Package 'RunuranGUI'

March 23, 2018

Type Package

Title A GUI for the UNU.RAN Random Variate Generators

Version 0.3

Date 2018-03-23

Author Josef Leydold

Maintainer Josef Leydold <josef.leydold@wu.ac.at>

Depends Runuran (>= 0.15.0)

Imports cairoDevice, graphics, gWidgets (>= 0.0-54), gWidgetsRGtk2(>= 0.0-86), methods, RGtk2 (>= 2.20.34), rvgtest (>= 0.5.0), utils

Description A GUI (Graphical User Interface) for the UNU.RAN random variate generators. Thus it allows to build non-uniform random number generators interactively for quite arbitrary distributions.
In addition, R code for the required calls from package Runuran can be displayed and stored for later use. Some basic analysis like goodness-of-fit tests can be performed.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2018-03-23 10:54:37 UTC

R topics documented:

RunuranGUI-package *A GUI for the UNU.RAN random variate generators.*

Description

This package provides a GUI (Graphical User Interface) for the UNU.RAN random variate generators.

Details

Package:	RunuranGUI
Type:	Package
Version:	0.3
Date:	2018-01-26
License:	GPL 2 or later

This package allows to build generators for non-uniform random variates interactively for quite arbitrary distributions. Thus one can either select one of the predefined distributions, or one can build a distribution object from scratch by providing data like the density function.

R code for the required calls from package **Runuran** can be displayed and stored for later use. Some basic analysis like goodness-of-fit tests can be performed.

Acknowledgements

The author thanks John Verzani for many helpful hints.

Note

Package **RunuranGUI** is based on package **gWidgets**. It has been developed and tested using GUI toolkit `"RGtk2"`. It also works with other toolkits of the **gWidgets** family albeit the layout may look a bit sloppy and the images may not displayed properly. These can be used by installing the corresponding package and setting option `"guiToolkit"`, e.g.,

‘`options("guiToolkit"="tcltk")`’

The following toolkits should work:

toolkit	*package name*	*source*	
`"RGtk2"`	**gWidgetsRGtk2**	CRAN	(default)
`"tcltk"`	**gWidgetstcltk**	CRAN	
`"rJava"`	**gWidgetsrJava**	CRAN	
`"Qt"`	**gWidgetsrQt**	R-forge	

Author(s)

Josef Leydold <unuran@statmath.wu.ac.at>.

References

W. H\"ormann, J. Leydold, and G. Derflinger (2004): Automatic Nonuniform Random Variate Generation. Springer-Verlag, Berlin Heidelberg

J. Leydold and W. H\"ormann (2000-2010): UNU.RAN User Manual, see http://statmath.wu.ac.at/unuran/.

See Also

Start GUI by means of function unuran.gui. See package Runuran for help on the UNU.RAN library of universal non-uniform random variate generators. Tests are implemented in package rvgtest.

---

stage1 *Stage 1 of the GUI for the UNU.RAN random variate generators.*

Description

Stage 1 allows to select the type of distribution and the generation method.

Distribution

1. Select type of distribution:
   - 'continuous univariate', or
   - 'discrete univariate'.
2. Choose one of the alternatives:
   - **Runuran** 'built-in' distribution, or
   - create your own 'user-defined' distribution.
3. Select one of the built-in distributions (if requested).

The parameters of 'built-in' distributions and required input for the 'user-defined' distributions are inserted on Stage 2.

Generation Method

1. Select one of the alternatives:

   Automatic - UNU.RAN tries to find an appropriate method for the given distribution.
   Inversion - An appropriate inversion method is used.
   Rejection - An appropriate rejection method is used.
   Select method - Choose a method by your own.
2. In case of 'Select method' choose one of the generation metods from the list.

Proceed to Stage 2

Click on button 'Ok' to proceed to Stage 2.

stage2 *Stage 2 of the GUI for the UNU.RAN random variate generators.*

Description

Stage 2 allows to set all required parameters for the selected distribution and generation method.

Distribution

The input data for the distribution heavily depends on the chosen distribution, see the second page of this notebook for details.

- **Runuran** 'built-in' distributions require the parameters of the distributions. Default values are preset and can be changed.
- 'user-defined' distributions usually require data like the density function and/or the cumulative distribution function (CDF) as well as the center (mode) of the distribution.

In addition the domain of the distribution has to be set / may be changed. For 'built-in' distributions one can easily draw samples from truncated distributions.

Generation Method

Set parameters for the generation method. These depend on the particular choice of the method, see the third page of this notebook for details.

Perform

The following UNU.RAN object are created and can be assigned to R objects of the given names.

- distribution object
- generator object
- random sample of given size

The checked objects are stored in the calling environment (or the environment passed as argument when calling `unuran.gui`).

In addition one can display the R code that generates these UNU.RAN objects when checking the item

- show R code and validate generator

Then a notebook is created that shows R code as well as the properties of the generator object. Moreover, other pages may be shown that allow to investigate the quality of the generator object.

Create Object

Click on button 'Ok' to create the UNU.RAN objects (and show R code if requested).

unuran.gui *A GUI for the UNU.RAN random variate generators.*

Description

Starts a GUI that allows to build generators for non-uniform random variates interactively for quite arbitrary distributions.

Usage

```
unuran.gui(envir=parent.frame())
```

Arguments

envir	the environment in which generator object and other requested data is stored.

Details

The GUI allows to build and (to some extend) analyse generators or quite arbitrary distributions. This is done in three stages:

Stage 1: Select type of distribution (continuous or discrete) and genertion method (automatic, inversion, rejection, or select a particular algorithm). In addition one can pick a distribution from a list of built-in ones or decide to define a distribution from scratch.

Stage 2: Provide parameters for built-in distributions or required information (like the density) for user-defined distributions. In addition the domain of the distribution can be truncated. Furthermore, default parameters for the selected methods may be modified.

Stage 3 (optional): Show R code that creates generator object and properties of this 'Runuran' object. In addition it is possible to perform some tests (like goodness-of-fit tests) on the generator. Notice, however, that these tests ar not available for all types of generators. E.g., approximation errors are only shown for methods that are based on numerical inversion.

Author(s)

Josef Leydold <unuran@statmath.wu.ac.at>.

Examples

```
## start GUI to Runuran random number generator
unuran.gui()
```

Index