

CENG 222

Statistical Methods for Computer Engineering

Week 6

Chapter 5

Computer Simulations and Monte Carlo Methods

Outline

- Generation of random numbers from specific distributions
 - Discrete distributions
 - Continuous distributions
- Chebyshev's inequality (3.3.7)
- Solving problems by Monte Carlo methods
 - Estimating probabilities
 - Estimating means and standard deviations

Uniform Random Numbers

- Tables of random numbers
- Pseudo-random number generators
 - Long sequences of random-looking numbers
 - Seed: starting location in the sequence
 - May use system time as seed
- Many systems provide standard uniform random number generators
 - $\text{Uniform}(0,1)$
- Question: Can we generate random numbers from any distribution using $\text{Uniform}(0,1)$ rvs?

Bernoulli

- Let U be Uniform(0,1)
- $X = \begin{cases} 1, & \text{if } U < p \\ 0, & \text{if } U \geq p \end{cases}$
- $P(\text{success}) = P(U < p) = p$

Binomial

- Sum of n independent Bernoulli variables.
- Example

```
n = 20; p = 0.68;
```

```
U = rand(n,1);
```

```
% generates an nx1 vector
```

```
% of uniform random numbers
```

```
X = sum(U < p);
```

Geometric

- Iterate and count the number of generated rvs until first success

- Example:

```
p = 0.16; X = 1;
```

```
while rand > p;
```

```
    X = X+1;
```

```
end;
```

```
X
```

Negative Binomial

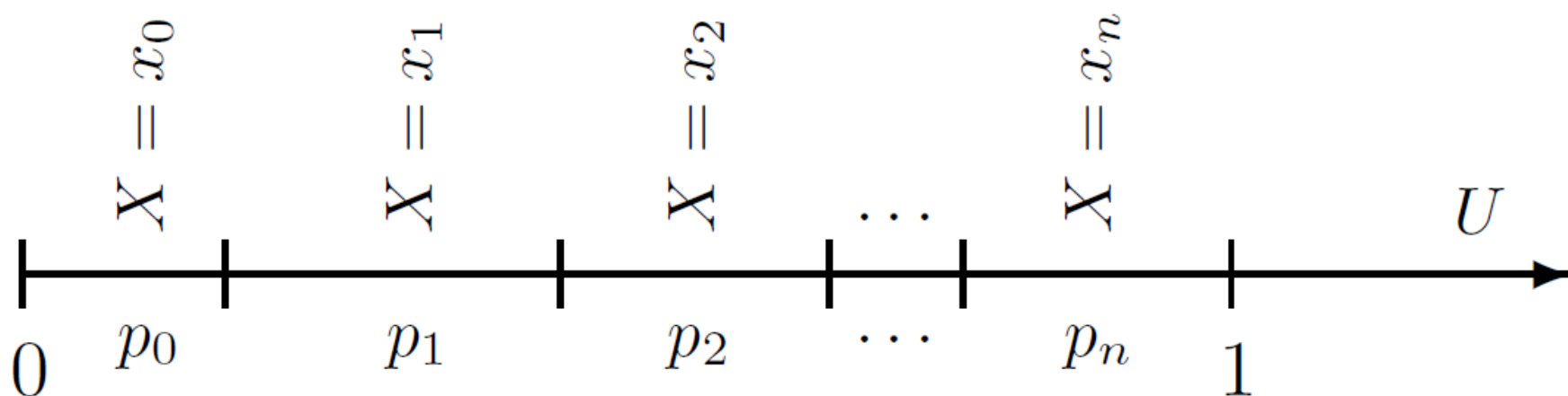
- Generate k independent $\text{Geometric}(p)$ random numbers and sum them to get a $\text{NegativeBinomial}(k,p)$ number.

- Example:

```
p = 0.16; X = 0; i = 0;
while i < k;
    G = 1;
    while rand > p;
        G = G+1;
    end;
    X = X+G;
end;
```

- How efficient is generating a Binomial, a Geometric, or a Negative Binomial random number?

Arbitrary discrete distributions



Algorithm 5.1

1. Divide the interval $[0,1]$ into subintervals A_i as follows:
 - $A_i = [p_0 + p_1 + \dots + p_{i-1}, p_0 + p_1 + \dots + p_i)$
2. Generate U , a standard uniform number
3. If U belongs to A_i then $X = x_i$
- How efficient is this method?
 - If you want to generate many X s, efficiency is important.
 - $O(n)$, $O(\log n)$, $O(1)$?
 - Check out the *Alias Method*, if you need an $O(1)$ method.

Poisson

- Using Algorithm 5.1 to generate Poisson numbers.
- Example:

```
lambda = 5;  
U = rand; i = 0;  
F = exp(-lambda); % F(0)  
while (U >= F);  
    i = i + 1;  
    F = F + exp(-lambda) * lambda^i / gamma(i+1);  
end;  
X = i;
```

Inverse transform method

- Theorem: $U = F_X(X)$ is Uniform(0,1)
- Proof:
 - Note that the standard uniform cdf is $F_U(u) = u$ (i.e., $F'_U(u) = f_U(u) = 1$). We will try to show this fact using the given definition of $U = F_X(X)$
 - $$\begin{aligned} F_U(u) &= P(U \leq u) \\ &= P(F_X(X) \leq u) \\ &= P(X \leq F_X^{-1}(u)) \\ &= F_X(F_X^{-1}(u)) \\ &= u \end{aligned}$$

Inverse transform method

- If $U = F_X(X)$ then $X = F_X^{-1}(U)$
- The method:
 - Generate a uniform random number
 - Plug it in F_X^{-1} to generate X (i.e. solve for X).
- Example 5.10 (Exponential):
 - $F_X(X) = 1 - e^{-\lambda X} = U$
 - $\rightarrow X = -\frac{1}{\lambda} \ln(1 - U)$
 - Can also use $X = -\frac{1}{\lambda} \ln(U)$ since $1-U$ is also Uniform(0,1).

Inverse transform method

- Difficult to use if the inverse of the cdf is not easy to compute
- For example, for discrete distributions, $F_X^{-1}(U)$ does not exist. $U = F_X(X)$ has no roots, because X (hence $F_X(X)$) is finite and countable; whereas U is continuous.
- Therefore, for discrete rvs, we use the inverse method with a slight modification:
 - $X = \min \{x \in S \text{ such that } F(x) > U\}$ where S is the set of possible values of X .

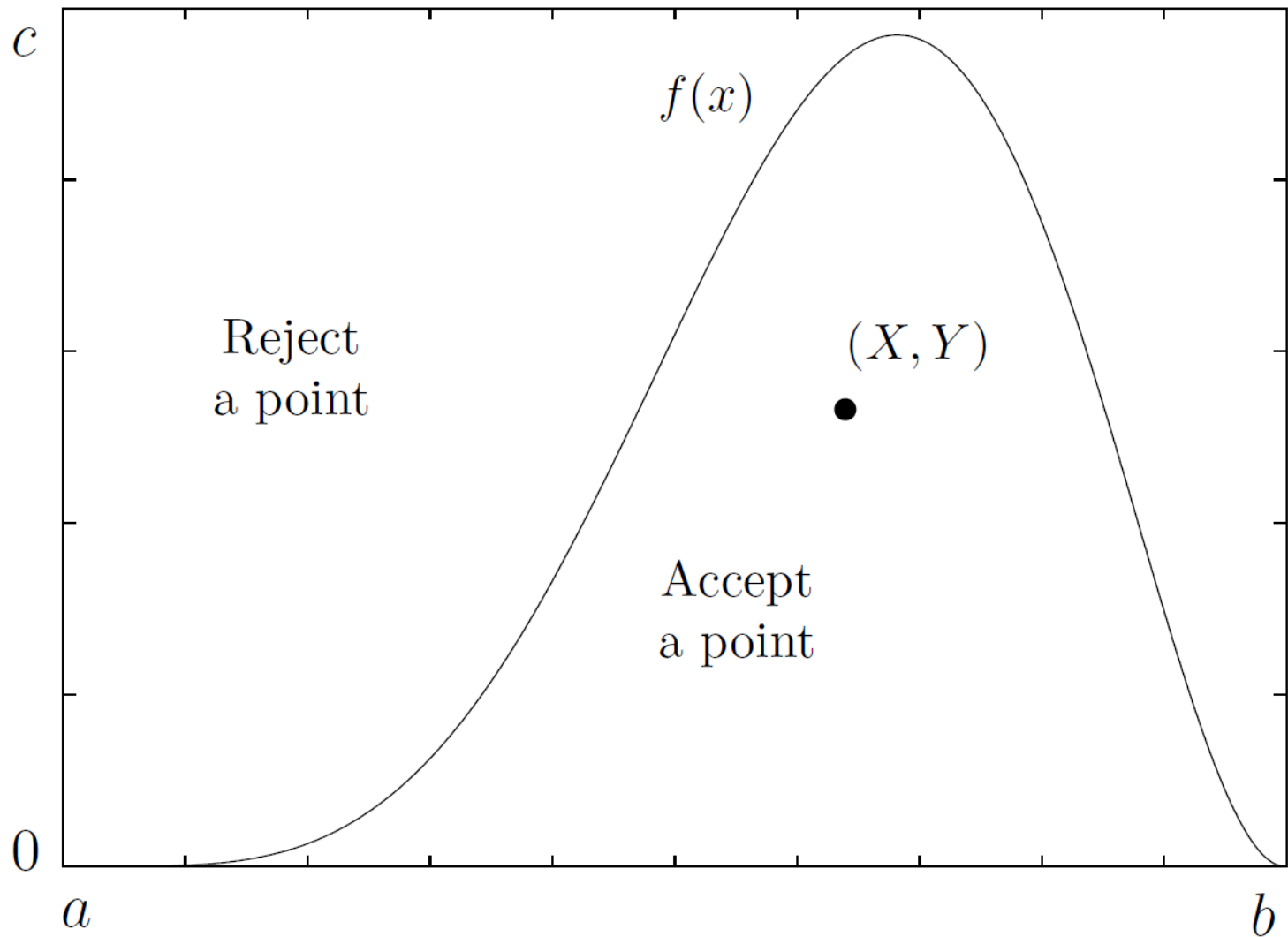
Example 5.12

- Using the inverse transform method for generating Geometric variables
- $X = \left\lceil \frac{\ln(1-U)}{\ln(1-p)} \right\rceil$
- The geometric variable is the ceiling of the exponential variable with $\lambda = -\ln(1 - p)$
 - Exponential is the continuous analogue of geometric
 - Both have the memoryless property.

Rejection method

- When the cdf is difficult to solve for X and the pdf f_X is available, the rejection method can be used to generate random numbers from f_X .
- Idea:
 - Generate 2D uniform coordinates (X, Y) in the bounding box of f_X and if $Y \leq f_X(X)$ output X .

Rejection method



Example

- The figure in the previous slide is the pdf of Beta($\alpha=5.5, \beta=3.1$)

$$- f_X = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1} \quad \text{for } 0 \leq x \leq 1$$

- Bounding box: $m = 2.5, s = 0, t = 1$.

```
a=5.5; b=3.1; s=0; t=1; m=2.5;
```

```
X = 0; Y = m;
```

```
F = gamma(a+b)/gamma(a)/gamma(b)*X^(a-1)*(1-X)^(b-1);
```

```
while (Y > F);
```

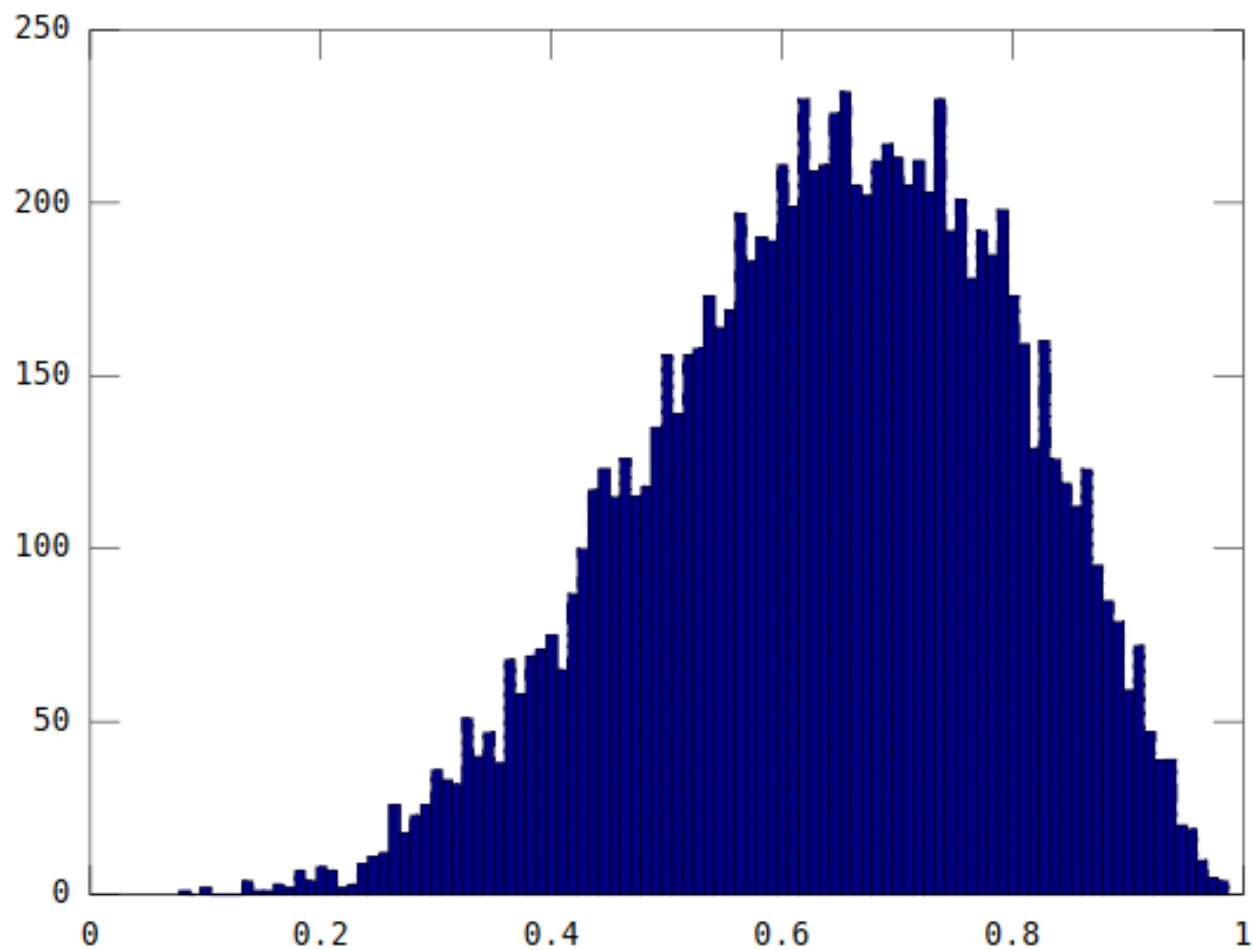
```
    U = rand; V = rand;
```

```
    X = s+(t-s)*U; Y = m*V;
```

```
    F = ... % same as above;
```

```
end; X
```

Example



Monte Carlo methods

- Generate many random variables from a distribution and estimate probabilities, means, standard deviations, etc. by simulating what happens in the long run.
- Question: How many numbers needed for acceptable results?
 - i.e., What will be the “size” of the Monte Carlo experiment?
 - Revisit Chebyshev’s Inequality

Chebyshev's Inequality (3.3.7)

- For any distribution with expectation μ and variance σ^2 and for any positive ε
 - $P(|X - \mu| > \varepsilon) \leq \left(\frac{\sigma}{\varepsilon}\right)^2$
 - In other words: any random variable X from the distribution is within ε distance of the μ with probability of at least $1 - (\sigma / \varepsilon)^2$

Estimating probabilities

- The probability $p=P(X \in A)$ can be estimated as \hat{p} by generating N random numbers and computing the proportion of random numbers that are in A .
 - How accurate is the estimator?
 - What is $E(\hat{p})$ and $\text{Std}(\hat{p})$?
 - The number of X_i that are in A among the N generated random numbers is $\text{Binomial}(N,p)$ with expectation Np and variance $Np(1-p)$
- $E(\hat{p}) = p$ (unbiased estimator)

$$\text{Std}(\hat{p}) = \sqrt{\frac{p(1-p)}{N}} \quad \text{the error in } \hat{p} \text{ decreases with } 1/\sqrt{N}$$

How large should N be?

- Given the error ε and the probability, α , to exceed this error limit
- If an intelligent guess p^* on the value of p is available:

$$- N \geq p^*(1 - p^*) \left(\frac{z_{\alpha/2}}{\varepsilon} \right)^2$$

- If not:

$$- N \geq 0.25 \left(\frac{z_{\alpha/2}}{\varepsilon} \right)^2$$

Example 5.14

How large should N be?

- If the N returned by these equations are not large enough for Binomial approximation, we may use Chebyshev's inequality:
 - If an intelligent guess p^* on the value of p is available:
 - $N \geq \frac{p^*(1-p^*)}{\alpha \varepsilon^2}$
 - If not:
 - $N \geq \frac{1}{4\alpha \varepsilon^2}$

Estimating means and standard deviations

- $\bar{X} = \frac{1}{N} (X_1 + \dots + X_N)$
 - also unbiased and its error decreases with $1/\sqrt{N}$
- $s^2 = \frac{1}{N-1} \sum_{i=1}^N (X_i - \bar{X})^2$
 - $1/N-1$ needed so that $\mathbf{E}(s^2) = \sigma^2$