Package ‘crossrun’

October 12, 2022

Version 0.1.1

Title Joint Distribution of Number of Crossings and Longest Run

Description Joint distribution of number of crossings and the longest run in a series of independent Bernoulli trials. The computations uses an iterative procedure where computations are based on results from shorter series. The procedure conditions on the start value and partitions by further conditioning on the position of the first crossing (or none).

Depends R (>= 3.5)

License GPL-3

URL https://github.com/ToreWentzel-Larsen/crossrun

Encoding UTF-8

LazyData true

Imports Rmpfr (>= 0.7-1)

RoxygenNote 7.1.2

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Tore Wentzel-Larsen [aut, cre], Jacob Anhøj [aut]

Maintainer Tore Wentzel-Larsen <tore.wentzellarsen@gmail.com>

Repository CRAN

Date/Publication 2022-04-13 07:32:33 UTC

R topics documented:

boxprobt ................................. 2
clshift .................................. 3
crossrunauto .................................. 3
crossrunbin .................................. 4
Description

A box cumulative sum is defined as the cumulative sum over a lower left rectangle. This function is primarily for use when the components are point probabilities for the number of crossings C and the longest run L, then component (c,l) in the result is the probability \( P(C \geq c, L \leq l) \).

Usage

\[ \text{boxprobt}(\text{mtrx}) \]

Arguments

\[ \text{mtrx} \quad \text{mpfr array} \]

Value

mpfr array
Examples

```r
nill <- Rmpfr::mpfr(0, 120)
one <- Rmpfr::mpfr(1, 120)
two <- Rmpfr::mpfr(2, 120)
contents <- c(one, nill, nill, one, one, one, two, two, two)
mtrx3 <- Rmpfr::mpfr2array(contents, dim = c(3, 3))
print(mtrx3)
print(boxprobt(mtrx3))
```

clshift

### Number of Crossings and Longest Run

**Description**

Auxiliary function for simclbin, computing the number of crossings (type=0) or longest run (type=2) in a sequence of independent normal observations. Crossings and runs are related to whether the observations are above a shift.

**Usage**

```r
clshift(seri, shift = 0, type = 0)
```

**Arguments**

- `seri` numeric; seri a sequence of random draws
- `shift` numeric; shift for the observations
- `type` numeric; 0 number of crossings, 1 longest run

**Value**

number of crossings or longest run, numeric

---

crossrunauto

### Joint Distribution for Crossings and Runs, autocorrelated Sequence

**Description**

Joint probability distribution for the number of crossings C and the longest run L in a sequence of n autocorrelated Bernoulli observations with success probability p. To enhance precision, results are stored in mpfr arrays and the probabilities are multiplied by $m^{n-1}$ for a multiplier m.
crossrunbin

Usage

crossrunauto(
  nmax = 100,
  prob = 0.5,
  changeprob = 0.5,
  mult = 2,
  prec = 120,
  printn = FALSE
)

Arguments

  nmax     max sequence length.
  prob     success probability p.
  changeprob    unrestricted change probability. If \( p \geq 0.5 \), probability of changing to success, if not probability of changing to failure.
  mult     multiplier for joint probabilities.
  prec     mpfr precision.
  printn   logical for progress output.

Value

  list of joint probabilities.

Examples

# p=0.6, independence
cr10.6 <- crossrunbin(nmax=10, prob=0.6, printn=TRUE)
cra10.6 <- crossrunauto(nmax=10, prob=0.6, changeprob=.6, printn=TRUE)
Rmpfr::asNumeric(cr10.6$pt[[10]])
Rmpfr::asNumeric(cra10.6$pt[[10]])
Rmpfr::asNumeric(cr10.6$pt[[10]]) - Rmpfr::asNumeric(cra10.6$pt[[10]]) # equal

# p=0.6, some dependence
cr10.6 <- crossrunbin(nmax=10, prob=0.6, printn=TRUE)
cra10.6.u.5 <- crossrunauto(nmax=10, prob=0.6, changeprob=.5, printn=TRUE)
round(Rmpfr::asNumeric(cr10.6$pt[[10]]),1)
round(Rmpfr::asNumeric(cra10.6.u.5$pt[[10]]),1) # not the same

crossrunbin  Joint Distribution for Crossings and Runs

Description

Joint probability distribution for the number of crossings \( C \) and the longest run \( L \) in a sequence of \( n \) independent Bernoulli observations with success probability \( p \). To enhance precision, results are stored in mpfr arrays and the probabilities are multiplied by \( m^{n-1} \) for a multiplier \( m \).
### Usage

```r
crossrunbin(nmax = 100, prob = 0.5, mult = 2, prec = 120, printn = FALSE)
```

### Arguments

- **nmax**: max sequence length.
- **prob**: success probability.
- **mult**: multiplier for joint probabilities.
- **prec**: mpft precision.
- **printn**: logical for progress output.

### Value

list of joint probabilities.

### Examples

```r
crb10.6 <- crossrunbin(nmax=10, prob=.6, printn=TRUE)
pdf(crb10.6$pt[[10]])
```

### Description

Joint probability distribution for the number of crossings C and the longest run L in a sequence of n independent Bernoulli observations with p possibly varying success probability. To enhance precision, results are stored in mpfr arrays and the probabilities are multiplied by $m^{n-1}$ for a multiplier $m$. 

### Usage

```r
crossrunchange(  
  nmax = 100,  
  prob = rep(0.5, 100),  
  mult = 2,  
  prec = 120,  
  printn = FALSE
)
```

### Arguments

- **nmax**: max sequence length.
- **prob**: success probabilities.
- **mult**: multiplier for joint probabilities.
- **prec**: mpft precision.
- **printn**: logical for progress output.
Value

List pt of joint probabilities. Cumulative probabilities qt within each row are also included. Further, mostly for code checking, lists pat and qat conditional on starting with a success, and pbt and qbt conditional of starting with a failure, are included.

Examples

prob10 <- c(rep(.5,5),rep(.7,5))
crchange10 <- crossrunchange(nmax=10, prob=prob10,printn=TRUE)
print(crchange10$pt[[10]])

crossrunem

Description

Joint probability distribution for the number of crossings C and the longest run L in a sequence of n Bernoulli observations where the number of successes is fixed at m, m between 0 and n. For fixed n, the joint distribution is computed for all m, this makes the computation demanding in terms of time and storage requirements. The joint distribution is computed separately for sequences where the first observation is, or is not, a success. The results are mainly intended for use when n is even and m=n/2, but computation in this case requires that all distributions are computed previously for all m, for all shorter sequences (lower n). In the case of even n and m=n/2, the distributions for sequences starting or not with a success are identical, and only the distribution among sequences starting with a success is used. In that case, this may be interpreted as the joint distribution for sequences around the empirical median.

Usage

crossrunem(nmax = 100, prec = 120, printn = FALSE)

Arguments

nmax       max sequence length.
prec       mpft precision.
printn     logical for progress output.

Value

nfi, number of sequences with m successes, starting with a success, and nfn, number of sequences with m successes, not starting with a success. Three-dimensional Rmpfr arrays for each n up to nmax, with dimensions n (C=0 to n-1), n (L=1 to n) and n+1 (m=0 to n). For n even and m=n/2, only nfi, and the part corresponding to C=1 to n-1 and L=1 and m=n/2 is non-zero and should be used.
**Examples**

crem14 <- crossrunem(nmax=14, printn=TRUE)
Rmpfr::asNumeric(crem14$nfi[[14]][,"m=7"])[,] # subsets of size 7=14/2
# restricted to possible values of C and L
Rmpfr::asNumeric(crem14$nfi[[14]][[-1,1:7,"m=7"]]) # same as stored data joint14em
Rmpfr::asNumeric(crem14$nfn[[14]][[-1,1:7,"m=7"]]) # the same

# subsets of sizes different from 14/2
# size 4, first observation included
Rmpfr::asNumeric(crem14$nfi[[14]][,"m=4"])
# size 14-4=10, first observation not included
Rmpfr::asNumeric(crem14$nfn[[14]][,"m=10"])

---

crossrunemcont  Continuation of an existing sequence of joint probabilities for crossings and longest run, based on the empirical median.

**Description**
Continuation of an existing sequence of the number of crossings $C$ and the longest run $L$ in a sequence of $n$ independent continuous observations classified as above or below the empirical median. To enhance precision, results are stored in mpfr arrays and the probabilities are multiplied by $\binom{n}{m}/2$ where $m=n/2$, even $n$ assumed. The probabilities are integers in this representation.

**Usage**
crossrunemcont(emstart, n1 = 61, nmax = 100, prec = 120, printn = FALSE)

**Arguments**
- **emstart**: existing sequence
- **n1**: sequence length for the first new case added
- **nmax**: max sequence length.
- **prec**: mpfr precision.
- **printn**: logical for including progress output.

**Value**
$nfi$, number of sequences with $m$ successes, starting with a success, and $nfn$, number of sequences with $m$ successes, not starting with a success.
crossrunshift

**Description**

wrapper for crossrunbin, success probability=\( \text{pnorm}(\text{shift}) \).

**Usage**

\[
\text{crossrunshift}(nmax = 100, \text{shift} = 0, \text{mult} = 2, \text{prec} = 120, \text{printn} = \text{FALSE})
\]

**Arguments**

- **nmax**: max sequence length.
- **shift**: mean of normal distribution.
- **mult**: multiplier for joint probabilities.
- **prec**: mpfr precision.
- **printn**: logical for progress output.

**Value**

list \( pt \) of joint probabilities. Cumulative probabilities \( qt \) within each row are also included. Further, mostly for code checking, lists \( \text{pat} \) and \( \text{qat} \) conditional on starting with a success, and \( \text{pbt} \) and \( \text{qbt} \) conditional of starting with a failure, are included.

**Examples**

\[
\text{crs15} \leftarrow \text{crossrunshift}(\text{nmax}=15, \text{printn}=\text{TRUE})
\]

\[
\text{print}(\text{crs15}\$\text{pt}[[15]])
\]

crossrunsymm

**Description**

Joint probability distribution for the number of crossings \( C \) and the longest run \( L \) in a sequence of \( n \) independent Bernoulli observations with success probability \( p \). To enhance precision, results are stored in mpfr arrays and the probabilities are multiplied by \( m^{n-1} \) for a multiplier \( m \). This is for the symmetric case with success probability 0.5, in which the multiplied probabilities are integers for the default value 2 of the multiplier.

**Usage**

\[
\text{crossrunsymm}(nmax = 100, \text{mult} = 2, \text{prec} = 120, \text{printn} = \text{FALSE})
\]
**Arguments**

- **nmax**: max sequence length.
- **mult**: multiplier for joint probabilities. Default 2.
- **prec**: mpfr precision.
- **printn**: logical for including progress output.

**Value**

pt, list of joint probabilities, multiplied with $m^{n-1}$. In addition cumulative probabilities qt within each row are also included.

**Examples**

```r
crs10 <- crossrunsymm(nmax=10, printn=TRUE)
```

---

### cumsumm

**Row-wise Cumulative Sums**

**Description**

Row-wise Cumulative Sums in mpfr Array.

**Usage**

```r
cumsumm(mtrx)
```

**Arguments**

- **mtrx**: mpfr two-dimensional array.

**Value**

mpfr array with row-wise cumulative sums, same dimension as the original array.

**Examples**

```r
null <- Rmpfr::mpfr(0, 120)
one <- Rmpfr::mpfr(1, 120)
two <- Rmpfr::mpfr(2, 120)
contents <- c(one, null, one, one, one, two, two, two)
mtrx3 <- Rmpfr::mpfr2array(contents, dim = c(3, 3))
print(mtrx3)
print(cumsumm(mtrx3))
```
cumsummcol  
*Column-Wise Cumulative Sums*

**Description**
Column-wise cumulative sums in mpfr array.

**Usage**
cumsummcol(mtrx)

**Arguments**
mtrx  
mpfr two-dimensional array.

**Value**
mpfr array with column-wise cumulative sums, same dimension as the original array.

**Examples**
nill <- Rmpfr::mpfr(0, 120)
one <- Rmpfr::mpfr(1, 120)
two <- Rmpfr::mpfr(2, 120)
contents <- c(one,nill,nill, one,one,one, two,two,two)
mtrx3 <- Rmpfr::mpfr2array(contents, dim = c(3, 3))
print(mtrx3)
print(cumsummcol(mtrx3))

---

**exactbin**  
*Exact Joint Probabilities for Low n*

**Description**
Exact joint probabilities, for low n, of the number of crossings C and the longest run L in n independent Bernoulli observations with success probability p. Probabilites are multiplied by $2^{n-1}$.

**Usage**
extactbin(n, p = 0.5, prec = 120)

**Arguments**
n  
number, length of sequence, at most 6.
p  
success probability.

prec  
precision in mpfr calculations. Default 120.
Value

mpfr array

Examples

exactbin(n=6)
exactbin(n=5, p=0.6)

---

joint100.6  
Joint probabilities, n=100, success probability 0.6

Description

The joint probabilities of the number C og crossings (0, ... 99) and the longest run L (1, ..., 100) in a series of n=100 independent Bernoulli observations for success probability 0.6. The probabilities are stored in the "times" representations, multiplied by $2^{100-1}$. Only the joint distributions for n=15, 60, 100 and success probabilities 0.5 and 0.6 are included in the package to avoid excessive storage, but many more cases may be generated by the function crossrunbin.

Usage

joint100.6

Format

matrix, 100 rows and 100 columns

Source

generated by the function crossrunbin and transformed from an Rmpfr array to a matrix

---

joint100symm  
Joint probabilities, n=100, symmetric case

Description

The joint probabilities of the number C og crossings (0, ... 99) and the longest run L (1, ..., 100) in a series of n=100 independent Bernoulli observations for the symmetric case (success probability 0.5). The probabilities are stored in the "times" representations, multiplied by $2^{100-1}$ and are integers in the symmetric case. Only the joint distributions for n=15, 60, 100 and success probabilities 0.5 and 0.6 are included in the package to avoid excessive storage, but many more cases may be generated by the function crossrunsymm.

Usage

joint100symm
Format
matrix, 100 rows and 100 columns

Source
generated by the function crossrunsymm and transformed from an Rmpfr array to a matrix

---

**joint14.6**

*Joint probabilities, n=14, success probability 0.6*

**Description**
The joint probabilities of the number C of crossings (0, ... 13) and the longest run L (1, ..., 14) in a series of n=14 independent Bernoulli observations for success probability 0.6. The probabilities are stored in the "times" representations, multiplied by $2^{14-1} = 8192$. Only the joint distributions for n=14, 60, 100 and success probabilities 0.5 and 0.6 are included in the package to avoid excessive storage, but many more cases may be generated by the function crossrunbin.

**Usage**

```
joint14.6
```

**Format**
matrix, 14 rows and 14 columns

**Source**
generated by the function crossrunbin and transformed from an Rmpfr array to a matrix

---

**joint14em**

*Joint probabilities, n=14, around the empirical median*

**Description**
Joint probabilities of the number C of crossings (1, ... 13) and the longest run L (1, ..., 17) in a series of n=60 Bernoulli observations around its empirical median. The probabilities are stored in the "times" representations, multiplied by (60 by 30)/2, the number of constellations starting above the median, and are integers. About the empirical median there is at least one crossing, and the longest run cannot exceed 14/2=7. Only the joint distributions for n=14, 60 are included in the package to avoid excessive storage, but many more cases may be generated by the function 'crossrunem. Since these computations are demanding in terms of storage and computation time, they are at present not performed for n much above 60.
**joint14symm**

**Usage**

joint14em

**Format**

matrix, 13 rows and 7 columns

**Source**

generated by the function crossrunsymm and transformed from an Rmpfr array to a matrix

---

**joint14symm** *Joint probabilities, n=14, symmetric case*

---

**Description**

Joint probabilities of the number $C$ of crossings ($0, \ldots, 13$) and the longest run $L$ ($1, \ldots, 14$) in a series of $n=14$ independent Bernoulli observations for the symmetric case (success probability $0.5$). The probabilities are stored in the "times" representations, multiplied by $2^{14-1} = 8192$ and are integers in the symmetric case. Only the joint distributions for $n=14$, $60$, $100$ and success probabilities $0.5$ and $0.6$ are included in the package to avoid excessive storage, but many more cases may be generated by the function crossrunsymm.

**Usage**

joint14symm

**Format**

matrix, 14 rows and 14 columns

**Source**

generated by the function crossrunsymm and transformed from an Rmpfr array to a matrix
Description

The joint probabilities of the number $C$ of crossings ($0, \ldots, 59$) and the longest run $L$ ($1, \ldots, 60$) in a series of $n=60$ independent Bernoulli observations for success probability $0.6$. The probabilities are stored in the "times" representations, multiplied by $2^{60} - 1$. Only the joint distributions for $n=15$, 60, 100 and success probabilities 0.5 and 0.6 are included in the package to avoid excessive storage, but many more cases are generated in the script crossrun1.R.

Usage

`joint60.6`

Format

matrix, 60 rows and 60 columns

Source

generated by the function `crossrunbin` and transformed from an Rmpfr array to a matrix

Description

Joint probabilities of the number $C$ of crossings ($1, \ldots, 59$) and the longest run $L$ ($1, \ldots, 30$) in a series of $n=14$ Bernoulli observations around its empirical median. The probabilities are stored in the "times" representations, multiplied by $(14 \times 7)/2=1716$, the number of constellations starting above the median, and are integers. About the empirical median there is at least one crossing, and the longest run cannot exceed $60/2=30$. Only the joint distributions for $n=14$, 60 are included in the package to avoid excessive storage, but many more cases may be generated by the function 'crossrunem. Since these computations are demanding in terms of storage and computation time, they are at present not performed for $n$ much above 60. '#'

Usage

`joint60em`

Format

matrix, 59 rows and 30 columns

Source

generated by the function `crossrunem` and transformed from an Rmpfr array to a matrix
**joint60symm**  
*Joint probabilities, n=60, symmetric case*

**Description**

The joint probabilities of the number of crossings (0, ..., 59) and the longest run (1, ..., 60) in a series of n=60 independent Bernoulli observations for the symmetric case (success probability 0.5). The probabilities are stored in the "times" representations, multiplied by $2^{60-1}$ and are integers in the symmetric case. Only the joint distributions for n=15, 60, 100 and success probabilities 0.5 and 0.6 are included in the package to avoid excessive storage, but many more cases may be generated by the function crossrunsymm.

**Usage**

`joint60symm`

**Format**

matrix, 60 rows and 60 columns

**Source**

generated by the function crossrunsymm and transformed from an Rmpfr array to a matrix

---

**simclbin**  
*Simulation of Independent Bernoulli Observations*

**Description**

Simulation of a sequence of independent Bernoulli Observations. To reduce the amount of random draws, each simulation is based on a sequence of standard normal variables, and whether each observation is above a shift defined by the binomial probabilities assumed.

**Usage**

`simclbin(nser = 100, nsim = 1e+05, probs = c(0.5, 0.6, 0.7, 0.8, 0.9))`

**Arguments**

- `nser` : length of sequence simulated
- `nsim` : number of simulations
- `probs` : binomial probabilities
Value

a data frame with the number of crossings and longest run for each probability. For instance the variables nc0.5 and lr0.5 are the number of crossings and the longest run for success probability 0.5. One row for each simulation.

Examples

```r
c130simbin <- simclbin(nser=30, nsim=100)
mean(c130simbin$nc0.5) # mean number of crossings, p=0.5
mean(c130simbin$lr0.9) # mean longest run, p=0.9
```

Description

Simulation of a sequence of n=2m observations around the median in the sequence. To be used for checking the results of crossrunem.

Usage

```r
simclem(m1 = 7, nsim = 1e+05)
```

Arguments

- `m1`: half the sequence length
- `nsim`: number of simulations

Value

data frame with cs, number of crossings and ls, longest run in the simulations.

Examples

```r
simclem14 <- simclem(nsim=sum(joint14em))
print(table(simclem14)) # joint distributions in the simulations
print(joint14em) # for comparison
```
Index

* datasets
  joint100.6, 11
  joint100symm, 11
  joint14.6, 12
  joint14em, 12
  joint14symm, 13
  joint60.6, 14
  joint60em, 14
  joint60symm, 15

boxprobt, 2

clshift, 3

crossrunauto, 3
crossrunbin, 4
crossrunchange, 5
crossrunem, 6
crossrunemcont, 7
crossrunshift, 8
crossrunsymm, 8
cumsumm, 9
cumsummcol, 10

exactbin, 10

joint100.6, 11
joint100symm, 11
joint14.6, 12
joint14em, 12
joint14symm, 13
joint60.6, 14
joint60em, 14
joint60symm, 15

simclbin, 15
simclem, 16