# Package 'RcppZiggurat'

March 22, 2025

**Type** Package

**Title** 'Rcpp' Integration of Different ``Ziggurat'' Normal RNG Implementations

**Version** 0.1.7

**Date** 2025-03-22

**Description** The Ziggurat generator for normally distributed random numbers, originally proposed by Marsaglia and Tsang (2000, <doi:10.18637/jss.v005.i08>) has been improved upon a few times starting with Leong et al (2005, <doi:10.18637/jss.v012.i07>). This package provides an aggregation in order to compare different implementations in order to provide a 'faster but good enough' alternative for use with R and C++ code. See the 'zigg' package for a lighter implementation for much easier use in other packages.

**License** GPL (>= 2)

**Depends** R (>= 3.0.0)

**Imports** Rcpp, parallel, graphics, stats, utils

**Suggests** rbenchmark, microbenchmark, lattice, knitr, rmarkdown, pinp, ggplot2

**URL** https://github.com/eddelbuettel/rcppziggurat, https://dirk.eddelbuettel.com/code/rcpp.ziggurat.html

**BugReports** https://github.com/eddelbuettel/rcppziggurat/issues

**VignetteBuilder** knitr

**LinkingTo** Rcpp, RcppGSL

**NeedsCompilation** yes

**Author** Dirk Eddelbuettel [aut, cre] (<https://orcid.org/0000-0001-6419-907X>)

**Maintainer** Dirk Eddelbuettel <edd@debian.org>

**Repository** CRAN

**Date/Publication** 2025-03-22 14:20:02 UTC

# Contents



---

`RcppZiggurat-package` *Collection and comparison of different Ziggurat RNGs*

---

## Description

Marsaglia and Tsang (JSS, 2000) introduced a very fast random number generator for drawing from a standard normal distribution. Leong, Zhang, Lee, Luk and Villasenor (JSS, 2005) suggested a simple improvement to provide better distributional properties.

This package implements both approaches, both in simple forms faithful to original papers, as well as in extended and modified versions of the C/C++ code provided by John Burkardt.

It also includes a variant which calls the fairly widely used Ziggurat implementation by Jochen Voss that is part of the GNU GSL. It uses the Mersenne-Twister as its uniform generator and does not suffer from the problem identified by Leong et al.

## Author(s)

Dirk Eddelbuettel

## References

George Marsaglia and Wai Wan Tsang. The Ziggurat Method for Generating Random Variables. Journal of Statistical Software, Vol 5, Iss 8, Oct 2000 doi:10.18637/jss.v005.i08.

Philip H W Leong, Ganglie Zhang, Dong-U Lee, Wayne Luk, and John Villasenor. A Comment on the Implementation of the Ziggurat method, Journal of Statistical Software, Vol 12, Iss 7, Feb 2005 doi:10.18637/jss.v012.i07.

Website of John Burkardt. https://people.sc.fsu.edu/~jburkardt/

Website of Jochen Voss. https://www.seehuhn.de/pages/ziggurat

## Examples

```
set.seed(42)
system.time(replicate(500, rnorm(10000)))

zsetseed(42)
system.time(replicate(500, zrnorm(10000)))
```

zrnorm *Collection of Ziggurat Normal RNGs*

## Description

This package regroups and provides a nummber of implementations of the Ziggurat generator for drawing (standard) normally distributed random numbers. Ziggurat was introduced by Marsaglia and Tsang (JSS, 2000) and improved by Leong, Zhang, et al (JSS, 2005).

## Usage

```
zrnorm(n)
zrexp(n)
zruni(n)
zrnormLZLLV(n)
zrnormMT(n)
zrnormV1(n)
zrnormVec(x)
zrnormVecV1(x)
zrnormGSL(n)
zrnormQL(n)
zrnormGl(n)
zrnormR(n)
zsetseed(s)
zsetseedV1(s)
zsetseedLZLLV(s)
zsetseedGSL(s)
zsetseedQL(s)
zsetseedGl(s)
zsetseedMT(s)
zgetseed()
zgetseedV1()
zgetpars()
zsetpars(p)
```

## Arguments

| | |
|---|---|
| n | An integer determining the length of the returned N(0,1) (or expoonential or uniform) vector. |
| x | A numeric vector, already allocated, that is to filled with N(0,1) draws. |
| s | An integer number used to seed the Ziggurat algorithm. |
| p | An numeric vector of length four with the state parameters. |

## Details

The 'MT' variants provide the original Marsaglia and Tsang implementation, updated to work in 32 and 64 bit environments. Their use is **not** recommended.

The 'LZLLV' variants provide the updated Marsaglia and Tsang implementation, based on the comment by Leong, Zhang, Luk, Lee and Villasenor. These versions should be suitable. The code has also been updated to work in 32 and 64 bit environments.

The 'V1' variants are based on an earlier implementation by John Burkardt. While fastest, they also correspond to just the Marsaglia and Zhang approach and should **not** be used.

The function `zrnorm()` (and corresponding seed setter `zsetseed()` and getter `zgetseed()`) is the recommended implementation based on merging John Burkardt earlier code with the Leong et al improvements.

The `zrnormGSL` functions use the GNU GSL implementation by Jochen Voss. They too can be recommended for use.

The functions `zrnormQL` and `zrnormGl` use, respectively, functions adapted from the GNU Gretl and QuantLib projects. They have been added primarily for comparison to the other engines.

The function `zrnormR` uses the `unif_rand` uniform generator from R (which defaults to the Mersenne Twister). It is useful when the Zigurrat generator is used as a user-supplied generator in R. However, it stores state in R's internal structures which makes it a little slower than other alternatives available here.

To turn an N(0,1)-distributed draw into N(mean,sd), multiply by sd and add mean. To turn and Exp(1)-distributed draw into Exp(rate), divide by lambda. Working out how to create a Unif(low, high) from a Unif(0,1) is left as an exercise to the reader. If in doubt, stick with functions from (base) R which are extremely well-tested and widely used.

## Value

The `zrnorm*` functions all return a vector of the requested size.

The `zsetseed*` functions do not return a value, but set the seed of the generator.

The `zgetseed*` functions return the (integer) seed. This is actually not sufficient to capture the state, so `zgetpars` `zsetpars` provide this functionality with a vector of size four. Use these functions with caution.

Following the original paper, `rexp` also provides a vector of random draws from the exponential distribution. In addition, `runi` offers random draws from the uniform distribution (which is used internally).

## Author(s)

Dirk Eddelbuettel

## References

George Marsaglia and Wai Wan Tsang. The Ziggurat Method for Generating Random Variables. Journal of Statistical Software, Vol 5, Iss 8, Oct 2000 doi:10.18637/jss.v005.i08.

Philip H W Leong, Ganglie Zhang, Dong-U Lee, Wayne Luk, and John Villasenor. A Comment on the Implementation of the Ziggurat method, Journal of Statistical Software, Vol 12, Iss 7, Feb 2005 doi:10.18637/jss.v012.i07.

Website of John Burkardt. https://people.sc.fsu.edu/~jburkardt/

Website of Jochen Voss. https://www.seehuhn.de/pages/ziggurat

## See Also

RcppZiggurat-package

## Examples

```
set.seed(42)
system.time(replicate(500, rnorm(10000)))

zsetseed(42)
system.time(replicate(500, zrnorm(10000)))
```

# Index