MATH-414 – Stochastic simulation

Lecture 1: Pseudo Random Number Generators

Prof. Fabio Nobile

EPFL

Outline

Pseudo Random Number Generators

Some examples

Empirical tests for RNGs

Uniform Random Number Generator

Definition. *A* **random number generator** *(RNG) is a procedure that produces an infinite stream of independent and identically distributed (i.i.d.) random variables* $U_1, U_2, \ldots \overset{iid}{\sim} \mu$ *according to some probability distribution* μ.

A RNG is called **uniform random number generator** *if* μ *is the uniform distribution in* $(0, 1)$.

All currently used RNGs are based on **algorithms**. As such, they produce a *purely deterministic* stream of numbers $U_1, U_2, \ldots$, which resembles a stream of iid random variables.

Algorithmic generators are called **Pseudo-Random Number Generators** (Pseudo-RNG).

General structure of a Pseudo-RNG

- $\mathcal{S}$: *finite* state space;
- $\mathcal{U}$: output space;
- $f : \mathcal{S} \to \mathcal{S}$ and $g : \mathcal{S} \to \mathcal{U}$: given functions;

Algorithm: Pseudo-RNG

take $X_0 \in \mathcal{S}$ `// seed`
for $k = 1, 2, \ldots$ **do**
  $X_k = f(X_{k-1})$ `// recursion on state variable` $X_k \in \mathcal{S}$
  $U_k = g(X_k)$ `// output` $U_k \in \mathcal{U}$
end

Remarks

- X_0 is called the **seed**.

  A Pseudo-RNG starting from a given seed will always produce the same sequence $U_1, U_2, \ldots$.

- Since the state space $\mathcal{S}$ is finite, the generator eventually revisits an already visited state. **All Pseudo-RNGs are periodic**.

  Good generators have period $p = |\mathcal{S}|$

Properties of a good uniform Pseudo-RNG

- *Have a large period*
- *Pass a battery of statistical tests* for uniformity and independence.
- *Be fast and efficient*
- *Be reproducible*
- *Have the possibility to generate multiple streams*.
- *Avoid producing the numbers 0 and 1*.

Linear Congruential Generator (LCG)

state space	$\mathcal{S} = \{0, 1, \dots, m-1\}$	
recursion	$X_k = (aX_{k-1} + b) \mod m,$	with $a, b \in \mathbb{N}$
output	$U_k = \frac{X_k}{m},\ k \geq 1$	

By construction $U_k \in \{0, \frac{1}{m}, \dots, \frac{m-1}{m}\} \subset [0, 1]$.

m should be very large to have a good coverage of $[0, 1]$.

Lewis-Goodman-Miller LCG (implemented in Matlab <= 5):
$a = 7^5 = 16807,\ b = 0,\ m = 2^{31} - 1$
Has maximal period $p = m - 1 \approx 4 \cdot 10^9$, too small for today's applications.

Multiple recursive generator (MRG) of order q

recursion: $X_k = (a_1 X_{k-1} + a_2 X_{k-2} + \cdots + a_q X_{k-q}) \mod m, \quad$ with $a_j \in \mathbb{N}$

output: $U_k = \frac{X_k}{m}, \;\; k \geq 1.$

Can be written in vectorial form with $\boldsymbol{X}^{(k)} = (X_{k-q+1}, \ldots, X_k)^\top$ as

$$\boldsymbol{X}^{(k)} = A\boldsymbol{X}^{(k-1)} \mod m, \qquad U_k = \frac{(\boldsymbol{X}^{(k)})_q}{m}, \;\; k \geq 1, \tag{1}$$

with integer matrix $A \in \mathbb{N}^{q \times q}$,

$$A = \begin{pmatrix} 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \\ a_q & a_{q-1} & \ldots & a_1 \end{pmatrix}.$$

- State space: $\{0, \ldots, m-1\}^q$; $\quad$ maximal period $p = m^q - 1$
- Generators of the general form (1) are called **Matrix Congruential Generators of order** q

EPFL

Combined generators

One can combine generators to obtain a better one.

Examples

- **Wichman-Hill**: combines 3 LCGs

$$\begin{aligned} X_k &= (a_1 X_{k-1}) \mod m_1 \\ Y_k &= (a_2 Y_{k-1}) \mod m_2 \\ Z_k &= (a_3 Z_{k-1}) \mod m_3 \end{aligned} \qquad \text{and} \qquad U_k = \frac{X_k}{m_1} + \frac{Y_k}{m_2} + \frac{Z_k}{m_3} \mod 1$$

- **MRG32k3a**: combines to MRGs

$$\begin{aligned} X_k &= (a_2 X_{k-2} + a_3 X_{k-3}) \mod m_1 \\ Y_k &= (b_1 Y_{k-1} + b_3 X_{k-3}) \mod m_2 \end{aligned} \qquad \text{and} \qquad U_k = \begin{cases} \frac{X_k - Y_k + m_1}{m_1 + 1}, & X_k \leq Y_k \\ \frac{X_k - Y_k}{m_1 + 1}, & X_k > Y_k \end{cases}$$

has a period of about $p \approx 10^{57}$ and is available as a generator in most softwares (Matlab, Mathematica, Intel's MKL library, etc)

Modulo 2 Linear Generators

These are MRGs (or Matrix Congruential Generators) with modulus $m = 2$. Multiplication and $\mod$ operations correspond to bit operations (very fast to evaluate).

Among these we mention the **Linear Feedback Shift Register (LFSR) Generator** and its generalization by **Mersenne Twister** now the default generator in Matlab, R. It has a period of $2^{19937} - 1$, is very fast and passes all practical statistical tests.

Python (numpy) uses a **Permutation Congruential Generator** (PCG XSL RR 128/64), a LFSR with a further permutation step to improve statistical performances. It has a period of 2^{128} with multiple streams and jump ahead capabilities.

Assessing the quality of a RNG

How to assess if a sequence $\boldsymbol{U} = (U_1, U_2, \ldots, U_n)$ produced by a (not-necessarily uniform) RNG has the right property, i.e. $U_i \overset{iid}{\sim} \mu$?

The suite *TestU01* developed by L'Ecuyer and Simard contains a comprehensive collection of statistical goodness-of-fit and independence tests.

- Let $U \in I \subset \mathbb{R}$ be a random variable with cumulative distribution function (CDF) $F(x) = \mathbb{P}(U \leq x)$.
- Let $\boldsymbol{U} = (U_1, \ldots, U_n)$ be a sample produced by a RNG with *empirical distribution function*

$$\hat{F}_n(x) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{\{U_i \leq x\}} = \frac{\#\{U_i \leq x\}}{n}$$

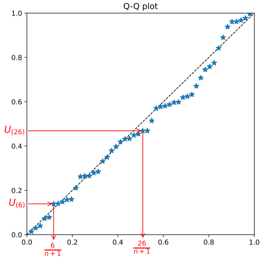
We want to test the hypothesis H_0 that $\boldsymbol{U}$ has been drawn independently from the distribution F.

Some non-parametric goodness-of-fit tests

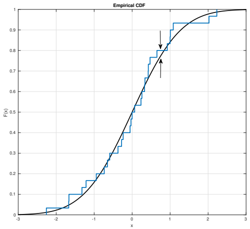
- **Q-Q plot**: plot the quantiles of $\hat{F}_n$ versus the corresponding quantiles of F. In particular, if $(U^{(1)}, U^{(2)}, \ldots, U^{(n)})$ denotes the ordered sample, then $U^{(j)}$ is a good estimator of the $\frac{j}{n+1}$ quantile, i.e.

  estimator: $$\hat{q}_{\frac{j}{n+1}} = U^{(j)},$$

  exact quantile: $$q_{\frac{j}{n+1}} = \underset{x}{\operatorname{argmin}}\{F(x) \geq \frac{j}{n+1}\}$$



- **Kolmogorov-Smirnov Test**: compute $D_n = \sup_x |\hat{F}_n(x) - F(x)|$.
  It is known that, under the null hypothesis H_0, $\sqrt{n}D_n$ converges in distribution to a Kolmogorov random variable, so we can reject H_0 at level α if $\sqrt{n}D_n > K_\alpha$ with K_α the α-quantile of K.



Some non-parametric goodness-of-fit tests

- χ^2 **test**: essentially, compares the histogram of the sample with the exact one. Let I_j, $j = 1, \ldots, m+1$ be a partition of I and let

$$p_j = \mathbb{P}(U \in I_j), \qquad N_j = \sum_{i=1}^{n} \mathbb{1}_{\{U_i \in I_j\}} = \#\{U_i \text{ that fall in } I_j\}$$

Then, under H_0, we have $\mathbb{E}[N_j] = np_j$ and

$$\hat{Q}_m = \sum_{j=1}^{m+1} \frac{(N_j - np_j)^2}{np_j}$$

has an asymptotic $\chi^2(m)$ distribution with m degrees of freedom ($m = \#$ classes $- 1$). Hence, we can reject the null hypothesis H_0 at level α if $\hat{Q}_m > q_{1-\alpha}$ where $q_{1-\alpha}$ is the $1-\alpha$ quantile of the $\chi^2(m)$ distribution.

Testing for independence

Consider a sequence $\boldsymbol{U} = (U_1, \dots, U_n)$ generated by a *uniform* RNG. We want to test the null hypothesis H_0 that $\{U_i\}_i$ are mutually independent and uniformly distributed in $(0, 1)$.

- **Serial test**: The idea is to group the sample in k blocks of lenght d (such that $kd = n$), $\mathbf{V}_j = (U_{(j-1)d+1}, \dots, U_{jd})$, $j = 1, \dots, k$ and test, e.g. by a χ^2 test, that the sample $\mathbf{V} = (\mathbf{V}_1, \dots, \mathbf{V}_k)$ has a *joint uniform distribution*.
- **Gap test**: Let T_1, $T_2, \dots$ denote the times when the sequence $\{U_i\}_{i=1}^n$ visits a given interval $(\alpha, \beta) \subset (0, 1)$ and let $Z_i = T_i - T_{i-1} - 1$ be the gap length between two consecutive visits (here $T_0 = 0$). Under H_0, Z_i are iid with a geometric distribution with parameter $p = \beta - \alpha$, i.e.

$$\mathbb{P}(Z = j) = p(1-p)^j, \quad j = 0, 1, 2, \dots.$$

  One can use a $\chi^2(r+1)$ test to test whether the $\{Z_i\}_i$ have the right geometric distribution, using the classes $Z = 0, Z = 1, \dots, Z = r$, $Z > r$.