Performance of Three Popular Statistical Softwares in Generating Random Numbers

[1]Luqman Hakim Musa and [1,2]Mohd Bakri Adam

[1]Department of Mathematics
Faculty of Science
Universiti Putra Malaysia

[2]Institute for Mathematical Research
Universiti Putra Malaysia

[1]luqman@math.upm.edu.my

Abstract

Random numbers are essential ingredients in a statistical analysis. They can be generated easily through statistical software packages. Each statistical package varies in terms of performance. Our objective is to compare the performances of three statistical software packages, namely R, SAS and SPSS based on their accurateness and time consumption in generating and analyzing random numbers. We obtain some estimated statistics to assess their accuracies. Comparison of the times in generating the random numbers is also observed.

Introduction

Random numbers are essential components in a statistical analysis. The criteria in choosing a statistical package depend on an individual preference which involves familiarities and costs. A suitable statistical package leads to accurate results and less time consumption.

A random number is a number that is drawn from a set of numbers where each number in the set has equal probability to be chosen. When we have a sequence of random numbers, the numbers drawn are statistically independent. The importance of these numbers enables us to do certain tasks such as simulating, modeling and understanding complicated phenomena, generating keys for data encryption and selecting random samples. Simulation task requires us to find some means to generate random numbers in order to have randomness in our work. The advancement of computer technology helps researchers to generate random numbers faster.

There are two methods to generate random numbers using computers. There are:

1. True Random Number Generators (TRNGs) which capture phenomenon from real physical world such as radioactive decay, atmospheric noise, the variance in our computer's mouse movements to generate random numbers [1] etc.
2. Pseudo-Random Number Generators (PRNGs) are constructed algorithms from mathematical formulae. The numbers generated are not genuinely random.

All characteristics of random number generators are important when performing a specific randomization task. Three main characteristics distinguished between the PRNGs and TRNGs are:

1. *Deterministic* which means the numbers generated can be reproduced later.
2. *Periodic* which means given enough time, the sequence will repeat itself.
3. *Efficient* which means we can produce many numbers in a short time.

The PRNG is *deterministic*, *efficient* and *periodic* while the TRNG has the opposite characters, i.e. *in-deterministic*, *inefficient* and *a-periodic*. Although being *deterministic* and *efficient* are the advantages of PRNG over TRNG, its weakness is *periodic*. This weakness is small and therefore it can be ignored when compared to its strengths due to new developed algorithms which can produce longer periods. The PRNG is suitable for tasks such as simulation and modeling which require many random numbers to be generated and regenerated. Tasks that require unpredictable random numbers like generating keys for data encryption, games, gambling and random sampling are more suitably done by TRNG. Further discussions on these two methods are currently available on the websites such as *www.random.org* and *www.fourmilab.ch*.

There are quite a number of times statisticians' routines involve simulation and modeling. Thus, PRNGs are included in every computer software packages for doing statistical analysis. One might have some questions and issues in choosing a suitable statistical software package. Therefore, it is important to compare selected statistical software packages that exist in the market based on their performance in generating and processing random numbers. This is the main motivation of the study. We discuss the objectives of the study in the next section. Subsequent section provides brief introduction to all software packages used in this comparative study and the following sections discuss all methods performed to compare those statistical packages. We give our results and discuss them in the result and discussion section and end the paper with some conclusions in final section.

Objective

The first objective of the study is to compare the efficiency of the selected statistical packages. The statistical package that generates the same amount of random numbers in the shortest time is the most efficient. The determination of the accuracy of the statistical packages is our second objective. The statistical package which produces the closest number summaries to the theoretical distribution is the most accurate. Initially we compare the statistical packages according to their efficiency and accuracy separately, later we combine both efficiency and accuracy and compare them accordingly. The statistical package which produces random numbers with highest accuracy in the shortest time is the most efficient and accurate. After generating the random numbers, we analyze them to get some results. The final objective is the determination of which statistical package provide the most efficient time to do the analysis.

Selection of Popular Statistical Packages

Three statistical software packages are used in this comparison study. They are chosen based on the wide usage among users namely R, SAS and SPSS.

The R is a system for statistical computation and graphics. Its development is mainly based on two existing languages: S language and *Scheme* (www. schemers. org) [2]. Thus, R language is similar to S language and therefore, S-Plus (based on S), another commercial statistical software package is not included in this study. The R software can be obtained freely from its homepage *www.R-project.org*. The R has been developed such that there are versions suitable for operating systems like UNIX, Windows and Mac. In this paper, we are using R version 2.7.1 for Windows operating system (OS).

The Statistical Analysis Software, SAS has been developed since 1976 out of the need to analyze agricultural-research data and since then it has developed into one of the world's leading provider of software and services for business intelligence [3]. It is a commercialized product by a company named SAS Institute Inc. which is based in USA. See the homepage *www.sas.com*. We use SAS for Windows version 9.1

The history of the Statistical Package for the Social Sciences, SPSS can be traced back to 1968 at Stanford University as a developed tool to analyze volumes of social science data gathered through various methods of research and quickly has become reputable worldwide since then [4]. It is also a commercialized product by a USA based company, SPSS Inc. and interested readers are advised to surf this homepage *www.spss.com*. We use SPSS 16.0 for Windows.

Methodology

We compare the *efficiency* and *accuracy* of the selected statistical packages by performing each of the following steps:

Step 1: Generate a hundred sets of 100,000 random samples from Normal (100,5) and record the time taken to accomplish this step.
Step 2: Calculate the summaries of the 100,000 random samples in each set. The number summaries are minimum, maximum and mean and record the time taken to accomplish this step.

As today's computers tend to be more efficient and powerful, we need to generate a hundred sets of sizes 100,000 random numbers in order for the computer to report a generating time greater than 0 seconds. The Normal distribution with mean 100 and standard deviation 5 is chosen because Normal distribution is a common distribution and easy to handle. We eliminate factor of the machine by running each statistical package on a single computer with processor speed 2.66 GHz (dual processor) and 1 GB RAM. The operating system of this computer is Windows XP Professional 2002 service pack 3. The codes for implementing the steps (1-2) for each statistical package are given as follows.

R Code

```
set.seed (12345678)
for (i in 1:100)
{t<-system.time(b[i,1:100000]<-rnorm(100000,100,5))
ta<-system.time(a<-summary(b[i,]))
ta<-ta+system.time(res[i,7]<-round(sqrt(var(b[i,])),3))}
```

SAS Code

```
do sample=1 to 100;
do i=1 to 100000;
seed=12345678;
b=100+5*rannor(seed);output;
end;
```

SPSS Code

```
SET RNG=MT MTINDEX=12345678.
INPUT PROGRAM.
- VECTOR SAMPEL(100).
- LOOP #I = 1 TO 1000.
- LOOP #J = 1 TO 100.
-COMPUTE SAMPEL(#J) = RV.NORMAL(100,5).
- END LOOP.
- END CASE.
- END LOOP.
- END FILE.
END INPUT PROGRAM.
FREQUENCIES VARIABLES=SAMPEL1 TO
SAMPEL100
```

```
/STATISTICS=MINIMUM MAXIMUM MEAN
STDDEV
/NTILES=4
/FORMAT=NOTABLE.
```

The softwares use the PRNGs method to generate random numbers. Thus, each statistical package adopts a certain algorithm to generate these numbers as the default algorithm, except SPSS which requires the user to decide which algorithm to be used. The default algorithm in softwares R and SAS are *Mersenne-Twister* [5] while we set the random number generator in SPSS to Mersenne-Twister through the command SET RNG=MT. The Mersenne-Twister requires a starting point to initiate the algorithm. We set the starting points for the algorithm in all three softwares to a fixed number 12345678. We do this through the command set.seed( ) in R, seed= in SAS and MTINDEX= in SPSS. The purpose of setting the starting point to a fixed number is that we can reproduce the random numbers again in the future.

The function to generate one random number from Normal(100,5) in R is *rnorm(1,100,5)*. We use the function *summary( )* to get the total, mean and var( ) to get the standard deviation, sd.

The function system.time ( ) will return the time taken to perform any expression in the ( ). This is done by calling another function that will record the time at the beginning prior to evaluating the expression in ( ), and call the function once again after it has finished evaluating, thus, returning the difference between the two times. The output times in R are *user time* and *system time*. Through the help pages of R, they are defined as:

The 'user time' is the CPU time charged for the execution of user instructions of the calling process. The 'system time' is the CPU time charged for execution by the system on behalf of the calling process.

There are user *time* and *system time* for each of the 100 samples (not show) and we total up these times accordingly. These total *user time* and *system time* for R to generate the required random numbers and to calculate the number summaries are given in Table 1.

A complete SAS program contains two parts: *Data step* and *Proc* (procedure) *step*. In *Data step*, data are read and manipulated (categorized, sub-setting etc:) and random numbers are generated. For Normal distribution, SAS generates only one random number from Standard Normal distribution through the function rannor(seed). Therefore, we multiply the number by 5 (the standard deviation) and add 100 (the mean) to the product to get a random number from Normal(100,5). This process is repeated 100,000 times through a loop to get the required volume of random numbers. Note that the multiplication may lead to more biased results since any number multiplied by 5 results in a number with the last digit of either 5 or 0.

The numbers generated in *Data step* are processed in *Proc step*. The function *proc means* is used to get the required number summaries and they are given in Table 2 together with their means and standard deviations. SAS automatically returns the times to perform the two previous tasks in its *log file*. The times reported in the *log file* are *real time* and *cpu time*. SAS *help* page defines these times as:

Real time represents the clock time it took to execute a job or step; it is heavily dependent on the capacity of the system and the current load. As more users share a particular resource, less of that resource is available to you. CPU time represents the actual processing time required by the CPU to execute the job, exclusive of capacity and load factors.

The names are different than in R, but the real time and the cpu time are respectively equivalent to user time and system time in R. These times are reported as the total time to generate random numbers in the Data step and as the total time to execute the Proc means command in Proc step.

We use RV.NORMAL( ) to generate a random number from Normal distribution in SPSS and FREQUENCIES to get the required summaries. We obtain the time for SPSS to perform the command FREQUENCIES through its output. The time given are 'Processing Time' and 'Elapsed Time'. We are not able to determine the time for SPSS to generate the required random numbers since the command to do that is unavailable to us. See Table 3 for the SPPS result.

Tables 1-3 exhibit total times to generate and process random numbers.

Table 1: Total times (in seconds) to generate and process random numbers for R with means and standard deviations.

	User time to generate	System time to generate	User time to process	System time to process
Total	**2.40**	**0.05**	**2.46**	**0.33**
Mean	**0.0240**	**0.0005**	**0.0246**	**0.0033**
Sd	**0.0085**	**0.0030**	**0.0109**	**0.0068**

Table 2: Total times (in seconds) to generate and process random numbers for SAS.

	Real time to generate	CPU time to generate	Real time to process	CPU time to process
Total	**18.87**	**3.42**	**10.82**	**12.3**

Table 3: Total times (in seconds) to generate and process random numbers for SPSS.

	Processor time to generate	Elapsed time to generate	Processor time to process	Elapsed time to process
Total	NA	NA	22.78	22.34

Results and Discussion

Table 4: Summary for 100 generated samples with R, SAS and SPSS (rounded to the nearest integer values).

	Software	Generated Value
Minimum	**R** **SAS** **SPSS**	**73** **74** **73**
Mean	**R, SAS, SPSS**	**100**
Maximum	**R** **SAS** **SPSS**	**122**
sd	**R, SAS, SPSS**	**5**

Table 4 lists the summaries for R, SAS and SPSS respectively. For example, the minimum value generated by R is 73. From the table, we also find that the minimum values and maximum values generated by all three software's are quite similar. For example, the values generated by R, SAS and SPSS range from 73 to 122, 74 to 122 and 73 to 122 respectively. The table also shows that means and standard deviations of the random numbers generated are the same with our theoretical values, whose mean is 100, and standard deviation is 5.

Tables 1 and 2, show that R generates random numbers and analyzes them faster than SAS. As in Table 3, the command in SPSS is not available to us, we are not able to compare the time for generating the random numbers, but with 22.78 seconds to analyze them, SPSS is the slowest of the three. It is clear that we can rank the software through their speed of performing the required tasks.

Conclusion

We are not able to determine which software is more accurate in generating random numbers. All of them generate random numbers with equal accuracy and it is not an issue to choose which of the three to use. But when time is the important criteria, R is the fastest. We are not able to measure accurately the time for SPSS to generate the required random numbers but simply through observation, with 22.78 seconds to process random numbers, it is the slowest.

References

[1] http://www.random.org/ RANDOM.ORG - True Random Number Services.

[2] http://www.r-project.org/ The R Project for Statistical Computing.

[3] http://www.sas.com/ SAS-Business Intelligence Software and Predictive Analytics.

[4] http://www.spss.com/ SPSS, Data Mining, Statistical Analysis Software, Predictive Analysis, Predictive Analytics, Decision Support Systems.

[5] Matsumoto, M. and Y. Kurita. 1994 Variational methods in Mathematics, Twisted GFSR generators II. *ACM Transaction on Modeling and Computer Simulation*, **4**(3), 254-266.