# A Novel Method for Improving the Uniformity of Random Number Generator Based on Data Oriented Modeling

A. Habibizad Navin[1] M. Naghian Fesharaki[2] M.K.Mirnia[3] M. Teshnelab[4]

Islamic Azad University [1,3]
Tabriz Branch, Tabriz, Iran.

Islamic Azad university Science[2,4]
and Research Branch, Tehran, Iran.

**Summary**

In this paper we introduce a novel method for improving the uniformity of random number generator named uniformity improving method, or UIM in short. In this approach data-oriented model of uniform random variable named UDPD is simulated. The simulation results show that generated numbers by this new method have good uniformity.

***Key words:***

*RNG, Data-Oriented Modeling, Uniform random variable, Simulation, UIM, improving the uniformity.*

## 1. Introduction

In this paper a new method is introduced for generating uniform random numbers based on data oriented modeling. This modeling method of random variable is proposed in [1]. By using this method, uniform random variable is modeled with several data. Up to now, statisticians have used mathematical functions to model random variables. This approach conforms to human brain structure, in other words it is easier for human to use mathematical functions for modeling. In this approach for modeling random variable X, distribution function F(x) is denoted its scattering. In our approach random variable X is modeled with data structure, sizeable number of data. Data-oriented model of uniform random variable proposed in [1] and named UDPD. UDPD is a weighted digraph which is explained in detail in next section of this paper. The introduced method in this paper is a simulation method of UDPD model. This simulation method based on its result named uniformity improving method, or UIM in short. The implementation results show that the good uniformity of numbers generated with UIM in comparison with Matlab's rand. Basic definition to outline UDPU and implementation of UIM is explained in section 2. In the third section non-uniformity of current Matlab's method and UIM is compared by calculating two evaluation factors and it is shown that uniformity of random numbers generated by UIM is better than Matlab's method.

Manuscript received July 5, 2007
Manuscript revised July 25, 2007

## 2. Uniformity Improving Method

In this section we introduce new method for generating uniform random numbers. The simulation result shows the good uniformity of generated number by this new method. This is the reason we named it uniformity improving method, or UIM in short. This method is a simulation way of UDPD model. UDPD is a data-oriented model of uniform random variable [1]. Data-oriented modeling is a way that models concepts by data structures. Following definitions are presented to outline UDPD as a data oriented model of uniform random variable. Then simulation of UDPD leads us the way to making the UIM.

**Definition:**

Let G=(V,E) be a weighted digraph with nonempty and finite set V as vertices and set E as edges. Weight of each edge is the probability of that transition, for example, weight of edge b→c is the probability of transition from b to c. It is called transition probability and is denoted by $P_{bc}$ . If b and c are digits then $P_{bc}$ is called Digit Probability [2].

**Definition:**

As defined in [4, 5] weighted directed graph G is called probability digraph or prodigraph in short if and only if for any vertex $a \in V$ we have: $\sum_{b \in v} P_{ab} = 1$.

Note that $P_{ij}=0$ if and only if there is not any edge from i to j and then there exists edge $i \to j$ exists in prodigraph if $P_{ij}>0$. Therefore like the adjacency matrix we can represent prodigraph by vector of vertices $V = [v_i]$ and $P = [P_{ij}]_{|V|\times|V|}$.

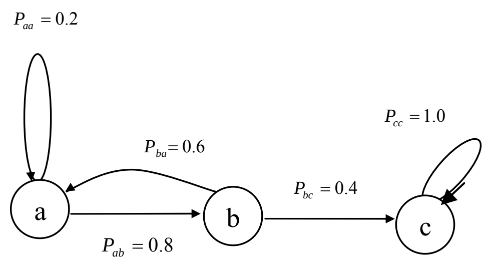


Fig.1 A prodigraph

The matrix P is called the probability matrix of prodigraph. Then a prodigraph can be denoted by $G = ([V],[P])$. For example prodigraph of figure1 can be denoted by:

$$G_P = \left( [a,b,c], \begin{bmatrix} 0.2 & 0.8 & 0.0 \\ 0.6 & 0.0 & 0.4 \\ 0.0 & 0.0 & 1.0 \end{bmatrix} \right).$$

**Definition:**

Let G=([V],[P]) be a prodigraph. Then G is a Digital Prodigraph if and only if V=[0., 0, 1, 2, …, 9].

**Definition:**

Let G=([V],[P]) be a Digital Prodigraph and w= 0., $a_1$, $a_2$, …,$a_n$ be a walk on this graph. Value Of Walk is denoted by VOW and is defined as follows:

$$VOW_w = a_1 \times 10^{-1} + a_2 \times 10^{-2} + \ldots + a_n \times 10^{-n}$$

$$= \sum_{i=1}^{n} a_i \times 10^{-i}$$

$$= 0.a_1 a_2 ... a_n.$$

In other words, VOW of walk $w = 0., a_1, a_2, ..., a_n$, is obtained by appending each vertex of w as we traverse digital prodigraph from 0. to $a_n$.

**Definition:**

Let G=([V],[P]) be a Digital Prodigraph, and w= 0., $a_1$, $a_2$, …,$a_n$ be a walk on this graph. Suppose $VOW_w = y = 0.a_1 a_2 ... a_n$ .Then $P_y$ is the Probability of $VOW_w$, if and only if:

$$P_y = P_{0.a_1} \times P_{a_1 a_2} \times \ldots \times P_{a_{n-1} a_n}$$

$$= P_{0.a_1} \prod_{i=2}^{n} P_{a_{i-1} a_i}.$$

**Digits uniformity Theorem:**

Let U be a random variable which is distributed uniformly in [0, 1] and $V_u = \{0,1,2,...,9\}$ be the set of digits then the probability of nth digit of U be equal to $v \in V_u$ is 0.1[2].

**Proof:**

We proved the digits uniformity theorem by induction on the probability of nth digit of U [2].

**Definition:**

Similar to random variable *U*, which is distributed uniformly in [0, 1], we define $G_u = (V_u, P_u)$ a Uniform Digital Probability Digraph, UDPD in short, as a data oriented model of *U*. We represent UDPD by the vector of vertices, $V_u$ and probability matrix $P_u$ based on digits uniformity theorem as follows:

$$V_u = [0.,0,1,2,...,8,9]$$

$$P_u = \begin{bmatrix} 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 \end{bmatrix}$$

UDPD is the same *U*. It models the *U* by data structures $V_u$ and $P_u$. Then UDPD is a data oriented model of *U*. The simulation of UDPD leads us to making UIM. In other words UIM is a simulation method of UDPD. By using this model to generate an n-digit number with uniform distribution, we only need to traverse UDPD, starting from "0." and making n random walks. We *can prove by induction, reverse of the digit uniformity theorem*, the VOW of this walk will be a random number which is distributed uniformly in range [0, 1]. In other words each digit of VOW is generated with the probability of 0.1 and by putting all these digits together we get an n-digit uniform random number [1]. This simulation method based on its result, good uniformity of generated number, named uniformity improving method, or UIM in short. In simulation of this method, to generate digits with probability of 0.1 we use other uniform random number generator. We name this generator the Digit Generator Engine, or DGE in short. The following method employs Matlab's rand function as DGE for generating n-digit number.

## 2.1Generating a number with UIM

In this paper the UIM is implemented by following steps to generate a uniform random number. Hereafter we call this method UIM_MATLAB because it uses the Matlab's rand function as its DGE.

Steps of UIM_MATLAB:

1. Generate n number $u_1, u_2, \ldots, u_n$ in interval [0, 1], by using Matlab's rand function as its DGE.
2. Divide the interval [0,1] to 10 non-overlapping sub-intervals with equal lengths $l_0, l_1, \ldots, l_9$.
3. Generate n digits $d_1, d_2, \ldots, d_n$ as follows:
   $d_j = k \quad if \quad u_j \in \; l_k$
4. By affixing these digits together and appending "0." to its beginning, a random number is generated.

In the next section MATLAB and UIM_MATLAB are implemented and the obtained results are compared.

## 3. Implementation and Comparison

In this section, the results of implementation of two methods, MATLAB and UIM_MATLAB are provided. To compare these results, We use two evaluation factors: $s_f^2$, frequency variance, and $l_f$, length of scattering, to measure the non-uniformity of generated numbers by each methods.

These two methods are implemented by the following steps and evaluation factors are calculated which are shown in table 1.

### 3.1 Implementation steps of the Methods

1. Generate 5000 random numbers with a method.
2. Divide the interval [0,1] in to 100 sub-intervals with equal lengths $a_1, a_2, \ldots, a_{100}$.
3. Calculate the frequency of numbers in each sub-interval and represent them consequently as: $f_1, f_2, \ldots, f_{100}$.
4. To measure non-uniformity of generated numbers we calculate evaluation factors $s_f^2$ and $l_f$ as follows:

$$s_f^2 = \frac{\sum_{i=1}^{100} (f_i - 50)^2}{99}$$

$$l_f = Max_f - \min_f$$
$$Max_f = Max\{f_1, f_2, ..., f_{100}\}$$
$$\min_f = \min\{f_1, f_2, ..., f_{100}\}$$

Lower the values of $s_f^2$ and $l_f$ represent better uniformity of the method and vice versa. Figure 2 shows the histogram of 5000 random numbers which are distributed with ideal uniformly. For such numbers we have: $s_f^2 = l_f = 0$.

Figures 3 and 4 represent the frequency histogram of numbers generated with each of two methods.

Simulation results in table 1 show that uniformity of numbers generated with UIM_MATLAB is better than MATLAB. This means that UIM improves the uniformity, of its digit generator engine.

Table 1. Performance of the methods

| Method | $Max_f$ | $Min_f$ | $l_f$ | $S_f^2$ |
|---|---|---|---|---|
| MATLAB | 76 | 32 | 44 | 67.1 |
| UIM_MATLAB | 66 | 33 | 33 | 56.58 |

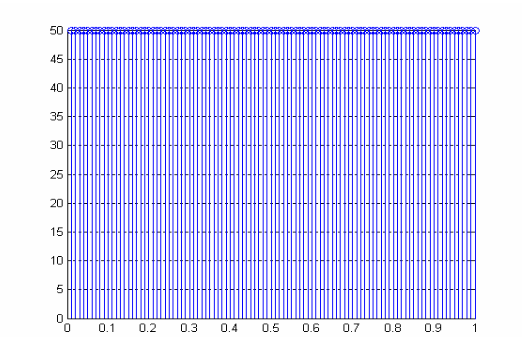

Fig.2 Frequency histogram of 5000 random numbers with ideal uniformity

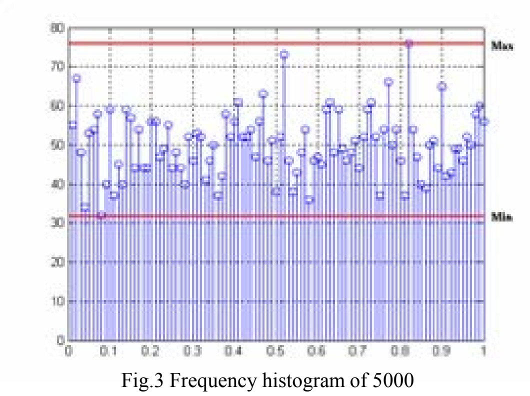


Fig.3 Frequency histogram of 5000 random numbers generated by MATLAB

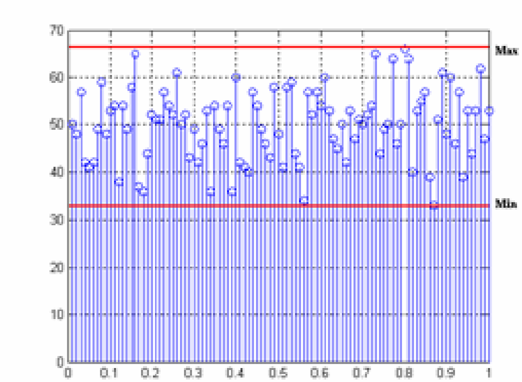


Fig.4 Frequency histogram of 5000 random numbers generated by UIM_MATLAB

## 4. Conclusion

In this paper we introduce a novel method for generating uniform random numbers. The simulation result shows the good uniformity of generated number by this new method. This is the reason we named it uniformity improving method, or UIM in short

The UIM is implemented based on UDPD which is a data-oriented model of uniform random variable. Simulation results show that numbers generated with UIM have better uniformity than its DGE. This means that the UIM can also be employed to increase the uniformity of other random number generator by using it as its digit generator engine. Good uniformity makes UIM a very suitable method where high uniformity is needed.

**Acknowledgments**

Thanks S. Rastgou Moghadam, and E. Shahamatnia for theirs help.

## References

[1] A.Habibizad Navin, M.Naghian, M.Teshnelab, and E.Shahamatnia. “Data Oriented Modeling of Uniform Random Variable: Applied Approach”, Accepted paper, Oral presentation, ICMS, 4th International Conference on Mathematics and Statistics, Vienna, Austria, May, 2007.

[2] A. Habibizad Navin , M. Naghian Fesharaki, Mohamad Teshnelab, “Digit Probability of Random Variable”, 5th Seminar on probability and Stochastic processes, Birjand-Iran, August 2005.

[3] Jerry Banks, John Carson, Barry L. Nelson, David Nicol ,Discrete-Event System Simulation, Prentice Hall, fourth edition, 2004.

[4] A. Habibizad Navin, M. Naghian Fesharaki, M. Lotfi Anhar, “Presenting a Probabilistic Problem and Solving It”, 35th Annual Iranian Mathematics Conference, Ahwaz-Iran, January 2005.

[5] A. Habibizad Navin , M. Naghian Fesharaki, Mohamad Teshnelab, “Probability Digraph and some of its important structures”, 5th Seminar on probability and Stochastic processes, Birjand-Iran, August 2005.