Hand and Spreadsheet Simulations

Dave Goldsman

Georgia Institute of Technology, Atlanta, GA, USA

8/16/21

Outline

Goal: Look at some examples of easy problems that we can simulate by hand (or almost by hand).

We'll start out with a simple method for solving a differential equation — a simulation with no randomness at all involved.

Then some small-to-medium problems involving a little randomness,

Finally, we'll show how to generate that randomness.

"Solving" a Differential Equation Numerically

Recall: If $f(x)$ is continuous, then it has the *derivative*

$$\frac{d}{dx}f(x) \equiv f'(x) \equiv \lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$

if the limit exists and is well-defined for any given x. Think of the derivative as the slope of the function.

Then for small h,

$$f'(x) \doteq \frac{f(x+h)-f(x)}{h}$$

so that

$$f(x+h) \doteq f(x) + hf'(x). \qquad (1)$$

Example: Suppose you have a differential equation of a population growth model, $f'(x) = 2f(x)$ with $f(0) = 10$.

This is really easy to solve analytically. To do so, we'll set $y = f(x)$ and rewrite the differential equation as $\frac{dy}{dx} = 2y$. Then we have $\frac{dy}{y} = 2\,dx$, so that $\int \frac{dy}{y} = \int 2\,dx$, so that $\ell\text{n}(y) = 2x + C$, where C is an arbitrary constant.

Exponentiating both sides yields $y = e^{2x+C} = Ke^{2x}$, where K is another arbitrary constant. We can solve for K by using the initial condition $f(0) = 10 = Ke^0$, forcing $K = 10$.

Thus, the analytical (exact) answer is $f(x) = 10e^{2x}$.

Now let's "solve" this same problem using a fixed-increment time approach with $h = 0.01$. (This is known as *Euler's method*.) By (1), we have

$$f(x+h) \ \doteq\ f(x) + hf'(x) \ =\ f(x) + 2hf(x) \ =\ (1+2h)f(x).$$

Similarly,

$$f(x+2h) \ =\ f((x+h)+h) \ \doteq\ (1+2h)f(x+h) \ \doteq\ (1+2h)^2 f(x).$$

Continuing,

$$f(x+ih) \ \doteq\ (1+2h)^i f(x) \quad i = 0, 1, 2, \ldots,$$

though the approximation may deteriorate as i gets large.

Plugging in $f(0) = 10$ and $h = 0.01$, we have

$$f(0.01i) \doteq 10(1.02)^i, \quad i = 0, 1, 2, \ldots. \tag{2}$$

The approximation (2) makes sense since for small z,

$$e^z = \sum_{\ell=0}^{\infty} \frac{z^\ell}{\ell!} \doteq 1 + z \doteq (1+z)^i \text{ for small } i,$$

and in particular, $10e^{0.02} \doteq 10(1.02)^i$, so that (2) pretty much matches the exact answer $f(x) = 10e^{2x}$.

In any case, let's see how well the approximation does....

$x = ih = 0.01i$	0	0.01	0.02	0.03	0.04	$\cdots$	0.10
approx $f(x) \doteq 10(1.02)^i$	10	10.20	10.40	10.61	10.82	$\cdots$	12.19
true $f(x) = 10e^{2x} = 10e^{0.02i}$	10	10.20	10.41	10.62	10.83	$\cdots$	12.21

Not bad at all (at least for small i)! $\square$

Outline

Monte Carlo Integration

The previous differential equation example didn't involve any randomness. That will change with the current Monte Carlo application. Let's integrate

$$I = \int_a^b g(x)\,dx = (b-a)\int_0^1 g(a+(b-a)u)\,du,$$

where we get the last equality by substituting $u = (x-a)/(b-a)$.

Of course, we can often do this by analytical methods that we learned back in calculus class, or by numerical methods like the trapezoid rule or something like Gauss-Laguerre integration. But if these methods aren't possible, you can always use MC integration. . . .

Suppose $U_1, U_2, \ldots$ are iid Unif(0,1), and define

$$I_i \equiv (b-a)g(a+(b-a)U_i) \quad \text{for } i = 1, 2, \ldots, n.$$

We can use the sample average as an estimator for I:

$$\bar{I}_n \equiv \frac{1}{n}\sum_{i=1}^{n} I_i = \frac{b-a}{n}\sum_{i=1}^{n} g(a+(b-a)U_i).$$

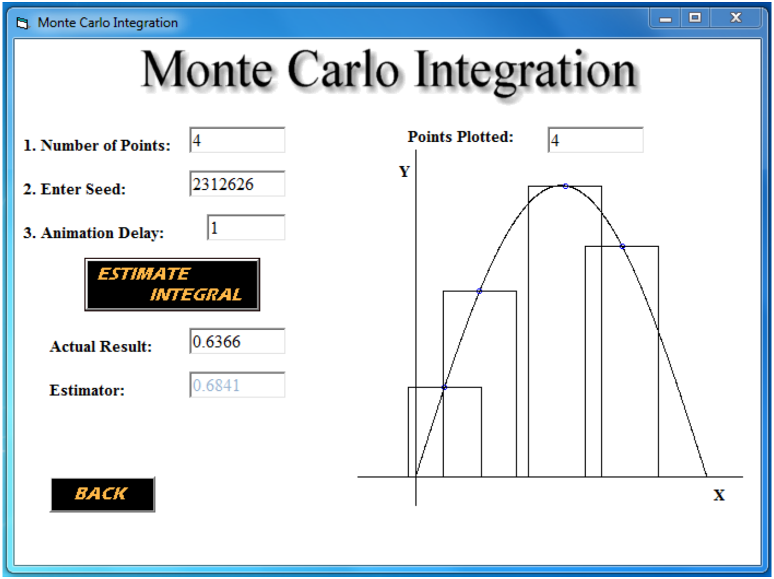
Why is this okey dokey? Let's appeal to our old friend, the Law of Large Numbers: If an estimator is asymptotically unbiased and its variance goes to zero, then things are good.

First, by the Law of the Unconscious Statistician, notice that

$$\begin{aligned}
\mathrm{E}[\bar{I}_n] &= (b-a)\mathrm{E}[g(a+(b-a)U_i)] \\
&= (b-a)\int_{\mathbb{R}} g(a+(b-a)u)f(u)\,du \\
&\quad \text{(where } f(u) \text{ is the Unif(0,1) pdf)} \\
&= (b-a)\int_0^1 g(a+(b-a)u)\,du \;=\; I.
\end{aligned}$$

So $\bar{I}_n$ is unbiased for I.

Since it can also be shown that $\mathrm{Var}(\bar{I}_n) = O(1/n)$, the LLN implies $\bar{I}_n \to I$ as $n \to \infty$.



Monte Carlo Integration

1. Number of Points: 4

2. Enter Seed: 2312626

3. Animation Delay: 1

Actual Result: 0.6366

Estimator: 0.6841

Points Plotted: 4

Approximate Confidence Interval for I:

By the CLT, for $n \geq 30$-ish, we have

$$\bar{I}_n \;\approx\; \mathrm{Nor}\big(\mathrm{E}[\bar{I}_n], \mathrm{Var}(\bar{I}_n)\big) \;\sim\; \mathrm{Nor}\big(I, \mathrm{Var}(I_i)/n\big).$$

This suggests that a reasonable approximate $100(1-\alpha)\%$ confidence interval for I is

$$I \;\in\; \bar{I}_n \pm z_{\alpha/2}\sqrt{S_I^2/n}, \tag{3}$$

where $z_{\alpha/2}$ is the usual standard normal quantile, and $S_I^2 \equiv \frac{1}{n-1}\sum_{i=1}^{n}(I_i - \bar{I}_n)^2$ is the sample variance of the I_i's.

Example: Suppose $I = \int_0^1 \sin(\pi x)\,dx$ (and pretend we don't know the actual answer, $2/\pi \doteq 0.6366$).

Let's take $n = 30$ Unif(0,1) observations, $U_1, U_2, \ldots, U_{30}$:

0.635	0.873	0.388	0.713	0.373	0.315	0.870	0.815	0.253	0.788
0.106	0.939	0.450	0.845	0.009	0.056	0.156	0.894	0.846	0.428
0.957	0.160	0.583	0.352	0.036	0.188	0.714	0.342	0.560	0.063

Since

$$I_i \;=\; (b-a)g(a+(b-a)U_i) \;=\; g(U_i) \;=\; \sin(\pi U_i),$$

we obtain

$$\bar{I}_n \;=\; \frac{1}{4}\sum_{i=1}^{30} I_i \;=\; \frac{1}{30}\sum_{i=1}^{30}\sin(\pi U_i) \;=\; 0.582,$$

which is sort of close to $2/\pi$. (We'll do better as n gets bigger.)

Moreover, the approximate 95% confidence interval for I from (3) is

$$\begin{aligned} I &\in \bar{I}_n \pm z_{\alpha/2}\sqrt{S_I^2/n} \\ &= 0.582 \pm 1.96\sqrt{0.0956/30} \\ &= [0.471, 0.693]. \end{aligned}$$

We'll usually get a smaller CI as n increases a bit — though sometimes the convergence is choppy due to good or bad luck. $\square$

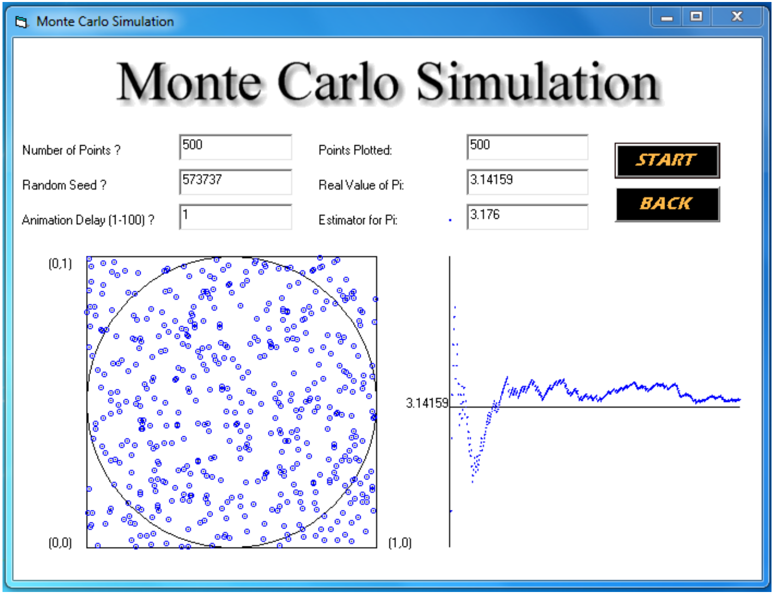
Outline

Making Some π

Consider a unit square (with area one). Inscribe in the square a circle with radius 1/2 (with area $\pi/4$). Suppose we toss darts randomly at the square. The probability that a particular dart will land in the inscribed circle is obviously $\pi/4$ (the ratio of the two areas). We can use this fact to estimate π.

Toss n such darts at the square and calculate the proportion $\hat{p}_n$ that land in the circle. Then an estimate for π is $\hat{\pi}_n = 4\hat{p}_n$, which converges to π as n becomes large by the LLN.

For instance, suppose that we throw $n = 500$ darts at the square and 397 of them land in the circle. Then $\hat{p}_n = 0.794$, and our estimate for π is $\hat{\pi}_n = 3.176$ — not so bad.



Monte Carlo Simulation

Monte Carlo Simulation

Number of Points ?	500	Points Plotted:	500
Random Seed ?	573737	Real Value of Pi:	3.14159
Animation Delay (1-100) ?	1	Estimator for Pi:	3.176

How would we actually conduct such an experiment?

To simulate a dart toss, suppose U_1 and U_2 are iid Unif(0,1). Then (U_1, U_2) represents the random position of the dart on the unit square. The dart lands in the circle if

$$\left(U_1 - \frac{1}{2}\right)^2 + \left(U_2 - \frac{1}{2}\right)^2 \le \frac{1}{4}.$$

Generate n such pairs of uniforms and count up how many of them fall in the circle. Then plug into $\hat{\pi}_n$. $\square$

Outline

Single-Server Queue

Customers arrive at a single-server Q with iid interarrival times and iid service times. Customers must wait in a FIFO Q if the server is busy.

We will estimate the expected customer waiting time, the expected number of people in the system, and the server utilization (proportion of busy time). First, some notation.

Interarrival time between customers $i-1$ and i is I_i
Customer i's arrival time is $A_i = \sum_{j=1}^{i} I_j$
Customer i starts service at time $T_i = \max(A_i, D_{i-1})$
Customer i's waiting time is $W_i^Q = T_i - A_i$
Customer i's time in the system is $W_i = D_i - A_i$
Customer i's service time is S_i
Customer i's departure time is $D_i = T_i + S_i$

Example: Suppose we have the following sequence of events…

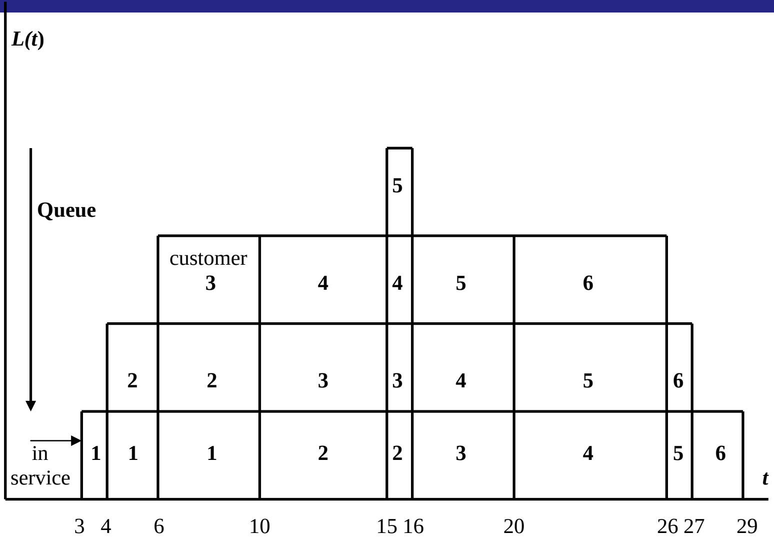
i	I_i	A_i	T_i	W_i^Q	S_i	D_i
1	3	3	3	0	7	10
2	1	4	10	6	6	16
3	2	6	16	10	4	20
4	4	10	20	10	6	26
5	5	15	26	11	1	27
6	5	20	27	7	2	29

The average waiting time for the six customers is obviously $\sum_{i=1}^{6} W_i^Q/6 = 7.33$.

How do we get the average number of people in the system (in line + in service)?

Note that arrivals and departures are the only possible times for the number of people in the system, $L(t)$, to change. These times (and the associated things that happen) are called events.

time t	event	$L(t)$
0	simulation begins	0
3	customer 1 arrives	1
4	2 arrives	2
6	3 arrives	3
10	1 departs; 4 arrives	3
15	5 arrives	4
16	2 departs	3
20	3 departs; 6 arrives	3
26	4 departs	2
27	5 departs	1
29	6 departs	0



The average number in the system is $\bar{L} = \frac{1}{29}\int_0^{29} L(t)\,dt = \frac{70}{29}$.

Another way to get the average number in the system is to calculate

$$\begin{aligned}\bar{L} &= \frac{\text{total person-time in system}}{29} \\ &= \frac{\sum_{i=1}^{6}(D_i - A_i)}{29} \\ &= \frac{7+12+14+16+12+9}{29} = \frac{70}{29}.\end{aligned}$$

Finally, to obtain the estimated server utilization, we easily see (from the picture) that the proportion of time that the server is busy is $\hat{\rho} = \frac{26}{29}$. $\Box$

Example: Same events, but *last*-in-first-out (LIFO) services…

i	I_i	A_i	T_i	W_i^Q	S_i	D_i
1	3	3	3	0	7	10
2	1	4	23	19	6	29
3	2	6	17	11	4	21
4	4	10	10	0	6	16
5	5	15	16	1	1	17
6	5	20	21	1	2	23

The average waiting time for the six customers is 5.33, and the average number of people in the system turns out to be $\frac{58}{29} = 2$, which in this case turn out to be better than FIFO. $\square$

Outline

(s, S) Inventory System

A store sells a product at \$$d$/unit. Our inventory policy is to have at least s units in stock at the start of each day. If the stock slips to less than s by the end of the day, we place an order with our supplier to push the stock back up to S by the beginning of the next day.

Let I_i denote the inventory at the *end* of day i, and let Z_i denote the order that we place at the end of day i. Clearly,

$$Z_i = \begin{cases} S - I_i & \text{if } I_i < s \\ 0 & \text{otherwise} \end{cases} .$$

If an order is placed to the supplier at the end of day i, it costs the store $K + cZ_i$. It costs \$$h$/unit for the store to hold unsold inventory overnight, and a penalty cost of \$$p$/unit if demand can't be met. No backlogs are allowed. Demand on day i is D_i.

How much money does the store make on day i?

$$\begin{aligned}
&\text{Total Profit} \\
&= \text{Sales} - \text{Ordering Cost} - \text{Holding Cost} - \text{Penalty Cost} \\
&= d\,\min(D_i, \text{inventory at beginning of day } i) \\
&\quad - \begin{cases} K + cZ_i & \text{if } I_i < s \\ 0 & \text{otherwise} \end{cases} \\
&\quad -hI_i - p\,\max(0, D_i - \text{inventory at beginning of day } i) \\
&= d\,\min(D_i, I_{i-1} + Z_{i-1}) \\
&\quad - \begin{cases} K + cZ_i & \text{if } I_i < s \\ 0 & \text{otherwise} \end{cases} \\
&\quad -hI_i - p\,\max(0, D_i - (I_{i-1} + Z_{i-1})).
\end{aligned}$$

Example: Suppose

$$d = 10, \quad s = 3, \quad S = 10, \quad K = 2, \quad c = 4, \quad h = 1, \quad p = 2.$$

Consider the following sequence of demands:

$$D_1 = 5, \quad D_2 = 2, \quad D_3 = 8, \quad D_4 = 6, \quad D_5 = 2, \quad D_6 = 1.$$

Suppose that we start out with an initial stock of $I_0 + Z_0 = 10$.

Day i	begin stock	D_i	I_i	Z_i	sales rev	order cost	hold cost	penalty cost	TOTAL profit
1	10	5	5	0	50	0	-5	0	45
2	5	2	3	0	20	0	-3	0	17
3	3	8	0	10	30	-42	0	-10	-22
4	10	6	4	0	60	0	-4	0	56
5	4	2	2	8	20	-34	-2	0	-16
6	10	1	9	0	10	0	-9	0	1

Outline

Simulating Random Variables

Example (Discrete Uniform): Consider a D.U. on $\{1, 2, \ldots, n\}$, i.e., $X = i$ with probability $1/n$ for $i = 1, 2, \ldots, n$. (Think of this as an n-sided dice toss for you Dungeons and Dragons fans.)

If $U \sim \text{Unif}(0, 1)$, we can obtain a D.U. random variate simply by setting $X = \lceil nU \rceil$, where $\lceil \cdot \rceil$ is the "ceiling" (or "round up") function.

For example, if $n = 10$ and we sample a Unif(0,1) random variable $U = 0.73$, then $X = \lceil 7.3 \rceil = 8$. $\Box$

Example (Another Discrete Random Variable):

$$P(X = x) = \begin{cases} 0.25 & \text{if } x = -2 \\ 0.10 & \text{if } x = 3 \\ 0.65 & \text{if } x = 4.2 \\ 0 & \text{otherwise} \end{cases}$$

Can't use a die toss to simulate this random variable. Instead, use what's called the *inverse transform method*.

x	$P(X = x)$	$P(X \leq x)$	Unif(0,1)'s
-2	0.25	0.25	$[0.00, 0.25]$
3	0.10	0.35	$(0.25, 0.35]$
4.2	0.65	1.00	$(0.35, 1.00)$

Sample $U \sim \text{Unif}(0, 1)$. Choose the corresponding x-value, i.e., $X = F^{-1}(U)$. For example, $U = 0.46$ means that $X = 4.2$. $\Box$

Now we'll use the inverse transform method to generate a continuous random variable. Recall…

Theorem: If X is a continuous random variable with cdf $F(x)$, then the random variable $F(X) \sim \mathrm{Unif}(0,1)$.

This suggests a way to generate realizations of the RV X. Simply set $F(X) = U \sim \mathrm{Unif}(0,1)$ and solve for $X = F^{-1}(U)$.

Old Example: Suppose $X \sim \mathrm{Exp}(\lambda)$. Then $F(x) = 1 - e^{-\lambda x}$ for $x > 0$. Set $F(X) = 1 - e^{-\lambda X} = U$. Solve for X,

$$X \ = \ \frac{-1}{\lambda}\,\ell\mathrm{n}(1-U) \ \sim \ \mathrm{Exp}(\lambda). \quad \Box$$

Example (Generating Uniforms): All of the above RV generation examples relied on our ability to generate a Unif(0,1) RV. For now, let's assume that we can generate numbers that are "practically" iid Unif(0,1).

If you don't like programming, you can use Excel function `RAND()` or something similar to generate Unif(0,1)'s.

Here's an algorithm to generate *pseudo-random numbers (PRN's)*, i.e., a series $R_1, R_2, \ldots$ of *deterministic* numbers that *appear* to be iid Unif(0,1). Pick a *seed* integer X_0, and calculate

$$X_i \;=\; 16807 X_{i-1} \mathrm{mod}(2^{31}-1), \quad i = 1, 2, \ldots.$$

Then set $R_i = X_i/(2^{31}-1)$, $i = 1, 2, \ldots$.

Here's an easy FORTRAN implementation of the above algorithm (from Bratley, Fox, and Schrage).

```
FUNCTION UNIF(IX)
K1 = IX/127773    (this division truncates, e.g., 5/3 = 1.)
IX = 16807*(IX - K1*127773) - K1*2836    (update seed)
IF(IX.LT.0)IX = IX + 2147483647
UNIF = IX * 4.656612875E-10
RETURN
END
```

In the above function, we input a positive integer `IX` and the function returns the PRN `UNIF`, as well as an updated `IX` that we can use again. □

Outline

Spreadsheet Simulation

Let’s simulate a fake stock portfolio consisting of 6 stocks from different sectors, as laid out in my Excel file Spreadsheet Stock Portfolio. We start out with $5000 worth of each stock, and each increases or decreases in value each year according to

$$\text{Previous Value} * \max\left[\, 0, \text{Nor}\big(\mu_i,\, \sigma_i^2\big) * \text{Nor}\big(1, (0.2)^2\big) \,\right],$$

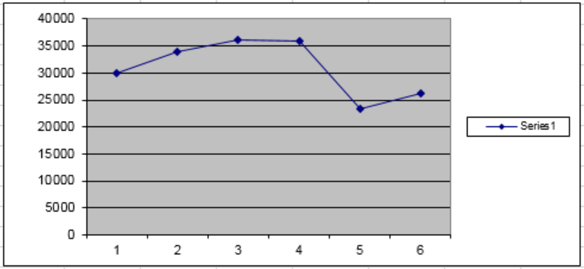
where the first normal term denotes the natural stock fluctuation for stock i, and the second normal denotes natural market conditions (that affect all stocks).

To generate a normal in Excel, you can use

$$\texttt{NORM.INV(RAND(),}\mu\texttt{,}\sigma\texttt{)} \quad ,$$

where `RAND()` is Unif(0,1), so that `NORM.INV` uses the inverse transform method.

	mean	sd	year0	year1	year2	year3	year4	year5
general year performance (so stocks are correlated)				1.12	1.08	1.00	0.73	1.02
energy	0.05	0.30	5000	4094	4507	3607	2991	3156
pharmaceuticals	0.06	0.20	5000	4664	3680	3988	2737	3130
entertainment	0.04	0.10	5000	5877	6974	6548	4670	4501
insurance	0.07	0.05	5000	5729	6454	6797	5597	6141
banking	0.06	0.30	5000	6922	8870	7611	4758	5729
computer technology	0.18	0.30	5000	6557	5761	7340	2657	3568
Totals			30000	33841.95	36246.22	35890.87	23409.34	26224.46


Series1