Package 'sse'

July 2, 2014

Type Package

Title Sample size estimation

Version 0.5-1

Date 2014-05-10

Author Thomas Fabbro

Maintainer Thomas Fabbro <thomas.fabbro@unibas.ch>

Description The sse package takes user-defined power functions, evaluates them for a parameter range, and draws a sensitivity plot. It also provides a resampling procedure for semi-parametric sample size estimation and methods for adding informations to a Sweave report.

License GPL-3

LazyLoad yes

Depends R (>= 2.5.0), methods, grid, lattice, graphics, base, stats

Repository CRAN

Repository/R-Forge/Project power

Repository/R-Forge/Revision 12

Repository/R-Forge/DateTimeStamp 2014-05-14 07:39:06

Date/Publication 2014-05-19 09:59:41

NeedsCompilation no

R topics documented:

`exDat-methods` *Methods for Function exDat*

Description

Methods for function `exDat`

Methods

describe this method here

`inspect-methods` *Methods for Function inspect*

Description

Methods for function `inspect`

Methods

describe this method here

n *Extracting the actual n*

Description

Extracting the actual n from a powPar-object. This method is needed within the function (referred to as power-function) needed for evaluating the power.

Usage

```
n(object, ...)
```

Arguments

object	An object of class powPar.
...	.

Details

During the evaluation process every n, from a sequence defined in the powPar-object, needs to be evaluated. This function extracts the actual n that varies during the evaluation process. When a powPar-object is created, the first element of n is also set to be the actual. This allows to use this method also outside the evaluation for testing the power function.

Value

An integer.

Note

Do not use the method pp inside the power-function because this would extract the whole sequence of n.

Examples

```
psi <- powPar(muA = 0,
              muB = 1,
              sA = 1,
              sB = round(seq(from = 0.5, to = 1.5, by = 0.05), 2),
              n = round(seq(from = 10, to = 30, by = 2),0),
              theta.name = "sB")
## extracting all elements of psi individually
n(psi)
## extracting all elements of psi
pp(psi, name = "n")
## an example of usage
powFun <- function(psi){
  power.t.test(n = n(psi),
               delta = pp(psi, "muA") - pp(psi, "muB"),
```

```
                sd = theta(psi)
                )$power
}
## testing the power-function
powFun(psi)
```

---

plot-methods *Plot functions*

Description

A sensitivity plot for the sample size calcualtion indicating, using a contour for a given power, how sample size changes if `theta` is varied.

Usage

```
## S4 method for signature 'power,missing'
plot(x, y, ...)
```

Arguments

`x`	The object of class `power` used for plotting
`y`	Not used
`...`	additional arguments implemented:

- `at` numeric vector giving breakpoints along the power range. Contours (if any) will be drawn at these power. The first number indicates, which contour should be emphasised.
- `smooth` a logical that indicates if the countour should be smoothed. If TRUE a span of 0.5 will be used alternatively the argument smooth can also take a numeric value between 0 and 1 that will be used for smoothing. See the documentation of the function `loess` for details.
- `example` a logical indicating if an example should be drawn or not. An example is an arrow that points from the particular `theta` on the x-axis to the contour line and to the sample size on the y-axis.

Methods

`signature(x = "ANY", y = "ANY")` to follow

`signature(x = "power", y = "missing")` Generates a contourplot with theta on the x-axis and n on the y-axis and the contours for the power indicated with the argument at.

powCalc *Power calculation*

Description

The power for the whole range of theta, xi and n, as specified in the powPar-object, is calculated using the user-defined power-function.

Usage

```
powCalc(object, statistic, ...)
```

Arguments

object	An object of class powPar.
statistic	A function that takes an object of class powPar as argument. Ideally this is also the only argument. The function should return a vector of numeric values or a vector of logicals, depending on the type. See Details.
...	• typeA character string specifying the type of power calculation. There are two forms implemented: '"power"' for power-functions that return a numeric value, the power and '"resample"' for power-functions that return logicals. In the latter case the argument n.iter specifies how often the power-function is evaluated for estimating the number of successes, the power. • n.iterA number specifying how often the power-function is evaluated. • clusterdocumentation follows

Details

none

Value

An object of class calc.

See Also

The function merge will, together with an object of class powEx form an object of class power

Examples

```
## no example so far
```

powCalc-class *Class "powCalc"*

Description

A class for storing information of the power calculation. It inherits from "powPar", that contains the information for the calculation.

Objects from the Class

Objects can be created by calling the creator "powCalc"

Slots

core: Object of class "array" with 4 dimentions, n, theta, xi and endpoint

statistic: Object of class "function", this function was used for calculating the power

endpoint.name: Object of class "character" names of the endpoints

n.iter: Object of class "integer" how many iterations were used to estimate the power

list: Object of class "list" from class "powPar"

theta: Object of class "numeric" from class "powPar"

theta.name: Object of class "character" from class "powPar"

theta.act: Object of class "numeric" from class "powPar"

xi: Object of class "numeric" from class "powPar"

xi.name: Object of class "character" from class "powPar"

xi.act: Object of class "numeric" from class "powPar"

n: Object of class "integer" from class "powPar"

n.act: Object of class "integer" from class "powPar"

Extends

Class "Resample", directly. Class "powPar", directly.

Methods

merge `signature(x = "powCalc", y = "powEx")`: ...

Note

none

Author(s)

<thomas.fabbro@unibas.ch>

References

none

Examples

```
showClass("powCalc")
```

`powCalc-methods` *Methods for Function powCalc*

Description

Methods for function `powCalc`

Methods

describe this method here

`power-class` *Class "power"*

Description

follows

Objects from the Class

are ..

Slots

`core`: Object of class `"array"` ~~

`endpoint.name`: Object of class `"character"` ~~

`n.iter`: Object of class `"integer"` ~~

`list`: Object of class `"list"` ~~

`theta`: Object of class `"numeric"` ~~

`theta.name`: Object of class `"character"` ~~

`theta.act`: Object of class `"numeric"` ~~

`xi`: Object of class `"numeric"` ~~

`xi.name`: Object of class `"character"` ~~

`xi.act`: Object of class `"numeric"` ~~

`n`: Object of class `"integer"` ~~

`n.act`: Object of class "integer" ~~

`theta.example`: Object of class "numeric" ~~

`xi.example`: Object of class "numeric" ~~

`endpoint.example`: Object of class "character" ~~

`power.example`: Object of class "numeric" ~~

`drop`: Object of class "numeric" ~~

Extends

Class "powCalc", directly. Class "powEx", directly. Class "Resample", by class "powCalc", distance 2. Class "powPar", by class "powCalc", distance 2.

Methods

plot `signature(x = "power", y = "missing")`: ...

show `signature(object = "power")`: ...

Author(s)

<thomas.fabbro@unibas.ch>

Examples

```
showClass("power")
```

`powEx` *Constructing an object of class powEx.*

Description

A function for constructing an object of class `powEx` used for drawing an example in a sensitivity plot and for evaluating sample size.

Usage

```
powEx(theta, xi = NA, endpoint = NA, power = 0.9, drop = 0,
method = c("default", "lm", "step"), lm.range = 0.2, forceDivisor = FALSE)
```

Arguments

`theta`	a numeric value indicating for which `theta` to draw the example in the sensitivity plot and evaluate sample size. It makes only sense to use a `theta` in the range evaluated.
`xi`	a numeric value, as `theta` but for `xi`
`endpoint`	Object of class `character`, indidating for which endpoint sample size should be evaluated

power — Object of class numeric, indicating for what power samle size should be evaluated

method — Defining the method how the sample size for the indicated theta, xi, and power is "estimated". If the power object was created using resampling the "default" evaluates to "lm", otherwise to "step". For method "lm" a linear model is fit as lm(sample.size ~ fisher-transform(power)) with all data where theta, and xi are equal to the theta and xi of the example and within the power-range as defined by the argument lm.range. This model is then used for predicting the sample size. The method "step" returns the last element in the sequence of sample sizes - power pairs, sorted with decreasing power, where the power is above the power defined for the example.

lm.range — The range of evaluations that are used for estimating the sample size if the method is set or evaluates to "lm". For the default lm.range = 0.2 this means from80 to 120 % of the power in the example, e.g. for the power of 0.9 this is a range from 0.72 to 1.08. Note that the range is cut at 0 and 1.

drop — Object of class numeric (range: 0 to 1), indicating how many drop outs are expected in the study. This information is used to calculate the number of subject that should be recruited (addressed e.g. by the function tex using type nRec).

forceDivisor — If TRUE the biggest common divisor of all evaluated sample sizes is used as divisor and the estimated sample size is increased to be divisible by this divisor. If an integer is provided it is used as divisor.

Details

An object of class powPar is used to evaluate the power for a range of n and theta and optionally for several xi and several endpoints. To be able to calculate the sample size for one particular situation (corresponding to a particular theta, xi, and endpoint and object of class powEx is needed and such an object can be constructed using the function powEx.

The method for choosing a particular sample size

Examples

```
ex <- powEx(theta = 1, drop = 0.1)
```

---

powEx-class *Class "powEx"*

Description

The power is always estimated for a range of

Objects from the Class

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

Slots

`theta.example`: Object of class `"numeric"`, the value of the parameter theta used for illustration and for estimating sample size.

`xi.example`: Object of class `"numeric"` Like theta.example but for xi-

`endpoint.example`: Object of class `"character"`, the endpoint to be used, only if several endpoints were evaluated.

`power.example`: Object of class `"numeric"`

`method`: Object of class `"character"` ~~

`lm.range`: Object of class `"numeric"` ~~

`drop`: Object of class `"numeric"` ~~

`forceDivisor`: Object of class `"logical"` ~~

`divisor`: Object of class `"logical"` ~~

Methods

merge `signature(x = "powCalc", y = "powEx")`: ...

Author(s)

<thomas.fabbro@unibas.ch>

Examples

```
##showClass("powEx")
```

---

powPar *Constructing an object of class powPar.*

Description

A function for constructing an object of class `powPar`. Such an object is used for evaluating the user defined"power function" for a parameter range. All information that is needed for calculating the power should be provided to this function by making use of the `...` argument.

Usage

```
powPar(n, theta = NA, xi = NA, ...)
```

Arguments

n	A numeric vector, indicating for which sample sizes to evaluate the power function.
theta	A numeric vector that will be used for evaluating the power function. The method theta can be used within the power function to extract the elements of this vector one by one.
xi	A numeric vector that will be used for evaluating the power function. Since for every element ofxi an individuall sensitivity plot has to be constructed, the length of the xi vector is usually short.
...	This arguemt is used to provide all parameters needed by the power function for calculating the power.

Details

An object of class powPar is used to evaluate the power function for a range of n and theta and optionally for several xi values.

The user can write a power function and extract the individual elements using the methods n, theta, and xi.

It is a good practice to include everything that is needed for the calculation, also data sets etc.

To extract the vector of theta, instead of individual values, you can use the method pp with the name theta.

For historical reasons: If the argument theta is NA the argument theta.name (a character) has to be used, to indicate the name of a numeric vector that was passed to the argument (...). The same is true for the argument xi.

Value

An object of the class powPar

Examples

```
library(sse)
psi <- powPar(n = seq(from = 20, to = 60, by = 2),
              theta = seq(from = 0.5, to = 1.5, by = 0.05)
              )
powFun <- function(psi){
  return(power.t.test(n = n(psi)/2, delta = theta(psi), sig.level = 0.05)$power)
}
pp(psi, name = "theta")
calc <- powCalc(psi, statistic = powFun)
ex <- powEx(theta = 1)
pow <- merge(calc, ex)
plot(pow, example = FALSE)
inspect(pow)
tex(pow, "nEval")  # How many subjects should be available for analysis

psi2 <- powPar(n = seq(from = 20, to = 60, by = 2),
```

```
                theta = seq(from = 0.5, to = 1.5, by = 0.05),
                xi = c(0.5, 1, 1.5)
                )

calc2 <- powCalc(psi2, statistic = powFun)
ex2 <- powEx(theta = 1, xi = 0.5)
pow2 <- merge(calc2, ex2)
plot(pow2)
tex(pow2, "nEval")  # How many subjects should be available for analysis
```

---

powPar-class *Class "powPar"*

Description

A class needed to calculate the power array. It containing all the parameters needed to feed a function that calculates the power. These are either numbers or vectors of numbers.

Objects from the Class

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

Slots

`list`: Object of class `"list"` ~~

`theta`: An `"numeric"` vector with possible values of a parameter that influences the power. The power will be calculated for every element of this vector to see how the power estimate depends on theta.

`theta.name`: Object of class `"character"` ~~

`theta.act`: Object of class `"numeric"` ~~

`xi`: Object of class `"numeric"` ~~

`xi.name`: Object of class `"character"` ~~

`xi.act`: Object of class `"numeric"` ~~

`n`: Object of class `"integer"` ~~

`n.act`: Object of class `"integer"` ~~

Methods

dim `signature(x = "powPar")`: ...

Note

none

Author(s)

<thomas.fabbro@unibas.ch>

References

none

See Also

powPar,power

Examples

```
## showClass("powPar")
```

---

pp *Extracting all elements of an object of class powPar by their names except theta and n.*

Description

All information needed for the power-function should be provided by an object of class powPar. To extract this information the pp should be used.

Usage

```
pp(object, name)
```

Arguments

object	An object of class powPar.
name	A character indicating the name of the object to be extracted.

Details

none

Value

Everything that can be stored within a list is possible.

Note

The name pp is an abbreviation for power parameter

See Also

For extracting individual elements of n, theta and xi the functions n, theta, theta should be used.

Examples

```
psi <- powPar(muA = 0,
              muB = 1,
              sA = 1,
              sB = round(seq(from = 0.5, to = 1.5, by = 0.05), 2),
              n = round(seq(from = 10, to = 30, by = 2),0),
              theta.name = "sB")
pp(psi, name = "muA")
## an example of usage
powFun <- function(psi){
  power.t.test(n = n(psi),
               delta = pp(psi, "muA") - pp(psi, "muB"),
               sd = theta(psi)
               )$power
}
## testing the power-function
powFun(psi)
```

---

pp-methods *Methods for Function pp*

Description

Methods for function pp

Methods

describe this method here

Resample-class *Class "Resample"*

Description

A concise (1-5 lines) description of what the class is.

Objects from the Class

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

Slots

`n.iter`**:** Object of class `"integer"`

Methods

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

Author(s)

<thomas.fabbro@unibas.ch>

See Also

"powCalc"

Examples

```
showClass("Resample")
```

SampleSize-class *Class "SampleSize"*

Description

follow

Objects from the Class

none

Slots

estimate: Object of class "integer" ~~

Methods

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

Author(s)

<thomas.fabbro@unibas.ch>

See Also

none

Examples

```
showClass("SampleSize")
```

sampleSize-methods *Methods for Function sampleSize*

Description

Methods for function `sampleSize`

Methods

describe this method here

tex-methods *Methods for Function tex*

Description

Methods for function `tex`

Usage

```
## S4 method for signature 'power,character'
tex(x, type, ...)
```

Arguments

`x`	The object of class `power` used for extraction
`type`	Currently available: • `drop` • `nRec` • `nEval` • `n.iter` • `power` • `sampling` • `theta`
`...`	Not used so far

Methods

describe this method here

theta-methods *Extracting the actual theta.*

Description

Extracting the actual theta from a powPar-object. This method is needed within the function (referred to as power-function) needed for evaluating the power.

Usage

```
## S4 method for signature 'powPar'
theta(object, ...)
```

Arguments

object	An object of class powPar.
...	Not used.

Details

During the evaluation process every theta, from a sequence defined in the powPar-object, needs to be evaluated. This function extracts the actual theta that varies during the evaluation process. When a powPar-object is created, the first element of theta is also set to be the actual. This allows to use this method also outside the evaluation for testing the power function.

Value

A numeric value.

Note

Do not use the method pp with argument name = "theta" inside the power-function unless you really need to extract the whole sequence of theta.

Examples

```
psi <- powPar(muA = 0,
              muB = 1,
              sA = 1,
              sB = round(seq(from = 0.5, to = 1.5, by = 0.05), 2),
              n = round(seq(from = 10, to = 30, by = 2),0),
              theta.name = "sB")
theta(psi)
## an example of usage
powFun <- function(psi){
  power.t.test(n = n(psi),
               delta = pp(psi, "muA") - pp(psi, "muB"),
               sd = theta(psi)
```

```
              )$power
}
## testing the power-function
powFun(psi)
```

---

xi-methods *Extracting the actual xi.*

Description

Extracting the actual xi from a powPar-object. This method is needed within the function (referred to as power-function) needed for evaluating the power.

Usage

```
## S4 method for signature 'powPar'
xi(object, ...)
```

Arguments

object	An object of class powPar.
...	Not used.

Details

During the evaluation process every xi, from a sequence defined in the powPar-object, needs to be evaluated. This function extracts the actual xi that varies during the evaluation process. When a powPar-object is created, the first element of xi is also set to be the actual. This allows to use this method also outside the evaluation for testing the power function.

Value

A numeric value.

Note

Do not use the method pp with argument name = "xi" inside the power-function unless you really need to extract the whole sequence of xi.

Examples

```
psi <- powPar(theta = seq(from = 0.5, to = 1.5, by = 0.05), #muA
              muB = 1,
              sA = 1,
              xi = c(1, 1.5), #sB
              n = seq(from = 10, to = 30, by = 2))
xi(psi)
## an example of usage
powFun <- function(psi){
```

```
  power.t.test(n = n(psi),
               delta = theta(psi) - pp(psi, "muB"),
               sd = xi(psi)
               )$power
}
## testing the power-function
powFun(psi)
## evaluation
calc <- powCalc(psi, powFun)
```

Index