Package 'randomizeBE'

June 20, 2018

Type Package

Title Create a Random List for Crossover Studies

Version 0.3-4

Date 2018-06-20

Author D. Labes

Maintainer D. Labes <detlewlabes@gmx.de>

Description Contains a function to randomize subjects, patients in groups of sequences (treatment sequences).
If a blocksize is given, the randomization will be done within blocks.
The randomization may be controlled by a Wald-Wolfowitz runs test.
Functions to obtain the p-value of that test are included.
The package is mainly intended for randomization of bioequivalence studies but may be used also for other clinical crossover studies.
Contains two helper functions sequences() and williams() to get the sequences of commonly used designs in BE studies.

Imports stats

License GPL (>= 2.0)

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-20 07:13:55 UTC

R topics documented:

print.rl4 *S3 method* print *for class "rl4"*

Description

Prints the randomization list including a summary if needed.

Usage

```
## S3 method for class 'rl4'
print(x, sumry=FALSE, ...)
```

Arguments

x	Object of class "rl4".
sumry	If set to TRUE a summary of the randomization will be shown. This summary contains the p-value of runs test of randomness if 2 sequences are used.
...	Additional arguments. None used.

Value

Returns invisible x.

Author(s)

D. Labes

See Also

RL4

Examples

```
# block randomization of 12 subjects for a reference replicate study
# and demonstration of the S3 print method
rl <- RL4(nsubj=12, blocksize=6, seqs=c("TRR","RTR","RRT"), seed=0)
print(rl, sumry=TRUE)
```

pruns.exact *Exact cumulative distribution function of runs test*

Description

This function calculates the exact cumulative conditional distribution of the Wald-Wolfowitz runs.

Usage

```
pruns.exact(r, n1, n2, tail = c("2-sided", "lower", "upper"))
```

Arguments

r	Number of runs observed.
n1	Number of +1 items in the sequence.
n2	Number of -1 items in the sequence.
tail	Tail of the cumulative distribution function. Default is the 2-tailed value.

Value

Numeric value of the cumulative distribution function according to the chosen tail.

Note

The 2-sided exact p-value is defined as P(abs(R-E(R)>=abs(r-E(R)).
The lower (left) tail p-value is defined as P(R<=r).
The upper (right) tail p-value is defined as P(R>=r).
r is the observed value of the random variable R.

Author(s)

D. Labes

Examples

```
# SPSS "Exact Tests": small sample example, exact p:  0.071
# x <- c(1, 1, 1, 1, 0, 0, 0, 0, 1, 1)
pruns.exact(r=3, n1=4, n2=6)
# 0.07142857
# left tail P(R<=3)=0.04761905
pruns.exact(r=3, n1=4, n2=6, tail="lower")
# right tail P(R>=3)=0.9904762
pruns.exact(r=3, n1=4, n2=6, tail="upper")
# or via runs.pvalue (2-sided)
x <- c(1, 1, 1, 1, 0, 0, 0, 0, 1, 1)
runs.pvalue(x, pmethod="ex")
```

RL4 *(Block) randomization of subjects, patients into sequence groups*

Description

This function creates a randomization list of subjects, patients into sequences of treatments.
It is mainly intended for use in crossover studies but may be used also for parallel group designs if for sequences c("T","R") is used.

Usage

```
RL4(nsubj, seqs = c("TR", "RT"), blocksize, seed=runif(1,max=1E7),
    randctrl=TRUE, pmethod=c("normal", "exact", "cc"), alpha=0.025)
```

Arguments

nsubj	Number of subjects, patients ... to be randomized. Or a vector of the subject numbers (f.i. 1001:1012)
seqs	Character representation of the sequences. In BE studies for a 2x2 cross-over usually something like `c("TR", "RT")`. If you prefer the ABC notation write down here f.i. `c("AB", "BA")`.
blocksize	Size of the blocks to randomize within. If blocksize is missing it defaults to 2*number of sequences. `blocksize` may be a vector of numbers. In that case the sizes of the blocks are randomly chosen from that vector. If `blocksize=0` then the randomization takes place in the one block with all subjects.
seed	An integer for the initialisation of the random number generator.
randctrl	Shall the creation of the randomlist controlled by a runs test of randomness? Defaults to `TRUE`.
pmethod	A character string describing the method for the p-value calculation of the runs test. `"normal"` denotes the normal approximation like the function(s) `runs.test()` of the packages `tseries` or `lawstat`. `"exact"` chooses the calculation via exact distribution of the # of runs. `"cc"` chooses the continuity correction to the large sample approximation like in the statistical software SPSS. Default is `pmethod="normal"`.
alpha	Critical alpha value for re-creation of the randomlist based on the runs test. Recommended is `alpha=0.025`.

Details

As default the randomization is done as block randomization.
If `blocksize` is given as a vector of numbers the sizes of the blocks are randomly chosen from that vector.
If `blocksize=0` then the randomization takes place in the one block with all subjects.

The randomization is in the default settings controlled by a Wald-Wolfowitz runs test of randomness, i.e. if this test shows significant deviation from randomness (p-value<0.025 as default) the randomization list is recreated.
This behavior may be switched off by setting `randctrl=FALSE` if you don't see any needfulness for such a control.
The reason for such a control was originally to avoid randomlists with only 'alternating' sequences f.i. "TR" "RT" "TR" "RT" ...
See http://forum.bebac.at/mix_entry.php?id=8745

In its original form the runs test is only defined for dichotomous outcomes (i.e if 2 sequences are used).
If more than 2 sequences are used the runs test is modified by dichotomizing the sample of sequence numbers into cases < median and > median.
See package `lawstat` or http://de.wikipedia.org/wiki/Run-Test for this.

Value

Returns a list of class "rl4". This list has the components

`rl`	A data.frame with the random list.
`seed`	The used seed.
`blocksize`	The used blocksize.
`ninseqs`	A named vector containing the number of subjects in the sequence groups.
`runs.pvalue`	The p-value of a runs test of randomness.
`date`	Date and time of creation.

The class rl4 has the S3 method `print.rl4`.

Note

The blocksize(s) should be a multiple of the used # of sequences. If this is not the case it (they) will be adjusted accordingly.

The number of subjects should be a multiple of the used # of sequences. If this is not the case a warning is thrown that the design is not balanced.

The default for `pmethod` is the calculation via standard normal approximation. This has shown the 'best' potential in rejecting the randomness for alternating random lists.
The randomness control does not work properly if more than 2 sequences are used. In that case a test of repeated patterns of sequences in blocks of length=nseq is done additionally to the runs test. Example (seqno) 1 2 3 1 2 3 ...

Author(s)

D. Labes
Part of the code for using the runs test for randomness according to a suggestion by Helmut Schuetz.

See Also

print.rl4, runs.pvalue

Examples

```
# block randomization of 12 subjects for a 2x2 cross-over using the defaults
# seqs = c("TR", "RT"), blocksize=4 and seed from Sys.time()
RL4(nsubj=12)

# block randomization of a full replicate design with randomly
# chosen blocksizes 2 or 4
rl <- RL4(nsubj=12, blocksize=c(2, 4), seqs=c("TRRT","RTTR"))
print(rl)

# randomization of 12 subjects for a 2x2 cross-over
# where the original random list don't pass the runs test
# watch the message.
RL4(nsubj=12, blocksize=0, seed=10)
#if you don't worry about some 'extreme' lists
RL4(nsubj=12, blocksize=0, seed=10, randctrl=FALSE)

# randomness control does'nt work that good in case of more
# than 2 sequences: 3x3 latin square example
rl <- RL4(nsubj=12, seq <- c("ABC","BCA","CAB"), blocksize=3, seed=5125)
print(rl, sumry=TRUE)
# gives runs p.value=0.2502
```

---

runs.pvalue *2-sided p-value of the runs test*

Description

The function calculates the 2-sided p-value of the Wald-Wolfowitz runs test after dichotomizing the input vector

Usage

```
runs.pvalue(y, pmethod = c("exact", "normal", "cc"))
```

Arguments

y	Numeric vector of data values.
pmethod	A character string describing the method for the p-value calculation of the runs test. "exact" chooses the calculation via exact distribution of the # of runs. "normal" denotes the normal approximation like the function(s) runs.test() of the packages tseries or lawstat. "cc" chooses the continuity correction to the large sample approximation like in the statistical software SPSS.

Details

This function calculates the 2-sided p-value of the runs.test.
The large sample approximations are an adaption from the codes for runs.test() found in the R-packages lawstat and tseries.
The aim of this own was to avoid the heavy footprint of both packages for this small package.

The user can choose the application of a continuity correction to the normal approximation like a SAS implementation http://support.sas.com/kb/33/092.html uses or like SPSS if n<50.

The exact distribution of runs and the p-value based on it are described in the manual of SPSS "Exact tests" to be found f.i. http://www.sussex.ac.uk/its/pdfs/SPSS_Exact_Tests_21.pdf.
If pmethod="exact" is chosen and n>30 and n1>12 and n2>12 (see pruns.exact) the continuity corrected version of the normal approximation is used to save time and memory.

Value

Numeric p-value of the 2-sided test.

Author(s)

D. Labes
adapted from runs.test() package lawstat
Authors: Wallace Hui, Yulia R. Gel, Joseph L. Gastwirth, Weiwen Miao
and from runs.test() package tseries
Author: A. Trapletti

See Also

pruns.exact

Examples

```
# alternating sequence 1,2,1,2 ...
# maybe seen as numeric representation of 'TR','RT' ...
# and is used in that way here in this package
x <- rep(c(1, 2), 6)
runs.pvalue(x, pmethod="normal")
```

```
# should give 0.002464631
# exact p-value
runs.pvalue(x, pmethod="exact")
# should give 0.004329004
#
# same for 3 numbers (numeric representation of 3 sequences)
x <- rep(c(1, 2, 3),4)
runs.pvalue(x, pmethod="normal")
# should give 0.2502128
# i.e. is seen as compatible with a random sequence!
# exact p-value, default i.e. must not given exolicitely
runs.pvalue(x)
# should give 0.3212121
# i.e. is seen even more as compatible with a random sequence!
```

`sequences` *Obtain sequences for BE study designs*

Description

The function is intended as helper function to get the sequences for commonly used designs in BE studies.
These sequences may then subsequently used in function `RL4()`.

Usage

```
sequences(design, tmts = NULL)
```

Arguments

`design`	A character value describing the study design. For crossover studies it is "tmts x sequences x periods" without space. F.i. "2x2x2" is the standard 2-treatments-2-sequence-2-period crossover. The designs `"2x2x2"`, `"3x3x3"`, `"4x4x4"` may be abbreviated to `"2x2"`, `"3x3"`, `"4x4"`.
`tmts`	A character vector with the codes of the treatments.

Details

This function was implemented because I couldn't remember f.i. all the six sequences of a "3x6x3" design if had to be coded by T1,T2 and R.
The sequences are primarily build within the ABC... notation.
If treatment codes are given the "AB..." in the sequences are replaced by these codes.

The following designs are implemented:
`"parallel"` = 2-group parallel design
`"2x2"` = classical 2-treatments-2-sequence-2-period crossover
`"3x3"` = 3-treatments-3-sequence-3-period crossover (Latin square)
`"3x6x3"` = 3-treatments-6-sequence-3-period crossover (Williams design)

"4x4" = 4-treatments-4-sequence-4-period crossover (Latin square)
"2x2x3" = 2-sequence-3-period replicate crossover
"2x2x4" = 2-sequence-4-period full replicate crossover
"2x4x4" = 4-sequence-4-period full replicate crossover
"2x3x3" = 3-sequence-3-period partial (reference) replicate crossover
"2x4x2" = 4-sequence-2-period replicate crossover, Baalams design.

The sequences for the "3x3" and "4x4" designs are randomly derived from the standard Latin squares (in ABC notation)
ABC, BCA, CAB
or in case of the "4x4" design from
ABCD, BDAC, CADB, DCBA.

Value

Returns a character vector with the sequences.

Note

For the higher order designs (designs with more than 2 treatments or replicate crossover designs, respectively) only a selection of possible designs are covered.
The design sequences for "4x4" are not guaranteed to be a Williams design. If need a Williams design use function williams() instead.

Author(s)

Detlew Labes

References

Jones B, Kenward MG (2003). "Design and Analysis of Cross-Over Trials" 2nd edition. Chapman & Hall, London.

See Also

williams, RL4

Examples

```
# classical 2x2 crossover in TR notation (simple enough to remember
sequences("2x2", tmts=c("T","R"))
# 3-treatment-6-sequence-3-period Williams design in ABC notation
sequences("3x6x3")
# 3-treatment-3-sequence-3-period design with one Test and two References
sequences("3x3", tmts=c("T","R1","R2"))
# 4-treatment-4-period in TxRy notation, two Test and two Reference
sequences("4x4", tmts=c("T1","T2","R1","R2"))
```

`williams` *Construct sequences of a Williams design*

Description

The function constructs the sequences of a Williams design via the algorithm given by Sheehe and Bross.

Usage

```
williams(ntmt = 4, tmts = NULL)
```

Arguments

`ntmt`	Number of treatments. Must be an integer >1.
`tmts`	NULL or a vector of treatment codes.

Details

A Williams design is a crossover design in which each subject receives each treatment. The design is balanced over periods. The design has additionally the further feature that every treatment follows every other treatment the same number of times. It is said that these designs are balanced for first order carry over effects.
For an even number of treatments the design is a Latin square. For an odd number the design is a combination of two Latin squares.

Although the balance to first order carry over effects is not absolutely necessary in well planned BE studies with sufficient washout the Williams designs were traditionally used in `"3x6x3"` `"4x4"` crossover studies.

The sequences are created originally within the ABC... notation. If `tmts` are given the sequences are returned based on these treatment codes.

Value

Returns a character vector of the sequences.

Note

If ntmt>3 the design returned is randomly chosen from the possible designs.

A similar but simpler function was contained in the package `crossdes`. But this package was removed from the CRAN repository.

Author(s)

D. Labes

References

Williams, E. J. (1949) "Experimental designs balanced for the estimation of residual effects of treatments" Australian J. of Scientific Research, Ser. A 2, 149-168.

Sheehe PR, Bross IDJ (1961) "Latin Squares to Balance Immediate Residual and Other Effects." Biometrics, 17, 405-414.

Jones B, Kenward MG (2003). "Design and Analysis of Cross-Over Trials" 2nd edition. Chapman & Hall, London.

Bing-Shun Wang, Xiao-Jin Wang, Li-Kun Gong (2009) "The Construction of a Williams Design and Randomization in Cross-Over Clinical Trials Using SAS" J. of Statistical Software, Volume 29, Code Snippet 1

See Also

sequences, RL4

Examples

```
# Williams design for 4 treatments in ABC... notation
williams()
# The 6 sequences of the Williams design for 3 treatments
# same as sequences(design="3x6x3") except the ordering
williams(ntmt=3)
```

Index