STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

Variance Reduction Methods

SYS 611: Systems Modeling and Simulation

Paul T. Grogan, Ph.D.
Assistant Professor
School of Systems and Enterprises

Agenda

1. Buffon's Needle Model Implementation
2. Variance Reduction Methods

Reference: S.M. Ross, "Variance Reduction Techniques," Ch. 9 in *Simulation*, 2012.

Buffon's Needle Model

Modeling Buffon's Needle (1)

1. Identify **random variables** with probability distributions

D: distance from needle midpoint to nearest line

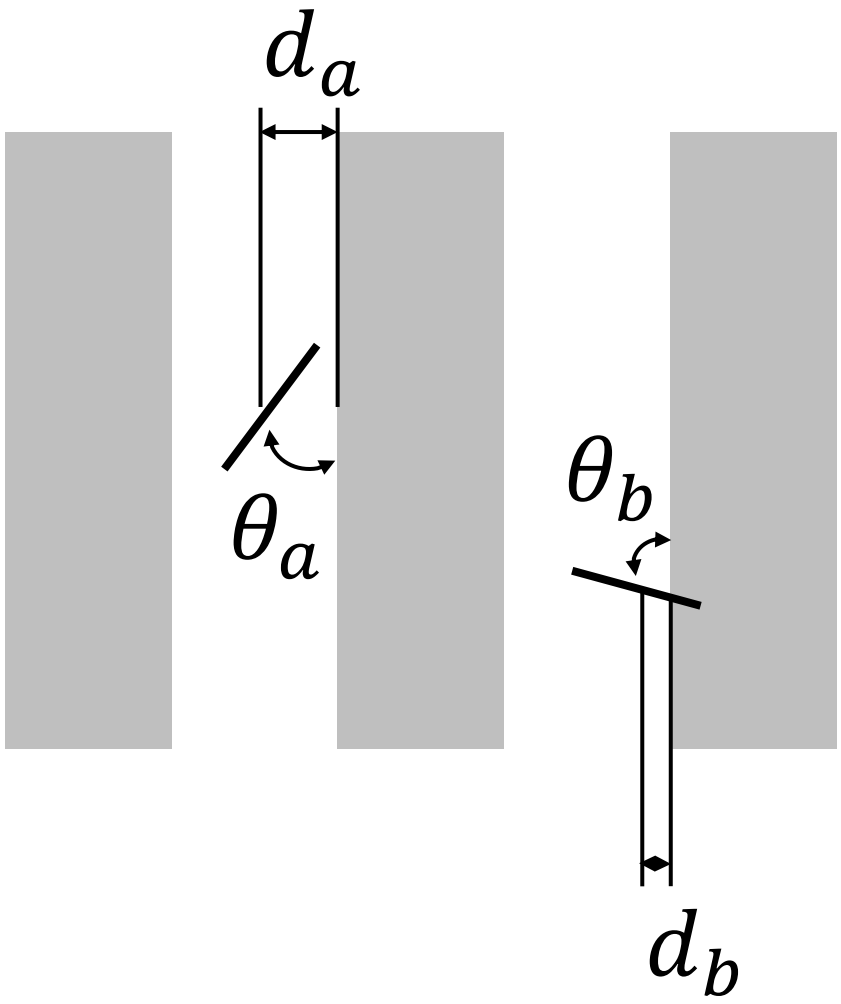
$$D \sim \text{uniform}(a = 0, b = T/2)$$

$$f(d) = 2/T$$

Θ: acute angle between needle and nearest line

$$\Theta \sim \text{uniform}(a = 0, b = \pi/2)$$

$$f(\theta) = 2/\pi$$



Modeling Buffon's Needle (2)

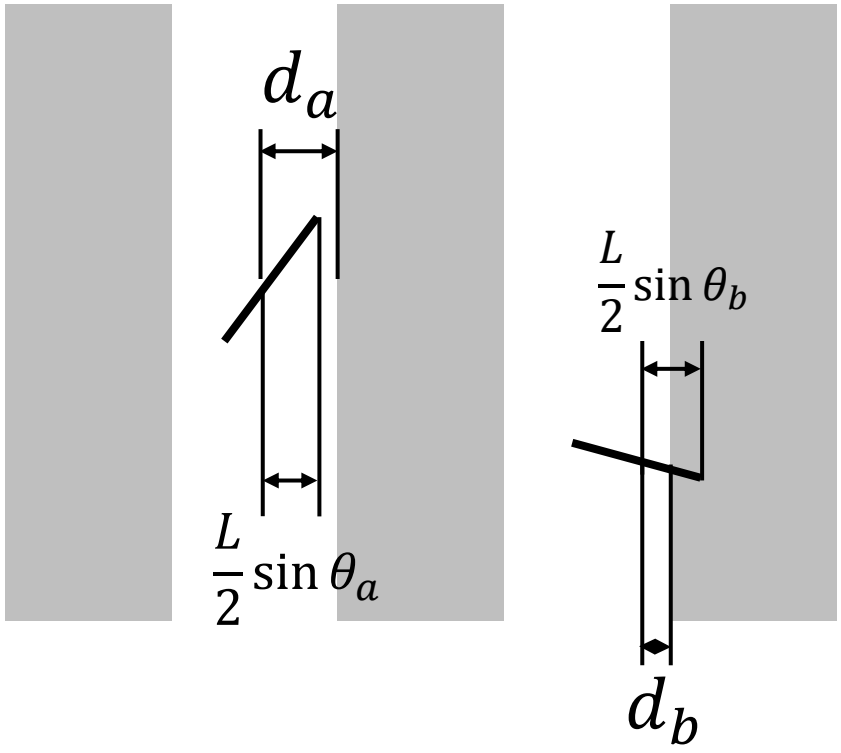
2. Identify **derived variables** and their functional form

$$X: \text{needle crosses line}$$

$$X = \begin{cases} 1 & \text{if } D \leq \frac{L}{2}\sin\Theta \\ 0 & \text{otherwise} \end{cases}$$

$$X \sim \text{bernoulli}(p)$$

(but do not know p!)



Modeling Buffon's Needle (3)

3. Determine **number of samples** required or other convergence criteria

- Want 95% confidence and ± 0.01 accuracy
  - $(1-\alpha)\cdot 100\%$ confidence interval: $\boldsymbol{\bar{x} \pm z_{1-\alpha/2} \frac{\sigma_x}{\sqrt{N}}}$
  - 95% confidence: $z_{1-\alpha/2} = z_{0.975} = 1.96$

$$0.01 = z_{1-\alpha/2}\frac{\sigma_x}{\sqrt{N}} \qquad \Rightarrow N = \left(\frac{1.96\cdot\sigma_x}{0.01}\right)^2 = 38416\cdot{\sigma_x}^2$$

  - Variance of Bernoulli distribution: $\sigma_x^2 = p\cdot(1-p)$

$$p \approx 0.5 \qquad \Rightarrow N = 38416\cdot 0.5(1-0.5) = 9604$$

Buffon's Needle Simulation

4. For each sample, **generate RVs** and compose and record **derived variables**

5. Visualize statistics from derived state variables

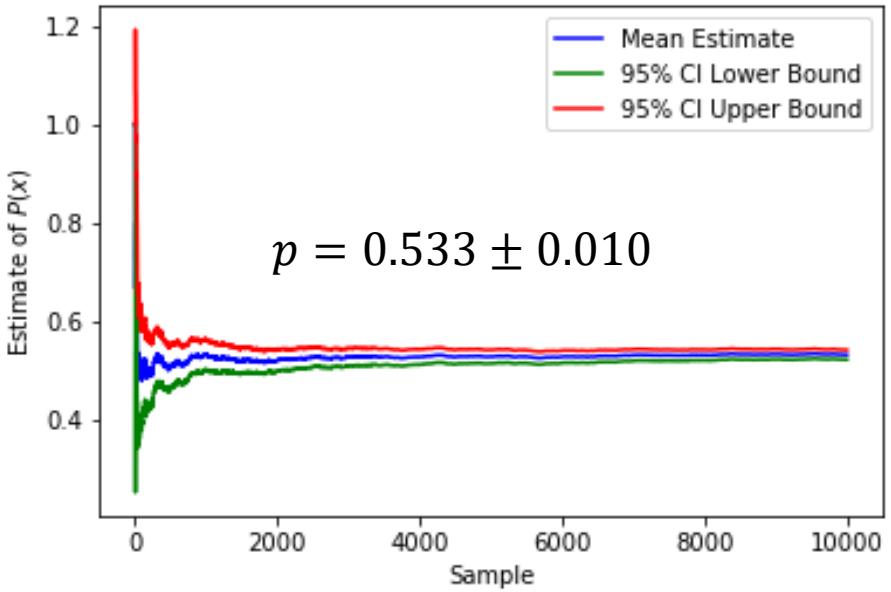
```
import numpy as np
import scipy.stats as stats

line_width = 3.0
needle_length = 2.5

def drop_needle():
  d = np.random.rand()*line_width/2
  theta = np.random.rand()*np.pi/2
  if d < needle_length/2*np.sin(theta):
    return True
  else:
    return False

samples = [drop_needle()
           for i in range(10000)]

print(np.mean(samples))
print(1.96*stats.sem(samples))
```



Analytical Solution

$$p = \iint_{d,\theta} X \cdot f(d,\theta)\, dd\, d\theta$$

$$= \int_{\theta=0}^{\pi/2} \int_{d=0}^{\frac{L}{2}\sin\theta} 1 \cdot f(d) \cdot f(\theta)\, dd\, d\theta$$

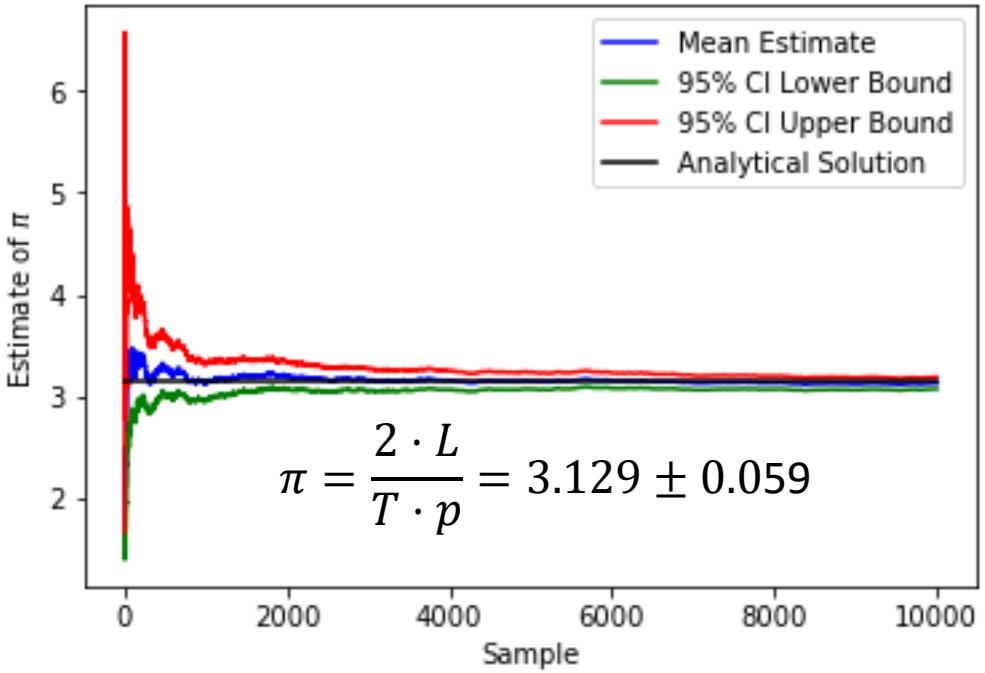
$$= \int_{\theta=0}^{\pi/2} \int_{d=0}^{\frac{L}{2}\sin\theta} \frac{2}{T} \cdot \frac{2}{\pi}\, dd\, d\theta$$

$$= \int_{\theta=0}^{\pi/2} \frac{2 \cdot L}{T \cdot \pi} \cdot \sin\theta\, d\theta$$

$$= -\frac{2 \cdot L}{T \cdot \pi} \cos\theta \Big|_{\theta=0}^{\frac{\pi}{2}} = \frac{2 \cdot L}{T \cdot \pi}$$

$$= \frac{2 \cdot 2.5}{3 \cdot \pi} = 0.5305$$

$$\Rightarrow \boldsymbol{\pi = \frac{2 \cdot L}{T \cdot p}}$$



Excel Implementation (1)

Define columns for the random variables (using process generators)

$$D \sim U(0, T/2) \Rightarrow f(d) = \frac{2}{T}, F(d) = \frac{2 \cdot d}{T} \Rightarrow \boldsymbol{d = \frac{r \cdot T}{2}}$$

$$\Theta \sim U(0, \pi/2) \Rightarrow f(\theta) = \frac{2}{\pi}, F(\theta) = \frac{2 \cdot \theta}{\pi} \Rightarrow \boldsymbol{\theta = \frac{r \cdot \pi}{2}}$$

	A	B	C	D	E	F
1	r1	d	r2	theta		
2	0.566645	=A2*3/2	0.358502	0.563134		
3	0.483976	0.725964	0.795111	1.248957		
4	0.181968	0.272952	0.729009	1.145125		
5	0.888497	1.332746	0.801691	1.259293		
6	0.215872	0.323808	0.109275	0.171648		
7	0.020329	0.030493	0.873607	1.372259		

	A	B	C	D	E	F
1	r1	d	r2	theta		
2	0.733606	1.100409	0.80425	=C2*PI()/2		
3	0.971178	1.456768	0.316316	0.496868		
4	0.584758	0.877136	0.22941	0.360356		
5	0.5151	0.772651	0.110767	0.173992		
6	0.557662	0.836492	0.286458	0.449967		
7	0.657947	0.98692	0.03508	0.055104		

Excel Implementation (2)

Define a column with the derived state variable

$$X = \begin{cases} 1, & \text{if } d \le \frac{L}{2}\sin\theta \\ 0, & \text{otherwise} \end{cases}$$

	A	B	C	D	E	F	G
1	r1	d	r2	theta	x		
2	0.5261	0.78915	0.169406	0.266102	=IF(B2<=2.5/2*SIN(D2),1,0)		
3	0.248706	0.373059	0.092523	0.145335	0		
4	0.150005	0.225008	0.032889	0.051662	0		
5	0.677325	1.015987	0.452693	0.711089	0		
6	0.906479	1.359719	0.858466	1.348475	0		
7	0.435767	0.65365	0.794133	1.247421	1		

Excel Implementation (3)

- Fill down the equations to generate samples
- Compute statistics and plots as needed

	A	B	C	D	E	F	G	H	I
1	r1	d	r2	theta	x				
2	0.100125	0.150187	0.797385	1.252529	1		x_bar =	=AVERAGE(E:E)	
3	0.407326	0.610989	0.088386	0.138837	0		st.dev =	0.506	
4	0.321878	0.482817	0.420854	0.661076	1		sem =	0.0175	
5	0.362576	0.543864	0.553215	0.868988	1				
6	0.184373	0.276559	0.149052	0.23413	1				
7	0.9891	1.48365	0.367504	0.577274	0				

Python Implementation (1)

Define functions to sample the elementary random variables (using process generators)

$$D \sim U(0, T/2) \Rightarrow f(d) = \frac{2}{T}, F(d) = \frac{2 \cdot d}{T} \Rightarrow \boldsymbol{d = \frac{r \cdot T}{2}}$$

$$\Theta \sim U(0, \pi/2) \Rightarrow f(\theta) = \frac{2}{\pi}, F(\theta) = \frac{2 \cdot \theta}{\pi} \Rightarrow \boldsymbol{\theta = \frac{r \cdot \pi}{2}}$$

```
import numpy as np
def generate_d():
    return np.random.rand()*3/2
def generate_theta():
    return np.random.rand()*np.pi/2
```

Python Implementation (2)

Define a function for the derived state variable

$$X = \begin{cases} 1, & \text{if } d \leq \frac{L}{2}\sin\theta \\ 0, & \text{otherwise} \end{cases}$$

```
def generate_x():
    d = generate_d()
    theta = generate_theta()
    if d <= 2.5/2*np.sin(theta):
        return 1
    else:
        return 0
```

Python Implementation (3)

- Iterate over a list to generate samples
- Compute statistics and plots as needed

```
samples = [generate_x() for i in range(30)]
print np.mean(samples)
print np.std(samples,ddof=1)
import scipy.stats as stats
print stats.sem(samples)
```

Variance Reduction Methods

Variance Reduction Methods

Basic Monte Carlo simulation requires many samples for accurate estimates (can be improved)

- **Antithetic variables**: leverage correlation in observations to get better estimates of population mean
- **Control variables**: replace the estimation of unknown quantity with the difference between two quantities, one of which has a known expected value
- **Importance sampling**: purposefully draw samples from a different (better) distribution and correct for bias
- **Stratified sampling**: purposefully draw samples from segmented regions of the sample space

Antithetic Variables: Theory

Antithetic variables help make more accurate estimates of **expected value** or **population mean**

- Recall the Central Limit Theorem:

$$\bar{X} \sim \text{normal}(\mu_x, \frac{\sigma_x}{\sqrt{N}})$$

- Reduce σ_x to improve the accuracy of estimates
- Find a *different* process that generates samples with lower variance (σ_x^2) but without changing sample mean (μ_x)

Antithetic Variables: Concept

Replace sample X with average of X_1 and X_2:

$$X = \frac{X_1 + X_2}{2}$$

$$E[X] = E\left[\frac{X_1 + X_2}{2}\right] = \frac{E[X_1] + E[X_2]}{2}$$

$$\mathrm{Var}(X) = \mathrm{Var}\left(\frac{X_1 + X_2}{2}\right) = \frac{\mathrm{Var}(X_1) + \mathrm{Var}(X_2) + 2 \cdot \mathrm{Cov}(X_1, X_2)}{4}$$

Select X_1 and X_2 such that:

$$\mathrm{Cov}(X_1, X_2) < 0, \qquad \text{and,} \qquad \frac{E[X_1] + E[X_2]}{2} = E[X]$$

Example: Needle Distance

Basic process generator samples $D = d$ as:

$$d = \frac{r \cdot T}{2}, \qquad r \sim \text{uniform}(0,1)$$

Antithetic form decomposes into two parts:

$$d = \frac{d_1 + d_2}{2}$$

Simplest negative correlation from re-using r:

$$d_1 = \frac{r \cdot T}{2}, \qquad d_2 = \frac{(1-r) \cdot T}{2}, \qquad r \sim \text{uniform}(0,1)$$

However, this application of antithetic variables is not that interesting, because we already know that $E[D] = T/4$

Antithetic Buffon's Needle

Generate two random values for distance and angle:

$$r_1 \sim \text{uniform}(0,1), \qquad r_2 \sim \text{uniform}(0,1)$$

Generate antithetic distance and angle samples

$$d_1 = \frac{r_1 \cdot T}{2}, d_2 = \frac{(1 - r_1) \cdot T}{2} \qquad \theta_1 = \frac{r_2 \cdot \pi}{2}, \theta_2 = \frac{(1 - r_2) \cdot \pi}{2}$$

Compute antithetic derived variable:

$$X_1 = 1 \text{ if } d_1 \leq \frac{L}{2} \sin\theta_1 \text{ else } 0, \qquad X_2 = 1 \text{ if } d_2 \leq \frac{L}{2} \sin\theta_2 \text{ else } 0$$

$$X = \frac{X_1 + X_2}{2}$$

Antithetic Variance Checks

In original model:

- $\mathrm{Var}(X) \approx s_x^2 = 0.25$
- 95% confidence with ± 0.01 error:

$$N = \left(\frac{1.96 \cdot \sigma_x}{0.01}\right)^2 \approx 38416 \cdot 0.25 = 9604$$

In model with antithetic variables:

- $\mathrm{Var}(X_1) \approx s_{x_1}^2 = 0.25$
- $\mathrm{Var}(X_2) \approx s_{x_2}^2 = 0.25$
- $\mathrm{Cov}(X_1, X_2) \approx -0.2$
- $\mathrm{Var}(X) = \frac{\mathrm{Var}(X_1)+\mathrm{Var}(X_2)+2\mathrm{Cov}(X_1,X_2)}{4}$
  $\approx \frac{0.25 + 0.25 + 2 \cdot (-0.2)}{4} = 0.025$
- 95% confidence with ± 0.01 error:

$$N = \left(\frac{1.96 \cdot \sigma_x}{0.01}\right)^2 \approx 38416 \cdot 0.025 = 960$$

Results of Antithetic Variable

Using Buffon's Needle Monte Carlo simulation:

- Basic Monte Carlo simulation ($N = 10000$):

$$\bar{x} = 0.533, \qquad s_x = 0.499, \qquad SEM = 0.005$$

- Antithetic Monte Carlo simulation ($N = 10000$):

$$\bar{x} = 0.527, \qquad s_x = 0.156, \qquad SEM = 0.0016$$

- Antithetic Monte Carlo simulation ($N = 850$):

$$\bar{x} = 0.522, \qquad s_x = 0.147, \qquad SEM = 0.005$$