object of class "L2ParamFamily" which represents a Binomial family where the probability of success is the parameter of interest.

Usage

```
BinomFamily(size = 1, prob = 0.5, trafo)
```

Arguments

size number of trials
prob probability of success

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Binom-class
```

```
(B1 <- BinomFamily(size = 25, prob = 0.25))
plot(B1)
FisherInfo(B1)
checkL2deriv(B1)</pre>
```

16 checkIC

checkIC

Generic Function for Checking ICs

Description

Generic function for checking centering and Fisher consistency of ICs.

Usage

```
checkIC(IC, L2Fam, ...)
```

Arguments

IC object of class "IC"

L2Fam L2-differentiable family of probability measures.

... additional parameters

Details

The precisions of the centering and the Fisher consistency are computed.

Value

The maximum deviation from the IC properties is returned.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, IC-class
```

```
IC1 <- new("IC")
checkIC(IC1)</pre>
```

checkL2deriv 17

checkL2deriv

Generic function for checking L2-derivatives

Description

Generic function for checking the L2-derivative of an L2-differentiable family of probability measures.

Usage

```
checkL2deriv(L2Fam, ...)
```

Arguments

L2Fam L2-differentiable family of probability measures
... additional parameters

Details

The precisions of the centering and the Fisher information are computed.

Value

The maximum deviation is returned.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class
```

```
F1 <- new("L2ParamFamily")
checkL2deriv(F1)</pre>
```

18 ContIC

_	
Con	+ T C
COI	CIC

Generating function for ContIC-class

Description

Generates an object of class "ContIC"; i.e., an influence curves η of the form

$$\eta = (A\Lambda - a)\min(1, b/|A\Lambda - a|)$$

with clipping bound b, centering constant a and standardizing matrix A. Λ stands for the L2 derivative of the corresponding L2 differentiable parametric family which can be created via CallL2Fam.

Usage

Arguments

name	object of class "character".
CallL2Fam	object of class "call": creates an object of the underlying L2-differentiable parametric family.
Curve	object of class "EuclRandVarList"
Risks	object of class "list": list of risks; cf. RiskType-class.
Infos	matrix of characters with two columns named method and message: additional informations. $ \\$
clip	positive real: clipping bound.
cent	real: centering constant
stand	matrix: standardizing matrix
lowerCase	optional constant for lower case solution.
neighborRadius	radius of the corresponding (unconditional) contamination neighborhood.

Value

Object of class "ContIC"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

ContIC-class 19

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
IC-class, ContIC
```

Examples

```
IC1 <- ContIC()
plot(IC1)</pre>
```

ContIC-class

Influence curve of contamination type

Description

Class of (partial) influence curves of contamination type; i.e., influence curves η of the form

$$\eta = (A\Lambda - a)\min(1, b/|A\Lambda - a|)$$

with clipping bound b, centering constant a and standardizing matrix A. Λ stands for the L2 derivative of the corresponding L2 differentiable parametric family created via the call in the slot CallL2Fam.

Objects from the Class

Objects can be created by calls of the form new("ContIC", ...). More frequently they are created via the generating function ContIC, respectively via the method generateIC.

Slots

CallL2Fam: object of class "call": creates an object of the underlying L2-differentiable parametric family.

name: object of class "character"

Curve: object of class "EuclRandVarList"

Risks: object of class "list": list of risks; cf. RiskType-class.

Infos: object of class "matrix" with two columns named method and message: additional informations.

clip: object of class "numeric": clipping bound.

cent: object of class "numeric": centering constant.

stand: object of class "matrix": standardizing matrix.

lowerCase: object of class "OptionalNumeric": optional constant for lower case solution.

neighborRadius: object of class "numeric": radius of the corresponding (unconditional) contamination neighborhood.

20 ContIC-class

Extends

```
Class "IC", directly.
Class "InfluenceCurve", by class "IC".
```

Methods

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
IC-class, ContIC
```

```
IC1 <- new("ContIC")
plot(IC1)</pre>
```

ContNeighborhood 21

ContNeighborhood

Generating function for ContNeighborhood-class

Description

Generates an object of class "ContNeighborhood".

Usage

```
ContNeighborhood(radius = 0)
```

Arguments

radius

non-negative real: neighborhood radius.

Value

Object of class "ContNeighborhood"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

ContNeighborhood-class

```
ContNeighborhood()
## The function is currently defined as
function(radius = 0){
    new("ContNeighborhood", radius = radius)
}
```

ContNeighborhood-class

Contamination Neighborhood

Description

Class of (unconditional) contamination neighborhoods.

Objects from the Class

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

Slots

```
type: Object of class "character": "(uncond.) convex contamination neighborhood". radius: Object of class "numeric": neighborhood radius.
```

Extends

```
Class "UncondNeighborhood", directly.
Class "Neighborhood", by class "UncondNeighborhood".
```

Methods

No methods defined with class "ContNeighborhood" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

ContNeighborhood, UncondNeighborhood-class

```
new("ContNeighborhood")
```

evalIC 23

evalIC

Generic function for evaluating ICs

Description

Generic function for evaluating ICs.

Usage

```
evalIC(IC, x)
```

Arguments

IC object of class "IC"

x numeric vector or matrix

Details

The list of random variables contained in the slot Curve is evaluated at x.

Value

In case x is numeric a vector and in case x is matrix a matrix is returned.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

IC-class

24 EvenSymmetric-class

EvenSymmetric

Generating function for EvenSymmetric-class

Description

Generates an object of class "EvenSymmetric".

Usage

```
EvenSymmetric(SymmCenter = 0)
```

Arguments

SymmCenter numeric: center of symmetry

Value

Object of class "EvenSymmetric"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

EvenSymmetric-class, FunctionSymmetry-class

Examples

```
EvenSymmetric()
## The function is currently defined as
function(SymmCenter = 0){
    new("EvenSymmetric", SymmCenter = SymmCenter)
}
```

EvenSymmetric-class

Class for Even Functions

Description

Class for even functions.

Objects from the Class

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

ExpScaleFamily 25

Slots

```
type: Object of class "character": contains "even function"

SymmCenter: Object of class "numeric": center of symmetry
```

Extends

```
Class "FunctionSymmetry", directly.
Class "Symmetry", by class "FunctionSymmetry".
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

```
EvenSymmetric, FunctionSymmetry-class
```

Examples

```
new("EvenSymmetric")
```

ExpScaleFamily

Generating function for exponential scale families

Description

Generates an object of class "L2ParamFamily" which represents an exponential scale family.

Usage

```
ExpScaleFamily(rate = 1, trafo)
```

Arguments

rate rate

trafo matrix: optional transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled. The scale parameter corresponds to 1/rate.

Value

```
Object of class "L2ParamFamily"
```

26 fiBias

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Exp-class
```

Examples

```
(E1 <- ExpScaleFamily())
plot(E1)
Map(L2deriv(E1)[[1]])
checkL2deriv(E1)</pre>
```

fiBias

Generating function for fiBias-class

Description

Generates an object of class "fiBias".

Usage

```
fiBias()
```

Value

```
Object of class "fiBias"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiBias-class
```

fiBias-class 27

Examples

```
fiBias()
## The function is currently defined as
function(){ new("fiBias") }
```

fiBias-class

Finite-sample Bias

Description

Class of finite-sample bias.

Objects from the Class

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

Slots

```
type: Object of class "character": "finite-sample bias".
```

Extends

```
Class "fiRisk", directly.
Class "RiskType", by class "fiRisk".
```

Methods

No methods defined with class "fiBias" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiRisk-class, fiBias
```

```
new("fiBias")
```

28 fiCov-class

fiCov

Generating function for fiCov-class

Description

Generates an object of class "fiCov".

Usage

```
asCov()
```

Value

Object of class "fiCov"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiCov-class
```

Examples

```
fiCov()
## The function is currently defined as
function(){ new("fiCov") }
```

fiCov-class

Finite-sample covariance

Description

Class of finite-sample covariance.

Objects from the Class

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

fiHampel 29

Slots

```
type: Object of class "character": "finite-sample covariance".
```

Extends

```
Class "fiRisk", directly.
Class "RiskType", by class "fiRisk".
```

Methods

No methods defined with class "fiCov" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiRisk-class, fiCov
```

Examples

```
new("fiCov")
```

fiHampel

Generating function for fiHampel-class

Description

```
Generates an object of class "fiHampel".
```

Usage

```
fiHampel(bound = Inf)
```

Arguments

bound

positive real: bias bound

Value

Object of class fiHampel

30 fiHampel-class

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiHampel-class
```

Examples

```
fiHampel()
## The function is currently defined as
function(bound = Inf){ new("fiHampel", bound = bound) }
```

fiHampel-class

Finite-sample Hampel risk

Description

Class of finite-sample Hampel risk which is the trace of the finite-sample covariance subject to a given bias bound (bound on gross error sensitivity).

Objects from the Class

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

Slots

```
type: Object of class "character": "trace of finite-sample covariance for given bias bound". bound: Object of class "numeric": given positive bias bound.
```

Extends

```
Class "fiRisk", directly.
Class "RiskType", by class "fiRisk".
```

Methods

```
bound signature(object = "fiHampel"): accessor function for slot bound.
show signature(object = "fiHampel")
```

fiMSE 31

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiRisk-class, fiHampel
```

Examples

```
new("fiHampel")
```

fiMSE

Generating function for fiMSE-class

Description

Generates an object of class "fiMSE".

Usage

```
asMSE()
```

Value

Object of class "fiMSE"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiMSE-class
```

```
fiMSE()
## The function is currently defined as
function(){ new("fiMSE") }
```

32 fiMSE-class

fiMSE-class

Finite-sample mean square error

Description

Class of asymptotic mean square error.

Objects from the Class

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

Slots

```
type: Object of class "character": "finite-sample mean square error".
```

Extends

```
Class "fiRisk", directly.
Class "RiskType", by class "fiRisk".
```

Methods

No methods defined with class "fiMSE" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiRisk-class, fiMSE
```

```
new("fiMSE")
```

fiRisk-class 33

fiRisk-class

Finite-sample risk

Description

Class of finite-sample risks.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
type: Object of class "character".
```

Extends

Class "RiskType", directly.

Methods

No methods defined with class "fiRisk" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

RiskType-class

fiUnOvShoot

Generating function for fiUnOvShoot-class

Description

Generates an object of class "fiUnOvShoot".

Usage

```
fiUnOvShoot(width = 1.960)
```

34 fiUnOvShoot-class

Arguments

width

positive real: half the width of given confidence interval.

Value

```
Object of class "fiUnOvShoot"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278

Rieder, H. (1989) A finite-sample minimax regression estimator. Statistics 20(2): 211-221.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

fiUnOvShoot-class

Examples

```
fiUnOvShoot()
## The function is currently defined as
function(width = 1.960){ new("fiUnOvShoot", width = width) }
```

fiUnOvShoot-class

Finite-sample under-/overshoot probability

Description

Class of finite-sample under-/overshoot probability.

Objects from the Class

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

Slots

```
type: Object of class "character": "finite-sample under-/overshoot probability". width: Object of class "numeric": half the width of given confidence interval.
```

FixRobModel 35

Extends

```
Class "fiRisk", directly.
Class "RiskType", by class "fiRisk".
```

Methods

```
width signature(object = "fiUnOvShoot"): accessor function for slot width.
show signature(object = "fiUnOvShoot")
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278.

Rieder, H. (1989) A finite-sample minimax regression estimator. Statistics 20(2): 211–221.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

Ruckdeschel, P. and Kohl, M. (2005) Computation of the Finite Sample Risk of M-estimators on Neighborhoods.

See Also

```
fiRisk-class
```

Examples

```
new("fiUnOvShoot")
```

FixRobModel

Generating function for FixRobModel-class

Description

Generates an object of class "FixRobModel".

Usage

```
FixRobModel(center = ParamFamily(), neighbor = ContNeighborhood())
```

Arguments

center object of class "ProbFamily"

neighbor object of class "UncondNeighborhood"

36 FixRobModel-class

Value

```
Object of class "FixRobModel"
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
FixRobModel-class
```

Examples

```
(M1 <- FixRobModel())
## The function is currently defined as
function(center = ParamFamily(), neighbor = ContNeighborhood()){
   new("FixRobModel", center = center, neighbor = neighbor)
}</pre>
```

FixRobModel-class

Robust model with fixed (unconditional) neighborhood

Description

Class of robust models with fixed (unconditional) neighborhoods.

Objects from the Class

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

Slots

```
center: Object of class "ProbFamily".
neighbor: Object of class "UncondNeighborhood".
```

Extends

```
Class "RobModel", directly.
```

Methods

```
neighbor<- signature(object = "FixRobModel"): replacement function for slot neighbor<-
show signature(object = "FixRobModel")</pre>
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

ProbFamily-class, UncondNeighborhood-class, FixRobModel

Examples

```
new("FixRobModel")
```

FunctionSymmetry-class

Class of Symmetries for Functions

Description

Class of symmetries for functions.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
type: Object of class "character": discribes type of symmetry.
SymmCenter: Object of class "OptionalNumeric": center of symmetry.
```

Extends

```
Class "Symmetry", directly.
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

38 FunSymmList

See Also

Symmetry-class, OptionalNumeric-class

FunSymmList

Generating function for FunSymmList-class

Description

Generates an object of class "FunSymmList".

Usage

```
FunSymmList(...)
```

Arguments

... Objects of class "FunctionSymmetry" which shall form the list of symmetry types.

Value

```
Object of class "FunSymmList"
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

See Also

```
FunSymmList-class
```

FunSymmList-class 39

FunSymmList-class

List of Symmetries for a List of Functions

Description

Create a list of symmetries for a list of functions

Objects from the Class

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

Slots

```
.Data: Object of class "list". A list of objects of class "FunctionSymmetry".
```

Extends

```
Class "list", from data part.
Class "vector", by class "list".
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

```
FunctionSymmetry-class
```

Examples

GammaFamily

Generating function for Gamma families

Description

Generates an object of class "L2ParamFamily" which represents a Gamma family.

```
GammaFamily(scale = 1, shape = 1, trafo)
```

40 generateIC

Arguments

scale positive real: scale parameter shape positive real: shape parameter

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Gammad-class
```

Examples

```
distrExOptions("EupperTruncQuantile" = 1e-15) # problem with q(Gamma())(1) = NaN
(G1 <- GammaFamily())
FisherInfo(G1)
checkL2deriv(G1)
distrExOptions("EupperTruncQuantile" = 0) # default</pre>
```

generateIC

Generic function for the generation of influence curves

Description

This function is rarely called directly. It is used by other functions to create objects of class "IC".

```
generateIC(neighbor, L2Fam, ...)
```

getAsRisk 41

Arguments

neighbor Object of class "Neighborhood".

L2Fam L2-differentiable family of probability measures.

... additional parameters

Value

Object of class "IC"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

IC-class, ContIC-class, TotalVarIC-class

getAsRisk

Generic Function for Computation of Asymptotic Risks

Description

Generic function for the computation of asymptotic risks. This function is rarely called directly. It is used by other functions.

```
getAsRisk(risk, L2deriv, neighbor, ...)

## S4 method for signature 'asMSE,UnivariateDistribution,Neighborhood'
getAsRisk(risk, L2deriv, neighbor, clip, cent, stand, trafo)

## S4 method for signature 'asMSE,EuclRandVariable,Neighborhood'
getAsRisk(risk, L2deriv, neighbor, clip, cent, stand, trafo)

## S4 method for signature 'asBias,UnivariateDistribution,ContNeighborhood'
getAsRisk(risk, L2deriv, neighbor, trafo)

## S4 method for signature 'asBias,UnivariateDistribution,TotalVarNeighborhood'
getAsRisk(risk, L2deriv, neighbor, trafo)
```

42 getAsRisk

```
## S4 method for signature 'asBias, RealRandVariable, ContNeighborhood'
getAsRisk(risk, L2deriv, neighbor, Distr, L2derivDistrSymm, trafo,
             z.start, A.start, maxiter, tol)
## S4 method for signature 'asCov,UnivariateDistribution,ContNeighborhood'
getAsRisk(risk, L2deriv, neighbor, clip, cent, stand)
## S4 method for signature 'asCov,UnivariateDistribution,TotalVarNeighborhood'
getAsRisk(risk, L2deriv, neighbor, clip, cent, stand)
## S4 method for signature 'asCov,RealRandVariable,ContNeighborhood'
getAsRisk(risk, L2deriv, neighbor, Distr, clip, cent, stand)
## S4 method for signature 'trAsCov,UnivariateDistribution,UncondNeighborhood'
getAsRisk(risk, L2deriv, neighbor, clip, cent, stand)
## S4 method for signature 'trAsCov,RealRandVariable,ContNeighborhood'
getAsRisk(risk, L2deriv, neighbor, Distr, clip, cent, stand)
## S4 method for signature
## 'asUnOvShoot,UnivariateDistribution,UncondNeighborhood'
getAsRisk(risk, L2deriv, neighbor, clip, cent, stand, trafo)
```

Arguments

object of class "asRisk". risk L2-derivative of some L2-differentiable family of probability distributions. L2deriv object of class "Neighborhood". neighbor additional parameters. . . . clip optimal clipping bound. optimal centering constant. cent standardizing matrix. stand trafo matrix: transformation of the parameter. Distr object of class "Distribution". L2derivDistrSymm object of class "DistrSymmList". initial value for the centering constant. z.start A.start initial value for the standardizing matrix. maxiter the maximum number of iterations tol the desired accuracy (convergence tolerance).

Value

The asymptotic risk is computed.

getAsRisk 43

Methods

- risk = "asMSE", L2deriv = "UnivariateDistribution", neighbor = "Neighborhood": computes asymptotic mean square error in methods for function getInfRobIC.
- risk = "asMSE", L2deriv = "EuclRandVariable", neighbor = "Neighborhood": computes asymptotic mean square error in methods for function getInfRobIC.
- risk = "asBias", L2deriv = "UnivariateDistribution", neighbor = "ContNeighborhood": computes standardized asymptotic bias in methods for function getInfRobIC.
- risk = "asBias", L2deriv = "UnivariateDistribution", neighbor = "TotalVarNeighborhood": computes standardized asymptotic bias in methods for function getInfRobIC.
- risk = "asBias", L2deriv = "RealRandVariable", neighbor = "ContNeighborhood": computes standardized asymptotic bias in methods for function getInfRobIC.
- risk = "asCov", L2deriv = "UnivariateDistribution", neighbor = "ContNeighborhood": computes asymptotic covariance in methods for function getInfRobIC.
- risk = "asCov", L2deriv = "UnivariateDistribution", neighbor = "TotalVarNeighborhood": computes asymptotic covariance in methods for function getInfRobIC.
- risk = "asCov", L2deriv = "RealRandVariable", neighbor = "ContNeighborhood": computes asymptotic covariance in methods for function getInfRobIC.
- risk = "trAsCov", L2deriv = "UnivariateDistribution", neighbor = "UncondNeighborhood": computes trace of asymptotic covariance in methods for function getInfRobIC.
- risk = "trAsCov", L2deriv = "RealRandVariable", neighbor = "ContNeighborhood": computes trace of asymptotic covariance in methods for function getInfRobIC.
- risk = "asUnOvShoot", L2deriv = "UnivariateDistribution", neighbor = "UncondNeighborhood": computes asymptotic under-/overshoot risk in methods for function getInfRobIC.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Ruckdeschel, P. and Rieder, H. (2004) Optimal Influence Curves for General Loss Functions. Statistics & Decisions (submitted).

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

asRisk-class

44 getFiRisk

getFiRisk

Generic Function for Computation of Finite-Sample Risks

Description

Generic function for the computation of finite-sample risks. This function is rarely called directly. It is used by other functions.

Usage

Arguments

risk object of class "RiskType". Distr object of class "Distribution". neighbor object of class "Neighborhood". additional parameters. clip positive real: clipping bound stand standardizing constant/matrix. sampleSize integer: sample size. "A" or "B". Algo "left" or "right". cont

Details

The computation of the finite-sample under-/overshoot risk is based on FFT. For more details we refer to Section 11.3 of Kohl (2005).

Value

The finite-sample risk is computed.

Methods

```
risk = "fiUnOvShoot", Distr = "Norm", neighbor = "ContNeighborhood" computes finite-sample under-/overshoot risk in methods for function getFixRobIC.
```

risk = "fiUnOvShoot", Distr = "Norm", neighbor = "TotalVarNeighborhood" computes finite-sample under-/overshoot risk in methods for function getFixRobIC.

getFixClip 45

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

Ruckdeschel, P. and Kohl, M. (2005) Computation of the Finite Sample Risk of M-estimators on Neighborhoods.

See Also

fiRisk-class

getFixClip

Generic Function for the Computation of the Optimal Clipping Bound

Description

Generic function for the computation of the optimal clipping bound in case of robust models with fixed neighborhoods. This function is rarely called directly. It is used to compute optimally robust ICs.

Usage

```
getFixClip(clip, Distr, risk, neighbor, ...)
## S4 method for signature 'numeric,Norm,fiUnOvShoot,ContNeighborhood'
getFixClip(clip, Distr, risk, neighbor)
## S4 method for signature 'numeric,Norm,fiUnOvShoot,TotalVarNeighborhood'
getFixClip(clip, Distr, risk, neighbor)
```

Arguments

```
clip positive real: clipping bound
Distr object of class "Distribution".
risk object of class "RiskType".
neighbor object of class "Neighborhood".
... additional parameters.
```

Value

The optimal clipping bound is computed.

46 getFixRobIC

Methods

```
clip = "numeric", Distr = "Norm", risk = "fiUnOvShoot", neighbor = "ContNeighborhood"
    optimal clipping bound for finite-sample under-/overshoot risk.
```

clip = "numeric", Distr = "Norm", risk = "fiUnOvShoot", neighbor = "TotalVarNeighborhood"
 optimal clipping bound for finite-sample under-/overshoot risk.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
ContIC-class, TotalVarIC-class
```

getFixRobIC

Generic Function for the Computation of Optimally Robust ICs

Description

Generic function for the computation of optimally robust ICs in case of robust models with fixed neighborhoods. This function is rarely called directly.

Usage

Arguments

```
Distr object of class "Distribution".

risk object of class "RiskType".

neighbor object of class "Neighborhood".

... additional parameters.

sampleSize integer: sample size.
```

upper upper bound for the optimal clipping bound.

getIneffDiff 47

maxiter	the maximum number of iterations.
tol	the desired accuracy (convergence tolerance).
warn	logical: print warnings.
Algo	"A" or "B".
cont	"left" or "right".

Value

The optimally robust IC is computed.

Methods

Distr = "Norm", risk = "fiUnOvShoot", neighbor = "UncondNeighborhood" computes the optimally robust influence curve for one-dimensional normal location and finite-sample under-/overshoot risk.

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

FixRobModel-class

getIneffDiff

Generic Function for the Computation of Inefficiency Differences

Description

Generic function for the computation of inefficiency differencies. This function is rarely called directly. It is used to compute the radius minimax IC and the least favorable radius.

48 getIneffDiff

Arguments

radius neighborhood radius.

L2Fam L2-differentiable family of probability measures.

neighbor object of class "Neighborhood".
risk object of class "RiskType".

... additional parameters

the lower end point of the interval to be searched.

upRad the upper end point of the interval to be searched.

loRisk the risk at the lower end point of the interval.

upRisk the risk at the upper end point of the interval.

z.start initial value for the centering constant.

A.start initial value for the standardizing matrix.

upper.b upper bound for the optimal clipping bound.

MaxIter the maximum number of iterations

eps the desired accuracy (convergence tolerance).

warn logical: print warnings.

Value

The inefficieny difference between the left and the right margin of a given radius interval is computed.

Methods

radius = "numeric", L2Fam = "L2ParamFamily", neighbor = "UncondNeighborhood", risk = "asMSE": computes difference of asymptotic MSE-inefficiency for the boundaries of a given radius interval.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H., Kohl, M. and Ruckdeschel, P. (2001) The Costs of not Knowing the Radius. Submitted. Appeared as discussion paper Nr. 81. SFB 373 (Quantification and Simulation of Economic Processes), Humboldt University, Berlin; also available under www.uni-bayreuth.de/departments/math/org/mathe7/RIEDER/pubs/RR.pdf

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

radiusMinimaxIC, leastFavorableRadius

getInfCent 49

getInfCent	Generic Function for the Computation of the Optimal Centering Constant/Lower Clipping Bound

Description

Generic function for the computation of the optimal centering constant (contamination neighborhoods) respectively, of the optimal lower clipping bound (total variation neighborhood). This function is rarely called directly. It is used to compute optimally robust ICs.

Usage

```
getInfCent(L2deriv, neighbor, ...)
## S4 method for signature 'UnivariateDistribution,ContNeighborhood'
getInfCent(L2deriv, neighbor, clip, cent, tol.z, symm, trafo)
## S4 method for signature 'UnivariateDistribution,TotalVarNeighborhood'
getInfCent(L2deriv, neighbor, clip, cent, tol.z, symm, trafo)
## S4 method for signature 'RealRandVariable,ContNeighborhood'
getInfCent(L2deriv, neighbor, Distr, z.comp, stand, cent, clip)
```

Arguments

L2deriv	L2-derivative of some L2-differentiable family of probability measures.
neighbor	object of class "Neighborhood".
	additional parameters.
Distr	distribution of L2-differentiable family.
clip	optimal clipping bound.
cent	optimal centering constant.
stand	standardizing matrix.
tol.z	the desired accuracy (convergence tolerance).
symm	logical: indicating symmetry of L2deriv.
trafo	matrix: transformation of the parameter.
z.comp	logical vector: indication which components of the centering constant have to be computed.

Value

The optimal centering constant is computed.

50 getInfClip

Methods

L2deriv = "UnivariateDistribution", neighbor = "ContNeighborhood" computation of optimal centering constant.

L2deriv = "UnivariateDistribution", neighbor = "TotalVarNeighborhood" computation of optimal lower clipping bound.

L2deriv = "RealRandVariable", neighbor = "ContNeighborhood" computation of optimal centering constant.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

ContIC-class, TotalVarIC-class

getInfClip

Generic Function for the Computation of the Optimal Clipping Bound

Description

Generic function for the computation of the optimal clipping bound in case of infinitesimal robust models. This function is rarely called directly. It is used to compute optimally robust ICs.

```
getInfClip(clip, L2deriv, risk, neighbor, ...)

## S4 method for signature
## 'numeric,UnivariateDistribution,asMSE,ContNeighborhood'
getInfClip(clip, L2deriv, risk, neighbor, cent, symm, trafo)

## S4 method for signature
## 'numeric,UnivariateDistribution,asMSE,TotalVarNeighborhood'
getInfClip(clip, L2deriv, risk, neighbor, cent, symm, trafo)

## S4 method for signature 'numeric,EuclRandVariable,asMSE,ContNeighborhood'
getInfClip(clip, L2deriv, risk, neighbor, Distr, stand, cent, trafo)

## S4 method for signature
## 'numeric,UnivariateDistribution,asUnOvShoot,UncondNeighborhood'
getInfClip(clip, L2deriv, risk, neighbor, cent, symm, trafo)
```

getInfClip 51

Arguments

clip positive real: clipping bound

L2-derivative of some L2-differentiable family of probability measures.

risk object of class "RiskType".
neighbor object of class "Neighborhood".

... additional parameters.

cent optimal centering constant.

stand standardizing matrix.

Distr object of class "Distribution".

symm logical: indicating symmetry of L2deriv.
trafo matrix: transformation of the parameter.

Value

The optimal clipping bound is computed.

Methods

clip = "numeric", L2deriv = "UnivariateDistribution", risk = "asMSE", neighbor = "ContNeighborhood" optimal clipping bound for asymtotic mean square error.

clip = "numeric", L2deriv = "UnivariateDistribution", risk = "asMSE", neighbor = "TotalVarNeighborhood" optimal clipping bound for asymtotic mean square error.

clip = "numeric", L2deriv = "EuclRandVariable", risk = "asMSE", neighbor = "ContNeighborhood" optimal clipping bound for asymtotic mean square error.

clip = "numeric", L2deriv = "UnivariateDistribution", risk = "asUnOvShoot", neighbor = "UncondNeighborhood" optimal clipping bound for asymtotic under-/overshoot risk.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

ContIC-class, TotalVarIC-class

52 getInfGamma

getInfGamma

Generic Function for the Computation of the Optimal Clipping Bound

Description

Generic function for the computation of the optimal clipping bound. This function is rarely called directly. It is called by getInfClip to compute optimally robust ICs.

Usage

```
getInfGamma(L2deriv, risk, neighbor, ...)

## S4 method for signature 'UnivariateDistribution,asMSE,ContNeighborhood'
getInfGamma(L2deriv, risk, neighbor, cent, clip)

## S4 method for signature
## 'UnivariateDistribution,asGRisk,TotalVarNeighborhood'
getInfGamma(L2deriv, risk, neighbor, cent, clip)

## S4 method for signature 'RealRandVariable,asMSE,ContNeighborhood'
getInfGamma(L2deriv, risk, neighbor, Distr, stand, cent, clip)

## S4 method for signature
## 'UnivariateDistribution,asUnOvShoot,ContNeighborhood'
getInfGamma(L2deriv, risk, neighbor, cent, clip)
```

Arguments

L2deriv	L2-derivative of some L2-differentiable family of probability measures.
risk	object of class "RiskType".
neighbor	object of class "Neighborhood".
	additional parameters
cent	optimal centering constant.
clip	optimal clipping bound.
stand	standardizing matrix.
Distr	object of class "Distribution".

Details

The function is used in case of asymptotic G-risks; confer Ruckdeschel and Rieder (2004).

getInfRobIC 53

Methods

```
L2deriv = "UnivariateDistribution", risk = "asMSE", neighbor = "ContNeighborhood" used by getInfClip.
```

L2deriv = "UnivariateDistribution", risk = "asGRisk", neighbor = "TotalVarNeighborhood" used by getInfClip.

L2deriv = "RealRandVariable", risk = "asMSE", neighbor = "ContNeighborhood" used by getInfClip.

L2deriv = "UnivariateDistribution", risk = "asUnOvShoot", neighbor = "ContNeighborhood" used by getInfClip.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Ruckdeschel, P. and Rieder, H. (2004) Optimal Influence Curves for General Loss Functions. Statistics & Decisions (submitted).

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

asGRisk-class, asMSE-class, asUnOvShoot-class, ContIC-class, TotalVarIC-class

getInfRobIC

Generic Function for the Computation of Optimally Robust ICs

Description

Generic function for the computation of optimally robust ICs in case of infinitesimal robust models. This function is rarely called directly.

```
getInfRobIC(L2deriv, risk, neighbor, ...)
## S4 method for signature 'UnivariateDistribution,asCov,ContNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, Finfo, trafo)
## S4 method for signature 'UnivariateDistribution,asCov,TotalVarNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, Finfo, trafo)
## S4 method for signature 'RealRandVariable,asCov,ContNeighborhood'
```

54 getInfRobIC

```
getInfRobIC(L2deriv, risk, neighbor, Distr, Finfo, trafo)
## S4 method for signature 'UnivariateDistribution,asBias,ContNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, symm, Finfo, trafo,
             upper, maxiter, tol, warn)
## S4 method for signature 'UnivariateDistribution,asBias,TotalVarNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, symm, Finfo, trafo,
             upper, maxiter, tol, warn)
## S4 method for signature 'RealRandVariable,asBias,ContNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, Distr, DistrSymm, L2derivSymm,
          L2derivDistrSymm, Finfo, z.start, A.start, trafo, upper, maxiter, tol, warn)
## S4 method for signature 'UnivariateDistribution,asHampel,UncondNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, symm, Finfo, trafo,
             upper, maxiter, tol, warn)
## S4 method for signature 'RealRandVariable,asHampel,ContNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, Distr, DistrSymm, L2derivSymm,
          L2derivDistrSymm, Finfo, trafo, z.start, A.start, upper, maxiter, tol, warn)
## S4 method for signature 'UnivariateDistribution,asGRisk,UncondNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, symm, Finfo, trafo,
             upper, maxiter, tol, warn)
## S4 method for signature 'RealRandVariable,asGRisk,ContNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, Distr, DistrSymm, L2derivSymm,
          L2derivDistrSymm, Finfo, trafo, z.start, A.start, upper, maxiter, tol, warn)
## S4 method for signature
## 'UnivariateDistribution,asUnOvShoot,UncondNeighborhood'
getInfRobIC(L2deriv, risk, neighbor, symm, Finfo, trafo,
             upper, maxiter, tol, warn)
```

Arguments

L2deriv L2-derivative of some L2-differentiable family of probability measures.

risk object of class "RiskType".

object of class "Neighborhood". neighbor

additional parameters. . . .

object of class "Distribution". Distr

logical: indicating symmetry of L2deriv. symm object of class "DistributionSymmetry". DistrSymm

L2derivSymm object of class "FunSymmList".

L2derivDistrSymm

object of class "DistrSymmList".

getInfRobIC 55

Finfo	Fisher information matrix.
z.start	initial value for the centering constant.
A.start	initial value for the standardizing matrix.
trafo	matrix: transformation of the parameter.
upper	upper bound for the optimal clipping bound.
maxiter	the maximum number of iterations

maxiter the maximum number of iterations.

tol the desired accuracy (convergence tolerance).

warn logical: print warnings.

Value

The optimally robust IC is computed.

Methods

- **L2deriv = "UnivariateDistribution", risk = "asCov", neighbor = "ContNeighborhood"** computes the classical optimal influence curve for L2 differentiable parametric families with unknown one-dimensional parameter.
- **L2deriv = "UnivariateDistribution", risk = "asCov", neighbor = "TotalVarNeighborhood"** computes the classical optimal influence curve for L2 differentiable parametric families with unknown one-dimensional parameter.
- **L2deriv = "RealRandVariable", risk = "asCov", neighbor = "ContNeighborhood"** computes the classical optimal influence curve for L2 differentiable parametric families with unknown k-dimensional parameter (k > 1) where the underlying distribution is univariate.
- **L2deriv = "UnivariateDistribution", risk = "asBias", neighbor = "ContNeighborhood"** computes the bias optimal influence curve for L2 differentiable parametric families with unknown one-dimensional parameter.
- **L2deriv = "UnivariateDistribution", risk = "asBias", neighbor = "TotalVarNeighborhood"** computes the bias optimal influence curve for L2 differentiable parametric families with unknown one-dimensional parameter.
- **L2deriv = "RealRandVariable", risk = "asBias", neighbor = "ContNeighborhood"** computes the bias optimal influence curve for L2 differentiable parametric families with unknown k-dimensional parameter (k > 1) where the underlying distribution is univariate.
- **L2deriv** = "UnivariateDistribution", risk = "asHampel", neighbor = "UncondNeighborhood" computes the optimally robust influence curve for L2 differentiable parametric families with unknown one-dimensional parameter.
- **L2deriv = "RealRandVariable", risk = "asHampel", neighbor = "ContNeighborhood"** computes the optimally robust influence curve for L2 differentiable parametric families with unknown k-dimensional parameter (k > 1) where the underlying distribution is univariate.
- **L2deriv = "UnivariateDistribution", risk = "asGRisk", neighbor = "UncondNeighborhood"** computes the optimally robust influence curve for L2 differentiable parametric families with unknown one-dimensional parameter.
- **L2deriv = "RealRandVariable", risk = "asGRisk", neighbor = "ContNeighborhood"** computes the optimally robust influence curve for L2 differentiable parametric families with unknown k-dimensional parameter (k > 1) where the underlying distribution is univariate.

56 getInfStand

L2deriv = "UnivariateDistribution", risk = "asUnOvShoot", neighbor = "UncondNeighborhood" computes the optimally robust influence curve for one-dimensional L2 differentiable parametric families and asymptotic under-/overshoot risk.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

InfRobModel-class

getInfStand

Generic Function for the Computation of the Standardizing Matrix

Description

Generic function for the computation of the standardizing matrix which takes care of the Fisher consistency of the corresponding IC. This function is rarely called directly. It is used to compute optimally robust ICs.

Usage

```
getInfStand(L2deriv, neighbor, ...)

## S4 method for signature 'UnivariateDistribution,ContNeighborhood'
getInfStand(L2deriv, neighbor, clip, cent, trafo)

## S4 method for signature 'UnivariateDistribution,TotalVarNeighborhood'
getInfStand(L2deriv, neighbor, clip, cent, trafo)

## S4 method for signature 'RealRandVariable,ContNeighborhood'
getInfStand(L2deriv, neighbor, Distr, A.comp, stand, clip, cent, trafo)
```

Arguments

L2-derivative of some L2-differentiable family of probability measures.

neighbor object of class "Neighborhood"

... additional parameters

getRiskIC 57

clip	optimal clipping bound.
cent	optimal centering constant.
stand	standardizing matrix.
Distr	object of class "Distribution".
trafo	matrix: transformation of the parameter.
A.comp	matrix: indication which components of the standardizing matrix have to be computed.

Value

The standardizing matrix is computed.

Methods

L2deriv = "UnivariateDistribution", neighbor = "ContNeighborhood" computes standardizing matrix.

L2deriv = "UnivariateDistribution", neighbor = "TotalVarNeighborhood" computes standardizing matrix.

L2deriv = "RealRandVariable", neighbor = "ContNeighborhood" computes standardizing matrix.

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

ContIC-class, TotalVarIC-class

Description

Generic function for the computation of a risk for an IC.

58 getRiskIC

Usage

```
getRiskIC(IC, risk, neighbor, L2Fam, ...)
## S4 method for signature 'IC,asCov,missing,missing'
getRiskIC(IC, risk, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,asCov,missing,L2ParamFamily'
getRiskIC(IC, risk, L2Fam, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC, trAsCov, missing, missing'
getRiskIC(IC, risk, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,trAsCov,missing,L2ParamFamily'
getRiskIC(IC, risk, L2Fam, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,asBias,ContNeighborhood,missing'
getRiskIC(IC, risk, neighbor, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,asBias,ContNeighborhood,L2ParamFamily'
getRiskIC(IC, risk, neighbor, L2Fam, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,asBias,TotalVarNeighborhood,missing'
getRiskIC(IC, risk, neighbor, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,asBias,TotalVarNeighborhood,L2ParamFamily'
getRiskIC(IC, risk, neighbor, L2Fam, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,asMSE,UncondNeighborhood,missing'
getRiskIC(IC, risk, neighbor, tol = .Machine$double.eps^0.25)
## S4 method for signature 'IC,asMSE,UncondNeighborhood,L2ParamFamily'
getRiskIC(IC, risk, neighbor, L2Fam, tol = .Machine$double.eps^0.25)
## S4 method for signature 'TotalVarIC,asUnOvShoot,UncondNeighborhood,missing'
getRiskIC(IC, risk, neighbor)
## S4 method for signature 'IC,fiUnOvShoot,ContNeighborhood,missing'
getRiskIC(IC, risk, neighbor, sampleSize, Algo = "A", cont = "left")
## S4 method for signature 'IC, fiUnOvShoot, TotalVarNeighborhood, missing'
getRiskIC(IC, risk, neighbor, sampleSize, Algo = "A", cont = "left")
```

Arguments

IC object of class "InfluenceCurve" risk object of class "RiskType".
neighbor object of class "Neighborhood".

getRiskIC 59

cobject of class "L2ParamFamily".

additional parameters

tol the desired accuracy (convergence tolerance).

sampleSize integer: sample size.

Algo "A" or "B".

cont "left" or "right".

Details

To make sure that the results are valid, it is recommended to include an additional check of the IC properties of IC using checkIC.

Value

The risk of an IC is computed.

Methods

- IC = "IC", risk = "asCov", neighbor = "missing", L2Fam = "missing" asymptotic covariance
 of IC.
- IC = "IC", risk = "asCov", neighbor = "missing", L2Fam = "L2ParamFamily" asymptotic covariance of IC under L2Fam.
- IC = "IC", risk = "trAsCov", neighbor = "missing", L2Fam = "missing" asymptotic covariance
 of IC.
- IC = "IC", risk = "trAsCov", neighbor = "missing", L2Fam = "L2ParamFamily" asymptotic covariance of IC under L2Fam.
- IC = "IC", risk = "asBias", neighbor = "ContNeighborhood", L2Fam = "missing" asymptotic bias of IC under convex contaminations.
- IC = "IC", risk = "asBias", neighbor = "ContNeighborhood", L2Fam = "L2ParamFamily" asymptotic bias of IC under convex contaminations and L2Fam.
- IC = "IC", risk = "asBias", neighbor = "TotalVarNeighborhood", L2Fam = "missing" asymptotic bias of IC in case of total variation neighborhoods.
- IC = "IC", risk = "asBias", neighbor = "TotalVarNeighborhood", L2Fam = "L2ParamFamily" asymptotic bias of IC under L2Fam in case of total variation neighborhoods.
- IC = "IC", risk = "asMSE", neighbor = "UncondNeighborhood", L2Fam = "missing" asymptotic mean square error of IC.
- IC = "IC", risk = "asMSE", neighbor = "UncondNeighborhood", L2Fam = "L2ParamFamily" asymptotic mean square error of IC under L2Fam.
- IC = "TotalVarIC", risk = "asUnOvShoot", neighbor = "UncondNeighborhood", L2Fam = "missing" asymptotic under-/overshoot risk of IC.
- IC = "IC", risk = "fiUnOvShoot", neighbor = "ContNeighborhood", L2Fam = "missing" finitesample under-/overshoot risk of IC.
- IC = "IC", risk = "fiUnOvShoot", neighbor = "TotalVarNeighborhood", L2Fam = "missing" finite-sample under-/overshoot risk of IC.

60 Gumbel

Note

This generic function is still under construction.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278.

Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

Ruckdeschel, P. and Kohl, M. (2005) Computation of the Finite Sample Risk of M-estimators on Neighborhoods.

See Also

getRiskIC-methods, InfRobModel-class

Gumbel |

Generating function for Gumbel-class

Description

Generates an object of class "Gumbel".

Usage

```
Gumbel(loc = 0, scale = 1)
```

Arguments

loc real number: location parameter of the Gumbel distribution.

scale positive real number: scale parameter of the Gumbel distribution

Value

Object of class "Gumbel"

Note

The class "Gumbel" is based on the code provided by the package evd.

Gumbel-class 61

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

```
Gumbel-class, rgumbel
```

Examples

```
(G1 <- Gumbel(loc = 1, scale = 2))
plot(G1)
loc(G1)
scale(G1)
loc(G1) <- -1
scale(G1) <- 2
plot(G1)

E(Gumbel()) # Euler's constant
E(G1, function(x){x^2})

## The function is currently defined as function(loc = 0, scale = 1){
   new("Gumbel", loc = loc, scale = scale)
}</pre>
```

Gumbel-class

Gumbel distribution

Description

The Gumbel cumulative distribution function with location parameter $loc = \mu$ and scale parameter $scale = \sigma$ is

$$F(x) = \exp(-\exp[-(x - \mu)/\sigma])$$

for all real x, where $\sigma > 0$; c.f. rgumbel. This distribution is also known as extreme value distribution of type I; confer Chapter~22 of Johnson et al. (1995).

```
E(object, fun, cond, ...)
## S4 method for signature 'Gumbel,missing,missing'
E(object, low = NULL, upp = NULL, ...)
var(x, ...)
## S4 method for signature 'Gumbel'
var(x, ...)
skewness(x, ...)
## S4 method for signature 'Gumbel'
skewness(x, ...)
kurtosis(x, ...)
```

62 Gumbel-class

```
## S4 method for signature 'Gumbel'
kurtosis(x, ...)
```

Arguments

object of class "Distribution"

fun if missing the (conditional) expectation is computed else the (conditional) expection of fun is computed.

cond if not missing the conditional expectation given cond is computed.

low lower bound of integration range.

upp upper bound of integration range.

x object of class "UnivariateDistribution"

... additional arguments to fun

Objects from the Class

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

Slots

```
img Object of class "Reals".
param Object of class "GumbelParameter".
r rgumbel
d dgumbel
p pgumbel
q qgumbel
gaps (numeric) matrix or NULL
```

- .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 "AbscontDistribution", directly.
Class "UnivariateDistribution", by class "AbscontDistribution".
Class "Distribution", by class "AbscontDistribution".
```

Gumbel-class 63

Methods

```
initialize signature(.Object = "Gumbel"): initialize method.
loc signature(object = "Gumbel"): wrapped access method for slot loc of slot param.
scale signature(x = "Gumbel"): wrapped access method for slot scale of slot param.
loc<- signature(object = "Gumbel"): wrapped replace method for slot loc of slot param.
scale<- signature(x = "Gumbel"): wrapped replace method for slot scale of slot param.
+ signature(e1 = "Gumbel", e2 = "numeric"): result again of class "Gumbel"; exact.
* signature(e1 = "Gumbel", e2 = "numeric"): result again of class "Gumbel"; exact.

E signature(object = "Gumbel", fun = "missing", cond = "missing"): exact evaluation of expectation using explicit expressions.

var signature(x = "Gumbel"): exact evaluation of expectation using explicit expressions.
kurtosis signature(x = "Gumbel"): exact evaluation of expectation using explicit expressions.
median signature(x = "Gumbel"): exact evaluation of expectation using explicit expressions.
IQR signature(x = "Gumbel"): exact evaluation of expectation using explicit expressions.</pre>
```

Note

This class is based on the code provided by the package **evd**.

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Johnson et al. (1995) Continuous Univariate Distributions. Vol. 2. 2nd ed. New York: Wiley.

See Also

```
rgumbel, AbscontDistribution-class
```

```
(G1 <- new("Gumbel", loc = 1, scale = 2))
plot(G1)
loc(G1)
scale(G1)
loc(G1) <- -1
scale(G1) <- 2
plot(G1)</pre>
```

GumbelLocationFamily Generating function for Gumbel location families

Description

Generates an object of class "L2ParamFamily" which represents a Gumbel location family.

Usage

```
GumbelLocationFamily(loc = 0, scale = 1, trafo)
```

Arguments

loc location parameter scale scale parameter

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Gumbel-class
```

GumbelParameter-class 65

GumbelParameter-class Paramter of Gumbel distributions

Description

The class of the parameter of Gumbel distributions.

Objects from the Class

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

Slots

```
loc real number: location parameter of a Gumbel distribution.

scale positive real number: scale parameter of a Gumbel distribution.

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

Extends

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

Methods

```
loc signature(object = "GumbelParameter"): access method for slot loc.
scale signature(x = "GumbelParameter"): access method for slot scale.
loc<- signature(object = "GumbelParameter"): replace method for slot loc.
scale<- signature(x = "GumbelParameter"): replace method for slot scale.</pre>
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

```
Gumbel-class, Parameter-class
```

```
new("GumbelParameter")
```

66 IC

IC

Generating function for IC-class

Description

Generates an object of class "IC".

Usage

Arguments

name Object of class "character".

CallL2Fam object of class "call": creates an object of the underlying L2-differentiable

parametric family.

Curve object of class "EuclRandVarList".

Risks object of class "list": list of risks; cf. RiskType-class.

Infos matrix of characters with two columns named method and message: additional

informations.

Value

Object of class "IC"

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

IC-class

IC-class 67

Examples

IC-class

Influence curve

Description

Class of (partial) influence curves.

Objects from the Class

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

Slots

```
CallL2Fam: Object of class "call": creates an object of the underlying L2-differentiable parametric family.
name: Object of class "character".
Curve: Object of class "EuclRandVarList".
Risks: Object of class "list": list of risks; cf. RiskType-class.
Infos: Object of class "matrix" with two columns named method and message: additional informations.
```

Extends

Class "InfluenceCurve", directly.

68 InfluenceCurve

Methods

```
CallL2Fam signature(object = "IC"): accessor function for slot CallL2Fam.
CallL2Fam<- signature(object = "IC"): replacement function for slot CallL2Fam.
checkIC signature(IC = "IC", L2Fam = "missing"): check centering and Fisher consistency of IC assuming the L2-differentiable parametric family which can be generated via the slot CallL2Fam of IC.
checkIC signature(IC = "IC", L2Fam = "L2ParamFamily"): check centering and Fisher consistency of IC assuming the L2-differentiable parametric family L2Fam.
evalIC signature(IC = "IC", x = "numeric"): evaluate IC at x.
evalIC signature(IC = "IC", x = "matrix"): evaluate IC at the rows of x.
infoPlot signature(object = "IC"): Plot absolute and relative information of IC.
plot signature(object = "IC")
show signature(object = "IC")</pre>
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
InfluenceCurve-class, IC
```

Examples

```
IC1 <- new("IC")
plot(IC1)</pre>
```

InfluenceCurve

Generating function for InfluenceCurve-class

Description

Generates an object of class "InfluenceCurve".

InfluenceCurve 69

Usage

Arguments

name character string: name of the influence curve

Curve object of class "EuclRandVarList"

Risks list of risks

Infos matrix of characters with two columns named method and message: additional

informations

Value

Object of class "InfluenceCurve"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

InfluenceCurve-class

70 InfluenceCurve-class

```
InfluenceCurve-class Influence curve
```

Description

Class of influence curves (functions).

Objects from the Class

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

Slots

```
name: object of class "character"
Curve: object of class "EuclRandVarList"
Risks: object of class "list": list of risks; cf. RiskType-class.
Infos: object of class "matrix" with two columns named method and message: additional informations.
```

Methods

```
name signature(object = "InfluenceCurve"): accessor function for slot name.
name<- signature(object = "InfluenceCurve"): replacement function for slot name.</pre>
Curve signature(object = "InfluenceCurve"): accessor function for slot Curve.
Map signature(object = "InfluenceCurve"): accessor function for slot Map of slot Curve.
Domain signature(object = "InfluenceCurve"): accessor function for slot Domain of slot
    Curve.
Range signature(object = "InfluenceCurve"): accessor function for slot Range of slot
    Curve.
Infos signature(object = "InfluenceCurve"): accessor function for slot Infos.
Infos<- signature(object = "InfluenceCurve"): replacement function for slot Infos.</pre>
addInfo<- signature(object = "InfluenceCurve"): function to add an information to slot
     Infos.
Risks signature(object = "InfluenceCurve"): accessor function for slot Risks.
Risks<- signature(object = "InfluenceCurve"): replacement function for slot Risks.
addRisk<- signature(object = "InfluenceCurve"): function to add a risk to slot Risks.
show signature(object = "InfluenceCurve")
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

infoPlot 71

References

Hampel et al. (1986) *Robust Statistics*. The Approach Based on Influence Functions. New York: Wiley.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
InfluenceCurve, RiskType-class
```

Examples

```
new("InfluenceCurve")
```

infoPlot

Plot absolute and relative information

Description

Plot absolute and relative information of influence curves.

Usage

```
infoPlot(object)
```

Arguments

object

object of class "InfluenceCurve"

Details

Absolute information is defined as the square of the length of an IC. The relative information is defined as the absolute information of one component with respect to the absolute information of the whole IC; confer Section 8.1 of Kohl (2005).

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, IC-class
```

72 InfRobModel

Examples

```
N <- NormLocationScaleFamily(mean=0, sd=1)
IC1 <- optIC(model = N, risk = asCov())
infoPlot(IC1)</pre>
```

InfRobModel

Generating function for InfRobModel-class

Description

Generates an object of class "InfRobModel".

Usage

```
InfRobModel(center = L2ParamFamily(), neighbor = ContNeighborhood())
```

Arguments

```
center object of class "ProbFamily"
neighbor object of class "UncondNeighborhood"
```

Value

```
Object of class "FixRobModel"
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
RobModel-class, FixRobModel-class
```

```
(M1 <- InfRobModel())
## The function is currently defined as
function(center = L2ParamFamily(), neighbor = ContNeighborhood()){
    new("InfRobModel", center = center, neighbor = neighbor)
}</pre>
```

InfRobModel-class 73

InfRobModel-class

Robust model with infinitesimal (unconditional) neighborhood

Description

Class of robust models with infinitesimal (unconditional) neighborhoods; i.e., the neighborhood is shrinking at a rate of \sqrt{n} .

Objects from the Class

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

Slots

```
center: Object of class "ProbFamily".
neighbor: Object of class "UncondNeighborhood".
```

Extends

Class "RobModel", directly.

Methods

```
neighbor<- signature(object = "InfRobModel"): replacement function for slot neighbor<-
show signature(object = "InfRobModel")</pre>
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

ProbFamily-class, UncondNeighborhood-class, InfRobModel

```
new("InfRobModel")
```

74 ksEstimator

ksEstimator	Generic Function for the Computation of the Kolmogorov Minimum Distance Estimator
	Distance Estimator

Description

Generic function for the computation of the Kolmogorov(-Smirnov) minimum distance estimator.

Usage

```
ksEstimator(x, distribution, ...)
## S4 method for signature 'numeric,Binom'
ksEstimator(x, distribution, param, eps = .Machine$double.eps^0.5)
## S4 method for signature 'numeric,Pois'
ksEstimator(x, distribution, param, eps = .Machine$double.eps^0.5)
## S4 method for signature 'numeric,Norm'
ksEstimator(x, distribution, param, eps = .Machine$double.eps^0.5)
## S4 method for signature 'numeric,Lnorm'
ksEstimator(x, distribution, param, eps = .Machine$double.eps^0.5)
## S4 method for signature 'numeric,Gumbel'
ksEstimator(x, distribution, param, eps = .Machine$double.eps^0.5)
## S4 method for signature 'numeric,Exp'
ksEstimator(x, distribution, param, eps = .Machine$double.eps^0.5)
## S4 method for signature 'numeric,Gammad'
ksEstimator(x, distribution, param, eps = .Machine$double.eps^0.5)
```

Arguments

X	sample
distribution	object of class "Distribution"
	additional parameters
param	name of the unknown parameter. If missing all parameters of the corresponding distribution are estimated.
eps	the desired accuracy (convergence tolerance).

Details

In case of discrete distributions the Kolmogorov distance is computed and the parameters which lead to the minimum distance are returned. In case of absolutely continuous distributions ks.test is called and the parameters which minimize the corresponding test statistic are returned.

ksEstimator 75

Value

The Kolmogorov minimum distance estimator is computed. Returns a list with components named like the parameters of distribution.

Methods

```
    x = "numeric", distribution = "Binom" Binomial distributions.
    x = "numeric", distribution = "Pois" Poisson distributions.
    x = "numeric", distribution = "Norm" Normal distributions.
    x = "numeric", distribution = "Lnorm" Lognormal distributions.
    x = "numeric", distribution = "Gumbel" Gumbel distributions.
    x = "numeric", distribution = "Exp" Exponential distributions.
    x = "numeric", distribution = "Gamma" Gamma distributions.
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

Distribution-class

```
x \leftarrow rnorm(100, mean = 1, sd = 2)

ksEstimator(x=x, distribution = Norm()) # estimate mean and sd

ksEstimator(x=x, distribution = Norm(mean = 1), param = "sd") # estimate sd

ksEstimator(x=x, distribution = Norm(sd = 2), param = "mean") # estimate mean

mean(x)

median(x)

sd(x)

mad(x)
```

76 L2ParamFamily

L2ParamFamily Generating function for L2ParamFamily-class

Description

Generates an object of class "L2ParamFamily".

Usage

Arguments

name character string: name of the family

distribution object of class "Distribution": member of the family

distrSymm object of class "DistributionSymmetry": symmetry of distribution.

main numeric vector: main parameter
nuisance numeric vector: nuisance parameter
trafo matrix: transformation of the parameter

param object of class "ParamFamParameter": parameter of the family

props character vector: properties of the family

L2deriv object of class "EuclRandVariable": L2 derivative of the family

L2derivSymm object of class "FunSymmList": symmetry of the maps contained in L2deriv

L2derivDistr object of class "UnivarDistrList": distribution of L2deriv

L2derivDistrSymm

object of class "DistrSymmList": symmetry of the distributions contained in

L2derivDistr

FisherInfo object of class "PosDefSymmMatrix": Fisher information of the family

Details

If name is missing, the default "L2 differentiable parametric family of probability measures" is used. In case distrSymm is missing it is set to NoSymmetry(). If param is missing, the parameter is created via main, nuisance and trafo as described in ParamFamParameter. In case L2derivSymm is missing, it is filled with an object of class FunSymmList with entries NonSymmetric(). In case L2derivDistr is missing, it is computed via imageDistr. If L2derivDistrSymm is missing, it is set to an object of class DistrSymmList with entries NoSymmetry(). In case FisherInfo is missing, it is computed from L2deriv using E.

L2ParamFamily-class 77

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class
```

Examples

```
F1 <- L2ParamFamily()
plot(F1)</pre>
```

L2ParamFamily-class

L2 differentiable parametric family

Description

Class of L2 differentiable parametric families.

Objects from the Class

Objects can be created by calls of the form new("L2ParamFamily", ...). More frequently they are created via the generating function L2ParamFamily.

Slots

```
name: object of class "character": name of the family.
distribution: object of class "Distribution": member of the family.
distrSymm: Object of class "DistributionSymmetry": symmetry of distribution.
param: object of class "ParamFamParameter": parameter of the family.
props: object of class "character": properties of the family.
L2deriv: object of class "EuclRandVariable": L2 derivative of the family.
L2derivSymm: object of class "FunSymmList": symmetry of the maps included in L2deriv.
L2derivDistr: object of class "UnivarDistrList": list which includes the distribution of L2deriv.
L2derivDistrSymm: object of class "DistrSymmList": symmetry of the distributions included in L2derivDistr.
FisherInfo: object of class "PosDefSymmMatrix": Fisher information of the family.
```

Extends

```
Class "ParamFamily", directly.
Class "ProbFamily", by class "ParamFamily".
```

Methods

```
L2deriv signature(object = "L2ParamFamily"): accessor function for L2deriv.
```

L2derivSymm signature(object = "L2ParamFamily"): accessor function for L2derivSymm.

L2derivDistr signature(object = "L2ParamFamily"): accessor function for L2derivDistr.

L2derivDistrSymm signature(object = "L2ParamFamily"): accessor function for L2derivDistrSymm.

FisherInfo signature(object = "L2ParamFamily"): accessor function for FisherInfo.

checkL2deriv signature(object = "L2ParamFamily"): check centering of L2deriv and compute precision of Fisher information.

E signature(object = "L2ParamFamily", fun = "EuclRandVariable", cond = "missing"): expectation of fun under the distribution of object.

E signature(object = "L2ParamFamily", fun = "EuclRandMatrix", cond = "missing"): expectation of fun under the distribution of object.

E signature(object = "L2ParamFamily", fun = "EuclRandVarList", cond = "missing"): expectation of fun under the distribution of object.

plot signature(x = "L2ParamFamily"): plot of distribution and L2deriv.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
L2ParamFamily, ParamFamily-class
```

```
F1 <- new("L2ParamFamily")
plot(F1)</pre>
```

leastFavorableRadius 79

leastFavorableRadius Generic Function for the Computation of Least Favorable Radii

Description

Generic function for the computation of least favorable radii.

Usage

Arguments

L2Fam	L2-differentiable family of probability measures.
neighbor	object of class "Neighborhood".
risk	object of class "RiskType".
	additional parameters
upRad	the upper end point of the radius interval to be searched.
rho	The considered radius interval is: $[r\rho, r/\rho]$ with $\rho \in (0, 1)$.
z.start	initial value for the centering constant.
A.start	initial value for the standardizing matrix.
upper	upper bound for the optimal clipping bound.
maxiter	the maximum number of iterations
tol	the desired accuracy (convergence tolerance).
warn	logical: print warnings.

Value

The least favorable radius and the corresponding inefficiency are computed.

Methods

```
L2Fam = "L2ParamFamily", neighbor = "UncondNeighborhood", risk = "asGRisk" computation of the least favorable radius.
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

80 LnormScaleFamily

References

Rieder, H., Kohl, M. and Ruckdeschel, P. (2001) The Costs of not Knowing the Radius. Submitted. Appeared as discussion paper Nr. 81. SFB 373 (Quantification and Simulation of Economic Processes), Humboldt University, Berlin; also available under www.uni-bayreuth.de/departments/math/org/mathe7/RIEDER/pubs/RR.pdf

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

radiusMinimaxIC

Examples

LnormScaleFamily

Generating function for lognormal scale families

Description

Generates an object of class "L2ParamFamily" which represents a lognormal scale family.

Usage

```
LnormScaleFamily(meanlog = 0, sdlog = 1, trafo)
```

Arguments

meanlog mean of the distribution on the log scale

sdlog standard deviation of the distribution on the log scale

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

locMEstimator 81

References

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Lnorm-class
```

Examples

```
(L1 <- LnormScaleFamily())
plot(L1)
Map(L2deriv(L1)[[1]])
checkL2deriv(L1)</pre>
```

locMEstimator

Generic function for the computation of location M estimators

Description

Generic function for the computation of location M estimators.

Usage

```
locMEstimator(x, IC, ...)
## S4 method for signature 'numeric,InfluenceCurve'
locMEstimator(x, IC, eps = .Machine$double.eps^0.5)
```

Arguments

x sample

IC object of class "InfluenceCurve"

... additional parameters

eps the desired accuracy (convergence tolerance).

Value

Returns a list with component

loc M estimator of location

Methods

```
x = "numeric", IC = "InfluenceCurve" univariate location.
```

82 lowerCaseRadius

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1964) Robust estimation of a location parameter. Ann. Math. Stat. 35: 73-101.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

InfluenceCurve-class

lowerCaseRadius

Computation of the lower case radius

Description

The lower case radius is computed; confer Subsection 2.1.2 in Kohl (2005).

Usage

```
lowerCaseRadius(L2Fam, neighbor, risk, ...)
```

Arguments

L2Fam L2 differentiable parametric family neighbor object of class "Neighborhood" risk object of class "RiskType" ... additional parameters

Value

lower case radius

Methods

```
L2Fam = "L2ParamFamily", neighbor = "ContNeighborhood", risk = "asMSE" lower case radius for risk "asMSE" in case of "ContNeighborhood".
```

L2Fam = "L2ParamFamily", neighbor = "TotalVarNeighborhood", risk = "asMSE" lower case radius for risk "asMSE" in case of "TotalVarNeighborhood".

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

Neighborhood-class 83

References

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation

See Also

```
L2ParamFamily-class, Neighborhood-class
```

Examples

```
lowerCaseRadius(BinomFamily(size = 10), ContNeighborhood(), asMSE())
lowerCaseRadius(BinomFamily(size = 10), TotalVarNeighborhood(), asMSE())
```

Neighborhood-class

Neighborhood

Description

Class of neighborhoods of families of probability measures.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
type: Object of class "character": type of the neighborhood. radius: Object of class "numeric": neighborhood radius.
```

Methods

```
type signature(object = "Neighborhood"): accessor function for slot type.
radius signature(object = "Neighborhood"): accessor function for slot radius.
show signature(object = "Neighborhood")
```

Author(s)

Matthias Kohl <Matthias .Kohl@stamats.de>

References

```
Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.
```

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
ProbFamily-class
```

84 NonSymmetric-class

NonSymmetric

Generating function for NonSymmetric-class

Description

Generates an object of class "NonSymmetric".

Usage

```
NonSymmetric()
```

Value

Object of class "NonSymmetric"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

NonSymmetric-class, FunctionSymmetry-class

Examples

```
NonSymmetric()
## The function is currently defined as
function(){ new("NonSymmetric") }
```

NonSymmetric-class

Class for Non-symmetric Functions

Description

Class for non-symmetric functions.

Objects from the Class

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

Slots

```
type: Object of class "character": contains "non-symmetric function"
SymmCenter: Object of class "NULL"
```

NormLocationFamily 85

Extends

```
Class "FunctionSymmetry", directly.
Class "Symmetry", by class "FunctionSymmetry".
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

NonSymmetric

Examples

```
new("NonSymmetric")
```

NormLocationFamily

Generating function for normal location families

Description

Generates an object of class "L2ParamFamily" which represents a normal location family.

Usage

```
NormLocationFamily(mean = 0, sd = 1, trafo)
```

Arguments

mean mean

sd standard deviation

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Norm-class
```

Examples

```
(N1 <- NormLocationFamily())
plot(N1)
L2derivDistr(N1)</pre>
```

NormLocationScaleFamily

Generating function for normal location and scale families

Description

Generates an object of class "L2ParamFamily" which represents a normal location and scale family.

Usage

```
NormLocationScaleFamily(mean = 0, sd = 1, trafo)
```

Arguments

mean mean

sd standard deviation

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

NormScaleFamily 87

See Also

```
L2ParamFamily-class, Norm-class
```

Examples

```
(N1 <- NormLocationScaleFamily())
plot(N1)
FisherInfo(N1)
checkL2deriv(N1)</pre>
```

NormScaleFamily

Generating function for normal scale families

Description

Generates an object of class "L2ParamFamily" which represents a normal scale family.

Usage

```
NormScaleFamily(sd = 1, mean = 0, trafo)
```

Arguments

sd standard deviation

mean mean

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Norm-class
```

88 OddSymmetric

Examples

```
(N1 <- NormScaleFamily())
plot(N1)
FisherInfo(N1)
checkL2deriv(N1)</pre>
```

OddSymmetric

Generating function for OddSymmetric-class

Description

Generates an object of class "OddSymmetric".

Usage

```
OddSymmetric(SymmCenter = 0)
```

Arguments

```
SymmCenter numeric: center of symmetry
```

Value

Object of class "OddSymmetric"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

```
OddSymmetric-class, FunctionSymmetry-class
```

```
OddSymmetric()
## The function is currently defined as
function(SymmCenter = 0){
    new("OddSymmetric", SymmCenter = SymmCenter)
}
```

OddSymmetric-class 89

OddSymmetric-class

Class for Odd Functions

Description

Class for odd functions.

Objects from the Class

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

Slots

```
type: Object of class "character": contains "odd function"
SymmCenter: Object of class "numeric": center of symmetry
```

Extends

```
Class "FunctionSymmetry", directly.
Class "Symmetry", by class "FunctionSymmetry".
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

OddSymmetric, FunctionSymmetry-class

Examples

```
new("OddSymmetric")
```

 ${\tt oneStepEstimator}$

Generic function for the computation of one-step estimators

Description

Generic function for the computation of one-step estimators.

Usage

```
oneStepEstimator(x, IC, start)
```

90 optIC

Arguments

X	sample

IC object of class "InfluenceCurve"

start initial estimate

Details

Given an initial estimation start, a sample x and an influence curve IC the corresponding one-step estimator is computed

Value

The one-step estimation is computed.

Methods

```
    x = "numeric", IC = "InfluenceCurve", start = "numeric" univariate samples.
    x = "numeric", IC = "InfluenceCurve", start = "list" univariate samples.
    x = "matrix", IC = "InfluenceCurve", start = "numeric" multivariate samples.
    x = "matrix", IC = "InfluenceCurve", start = "list" multivariate samples.
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

InfluenceCurve-class

optIC

Generic function for the computation of optimally robust ICs

Description

Generic function for the computation of optimally robust ICs.

optIC 91

Usage

Arguments

model	probability model.
risk	object of class "RiskType".
	additional parameters.
z.start	initial value for the centering constant.
A.start	initial value for the standardizing matrix.
upper	upper bound for the optimal clipping bound.
maxiter	the maximum number of iterations.
tol	the desired accuracy (convergence tolerance).
warn	logical: print warnings.
sampleSize	integer: sample size.
Algo	"A" or "B".
cont	"left" or "right".

Details

In case of the finite-sample risk "fiUnovShoot" one can choose between two algorithms for the computation of this risk where the least favorable contamination is assumed to be left or right of some bound. For more details we refer to Section 11.3 of Kohl (2005).

Value

Some optimally robust IC is computed.

92 optRisk

Methods

model = "L2ParamFamily", risk = "asCov" computes classical optimal influence curve for L2 differentiable parametric families.

model = "InfRobModel", risk = "asRisk" computes optimally robust influence curve for robust models with infinitesimal neighborhoods and various asymptotic risks.

model = "InfRobModel", risk = "asUnOvShoot" computes optimally robust influence curve for robust models with infinitesimal neighborhoods and asymptotic under-/overshoot risk.

model = "FixRobModel", risk = "fiUnOvShoot" computes optimally robust influence curve for robust models with fixed neighborhoods and finite-sample under-/overshoot risk.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278.

Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

InfluenceCurve-class, RiskType-class

Examples

```
B <- BinomFamily(size = 25, prob = 0.25)
## classical optimal IC
IC0 <- optIC(model = B, risk = asCov())
plot(IC0) # plot IC
checkIC(IC0, B)</pre>
```

optRisk

Generic function for the computation of the minimal risk

Description

Generic function for the computation of the optimal (i.e., minimal) risk for a probability model.

optRisk 93

Usage

Arguments

model	probability model
risk	object of class RiskType
	additional parameters
z.start	initial value for the centering constant.
A.start	initial value for the standardizing matrix.
upper	upper bound for the optimal clipping bound.
maxiter	the maximum number of iterations
tol	the desired accuracy (convergence tolerance).
warn	logical: print warnings.
sampleSize	integer: sample size.
Algo	"A" or "B".
cont	"left" or "right".

Details

In case of the finite-sample risk "fiUnOvShoot" one can choose between two algorithms for the computation of this risk where the least favorable contamination is assumed to be left or right of some bound. For more details we refer to Section 11.3 of Kohl (2005).

Value

The minimal risk is computed.

Methods

```
    model = "L2ParamFamily", risk = "asCov" asymptotic covariance of L2 differentiable parameteric family.
    model = "InfRobModel", risk = "asRisk" asymptotic risk of a infinitesimal robust model.
    model = "FixRobModel", risk = "fiUnOvShoot" finite-sample under-/overshoot risk of a robust model with fixed neighborhood.
```

94 ParamFamily

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Huber, P.J. (1968) Robust Confidence Limits. Z. Wahrscheinlichkeitstheor. Verw. Geb. 10:269–278.

Rieder, H. (1980) Estimates derived from robust tests. Ann. Stats. 8: 106–115.

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
RiskType-class
```

Examples

```
optRisk(model = NormLocationScaleFamily(), risk = asCov())
```

ParamFamily

Generating function for ParamFamily-class

Description

Generates an object of class "ParamFamily".

Usage

Arguments

name character string: name of family

distribution object of class "Distribution": member of the family

distrSymm object of class "DistributionSymmetry": symmetry of distribution.

main numeric vector: main parameter
nuisance numeric vector: nuisance parameter
trafo matrix: transformation of the parameters

param object of class "ParamFamParameter": parameter of the family

props character vector: properties of the family

ParamFamily-class 95

Details

If name is missing, the default ""parametric family of probability measures"" is used. In case distrSymm is missing it is set to NoSymmetry(). If param is missing, the parameter is created via main, nuisance and trafo as described in ParamFamParameter.

Value

```
Object of class "ParamFamily"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

```
ParamFamily-class
```

Examples

ParamFamily-class

Parametric family of probability measures.

Description

Class of parametric families of probability measures.

Objects from the Class

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

96 ParamFamily-class

Slots

```
param: Object of class "ParamFamParameter": parameter of the family.

name: Object of class "character": name of the family.

distribution: Object of class "Distribution": member of the family.

distrSymm: Object of class "DistributionSymmetry": symmetry of distribution.

props: Object of class "character": properties of the family.
```

Extends

```
Class "ProbFamily", directly.
```

Methods

```
main signature(object = "ParamFamily"): wrapped accessor function for slot main of slot
    param.

nuisance signature(object = "ParamFamily"): wrapped accessor function for slot nuisance
    of slot param.

trafo signature(object = "ParamFamily"): wrapped accessor function for slot trafo of slot
    param.

param signature(object = "ParamFamily"): accessor function for slot param.

plot signature(x = "ParamFamily"): plot of slot distribution.

show signature(object = "ParamFamily")
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

See Also

```
Distribution-class
```

```
F1 <- new("ParamFamily") # prototype
plot(F1)</pre>
```

ParamFamParameter 97

ParamFamParameter	Generating function for ParamFamParameter-class
-------------------	---

Description

Generates an object of class "ParamFamParameter".

Usage

```
ParamFamParameter(name, main = numeric(0), nuisance, trafo)
```

Arguments

name character string: name of parameter
main numeric vector: main parameter
nuisance numeric vector: nuisance paramter
trafo matrix: transformation of the parameter

Details

If name is missing, the default ""parameter of a parametric family of probability measures"" is used. If nuisance is missing, the nuisance parameter is set to NULL. The number of columns of trafo have to be equal and the number of rows have to be not larger than the sum of the lengths of main and nuisance. If trafo is missing, no transformation to the parameter is applied; i.e., trafo is set to an identity matrix.

Value

```
Object of class "ParamFamParameter"
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

```
ParamFamParameter-class
```

```
ParamFamParameter(main = 0, nuisance = 1, trafo = diag(c(1,2)))

## The function is currently defined as
function(name, main = numeric(0), nuisance, trafo){
   if(missing(name))
        name <- "parameter of a parametric family of probability measures"
   if(missing(nuisance))
        nuisance <- NULL</pre>
```

98 ParamFamParameter-class

ParamFamParameter-class

Parameter of a parametric family of probability measures

Description

Class of the parameter of parametric families of probability measures.

Objects from the Class

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

Slots

```
main: Object of class "numeric": main parameter.
nuisance: Object of class "OptionalNumeric": optional nuisance parameter.
trafo: Object of class "matrix": transformation of the parameter.
name: Object of class "character": name of the parameter.
```

Extends

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

Methods

```
main signature(object = "ParamFamParameter"): accessor function for slot main.
main<- signature(object = "ParamFamParameter"): replacement function for slot main.
nuisance signature(object = "ParamFamParameter"): accessor function for slot nuisance.
nuisance<- signature(object = "ParamFamParameter"): replacement function for slot nuisance.
trafo signature(object = "ParamFamParameter"): accessor function for slot trafo.
trafo<- signature(object = "ParamFamParameter"): replacement function for slot trafo.
length signature(x = "ParamFamParameter"): sum of the lengths of main and nuisance.
show signature(object = "ParamFamParameter")</pre>
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

PoisFamily 99

See Also

```
Parameter-class
```

Examples

```
new("ParamFamParameter")
```

PoisFamily

Generating function for Poisson families

Description

Generates an object of class "L2ParamFamily" which represents a Poisson family.

Usage

```
PoisFamily(lambda = 1, trafo)
```

Arguments

lambda positive mean

trafo matrix: transformation of the parameter

Details

The slots of the corresponding L2 differentiable parameteric family are filled.

Value

```
Object of class "L2ParamFamily"
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
L2ParamFamily-class, Pois-class
```

```
(P1 <- PoisFamily(lambda = 4.5))
plot(P1)
FisherInfo(P1)
checkL2deriv(P1)</pre>
```

100 ProbFamily-class

ProbFamily-class

Family of probability measures

Description

Class of families of probability measures.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
name: Object of class "character": name of the family.
distribution: Object of class "Distribution": member of the family.
distrSymm: Object of class "DistributionSymmetry": symmetry of distribution.
props: Object of class "character": properties of the family.
```

Methods

```
name signature(object = "ProbFamily"): accessor function for slot name.
name<- signature(object = "ProbFamily"): replacement function for slot name.
distribution signature(object = "ProbFamily"): accessor function for slot distribution.
distrSymm signature(object = "ProbFamily"): accessor function for slot distrSymm.
props signature(object = "ProbFamily"): accessor function for slot props.
props<- signature(object = "ProbFamily"): replacement function for slot props.
addProp<- signature(object = "ProbFamily"): add a property to slot props.</pre>
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

See Also

Distribution-class

radiusMinimaxIC 101

radiusMinimaxIC	Generic function for the computation of the radius minimax IC	

Description

Generic function for the computation of the radius minimax IC.

Usage

Arguments

L2Fam	L2-differentiable family of probability measures.
neighbor	object of class "Neighborhood".
risk	object of class "RiskType".
	additional parameters.
loRad	the lower end point of the interval to be searched.
upRad	the upper end point of the interval to be searched.
z.start	initial value for the centering constant.
A.start	initial value for the standardizing matrix.
upper	upper bound for the optimal clipping bound.
maxiter	the maximum number of iterations
tol	the desired accuracy (convergence tolerance).
warn	logical: print warnings.

Value

The radius minimax IC is computed.

Methods

```
L2Fam = "L2ParamFamily", neighbor = "UncondNeighborhood", risk = "asGRisk": computation of the radius minimax IC for an L2 differentiable parametric family.
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

102 RiskType-class

References

Rieder, H., Kohl, M. and Ruckdeschel, P. (2001) The Costs of not Knowing the Radius. Submitted. Appeared as discussion paper Nr. 81. SFB 373 (Quantification and Simulation of Economic Processes), Humboldt University, Berlin; also available under www.uni-bayreuth.de/departments/math/org/mathe7/RIEDER/pubs/RR.pdf

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

radiusMinimaxIC

Examples

RiskType-class

Risk

Description

Class of risks; e.g., estimator risks.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
type: Object of class "character": type of risk.
```

Methods

```
type signature(object = "RiskType"): accessor function for slot type.
show signature(object = "RiskType")
```

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

RobModel-class 103

RobModel-class

Robust model

Description

Class of robust models. A robust model consists of family of probability measures center and a neighborhood neighbor about this family.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
center: Object of class "ProbFamily"
neighbor: Object of class "Neighborhood"
```

Methods

```
center signature(object = "RobModel"): accessor function for slot center.
center<- signature(object = "RobModel"): replacement function for slot center.
neighbor signature(object = "RobModel"): accessor function for slot neighbor.
neighbor<- signature(object = "RobModel"): replacement function for slot neighbor.</pre>
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

ProbFamily-class, Neighborhood-class

104 TotalVarIC

ROptEstOldConstants

Built-in Constants in package ROptEstOld

Description

Constants built into ROptEstOld.

Usage

EULERMASCHERONICONSTANT APERYCONSTANT

Details

ROptEstOld has a small number of built-in constants.

The following constants are available:

• EULERMASCHERONICONSTANT: the Euler Mascheroni constant

$$\gamma = -\Gamma'(1)$$

given in http://mathworld.wolfram.com/Euler-MascheroniConstant.html (48);

• APERYCONSTANT: the Apéry constant

$$\zeta(3) = \frac{5}{2} \left(\sum_{k \ge 1} \frac{(-1)^{k-1}}{k^3 \binom{2k}{k}} \right)$$

as given in http://mathworld.wolfram.com/AperysConstant.html, equation (8);

These are implemented as variables in the ROptEstOld name space taking appropriate values.

Examples

EULERMASCHERONICONSTANT APERYCONSTANT

TotalVarIC

Generating function for TotalVarIC-class

Description

Generates an object of class "TotalVarIC"; i.e., an influence curves η of the form

$$\eta = c \vee A\Lambda \wedge d$$

with lower clipping bound c, upper clipping bound d and standardizing matrix A. Λ stands for the L2 derivative of the corresponding L2 differentiable parametric family which can be created via CallL2Fam.

TotalVarIC 105

Usage

Arguments

name	object of class "character".
CallL2Fam	object of class "call": creates an object of the underlying L2-differentiable parametric family.
Curve	object of class "EuclRandVarList".
Risks	object of class "list": list of risks; cf. RiskType-class.
Infos	matrix of characters with two columns named method and message: additional informations.
clipLo	negative real: lower clipping bound.
clipUp	positive real: lower clipping bound.
stand	matrix: standardizing matrix
lowerCase	optional constant for lower case solution.

neighborRadius radius of the corresponding (unconditional) contamination neighborhood.

Value

Object of class "TotalVarIC"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
IC-class, ContIC
```

```
IC1 <- TotalVarIC()
plot(IC1)</pre>
```

106 Total VarIC-class

TotalVarIC-class

Influence curve of total variation type

Description

Class of (partial) influence curves of total variation type. i.e., an influence curves η of the form

$$\eta = c \vee A\Lambda \wedge d$$

with lower clipping bound c, upper clipping bound d and standardizing matrix A. Λ stands for the L2 derivative of the corresponding L2 differentiable parametric family which can be created via CallL2Fam.

Objects from the Class

Objects can be created by calls of the form new("TotalVarIC", ...). More frequently they are created via the generating function TotalVarIC, respectively via the method generateIC.

Slots

CallL2Fam: object of class "call": creates an object of the underlying L2-differentiable parametric family.

name: object of class "character".

Curve: object of class "EuclRandVarList".

Risks: object of class "list": list of risks; cf. RiskType-class.

Infos: object of class "matrix" with two columns named method and message: additional informations.

clipLo: object of class "numeric": lower clipping bound.

clipUp: object of class "numeric": upper clipping bound.

stand: object of class "matrix": standardizing matrix.

lowerCase object of class "OptionalNumeric": optional constant for lower case solution.

neighborRadius: object of class "numeric": radius of the corresponding (unconditional) contamination neighborhood.

Extends

```
Class "IC", directly.
Class "InfluenceCurve", by class "IC".
```

Total Var Neighborhood 107

Methods

```
CallL2Fam<- signature(object = "TotalVarIC"): replacement function for slot CallL2Fam.
clipLo signature(object = "TotalVarIC"): accessor function for slot clipLo.
clipLo<- signature(object = "TotalVarIC"): replacement function for slot clipLo.
clipUp signature(object = "TotalVarIC"): accessor function for slot clipUp.
clipUp<- signature(object = "TotalVarIC"): replacement function for slot clipUp.
stand signature(object = "TotalVarIC"): accessor function for slot stand.
stand<- signature(object = "TotalVarIC"): replacement function for slot neighborRadius.
neighborRadius signature(object = "TotalVarIC"): accessor function for slot neighborRadius.
neighborRadius<- signature(object = "TotalVarIC"): replacement function for slot neighborRadius.
generateIC signature(neighbor = "TotalVarIC"): replacement function for slot neighborRadius.
generateIC signature(neighbor = "TotalVarIC"): Rarely called directly.
show signature(object = "TotalVarIC")</pre>
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
IC-class, ContIC
```

Examples

```
IC1 <- new("TotalVarIC")
plot(IC1)</pre>
```

 ${\tt TotalVarNeighborhood-} {\it Cenerating function for TotalVarNeighborhood-} {\it class}$

Description

Generates an object of class "TotalVarNeighborhood".

Usage

```
TotalVarNeighborhood(radius = 0)
```

Arguments

radius non-negative real: neighborhood radius.

Value

Object of class "ContNeighborhood"

Author(s)

Matthias Kohl < Matthias . Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

TotalVarNeighborhood-class

Examples

```
TotalVarNeighborhood()
## The function is currently defined as
function(radius = 0){
    new("TotalVarNeighborhood", radius = radius)
}
```

TotalVarNeighborhood-class

Total variation neighborhood

Description

Class of (unconditional) total variation neighborhoods.

Objects from the Class

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

Slots

```
type: Object of class "character": "(uncond.) total variation neighborhood". radius: Object of class "numeric": neighborhood radius.
```

trAsCov 109

Extends

```
Class "UncondNeighborhood", directly.
Class "Neighborhood", by class "UncondNeighborhood".
```

Methods

No methods defined with class "TotalVarNeighborhood" in the signature.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) Numerical Contributions to the Asymptotic Theory of Robustness. Bayreuth: Dissertation.

See Also

TotalVarNeighborhood, UncondNeighborhood-class

Examples

```
new("TotalVarNeighborhood")
```

trAsCov

Generating function for trAsCov-class

Description

Generates an object of class "trAsCov".

Usage

trAsCov()

Value

Object of class "trAsCov"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

110 trAsCov-class

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

```
trAsCov-class
```

Examples

```
trAsCov()
## The function is currently defined as
function(){ new("trAsCov") }
```

trAsCov-class

Trace of asymptotic covariance

Description

Class of trace of asymptotic covariance.

Objects from the Class

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

Slots

```
type: Object of class "character": "trace of asymptotic covariance".
```

Extends

```
Class "asRisk", directly.
Class "RiskType", by class "asRisk".
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

trFiCov 111

See Also

```
asRisk-class, trAsCov
```

Examples

```
new("trAsCov")
```

trFiCov

Generating function for trFiCov-class

Description

Generates an object of class "trFiCov".

Usage

```
trFiCov()
```

Value

Object of class "trFiCov"

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
trFiCov-class
```

Examples

```
trFiCov()
## The function is currently defined as
function(){ new("trFiCov") }
```

112 trFiCov-class

trFiCov-class

Trace of finite-sample covariance

Description

Class of trace of finite-sample covariance.

Objects from the Class

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

Slots

type: Object of class "character": "trace of finite-sample covariance".

Extends

```
Class "fiRisk", directly.
Class "RiskType", by class "fiRisk".
```

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Ruckdeschel, P. and Kohl, M. (2005) How to approximate the finite sample risk of M-estimators.

See Also

```
fiRisk-class, trFiCov
```

Examples

```
new("trFiCov")
```

UncondNeighborhood-class

Unconditional neighborhood

Description

Class of unconditonal (errors-in-variables) neighborhoods.

Objects from the Class

A virtual Class: No objects may be created from it.

Slots

```
type: Object of class "character": type of the neighborhood. radius: Object of class "numeric": neighborhood radius.
```

Extends

Class "Neighborhood", directly.

Author(s)

Matthias Kohl <Matthias.Kohl@stamats.de>

References

Rieder, H. (1994) Robust Asymptotic Statistics. New York: Springer.

Kohl, M. (2005) *Numerical Contributions to the Asymptotic Theory of Robustness*. Bayreuth: Dissertation.

See Also

Neighborhood-class

Index

*Topic classes	Gumbel-class, 61
asBias-class, 5	GumbelParameter-class, 65
asCov-class, 6	ParamFamily, 94
asGRisk-class,7	*Topic models
asHampel-class, 9	BinomFamily, 15
asMSE-class, 11	ContNeighborhood, 21
asRisk-class, 12	ContNeighborhood-class, 22
asUnOvShoot-class, 14	ExpScaleFamily, 25
ContIC-class, 19	FixRobModel, 35
ContNeighborhood-class, 22	FixRobModel-class, 36
EvenSymmetric-class, 24	GammaFamily, 39
fiBias-class, 27	GumbelLocationFamily, 64
fiCov-class, 28	GumbelParameter-class, 65
fiHampel-class, 30	InfRobModel, 72
fiMSE-class, 32	InfRobModel-class, 73
fiRisk-class, 33	L2ParamFamily, 76
fiUnOvShoot-class, 34	L2ParamFamily-class, 77
FixRobModel-class, 36	LnormScaleFamily, 80
FunctionSymmetry-class, 37	Neighborhood-class, 83
FunSymmList-class, 39	NormLocationFamily, 85
IC-class, 67	NormLocationScaleFamily, 86
InfluenceCurve-class, 70	NormScaleFamily, 87
InfRobModel-class, 73	ParamFamily-class, 95
L2ParamFamily-class, 77	PoisFamily, 99
Neighborhood-class, 83	ProbFamily-class, 100
NonSymmetric-class, 84	RobModel-class, 103
OddSymmetric-class, 89	TotalVarNeighborhood, 107
ParamFamily-class, 95	TotalVarNeighborhood-class, 108
ParamFamParameter-class, 98	UncondNeighborhood-class, 113
ProbFamily-class, 100	*Topic robust
RiskType-class, 102	asBias, 4
RobModel-class, 103	asCov, 6
TotalVarIC-class, 106	asHampel, 8
TotalVarNeighborhood-class, 108	asMSE, 10
trAsCov-class, 110	asUnOvShoot, 13
trFiCov-class, 112	checkIC, 16
UncondNeighborhood-class, 113	checkL2deriv, 17
*Topic distribution	ContIC, 18
Gumbel, 60	evalIC, 23

EvenSymmetric, 24	addProp<-,ProbFamily-method
fiBias, 26	(ProbFamily-class), 100
fiCov, 28	addRisk<- (InfluenceCurve-class), 70
fiHampel, 29	addRisk<-,InfluenceCurve-method
fiMSE, 31	(InfluenceCurve-class), 70
fiUnOvShoot, 33	APERYCONSTANT (ROptEstOldConstants), 104
FunSymmList, 38	asBias, 4, 5
	asBias-class, 5
generateIC, 40	asCov, 6, 7
getAsRisk, 41	
getFiRisk, 44	asCov-class, 6
getFixClip, 45	asGRisk-class,7
getFixRobIC, 46	asHampel, 8, 10
getIneffDiff,47	asHampel-class, 9
getInfCent, 49	asMSE, 10, <i>11</i>
<pre>getInfClip, 50</pre>	asMSE-class, 11
getInfGamma, 52	asRisk-class, 12
getInfRobIC, 53	asUnOvShoot, 13
getInfStand, 56	asUnOvShoot-class, 14
getRiskIC, 57	
IC, 66	BinomFamily, 15
InfluenceCurve, 68	bound (asHampel-class), 9
infoPlot, 71	<pre>bound,asHampel-method(asHampel-class),</pre>
ksEstimator, 74	9
leastFavorableRadius, 79	bound, fiHampel-method (fiHampel-class),
	30
locMEstimator, 81	
lowerCaseRadius, 82	CallL2Fam(IC-class), 67
NonSymmetric, 84	CallL2Fam, IC-method (IC-class), 67
OddSymmetric, 88	CallL2Fam<- (IC-class), 67
oneStepEstimator, 89	CallL2Fam<-,ContIC-method
optIC, 90	(ContIC-class), 19
optRisk, 92	CallL2Fam<-, IC-method (IC-class), 67
ParamFamParameter, 97	CallL2Fam<-,TotalVarIC-method
radiusMinimaxIC, 101	(TotalVarIC-class), 106
RobModel-class, 103	cent (ContIC-class), 19
TotalVarIC, 104	cent, ContIC-method (ContIC-class), 19
trAsCov, 109	cent<- (ContIC-class), 19
trFiCov, 111	cent<-,ContIC-method (ContIC-class), 19
*Topic sysdata	center (RobModel-class), 103
ROptEstOldConstants, 104	center, RobModel-method
*, Gumbel, numeric-method (Gumbel-class),	(RobModel-class), 103
61	center<- (RobModel-class), 103
+, Gumbel, numeric-method (Gumbel-class),	, , , , , , , , , , , , , , , , , , , ,
61	center<-,RobModel-method
O1	(RobModel-class), 103
addra Cox (Ta Classical Control Contro	checkIC, 16
addInfo<- (InfluenceCurve-class), 70	checkIC,IC,L2ParamFamily-method
addInfo<-,InfluenceCurve-method	(IC-class), 67
(InfluenceCurve-class), 70	checkIC,IC,missing-method(IC-class),67
addProp<-(ProbFamily-class), 100	checkL2deriv, 17

checkL2deriv,L2ParamFamily-method	evalIC,IC,numeric-method(IC-class),67
(L2ParamFamily-class), 77	EvenSymmetric, 24, 25
clip (ContIC-class), 19	EvenSymmetric-class, 24
<pre>clip,ContIC-method(ContIC-class), 19</pre>	ExpScaleFamily, 25
<pre>clip<- (ContIC-class), 19</pre>	
<pre>clip<-,ContIC-method(ContIC-class), 19</pre>	fiBias, 26, 27
clipLo(TotalVarIC-class), 106	fiBias-class, 27
clipLo,TotalVarIC-method	fiCov, 28, 29
(TotalVarIC-class), 106	fiCov-class, 28
clipLo<- (TotalVarIC-class), 106	fiHampel, 29, 31
clipLo<-,TotalVarIC-method	fiHampel-class, 30
(TotalVarIC-class), 106	fiMSE, 31, 32
clipUp(TotalVarIC-class), 106	fiMSE-class, 32
clipUp,TotalVarIC-method	fiRisk-class, 33
(TotalVarIC-class), 106	FisherInfo (L2ParamFamily-class), 77
clipUp<-(TotalVarIC-class), 106	FisherInfo,L2ParamFamily-method
clipUp<-,TotalVarIC-method	(L2ParamFamily-class), 77
(TotalVarIC-class), 106	fiUnOvShoot, 33
ContIC, 18, 19, 20, 105, 107	fiUnOvShoot-class, 34
ContIC-class, 19	FixRobModel, 35, 37
ContNeighborhood, 21, 22	FixRobModel-class, 36
ContNeighborhood-class, 22	FunctionSymmetry-class, 37
Curve (InfluenceCurve-class), 70	FunSymmList, 38
Curve, InfluenceCurve-method	FunSymmList-class, 39
(InfluenceCurve-class), 70	Turisymmetric Class, 37
distribution (ProbFamily-class), 100	GammaFamily, 39
distribution, ProbFamily-method	generateIC, 40
(ProbFamily-class), 100	<pre>generateIC,ContNeighborhood,L2ParamFamily-method</pre>
distrSymm (ProbFamily-class), 100	(ContIC-class), 19
distrSymm, ProbFamily-method	<pre>generateIC,TotalVarNeighborhood,L2ParamFamily-method</pre>
(ProbFamily-class), 100	(TotalVarIC-class), 106
Domain, InfluenceCurve-method	getAsRisk, 41
(InfluenceCurve-class), 70	<pre>getAsRisk,asBias,RealRandVariable,ContNeighborhood-method</pre>
(IIII Idenceed) ve eldss), 70	(getAsRisk), 41
E (Gumbel-class), 61	getAsRisk,asBias,UnivariateDistribution,ContNeighborhood-m
E,Gumbel,missing,missing-method	(getAsRisk), 41
(Gumbel-class), 61	getAsRisk,asBias,UnivariateDistribution,TotalVarNeighborho
E,L2ParamFamily,EuclRandMatrix,missing-metho	
(L2ParamFamily-class), 77	<pre>getAsRisk,asCov,RealRandVariable,ContNeighborhood-method</pre>
E,L2ParamFamily,EuclRandVariable,missing-met	
(L2ParamFamily-class), 77	getAsRisk,asCov,UnivariateDistribution,ContNeighborhood-me
E,L2ParamFamily,EuclRandVarList,missing-meth	
(L2ParamFamily-class), 77	getAsRisk,asCov,UnivariateDistribution,TotalVarNeighborhoo
E-methods (Gumbel-class), 61	(getAsRisk), 41
EULERMASCHERONICONSTANT	getAsRisk,asMSE,EuclRandVariable,Neighborhood-method
(ROptEstOldConstants), 104	(getAsRisk), 41
evalIC, 23	getAsRisk,asMSE,UnivariateDistribution,Neighborhood-method
evalIC, IC, matrix-method (IC-class), 67	(getAsRisk), 41

```
getAsRisk,asUnOvShoot,UnivariateDistribution, detDmdQamingabdminvandimetDbstribution,asGRisk,TotalVarNeighbo
                         (getAsRisk), 41
                                                                                                                                                                             (getInfGamma), 52
getAsRisk,trAsCov,RealRandVariable,ContNeighbgetboofdGamentapddnivariateDistribution,asMSE,ContNeighborhood-
                         (getAsRisk), 41
                                                                                                                                                                              (getInfGamma), 52
getAsRisk,trAsCov,UnivariateDistribution,UncogetNEnig@abomapdodimenthatdeDistribution,asUnOvShoot,ContNeighbo
                         (getAsRisk), 41
                                                                                                                                                                             (getInfGamma), 52
                                                                                                                                                   getInfGamma-methods (getInfGamma), 52
getAsRisk-methods (getAsRisk), 41
getFiRisk, 44
                                                                                                                                                   getInfRobIC, 53
getFiRisk,fiUnOvShoot,Norm,ContNeighborhood-mgetDodfRobIC,RealRandVariable,asBias,ContNeighborhood-metho
                         (getFiRisk), 44
                                                                                                                                                                              (getInfRobIC), 53
getFiRisk, fiUnOvShoot, Norm, Total Var Neighborhog \textit{etimefRob} IC, Real Rand Variable, as Cov, Cont Neighborhood-method and the first of the contraction of the c
                         (getFiRisk), 44
                                                                                                                                                                             (getInfRobIC), 53
getFiRisk-methods (getFiRisk), 44
                                                                                                                                                   getInfRobIC, RealRandVariable, asGRisk, ContNeighborhood-meth
getFixClip, 45
                                                                                                                                                                             (getInfRobIC), 53
getFixClip,numeric,Norm,fiUnOvShoot,ContNeighbethoo6RobmEthRedalRandVariable,asHampel,ContNeighborhood-met
                         (getFixClip), 45
                                                                                                                                                                             (getInfRobIC), 53
getFixClip,numeric,Norm,fiUnOvShoot,TotalVarNgaghb6RdbDd,bwethadiateDistribution,asBias,ContNeighborhood
                         (getFixClip), 45
                                                                                                                                                                             (getInfRobIC), 53
getFixClip-methods (getFixClip), 45
                                                                                                                                                   getInfRobIC,UnivariateDistribution,asBias,TotalVarNeighbor
getFixRobIC, 46
                                                                                                                                                                             (getInfRobIC), 53
getFixRobIC,Norm,fiUnOvShoot,UncondNeighborhogetImetRodUC,UnivariateDistribution,asCov,ContNeighborhood-
                                                                                                                                                                             (getInfRobIC), 53
                         (getFixRobIC), 46
getFixRobIC-methods (getFixRobIC), 46
                                                                                                                                                   \tt getInfRobIC, UnivariateDistribution, asCov, Total Var Neighborh Start Star
getIneffDiff, 47
                                                                                                                                                                              (getInfRobIC), 53
getIneffDiff,numeric,L2ParamFamily,UncondNeighboIhb@dhI@MSEimenthadeDistribution,asGRisk,UncondNeighborh
                                                                                                                                                                             (getInfRobIC), 53
                         (getIneffDiff), 47
                                                                                                                                                   getInfRobIC, UnivariateDistribution, asHampel, UncondNeighbor
getIneffDiff-methods (getIneffDiff), 47
                                                                                                                                                                             (getInfRobIC), 53
getInfCent, 49
\texttt{getInfCent}, \texttt{RealRandVariable}, \texttt{ContNeighborhood-} \textbf{\textit{getInoC}} \texttt{RobIC}, \texttt{UnivariateDistribution}, \texttt{asUnOvShoot}, \texttt{UncondNeighborhood-} \textbf{\textit{getInoC}} \texttt{RobIC}, \texttt{UnivariateDistribution}, \texttt{asUnOvShoot}, \texttt{UncondNeighborhood-} \textbf{\textit{getInoC}} \texttt{RobIC}, \texttt{UnivariateDistribution}, \texttt{asUnOvShoot}, \texttt{UncondNeighborhood-} \textbf{\textit{getInoC}} \texttt{NobIC}, \texttt{UnivariateDistribution}, \texttt{uncondNeighborhood-} \texttt{uncondNeighborhood-} \texttt{uncondNeighborhood-} \textbf{\textit{getInoC}} \texttt{uncondNeighborhood-} \texttt{un
                                                                                                                                                                              (getInfRobIC), 53
                          (getInfCent), 49
getInfCent,UnivariateDistribution,ContNeighboghbbdfRetthGdmethods(getInfRobIC),53
                                                                                                                                                   getInfStand, 56
getInfCent,UnivariateDistribution,TotalVarNeigebomfisbalndeRealRandVariable,ContNeighborhood-method
                         (getInfCent), 49
                                                                                                                                                                             (getInfStand), 56
                                                                                                                                                   getInfStand, UnivariateDistribution, ContNeighborhood-method
getInfCent-methods (getInfCent), 49
getInfClip, 50
                                                                                                                                                                              (getInfStand), 56
getInfClip, numeric, EuclRandVariable, asMSE, CongeleIgfiSbænholpdnimeathioalteDistribution, TotalVarNeighborhood-me
                                                                                                                                                                              (getInfStand), 56
                         (getInfClip), 50
getInfClip, numeric, UnivariateDistribution, asMSEt, Dofistarighthethoods (getEnotStand), 56
                         (getInfClip), 50
                                                                                                                                                   getRiskIC, 57
getInfClip, numeric, UnivariateDistribution, asMSEtRoskalVarOeagBbashOodtNetblodorhood, L2ParamFamily-method
                         (getInfClip), 50
                                                                                                                                                                             (getRiskIC), 57
getInfClip,numeric,UnivariateDistribution,asUg@t/ShskIC,UnicoasBeagh@omtNedghledhlodod,missing-method
                         (getInfClip), 50
                                                                                                                                                                             (getRiskIC), 57
getInfClip-methods (getInfClip), 50
                                                                                                                                                   getRiskIC,IC,asBias,TotalVarNeighborhood,L2ParamFamily-met
getInfGamma, 52
                                                                                                                                                                             (getRiskIC), 57
getInfGamma,RealRandVariable,asMSE,ContNeighbgetNRibskin@thGdasBias,TotalVarNeighborhood,missing-method
```

(getRiskIC), 57

(getInfGamma), 52

<pre>getRiskIC,IC,asCov,missing,L2ParamFamily-met</pre>	h kos Estimator, numeric, Gumbel-method
(getRiskIC), 57	(ksEstimator), 74
<pre>getRiskIC,IC,asCov,missing,missing-method</pre>	ksEstimator, numeric, Lnorm-method
(getRiskIC), 57	(ksEstimator), 74
<pre>getRiskIC,IC,asMSE,UncondNeighborhood,L2Para</pre>	mkamiliymanedhoolumeric,Norm-method
(getRiskIC), 57	(ksEstimator), 74
<pre>getRiskIC,IC,asMSE,UncondNeighborhood,missin</pre>	gksÆstbiondator, numeric, Pois-method
(getRiskIC), 57	(ksEstimator), 74
<pre>getRiskIC,IC,fiUnOvShoot,ContNeighborhood,mi</pre>	sksingtimatood-methods (ksEstimator), 74
(getRiskIC), 57	kurtosis (Gumbel-class), 61
<pre>getRiskIC,IC,fiUnOvShoot,TotalVarNeighborhoo</pre>	dkunitssing, Guanthed - method (Gumbel-class),
(getRiskIC), 57	61
<pre>getRiskIC,IC,trAsCov,missing,L2ParamFamily-m</pre>	ekthotosis-methods (Gumbel-class), 61
(getRiskIC), 57	
<pre>getRiskIC,IC,trAsCov,missing,missing-method</pre>	L2deriv(L2ParamFamily-class),77
(getRiskIC), 57	L2deriv,L2ParamFamily-method
${\tt getRiskIC}, {\tt TotalVarIC}, {\tt asUnOvShoot}, {\tt UncondNeighbord}$	borhood, missing method, -class) 77
(getRiskIC), 57	L2derivDistr (L2ParamFamily-class), 77
<pre>getRiskIC-methods (getRiskIC), 57</pre>	L2derivDistr,L2ParamFamily-method
Gumbel, 60	(L2ParamFamily-class), 77
Gumbel-class, 61	L2derivDistrSymm (L2ParamFamily-class),
GumbelLocationFamily, 64	77
GumbelParameter-class, 65	L2derivDistrSymm,L2ParamFamily-method
,	(L2ParamFamily-class), 77
IC, 66, 68	L2derivSymm (L2ParamFamily-class), 77
IC-class, 67	L2derivSymm,L2ParamFamily-method
InfluenceCurve, 68, 71	(L2ParamFamily-class), 77
InfluenceCurve-class, 70	L2ParamFamily, 76, 78
infoPlot, 71	
infoPlot,IC-method(IC-class),67	L2ParamFamily-class, 77
Infos (InfluenceCurve-class), 70	leastFavorableRadius, 48, 79
Infos, InfluenceCurve-method	leastFavorableRadius,L2ParamFamily,UncondNeighborhood,asGF
(InfluenceCurve-class), 70	(leastFavorableRadius), 79
	leastFavorableRadius-methods
Infos<- (InfluenceCurve-class), 70	(leastFavorableRadius), 79
Infos<-, InfluenceCurve-method	length, ParamFamParameter-method
(InfluenceCurve-class), 70	(ParamFamParameter-class), 98
InfRobModel, 72, 73	LnormScaleFamily, 80
InfRobModel-class, 73	loc (GumbelParameter-class), 65
initialize, Gumbel-method	loc, Gumbel-method (Gumbel-class), 61
(Gumbel-class), 61	loc,GumbelParameter-method
	(GumbelParameter-class), 65
ksEstimator, 74	loc<- (GumbelParameter-class), 65
ksEstimator,numeric,Binom-method	<pre>loc<-,Gumbel-method(Gumbel-class),61</pre>
(ksEstimator), 74	loc<-,GumbelParameter-method
ksEstimator,numeric,Exp-method	(GumbelParameter-class), 65
(ksEstimator), 74	locMEstimator, 81
ksEstimator,numeric,Gammad-method	<pre>locMEstimator,numeric,InfluenceCurve-method</pre>
(ksEstimator), 74	(locMEstimator), 81

<pre>locMEstimator-methods (locMEstimator),</pre>	neighbor<-,RobModel-method
81	(RobModel-class), 103
lowerCase (ContIC-class), 19	Neighborhood-class, 83
<pre>lowerCase,ContIC-method(ContIC-class),</pre>	neighborRadius(ContIC-class), 19
19	neighborRadius,ContIC-method
<pre>lowerCase,TotalVarIC-method</pre>	(ContIC-class), 19
(TotalVarIC-class), 106	neighborRadius,TotalVarIC-method
<pre>lowerCase<- (ContIC-class), 19</pre>	(TotalVarIC-class), 106
lowerCase<-,ContIC-method	neighborRadius<- (ContIC-class), 19
(ContIC-class), 19	neighborRadius<-,ContIC-method
lowerCase<-,TotalVarIC-method	(ContIC-class), 19
(TotalVarIC-class), 106	neighborRadius<-,TotalVarIC-method
lowerCaseRadius, 82	(TotalVarIC-class), 106
lowerCaseRadius,L2ParamFamily,ContNeighbo	
(lowerCaseRadius), 82	NonSymmetric-class, 84
lowerCaseRadius,L2ParamFamily,TotalVarNei	
(lowerCaseRadius), 82	NormLocationScaleFamily, 86
lowerCaseRadius-methods	NormScaleFamily, 87
	nuisance (ParamFamParameter-class), 98
(lowerCaseRadius), 82	
main (ParamFamParameter-class), 98	nuisance, ParamFamily-method
main, ParamFamily-method	(ParamFamily-class), 95
•	nuisance, ParamFamParameter-method
(ParamFamily-class), 95	(ParamFamParameter-class), 98
main, ParamFamParameter-method	nuisance<- (ParamFamParameter-class), 98
(ParamFamParameter-class), 98	nuisance<-,ParamFamParameter-method
main<- (ParamFamParameter-class), 98	(ParamFamParameter-class), 98
main<-,ParamFamParameter-method	0.110
(ParamFamParameter-class), 98	OddSymmetric, 88, 89
Map,InfluenceCurve-method	OddSymmetric-class, 89
(InfluenceCurve-class), 70	oneStepEstimator, 89
	one Step Estimator, matrix, Influence Curve, list-method
name,InfluenceCurve-method	(oneStepEstimator), 89
(InfluenceCurve-class), 70	$one Step Estimator, \verb matrix , Influence Curve , \verb numeric-method $
name,ProbFamily-method	(oneStepEstimator), 89
(ProbFamily-class), 100	one Step Estimator, numeric, Influence Curve, list-method
<pre>name,RobModel-method(RobModel-class),</pre>	(oneStepEstimator), 89
103	oneStepEstimator,numeric,InfluenceCurve,numeric-method
name<-,InfluenceCurve-method	(oneStepEstimator), 89
(InfluenceCurve-class), 70	oneStepEstimator-methods
name<-,ProbFamily-method	(oneStepEstimator), 89
(ProbFamily-class), 100	optIC, 90
neighbor (RobModel-class), 103	optIC,FixRobModel,fiUnOvShoot-method
neighbor,RobModel-method	(optIC), 90
(RobModel-class), 103	optIC,InfRobModel,asRisk-method
neighbor<- (RobModel-class), 103	(optIC), 90
neighbor<-,FixRobModel-method	optIC,InfRobModel,asUnOvShoot-method
(FixRobModel-class), 36	(optIC), 90
neighbor<-,InfRobModel-method	optIC,L2ParamFamily,asCov-method
(InfRobModel-class), 73	(optIC), 90
(IIII KODPIOUET CLASS), /3	(Optic), 30

optIC-methods (optIC), 90	ROptEstOldConstants, 104
optRisk, 92	1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
<pre>optRisk,FixRobModel,fiUnOvShoot-method (optRisk), 92</pre>	<pre>scale,Gumbel-method(Gumbel-class),61 scale,GumbelParameter-method</pre>
optRisk,InfRobModel,asRisk-method	(GumbelParameter-class), 65
(optRisk), 92	scale<-, Gumbel-method (Gumbel-class), 61
optRisk,L2ParamFamily,asCov-method	scale<-,GumbelParameter-method
(optRisk), 92	(GumbelParameter-class), 65
optRisk-methods (optRisk), 92	show, asHampel-method (asHampel-class), 9
()	show, asUnOvShoot-method
param, ParamFamily-method	(asUnOvShoot-class), 14
(ParamFamily-class), 95	<pre>show, ContIC-method (ContIC-class), 19</pre>
ParamFamily, 94	<pre>show, fiHampel-method (fiHampel-class),</pre>
ParamFamily-class, 95	30
ParamFamParameter, 76, 95, 97	show,fiUnOvShoot-method
ParamFamParameter-class, 98	(fiUnOvShoot-class), 34
plot, IC, ANY-method (IC-class), 67	show,FixRobModel-method
plot,L2ParamFamily,ANY-method	(FixRobModel-class), 36
(L2ParamFamily-class), 77	show, IC-method (IC-class), 67
plot, ParamFamily, ANY-method	show, InfluenceCurve-method
(ParamFamily-class), 95	(InfluenceCurve-class), 70
PoisFamily, 99	show, InfRobModel-method
ProbFamily-class, 100	(InfRobModel-class), 73
props (ProbFamily-class), 100	show, Neighborhood-method
props, ProbFamily-method	(Neighborhood-class), 83
(ProbFamily-class), 100	show,ParamFamily-method
<pre>props<- (ProbFamily-class), 100</pre>	(ParamFamily-class), 95
<pre>props<-,ProbFamily-method</pre>	show, ParamFamParameter-method
(ProbFamily-class), 100	(ParamFamParameter-class), 98
	<pre>show,RiskType-method(RiskType-class),</pre>
radius (Neighborhood-class), 83	102
radius,Neighborhood-method	show,TotalVarIC-method
(Neighborhood-class), 83	(TotalVarIC-class), 106
radiusMinimaxIC, 48, 80, 101, 102	skewness (Gumbel-class), 61
$radius {\tt MinimaxIC}, {\tt L2ParamFamily}, {\tt UncondNeighbord}, {\tt Uncon$	^h srewaes B iskmmelhae thod(Gumbel-class),
(radiusMinimaxIC), 101	61
radiusMinimaxIC-methods	skewness-methods (Gumbel-class), 61
(radiusMinimaxIC), 101	stand (ContIC-class), 19
Range, InfluenceCurve-method	stand, ContIC-method (ContIC-class), 19
(InfluenceCurve-class), 70	stand,TotalVarIC-method
rgumbel, <i>61</i> , <i>63</i>	(TotalVarIC-class), 106
Risks (InfluenceCurve-class), 70	stand<- (ContIC-class), 19
Risks,InfluenceCurve-method	stand<-,ContIC-method(ContIC-class), 19
(InfluenceCurve-class), 70	stand<-,TotalVarIC-method
Risks<- (InfluenceCurve-class), 70	(TotalVarIC-class), 106
Risks<-,InfluenceCurve-method	
(InfluenceCurve-class), 70	TotalVarIC, 104
RiskType-class, 102	TotalVarIC-class, 106
RobModel-class, 103	TotalVarNeighborhood, 107, 109

```
TotalVarNeighborhood-class, 108
trafo (ParamFamParameter-class), 98
trafo, ParamFamily-method
        (ParamFamily-class), 95
trafo, ParamFamParameter-method
        (ParamFamParameter-class), 98
trafo<- (ParamFamParameter-class), 98</pre>
trafo<-,ParamFamParameter-method</pre>
        (ParamFamParameter-class), 98
trAsCov, 109, 111
trAsCov-class, 110
trFiCov, 111, 112
trFiCov-class, 112
type, Neighborhood-method
        (Neighborhood-class), 83
type,RiskType-method (RiskType-class),
        102
UncondNeighborhood-class, 113
var (Gumbel-class), 61
var, Gumbel-method (Gumbel-class), 61
var-methods (Gumbel-class), 61
width (asUnOvShoot-class), 14
width,asUnOvShoot-method
        (asUnOvShoot-class), 14
width,fiUnOvShoot-method
        (fiUnOvShoot-class), 34
```