New statistical method for generating sequences of pseudo-random numbers

M A Denisov[1], **E A Chzhan**[1], **A A Korneeva**[1], **V V Kukartsev**[1,2] **and V S Tynchenko**[1,2]

[1] Siberian Federal University, 79, Svobodny pr., Krasnoyarsk, 660041, Russia
[2] Siberian State University of Science and Technology, 31, Krasnoyarskiy Rabochiy Ave., Krasnoyarsk, 660037, Russia

E-mail: max_denisov00@mail.ru

Abstract. The paper describes the algorithm of the precision generator (hereinafter the P-generator) of random variables. Its feature is the ability to reproduce samples of large and small amounts of data with high accuracy knowing only the type of distribution. The created program module, which is based on the algorithm described in the work, is capable of generating a sample according to four distribution laws: normal, exponential, Laplace and Pareto. The proposed statistical method is based on calculating the number of points using priori known density distribution formula of the selected law. Further generation of the found number of points is implemented using the built into C# programming language random numbers generator. Computational experiments, which are given in the work, demonstrate the accuracy of the algorithm and also allow one to conduct a comparative analysis with existing alternatives (such popular mathematical and statistical packages as Mathcad and Excel).

1. Introduction

Random number generators are used in various fields of computer science: cryptography, algorithm testing, mathematical modeling problems, statistical analysis. The application area of the random number generator, presented in this work, is focused on tasks related to mathematical modeling where the input data simulates the operation of the system under study in real conditions [1].

There is a class of methods that implements the generation of pseudo-random numbers. The main task of generators is to create a sample according to a certain distribution law. It should be noted a variety of such laws. For example, the normal distribution law, which is used to describe the phenomena of various fields of science, in particular, in chemical texture analysis [2]; the exponential distribution law is widely used in problems of reliability analysis of systems [3], the Laplace distribution law is applied in physics in order to improve the denoising effect of the pulsar signal [4] and the Pareto is used in cases of flood frequency analysis [5].

During computational experiments, most often, the authors used built-in number generators in mathematical packages such as Excel, Mathcad and others. These programs are based on the Monte Carlo class of methods that uses the assumption of an infinite sample size of observations in the process of proving their convergence. This approach is ineffective when in practice there is a necessity to generate a sample of small size. This problem was the impetus for the creation of the P-generator

algorithm for random values that excludes this drawback, and it is implemented differently in comparison with classical methods.

The authors of this article propose a pseudo-random numbers generator which involves considering laws of distribution, namely: normal, exponential, Pareto and Laplace. Practically P-generator, described in the paper, may be actively used in solving such modeling problems as it has been considered in [6] or [7].

2. The problem statement

The problem statement: it is necessary to generate a sample $\{x_1,\ldots,x_N\}$ of a random variable distributed according to a given law with known parameters. The algorithm can be divided into several steps:

1 step. Enter general parameters: k – number of subintervals, N – required sample size, ε – accuracy (the smallest selected distribution density value). Enter the scale parameters of the distribution laws: λ – exponential and Laplace distributions, σ – normal distribution law, α – Pareto distribution law.

2 step. Determine the region of admissible values of a random variable which is defined by the boundaries [a, b]. The boundaries for each of the distribution laws are calculated by equalizing the accuracy ε and its distribution density.

Let us calculate the normal distribution boundaries [a, b] as follows:

$$a=\sqrt{-2\sigma^2\ln(\varepsilon\sigma\sqrt{2\pi})},\quad b=-\sqrt{-2\sigma^2\ln(\varepsilon\sigma\sqrt{2\pi})}\,. \tag{1}$$

For the exponential distribution law:

$$a=0,\quad b=-\ln(\varepsilon/\lambda)\cdot\lambda^{-1}. \tag{2}$$

For the Laplace distribution law:

$$a=\ln(\lambda/2\varepsilon)\cdot\lambda^{-1},\ b=-\ln(\lambda/2\varepsilon)\cdot\lambda^{-1}. \tag{3}$$

For the Pareto distribution law:

$$a=C_0,\quad b=(aC_0{}^{a}/\varepsilon)^{(a+1)^{-1}}, \tag{4}$$

where C_0 – initial boundary.

3 step. The interval [a, b] is divided into subintervals (their number is equal to k), the boundaries of which are calculated by the recurrent method using the formula:

$$l_i=l_{i-1}+\Delta,\ i=\overline{2,k}, \tag{5}$$

where $l_1=a$, $l_k=b$, step Δ is defined as:

$$\Delta=(b-a)/k. \tag{6}$$

4 step. It is necessary to determine the number of points that get into each of the subintervals. To do this, let us use the formula to determine the cell frequency:

$$n_i=p_i/N,\quad i=\overline{1,k}, \tag{7}$$

where probability is calculated using the density of the current distribution according to the formula:

$$p_i=\frac{f(l_{i-1})+f(l_i)}{2}(l_i-l_{i-1}),\quad i=\overline{2,k}\,. \tag{8}$$

5 step. Next, let us check the condition: does the number of points obtained at the 4th step coincide with the one that was set at the 1st step? If the sample size is larger, then remove the extra number of points from the ends of the interval [a, b]; if less, then add the required number using the built into a programming language pseudo-random number generator (using a uniform distribution law) in the

middle of this interval. This condition allows you to get a sample the distribution of which corresponds to the given even for small volumes.

6 step. The resulting numbers are mixed. This step is important for simulation modeling tasks.

7 step. Search for the estimation of the considered distribution law parameter to determine the accuracy of the sample obtained by the following formulas:

For the normal distribution law:

$$\sigma = \sqrt{\sum_{i=1}^{N} x_i^2 \Big/ N - 1}\,. \tag{9}$$

For the exponential distribution law:

$$\lambda = \sqrt{N - 1 \Big/ \sum_{i=1}^{N} x_i}\,. \tag{10}$$

For the Laplace distribution law:

$$\lambda = \sqrt{2\left(\sum_{i=1}^{N} x_i^2 \Big/ N - 1\right)^{-1}}\,. \tag{11}$$

For the Pareto distribution law:

$$\alpha = \left(\sum_{i=1}^{N} \ln(x_i / C_0) \Big/ N\right)^{-1}. \tag{12}$$

3. Computational experiments

Random number generators exist in specialized mathematically-oriented software, for example, Excel or Mathcad. Let us show the features of the algorithm described above with already existing implementations.

The computational experiment involves the generation of numbers according to the normal and exponential distribution laws. Accuracy is estimated by formulas (9) and (10). The experiment uses small sample sizes s=20 and s=100. The required accuracy of the algorithm for this computational experiment equals ε=0.005.

At the first stage, we consider the normal distribution law with the mean m_x=0 and variance σ^2=1. We will conduct a comparative analysis of the P-generator with the existing random number generator in Excel which is available in the "Data Analysis" panel, as well as in Mathcad using the rnorm() function. Figure 1 shows the results of the histogram approximation of the generated sample for the programs Excel, Mathcad and the P-generator with different sample sizes.

A set of ten experiments was carried out at each iteration of which the accuracy value (9) was calculated, in other words, the sample correspondence to the specified distribution parameters (m_x, σ^2). The results are shown in the figures below:

The results are shown below:

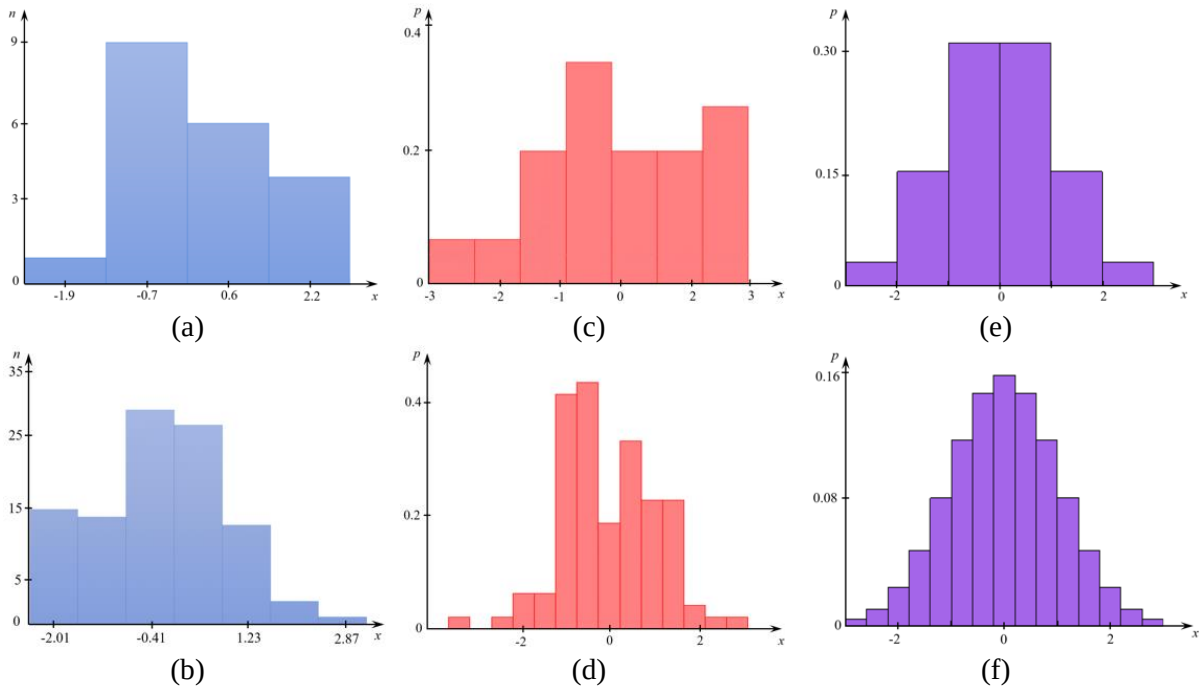


Figure 1. (a) Excel *s*=20; (b) Excel *s*=100, (c) *s*=20 Mathcad, (d) *s*=100 Mathcad, (e) *s*=20 P-generator, (f) *s*=100 P-generator.

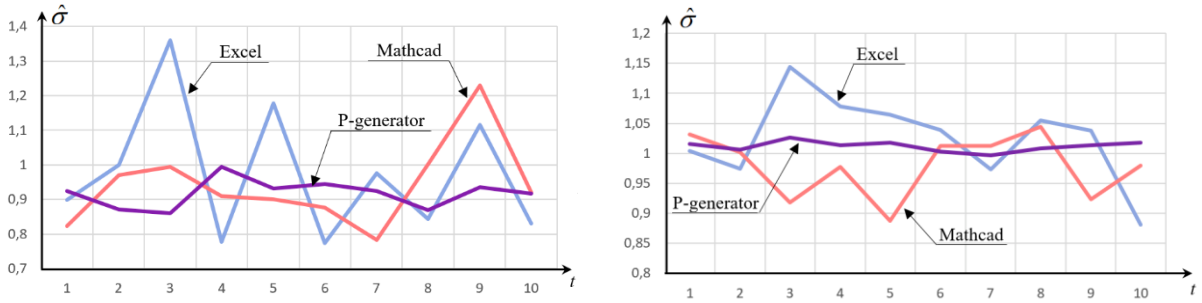


Figure 2. $\hat{\sigma}$ estimation experiments results: (a) *s*=20, (b) *s*=100.

Values $\hat{\sigma}$ for each software were averaged in order to calculate the absolute percentage error using the formula:

$$W = \left| \frac{\theta - \hat{\theta}}{\theta} \right| \cdot 100\%, \qquad (13)$$

where θ – distribution parameter value.

The final results are tabulated below.

Table 1. Error values (13) with different sample sizes

Software	*W* at *s*=20, %	*W* at *s*=100, %
Excel	2.5	2.5
Mathcad	5.8	2.1
P-generator	8.2	1.2

The highest accuracy of the approximation by the "bell-shaped" density of the normal distribution is shown in Figure 1 – (e), (f). For Excel and Mathcad, the question arises: are the numbers really normally distributed (due to their not "bell-shaped" type)?

The values presented in Table 1 show that the accuracy of the P-generator for the sample with the volume of *s*=20 is lower compared to similar algorithms. However, it is very important to pay attention to Figure 2. Two graphs show that the estimates values (9) for the Excel and Mathcad software have a high level of dispersion in comparison with the values of the P-generator estimates. Thus, by eliminating each other, the error values (13) in table 1 turned out to be lower. Nevertheless, the authors believe that for the current situation the stability of the algorithm is more important than its accuracy. The user must be sure that the sample generated by him at each new iteration will not significantly differ from the previous one. P-generator provides such assurance based on the above results.

The second stage of the computational experiment includes the analysis of the exponential law. In Excel, this type of distribution is not implemented, so the P-generator is compared only with the Mathcad software, which uses the rexp() function. The value is λ=1. The sample sizes are *s*=20 and *s*=100. The accuracy value is ε=0.05.

The computational experiment is identical to the previous one in its content. The results are shown below.

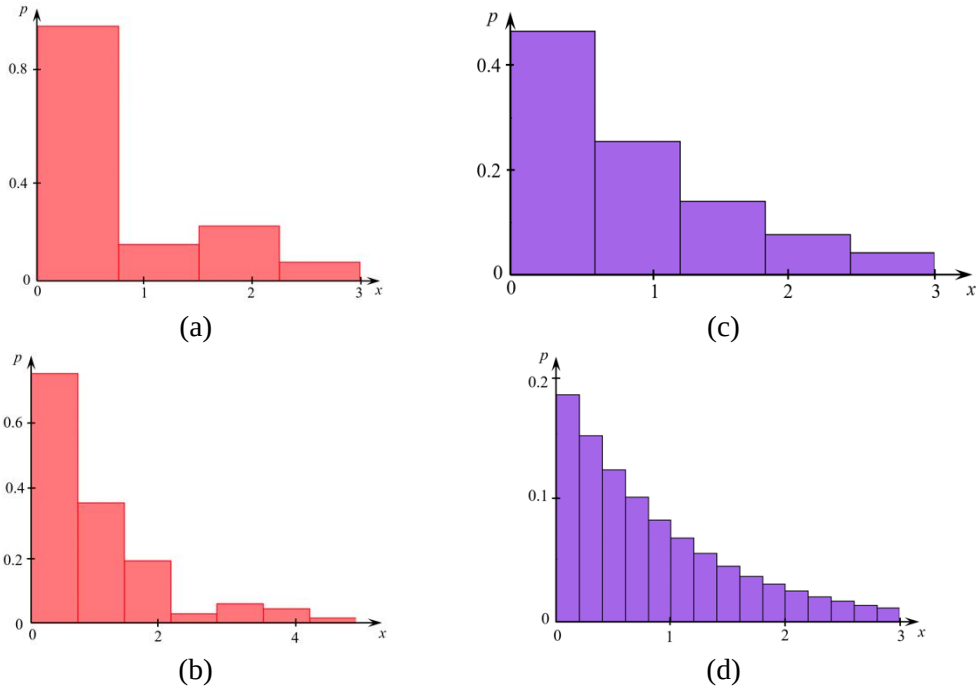
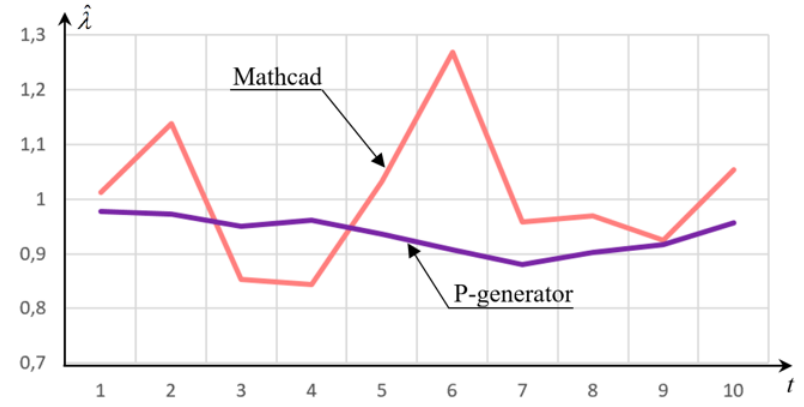
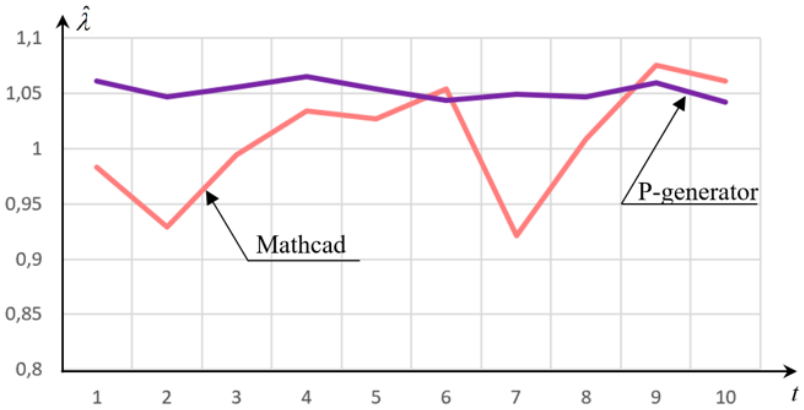


Figure 3. (a) *s* = 20 Mathcad, (b) *s*=100 Mathcad, (c) *s* = 20 P-generator, (d) *s*=100 P-generator.



(a)



(b)

Figure 4. $\hat{\lambda}$ estimation experiments results: (a) s=20, (b) s=100.

Table 2. Error values (13) with different sample sizes

Software	W at s=20, %	W at s=100, %
Mathcad	0.5	0.8
P-generator	6.4	5.2

4. Conclusion

The algorithm described in the paper (as well as software created on its basis) has a number of distinctive features that are not implemented in other standard statistical and mathematical software packages. The first thing is the creation of a sample with the required level of accuracy. The second is the existing possibility to vary the number intervals during the histogram construction, which ultimately has a positive effect on accuracy. Third - the implementation of the laws of the distribution of Laplace and Pareto, which are rarely can be found in statistical packages.

A computational experiments set has shown that with repeated attempts to generate a sample, the P-generator is more stable (the variance of the estimate of the selected distribution parameter is the smallest) in comparison with that with the analogue algorithms in Excel and Mathcad, which is another important feature.

References

[1] Prichard D and Theiler J 1994 *Phys. Rev. Lett.* **73** 951
[2] Savyolova T I and Filatov S V 2017 *J. Phys.: Conf. Ser.* **937** 012045

[3] Iskandar I 2018 *IOP Conf. Ser.: Mater. Sci. Eng.* **319** 012069
[4] Wang Wenbo, Zhao Yanchao and Wang Xiangli 2016 *AJ* **152** 131
[5] Zhou C R, Chen Y F, Huang Q and Gu S H 2017 *IOP Cong. Ser.: Earth Environ. Sci.* **82** 012031
[6] Denisov M A, Chzhan E A, Korneeva A A and Kukartsev V V 2018 *IOP Conf. Ser.: Mater. Sci. Eng.* **450** 042001
[7] Denisov M A, Chzhan E A and Korneeva A A 2019 *Regional Problems of Earth Remote Sensing (RPERS 2018)* **75** 01015