Package ‘rngSetSeed’

February 20, 2015

Type Package
Title Seeding the Default RNG with a Numeric Vector
Version 0.3-2
Date 2014-12-03
Author Petr Savicky
Maintainer Petr Savicky <savicky@cs.cas.cz>
Description A function setVectorSeed() is provided. Its argument is a numeric vector of an arbitrary nonzero length, whose components have integer values from [0, 2^32-1]. The input vector is transformed using AES (Advanced Encryption Standard) algorithm into an initial state of Mersenne-Twister random number generator. The function provides a better alternative to the R base function set.seed(), if the input vector is a single integer. Initializing a stream of random numbers with a vector is a convenient way to obtain several streams, each of which is identified by several integer indices.
License GPL-3

R topics documented:

  rngSetSeed-package .................................................. 2
generateInitialization ............................................... 2
setVectorSeed .......................................................... 4

Index
Seeding the Default RNG with a Numeric Vector

Description

The function `setVectorSeed(vseed)` is provided, which allows to initialize the R base Mersenne-Twister random number generator using a numeric vector `vseed` of an arbitrary nonzero length. The transformation of `vseed` into the initial state of Mersenne-Twister is computed using AES (Advanced Encryption Standard). The precise algorithm is described in `generateInitialization`.

The directory "rngSetSeed/tests" contains tests, which

1. compare the generated random numbers to stored precomputed ones,
2. compare the initial states obtained using AES in C-level functions with initial states obtained using a (slow) implementation of AES in R-level functions included for test purposes.

See Also

`setVectorSeed`, `generateInitialization`.

generateInitialization

Generates a random integer vector of a specified length using AES

Description

In a typical application of the package, this function is not called directly by the user and the function is called from `setVectorSeed()`. The function is made available in order to simplify testing correctness of the package and the documentation of this function explains the exact algorithm used by `setVectorSeed()`.

Usage

`generateInitialization(vseed, m)`

Arguments

- `vseed` Numeric vector of an arbitrary nonzero length, whose components have integer values from \([0, 2^{32} - 1]\).
- `m` Numeric, the length of the required output integer vector.
Details
The function transforms an input vector vseed of an arbitrary length to a random integer vector of
length \( m \) using Advanced Encryption Standard (AES) block cipher. The function setVectorSeed()
calls generateInitialization(vseed, 624) and uses its output as an initial state of the R base
Mersenne-Twister random number generator.

The vector vseed is first replaced by \( c(vseed, \text{length}(vseed)) \) in order to guarantee that if
vseed1 is a prefix of vseed2, but they have a different length, then the outputs are unrelated. If the
length of the resulting vector is not divisible by 8, the vector is padded by zeros to the nearest larger
length divisible by 8 in order to meet the requirements of the AES algorithm. The resulting vector
is splitted into blocks of length 8 and these blocks are used as 256-bit keys in AES. Each of these
keys is used to encrypt a counter sequence of length \( \text{ceiling}(m/4) \). The encrypted values of these
sequences are combined by XOR to a single sequence of \( \text{ceiling}(m/4) \) values, each of which is
a sequence of 16 bytes. These sequences are splitted into subsequences of 4 bytes, each of which
encodes a 32-bit integer in an endianness independent way. The first \( m \) of the obtained integers form
the output.

If \( \text{length}(vseed) \leq 7 \), then the above algorithm uses AES in counter mode suggested in Fortuna
random number generator as described at http://en.wikipedia.org/wiki/Fortuna_(PRNG) with
a key specified by the user. If \( \text{length}(vseed) \geq 8 \), the algorithm uses XOR of the outputs of sev-
eral Fortuna generators with keys formed from disjoint parts of the input vector and disjoint counter
sequences.

Value
Vector of length \( m \) of integer type suitable for substituting into the components of .Random.seed.
This means that the components are integers from the interval \([-2^{31} + 1, 2^{31} - 1]\) or NA,
which represents \(-2^{31}\). If \( m = 624 \), the output vector is suitable as the initial state of Mersenne-
Twister to be copied into .Random.seed[3:626].

If \( m_1 < m \), then generateInitialization(vseed, m1) is equal to the first \( m_1 \) components of
generateInitialization(vseed, m2).

References
wiki/Fortuna_(PRNG).

See Also
setVectorSeed

Examples
s1 <- generateInitialization(1, 3)
s2 <- generateInitialization(c(1, 0), 3)
s3 <- generateInitialization(c(1, 0, 0), 3)
stopifnot(s1 == c(2054882079, -83320600, -37036705))
stopifnot(s2 == c(-1435341980, 1760892082, 970286446))
stopifnot(s3 == c(1941187208, 915534877, -365000103))
setVectorSeed  

*Initialization of Mersenne-Twister RNG with a numeric vector of an arbitrary nonzero length*

**Description**

Initializes Mersenne-Twister random number generator, which is the default RNG in R, with a numeric vector of arbitrary nonzero length, whose components are interpreted as 32-bit integers. In order to guarantee that different input vectors yield unrelated streams of random numbers, Fortuna random number generator using AES (Advanced Encryption Standard) encryption algorithm is used for the transformation of `vseed` to the initial state of Mersenne-Twister. See `generateInitialization` for more detail of the algorithm.

**Usage**

```
setVectorSeed(vseed)
```

**Arguments**

- `vseed`: Numeric vector of arbitrary nonzero length with integer values from $[0, 2^{32}-1]$.

**Details**

The function calls `RNGkind("Mersenne-Twister")` and then replaces its state with `generateInitialization(vseed)` as an initial state. See `generateInitialization` for the description of the algorithm computing the initialization of length 624 from `vseed`.

**Value**

`NULL` invisibly.

**See Also**

`generateInitialization`.

**Examples**

```r
c1 <- c(1, 0)
c2 <- c(0, 1)
stopifnot(abs(x1 - c(0.30327915, 0.93045726, 0.20716215, 0.04424525, 0.07478261)) < 1e-8)
stopifnot(abs(x2 - c(0.2231465, 0.80036017, 0.27630612, 0.69594674, 0.02688734)) < 1e-8)
```

```
Index

generateInitialization, 2, 2, 4
rngSetSeed (rngSetSeed-package), 2
rngSetSeed-package, 2
setVectorSeed, 2, 3, 4