The Stata Journal (2011)
11, Number 2, pp. 299–304

Generating random samples from user-defined distributions

Katarína Lukácsy
Central European University
Budapest, Hungary
lukacsy_katarina@phd.ceu.hu

Abstract. Generating random samples in Stata is very straightforward if the distribution drawn from is uniform or normal. With any other distribution, an inverse method can be used; but even in this case, the user is limited to the built-in functions. For any other distribution functions, their inverse must be derived analytically or numerical methods must be used if analytical derivation of the inverse function is tedious or impossible. In this article, I introduce a command that generates a random sample from any user-specified distribution function using numeric methods that make this command very generic.

Keywords: st0229, rsample, random sample, user-defined distribution function, inverse method, Monte Carlo exercise

1 Introduction

In statistics, a probability distribution identifies the probability of a random variable. If the random variable is discrete, it identifies the probability of each value of this variable. If the random variable is continuous, it defines the probability that this variable's value falls within a particular interval. The probability distribution describes the range of possible values that a random variable can attain, further referred to as the support interval, and the probability that the value of the random variable is within any (measurable) subset of that range.

Random sampling refers to taking a number of independent observations from a probability distribution. Typically, the parameters of the probability distribution (often referred to as true parameters) are unknown, and the aim is to retrieve them using various estimation methods on the random sample generated from this probability distribution.

For example, consider a normally distributed population with true mean μ and true variance σ^2. Assume that we take a sample of a given size from this population and calculate its mean and standard deviation—these two statistics are called the sample mean and the sample standard deviation. Depending on the estimation method used, the sample size, and other factors, these two statistics can be shown to asymptotically converge to the true parameters of the original probability distribution and are thus good approximations of these values.

The researcher may wish to evaluate the accuracy of the estimation method prior to using it on real data. In such a case, a random sample generated from a distribution with known true parameters can be used, and its estimated sample parameters can be compared with the true parameters to establish the accuracy of the estimation method. Taking our previous example, we can use a random sample generated from a normal distribution with parameters $\mu = 0$ and $\sigma^2 = 1$; we can estimate its sample mean by taking the arithmetic average, and we can estimate its sample variance by taking the arithmetic average of the squared errors. If these two values are reasonably close enough to the true values of these parameters, we can conclude that our estimation method (arithmetic averaging) is accurate.

In this article, I introduce a command that generates random samples from user-specified probability distribution functions with known parameters that can be used, among other applications, in such simulation exercises.

2 Probability distributions in Stata

A probability distribution can take on any functional form as long as it fulfills certain conditions; namely, it must be nonnegative and it must integrate or sum to one. Some probability distributions have become so widely used that they have been given specific names. Two of the most important are the normal and the uniform distributions; among others are, for example, β, χ^2, F, γ, Bernoulli, Poisson, logarithmic, and lognormal distributions. For more information on basic statistics and random sampling, refer to Gentle (2003) and Lind, Marchal, and Wathen (2008).

Several built-in random-number functions such as `runiform()`, `rbeta(`*a,b*`)`, `rbinomial(`*n,p*`)`, `rchi2(`*df*`)`, etc., are available to generate a random sample in Stata 10 and later versions. However, if a sample is to be drawn from any other distribution function, an inverse cumulative distribution function method must be used to apply an inverse distribution function on a uniform random sample. This method builds on the fact that if x is a continuous random variable with cumulative distribution function F_x, and if $u = F_x(x)$, then u has a uniform distribution on $(0, 1)$. It holds then that if u has a uniform distribution on $(0, 1)$ and if x is defined as $x = F_x^{-1}(u)$, then x has cumulative distribution function F_x. This essentially means equation $u = F_x(x)$ must be solved for x. For more information on the inverse cumulative distribution function method and its applications to simulation experiments, refer to Ulrich and Watson (1987) and Avramidis and Wilson (1994).

To apply this method, a number of built-in inverse cumulative distributions are available, such as `invibeta(`*a,b,p*`)`, `invbinomial(`*n,k,p*`)`, `invchi2(`*n,p*`)`, `invF(`*n1,n2,p*`)`, `invnormal(`*p*`)`, etc. They can be applied to generate random samples as outlined in the following two examples, which generate a normally and a χ^2 distributed series `my_normal` and `my_chi2`, respectively:

```
generate my_normal = invnormal(runiform())
generate my_chi2 = invchi2(n,runiform())
```

However, a user may wish to draw a random sample from a distribution that is not built in Stata. In such a case, the inverse distribution function must be algebraically derived so that the inverse cumulative distribution function method can be used. But sometimes, the inverse of a function cannot be expressed by a formula. For example, if f is the function $f(x) = x + \sin(x)$, then f is a one-to-one function and therefore possesses an inverse function f^{-1}. However, there is no simple formula for this inverse, because the equation $y = x + \sin(x)$ cannot be solved algebraically for x. In this case, numerical methods must be applied to match the values of x and y.

3 The rsample command

The syntax of the `rsample` command is

`rsample` *pdf_function* [, `left(#) right(#) bins(#) size(#) plot(#)`]

This command generates the random variables into a new variable called `rsample`.

pdf_function is required. It is a string that specifies the probability distribution function that the random sample is to be drawn from. It must be formulated in terms of x, for example, `exp(-x^2)`, although no x variable needs to exist prior to command execution.

Two properties must normally hold for the probability distribution functions:

1. They must be nonnegative on the whole support interval of the random variable.
2. They must sum or integrate to 1.

The second condition does not have to be fulfilled in this case because the `rsample` command calculates what the *pdf_function* sums or integrates up to and uses that value as a rescaling factor. This way, a rescaled *pdf_function* is used that always integrates or sums to 1. The first condition, however, must be fulfilled. If it is violated, an error message appears that reminds the user to supply a nonnegative *pdf_function*. The rescaling constant always appears on the screen.

`left(#)` and `right(#)` specify the support interval, that is, all values that the random variable can attain. `left()` must be specified to be less than `right()`. If these values are not specified, they take on default values of `-2` and `2`, respectively.

`bins(#)` specifies the number of bins into which the support interval is split for the purposes of the algorithm. Essentially, it allows the user to specify the precision of the algorithm. If this value is not specified, it takes on a default value of `1000`. However, if this value (whether defined by the user or its default) exceeds the total number of observations (whether defined by the corresponding optional parameter or by the size of the existing Stata file), it is automatically set to equal one-fifth of the total number of observations.

`size(#)` specifies the number of observations to be generated. The default is `size(5000)`. If this value is specified even though the command is executed in an existing Stata session with a nonzero number of observations, then this value is not used and the original number of observations is preserved.

`plot(#)` allows the user to choose whether results will be presented graphically in a histogram or whether no graphical view is needed. It is a binary option; that is, it can only take on values `1` and `0`, where `1` is for "yes" and `0` is for "no". The default is `plot(1)`.

4 Examples

In this section, I provide several specific examples with various *pdf_function*s and various optional parameters specified. The graphical representations of the last three examples are presented in figures 1, 2, and 3 in the form of histograms of the generated random values. These three examples were generated from normal, lognormal, and Laplace distribution functions, respectively.

```
. rsample exp(-x^2), left(-2.5) right(2.5)
. rsample exp(-abs(x)), left(-4) right(4) bins(500)
. rsample exp(-abs(x)), left(-4) right(4) size(500)
. rsample exp(-abs(x)), plot(0)
. rsample exp(-log(x)^2), left(0.1) right(6) bins(300) size(3000) plot(0)
. rsample exp(-x^2)
. rsample exp(-log(x)^2), left(0.1) right(6)
. rsample exp(-abs(x)), left(-4) right(4)
```

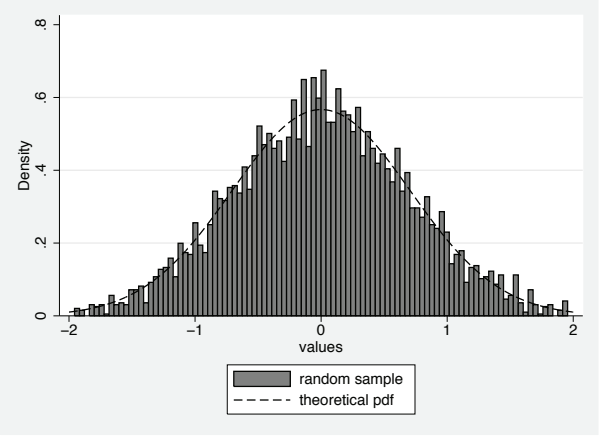

Figure 1. Random sample from a normal distribution function

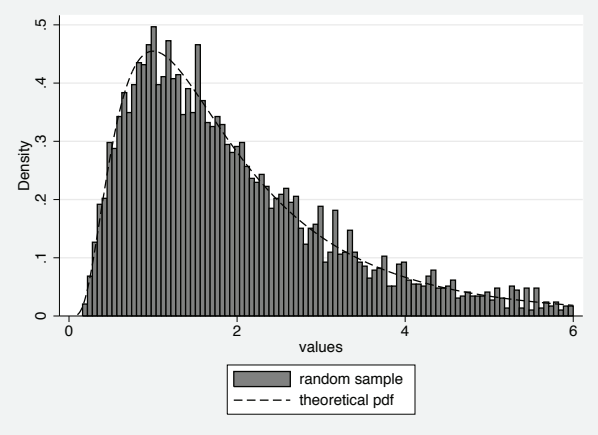

Figure 2. Random sample from a lognormal distribution function

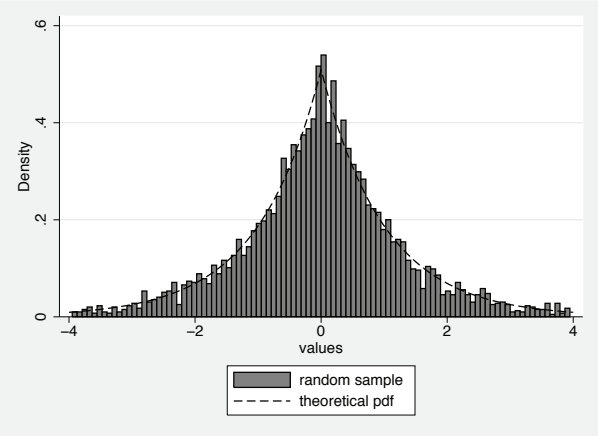

Figure 3. Random sample from a Laplace distribution function

5 References

Avramidis, A. N., and J. R. Wilson. 1994. A flexible method for estimating inverse distribution functions in simulation experiments. *ORSA Journal on Computing* 6: 342–355.

Gentle, J. E. 2003. *Random Number Generation and Monte Carlo Methods*. 2nd ed. New York: Springer.

Lind, D. A., W. G. Marchal, and S. A. Wathen. 2008. *Basic Statistics for Business and Economics*. 6th ed. New York: McGraw–Hill.

Ulrich, G., and L. T. Watson. 1987. A method for computer generation of variates from arbitrary continuous distributions. *SIAM Journal on Scientific Computing* 8: 185–197.

About the author

Katarína Lukácsy works as a statistical analyst for a large multinational company primarily focusing on improving the relevance of search results and related issues. Currently, she is finalizing her PhD dissertation on price-setting mechanisms from statistical, econometrical, and theoretical viewpoints.