भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Programming for AI

Sampling Techniques: Rejection Sampling (Accept-Reject Sampling), Pseudo-Random Number Generators (PRNGs)

Karthik P. N.
Assistant Professor, Department of AI
Email: pnkarthik@ai.iith.ac.in

17 April 2025

Inverse Transform Technique

Objective

Given a cumulative distribution function (CDF) $F : \mathbb{R} \to [0, 1]$, generate a sample $X \sim F$.

- Sample $U \sim \mathrm{Unif}(0, 1)$
- Define the function F^{-1} as

$$F^{-1}(u) := \min\{x \in \mathbb{R} : F(x) \geq u\}$$

- F^{-1} is called the quantile function
- Set $X = F^{-1}(U)$
- **Claim:** The CDF of X is exactly equal to F, i.e., $F_X = F$

Example

- Let X be a discrete random variable with the following PMF:

$$p_X(x) = \begin{cases} 0.1, & x = 10, \\ 0.2, & x = 20, \\ 0.3, & x = 30, \\ 0.4, & x = 40, \\ 0, & \text{otherwise.} \end{cases}$$

Use the inverse transform method to generate a sample from the above distribution.

Example

- **[Generating a Sample from Rayleigh Distribution]**

  The PDF of the Rayleigh distribution is given by

$$f(r) = r\, e^{-r^2/2}, \quad r > 0.$$

  Use the inverse transform method to generate a sample from the above distribution.

Gaussian Samples on Python via ITT

- Python's built-in module

$$\texttt{numpy.random.normal(loc, scale, size)}$$

generates n independent samples from $\mathcal{N}(\mu, \sigma^2)$, where

$$n = \texttt{size}, \qquad \mu = \texttt{loc}, \qquad \sigma = \texttt{scale}.$$

- In principle, the above module uses the inverse transform technique

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

$\mathcal{N}(\mu, \sigma^2)$ Samples on Python via ITT

1. Let $U_1, U_2 \overset{\text{i.i.d.}}{\sim} \text{Unif}(0, 1)$
2. Let R and Θ be two random variables defined via

$$R = F_1^{-1}(U_1), \qquad \Theta = 2\pi U_2,$$

   where F_1 is the CDF of the Rayleigh distribution
3. Let Y_1 and Y_2 be defined as

$$Y_1 = R\,\cos(\Theta), \qquad Y_2 = R\,\sin(\Theta).$$

4. Then, $Y_1, Y_2 \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$
5. To get $X \sim \mathcal{N}(\mu, \sigma^2)$, simply discard Y_2, and

$$X = \sigma\, Y_1 + \mu.$$

6. Repeat steps 1-5 a total of n times to get $X_1, X_2, \ldots, X_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu, \sigma^2)$

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Rejection Sampling (Accept-Reject) Technique

Rejection Sampling (Accept-Reject) Technique

- Many a times, computing F^{-1} is tedious

Rejection Sampling (Accept-Reject) Technique

- Many a times, computing F^{-1} is tedious
- For instance, consider the PDF

$$f(x) = 6x(1-x), \qquad x \in [0,1]$$

Rejection Sampling (Accept-Reject) Technique

- Many a times, computing F^{-1} is tedious
- For instance, consider the PDF

$$f(x) = 6x(1-x), \qquad x \in [0, 1]$$

- The CDF corresponding to the above PDF is given by

$$F(x) = \begin{cases} 0, & x < 0, \\ 3x^2 - 2x^3, & 0 \leq x < 1, \\ 1, & x \geq 1. \end{cases}$$

Rejection Sampling (Accept-Reject) Technique

- Many a times, computing F^{-1} is tedious
- For instance, consider the PDF

$$f(x) = 6x(1-x), \qquad x \in [0,1]$$

- The CDF corresponding to the above PDF is given by

$$F(x) = \begin{cases} 0, & x < 0, \\ 3x^2 - 2x^3, & 0 \le x < 1, \\ 1, & x \ge 1. \end{cases}$$

- Computation of F^{-1} is non-trivial; need to use numerical techniques

Rejection Sampling (Accept-Reject) Technique

- Many a times, computing F^{-1} is tedious
- For instance, consider the PDF

$$f(x) = 6x(1-x), \qquad x \in [0,1]$$

- The CDF corresponding to the above PDF is given by

$$F(x) = \begin{cases} 0, & x < 0, \\ 3x^2 - 2x^3, & 0 \leq x < 1, \\ 1, & x \geq 1. \end{cases}$$

- Computation of F^{-1} is non-trivial; need to use numerical techniques
- **Alternative solution:** Rejection sampling!

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

- Let U, Z be a pair of continuous random variables satisfying the following properties:

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \text{Unif}(0, 1)$.

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \mathrm{Unif}(0, 1)$.
  2. U is independent of Z.

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \mathrm{Unif}(0, 1)$.
  2. U is independent of Z.
  3. There exists a real number $a \geq 1$ such that

$$f(x) \leq a f_Z(x) \qquad \forall x \in \mathbb{R},$$

  where f_Z is the PDF of Z

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \text{Unif}(0, 1)$.
  2. U is independent of Z.
  3. There exists a real number $a \geq 1$ such that

     $$f(x) \leq a f_Z(x) \qquad \forall x \in \mathbb{R},$$

     where f_Z is the PDF of Z
- Generate (U, Z) satisfying 1-3 above

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \mathrm{Unif}(0, 1)$.
  2. U is independent of Z.
  3. There exists a real number $a \geq 1$ such that

     $$f(x) \leq a\, f_Z(x) \qquad \forall x \in \mathbb{R},$$

     where f_Z is the PDF of Z
- Generate (U, Z) satisfying 1-3 above
  (i) If $a\, U\, f_Z(Z) \leq f(Z)$, accept (U, Z), and set $X = Z$.

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Rejection Sampling (Accept-Reject) Technique

Objective

Given a PDF f with CDF F for which computing F^{-1} is tedious, generate a sample $X \sim f$.

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \mathrm{Unif}(0, 1)$.
  2. U is independent of Z.
  3. There exists a real number $a \geq 1$ such that

$$f(x) \leq a\, f_Z(x) \qquad \forall x \in \mathbb{R},$$

  where f_Z is the PDF of Z
- Generate (U, Z) satisfying 1-3 above
  (i) If $a\, U\, f_Z(Z) \leq f(Z)$, accept (U, Z), and set $X = Z$.
  (ii) If not, reject current (U, Z) and generate new (U, Z). Repeat till (i) holds.

Rejection Sampling (Accept-Reject) Technique

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \text{Unif}(0,1)$.
  2. U is independent of Z.
  3. There exists a real number $a \geq 1$ such that

$$f(x) \leq a f_Z(x) \qquad \forall x \in \mathbb{R},$$

     where f_Z is the PDF of Z
- Generate (U, Z) satisfying 1-3 above
  (i) If $a\, U f_Z(Z) \leq f(Z)$, accept (U, Z), and set $X = Z$.
  (ii) If not, reject current (U, Z) and generate new (U, Z). Repeat till (i) holds.

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Rejection Sampling (Accept-Reject) Technique

- Let U, Z be a pair of continuous random variables satisfying the following properties:
  1. $U \sim \mathrm{Unif}(0,1)$.
  2. U is independent of Z.
  3. There exists a real number $a \geq 1$ such that
  $$f(x) \leq a\, f_Z(x) \qquad \forall x \in \mathbb{R},$$
  where f_Z is the PDF of Z
- Generate (U, Z) satisfying 1-3 above
  (i) If $a\, U\, f_Z(Z) \leq f(Z)$, accept (U, Z), and set $X = Z$.
  (ii) If not, reject current (U, Z) and generate new (U, Z). Repeat till (i) holds.

Theorem (Rejection Sampling)

Let $E = \{a\, U\, f_Z(Z) \leq f(Z)\}$. Then,

$$F_X(x) = \mathbb{P}(X \leq x) = \mathbb{P}(Z \leq x \mid E).$$

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Example

- Design an algorithm to generate a sample $X \sim f$, where

$$f(x) = 6x(1-x), \qquad x \in [0,1].$$

Example

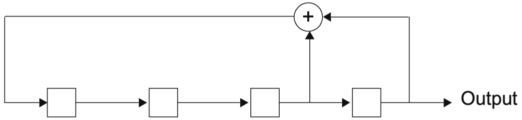
- For fixed constants $\lambda, t > 0$, the $\mathrm{Gamma}(\lambda, t)$ PDF is given by

$$f(x) = e^{-\lambda x} \frac{\lambda^t x^{t-1}}{\Gamma(t)}, \qquad x > 0.$$

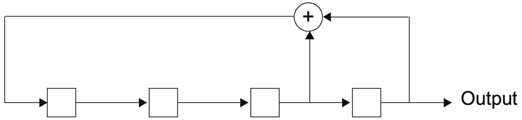
1. When $t \in \mathbb{N}$, suggest a technique to generate a sample $X \sim f$ via ITT.
2. When $t \notin \mathbb{N}$, design an algorithm to sample $X \sim f$ via rejection sampling.
   Hint: Take $Z \sim \mathrm{Exponential}(1/t)$.

Pseudo-Random Number Generators (PRNGs)

Binary PRNGs

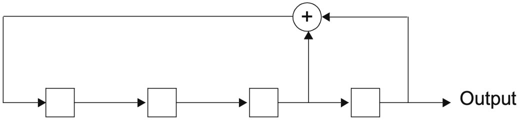


Binary PRNGs



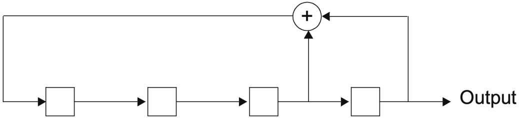
$S_0S_1S_2S_3$	Output
1111	1

Binary PRNGs



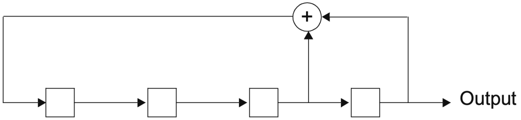
$S_0S_1S_2S_3$	Output
1111	1
0111	1

Binary PRNGs



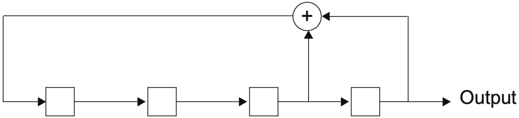
$S_0S_1S_2S_3$	Output
1111	1
0111	1
0011	1

Binary PRNGs



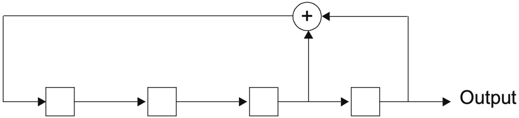
$S_0 S_1 S_2 S_3$	Output
1111	1
0111	1
0011	1
0001	1

Binary PRNGs



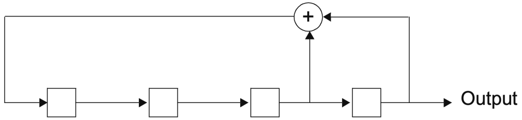
$S_0S_1S_2S_3$	Output
1111	1
0111	1
0011	1
0001	1
1000	0

Binary PRNGs



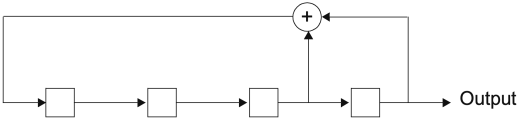
$S_0S_1S_2S_3$	Output
1111	1
0111	1
0011	1
0001	1
1000	0
0100	0

Binary PRNGs



$S_0S_1S_2S_3$	Output
1111	1
0111	1
0011	1
0001	1
1000	0
0100	0
0010	0

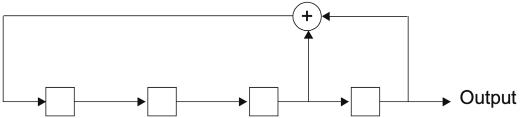
Binary PRNGs



$S_0S_1S_2S_3$	Output
1111	1
0111	1
0011	1
0001	1
1000	0
0100	0
0010	0

$S_0S_1S_2S_3$	Output
1001	1
1100	0
0110	0
1011	1
0101	1
1010	0
1101	1
1110	0

Binary PRNGs

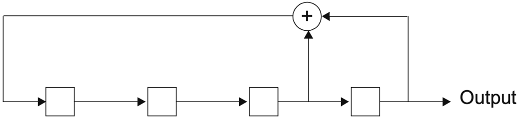


$S_0S_1S_2S_3$	Output
1111	1
0111	1
0011	1
0001	1
1000	0
0100	0
0010	0

$S_0S_1S_2S_3$	Output
1001	1
1100	0
0110	0
1011	1
0101	1
1010	0
1101	1
1110	0

Output (one period): 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0

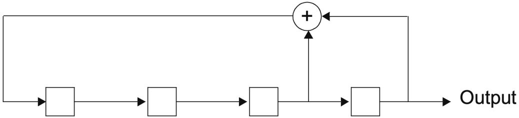
Properties of the Binary PRNG



Output (one period): $1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0$

- Number of zeros in one period $\approx$ number of ones in one period (desirable of uniform binary random number generator)

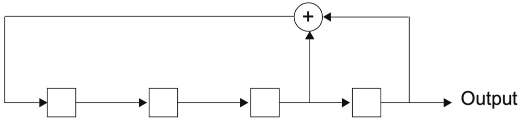
Properties of the Binary PRNG



Output (one period): $1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0$

- Number of zeros in one period $\approx$ number of ones in one period
  (desirable of uniform binary random number generator)
- Period = 15
  (not desirable of uniform binary random number generator)

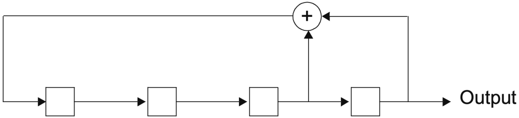
Properties of the Binary PRNG



Output (one period): $1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0$

- Number of zeros in one period $\approx$ number of ones in one period (desirable of uniform binary random number generator)
- Period = 15 (not desirable of uniform binary random number generator)

Properties of the Binary PRNG



Output (one period): $1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0$

- Number of zeros in one period $\approx$ number of ones in one period
  (desirable of uniform binary random number generator)
- Period = 15
  (not desirable of uniform binary random number generator)

Possible Workaround for Periodicity in Output

Increase the number of stages N.

N-Stage Binary PRNG

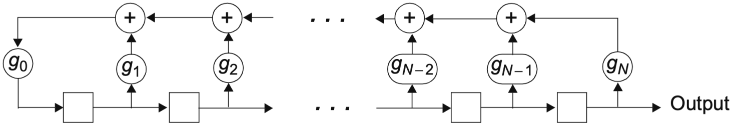


Figure: N-Stage, binary linear feedback shift register.

N-Stage Binary PRNG

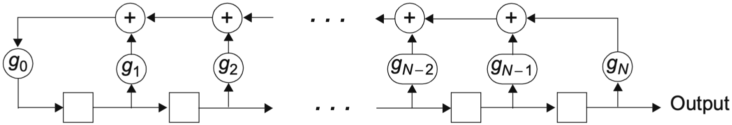


Figure: N-Stage, binary linear feedback shift register.

- $g_0 = g_N = 1$

N-Stage Binary PRNG

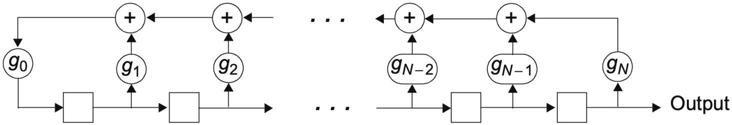


Figure: N-Stage, binary linear feedback shift register.

- $g_0 = g_N = 1$
- Adjust the tap gains $\{g_1, \ldots, g_{N-1}\}$ to achieve highest possible period $(= 2^N - 1)$

N-Stage Binary PRNG

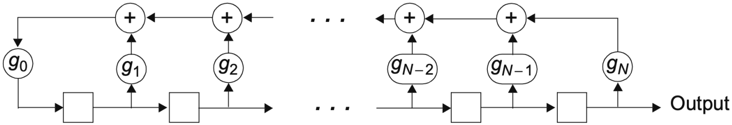


Figure: N-Stage, binary linear feedback shift register.

- $g_0 = g_N = 1$
- Adjust the tap gains $\{g_1, \ldots, g_{N-1}\}$ to achieve highest possible period $(= 2^N - 1)$
- E.g., for $N = 4$, set

$$(g_0, g_1, g_2, g_3, g_4) = (1, 0, 0, 1, 1) = (23)_8.$$

N-Stage Binary PRNG

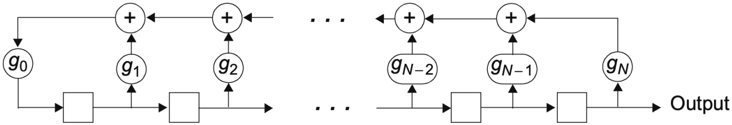


Figure: N-Stage, binary linear feedback shift register.

- $g_0 = g_N = 1$
- Adjust the tap gains $\{g_1, \ldots, g_{N-1}\}$ to achieve highest possible period ($= 2^N - 1$)
- E.g., for $N = 4$, set

$$(g_0, g_1, g_2, g_3, g_4) = (1, 0, 0, 1, 1) = (23)_8.$$

- Maximal period sequences are called m-sequences

Commonly Used Feedback Connections

SR Length, N	Feedback Connections (in Octal Format)
2	7
3	13
4	23
5	45, 67, 75
6	103, 147, 155
7	203, 211, 217, 235, 277, 313, 325, 345, 367
8	435, 453, 537, 543, 545, 551, 703, 747

Figure: Non-exhaustive list of feedback connections to obtain m-sequences.

Properties of m-Sequences

- Are periodic with period $= 2^N - 1$

Properties of m-Sequences

- Are periodic with period $= 2^N - 1$
- Contain approximately equal number of ones and zeros in any one period

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Properties of m-Sequences

- Are periodic with period $= 2^N - 1$
- Contain approximately equal number of ones and zeros in any one period
- Autocorrelation function is nearly identical to that of IID $\mathrm{Ber}(0.5)$ process

Non-Binary PRNGs

- How do we generate non-binary m-sequences?

Non-Binary PRNGs

- How do we generate non-binary m-sequences?
- One technique is the power residue method

Non-Binary PRNGs

- How do we generate non-binary m-sequences?
- One technique is the power residue method
- Given numbers (a, x_0, p), let

$$x_n = a x_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

Non-Binary PRNGs

- How do we generate non-binary m-sequences?
- One technique is the power residue method
- Given numbers (a, x_0, p), let

$$x_n = a x_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

  - p is typically a large prime (e.g., $p = 2^{31} - 1 = 2147483647$)

Non-Binary PRNGs

- How do we generate non-binary m-sequences?
- One technique is the power residue method
- Given numbers (a, x_0, p), let

$$x_n = ax_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

  - p is typically a large prime (e.g., $p = 2^{31} - 1 = 2147483647$)
  - $x_0 \neq 0$ is called the seed value (e.g., $x_0 = 12345$)

Non-Binary PRNGs

- How do we generate non-binary m-sequences?
- One technique is the power residue method
- Given numbers (a, x_0, p), let

$$x_n = ax_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

  - p is typically a large prime (e.g., $p = 2^{31} - 1 = 2147483647$)
  - $x_0 \neq 0$ is called the seed value (e.g., $x_0 = 12345$)
  - $a \neq 1$ is the multiplicative factor

Non-Binary PRNGs

- How do we generate non-binary m-sequences?
- One technique is the power residue method
- Given numbers (a, x_0, p), let

$$x_n = a x_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

  - p is typically a large prime (e.g., $p = 2^{31} - 1 = 2147483647$)
  - $x_0 \neq 0$ is called the seed value (e.g., $x_0 = 12345$)
  - $a \neq 1$ is the multiplicative factor
- Due to $(\bmod\ p)$ operation, $x_n \in \{1, \ldots, p-1\}$ for all n

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Non-Binary PRNGs

- How do we generate non-binary m-sequences?
- One technique is the power residue method
- Given numbers (a, x_0, p), let

$$x_n = ax_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

  - p is typically a large prime (e.g., $p = 2^{31} - 1 = 2147483647$)
  - $x_0 \neq 0$ is called the seed value (e.g., $x_0 = 12345$)
  - $a \neq 1$ is the multiplicative factor
- Due to $(\bmod\ p)$ operation, $x_n \in \{1, \ldots, p-1\}$ for all n
- The choice of (a, p) is crucial to obtain an m-sequence

Non-Binary PRNGs

Recursion

$$x_n = ax_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

- If $(a, p) = (4, 7)$, then output sequence (with $x_0 = 1$) is

Non-Binary PRNGs

Recursion

$$x_n = ax_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

- If $(a, p) = (4, 7)$, then output sequence (with $x_0 = 1$) is

$$(1, 4, 2, 1, 4, 2, \cdots)$$

- If $(a, p) = (3, 7)$, then output sequence (with $x_0 = 1$) is

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Non-Binary PRNGs

Recursion

$$x_n = ax_{n-1} \bmod p, \qquad n \in \mathbb{N}.$$

- If $(a, p) = (4, 7)$, then output sequence (with $x_0 = 1$) is

$$(1, 4, 2, 1, 4, 2, \cdots)$$

- If $(a, p) = (3, 7)$, then output sequence (with $x_0 = 1$) is

$$(1, 3, 2, 6, 4, 5, 1, 3, 2, 6, 4, 5, \cdots)$$

- In most programming languages:
  - $a = 7^5, \quad p = 2^{31} - 1.$
  - Output normalised to take values in $\left\{\frac{1}{p}, \frac{2}{p}, \ldots, \frac{p-1}{p}\right\}$