Lecture (12/2) Pre-Final Presentations

CME 270

Pseudo-number generation

(1) Given uniformly distributed and independent $U_i \sim U(0,1)$, we can simulate iid random variables X_i of any desired distribution.

↘ Ashwin checked, and the Python source code actually does this! Cool!

Ex. Want i.i.d $X_i \sim Exp(\lambda)$, $\lambda > 0$. Define $X_i = -1/\lambda \ln(U_i)$. Then

$$P(X_i \leq x) = P(-1/\lambda \ln(U_i) \leq x)$$

↑ number (const.)

$$= P(U_i \geq \exp(-\lambda x))$$

$$= 1 - P(U_i \leq \exp(-\lambda x))$$

$$= 1 - \exp(-\lambda x)$$

exactly the CDF of $Exp(\lambda)$

$U_i \sim U(0,1)$

CDF

$$f(U_i \leq x) = \frac{x-0}{1-0}$$

$$= x$$

Ex. 2 Given independent $U_1, U_2 \sim U(0,1)$, get independent $X_1, X_2 \sim N(0,1)$.

The **Box Muller** transformation sets

$$X_1 = R\cos(\theta) \qquad R = \sqrt{-2\log(U_1)}$$

$$X_2 = R\sin(\theta) \qquad \theta = 2\pi U_2.$$

We can show (by factoring joint PDF of X_1 and X_2) that $X_1, X_2 \sim N(0,1)$ and are independent. ← Good exercise; see "A First Course in Probability" by R. Sheldon, pg. 279-281

How are U_i generated?

- **Linear Congruential Generator**

Algorithm: User picks some integers a, c, m s.t. $0 < a < m$, $0 \leq c < m$. Then pick any Z_0 (integer) s.t. $0 \leq Z_0 < c < m$. This Z_0 is called the seed. The update formulas are

$$U_n = Z_n/m \quad \leftarrow \text{always between 0 and 1}$$

$$Z_{n+1} = (aZ_n + c) \text{ modulo } (m)$$

↑ $Z_n < m$ always

Heuristics:

- m large*
- a and c picked s.t. $\{Z_n\}_n$ "cover" whole interval $\{0, \ldots, m-1\}$

Ex 2. (Bad) $m = 6$

$a = 1$

$c = 3$

$Z_{n+1} = Z_n + c$

cycle length 2

For seed $Z_0 = 1$,
the #'s generated are $\{1, \underset{+3}{4}, 1, 4, 1, \ldots\}$

Idea: want a "cycle length" of m

Similarly, all seeds have cycle length 2 for this (a, c).

* If we pick modulo $m = 2^{64}$, then computer does modulo arithmetic automatically (assuming 64 bit precision).

3/3