1 Questions to Answer

Upload a Word Document with your answers to the D2L dropbox by 5/4/2018

Problem 1: Recall that a LCG is pseudo-random number generator of the form

$$x_{n+1} = ax_n + b \quad \mod m,$$

where $x_0 \in \{0, \ldots, m-1\}$ is a seed value. Rank the following LCG's in order of quality. Give some evidence to support your ranking.

(A)	$m = 2^{48}$;	$a = 3 + 2^{24}$;	$b = 0$
(B)	$m = 2^{31}$;	$a = 3 + 2^{16}$;	$b = 7$
(C)	$m = 2^{31} - 1$;	$a = 7^5$;	$b = 0$

Problem 2: Gambler's Ruin Algorithm. *What is the probability of doubling your money before losing of all your money?* Specifically, calculate this probability in each of the following scenarios:

(A) bet = 5 dollars;	initial money = 1000 dollars;	win probability = 0.4929 (craps)
(B) bet = 2 dollars;	initial money = 1000 dollars;	win probability = 0.4929
(C) bet = 5 dollars;	initial money = 10000 dollars;	win probability = 0.4929
(D) bet = 5 dollars;	initial money = 1000 dollars;	win probability = 0.4737 (roulette)

Problem 3: Three Dimensional Random Walk. See the following pages for specific info on the implementation.

What to calculate: The probability of reaching home $(0, 0, 0)$ in a true random walk within a reasonable time limit and a reasonable domain size. Specifically, find the probability of reaching home in each of the following scenarios:

(A)	`maxSteps`=1000;	Initial Position: $(5, 0, 0)$;	*Closed* universe of size 10.
(B)	`maxSteps`=1000;	Initial Position: $(5, 0, 0)$;	*Torus* universe of size 10.
(C)	`maxSteps`=10000;	Initial Position: $(5, 0, 0)$;	*Closed* universe of size 10.
(D)	`maxSteps`=1000;	Initial Position: $(5, 0, 0)$;	*Closed* universe of size 6.

Problem 4: Two Dimension Biased Random Walk. See the following pages for specific info on the implementation.

What to calculate: The probability of reaching home $(0, 0)$ in a biased random walk within a reasonable time limit in open domain. Specifically, calculate this probability in each of the following scenarios:

(A)	`maxSteps`=5000;	Initial Position: $(20, 0)$;	Bias Parameter $\alpha = 1.1$;
(B)	`maxSteps`=5000;	Initial Position: $(10, 0)$;	Bias Parameter $\alpha = 1.1$;
(C)	`maxSteps`=5000;	Initial Position: $(20, 0)$;	Bias Parameter $\alpha = 1.5$;

2 Specifications for `walk3D.m`

Input Arguments:

- x, y, z : Initial Position of the walker (integers.)
- `maxSteps`: Maximum number of steps to take before the walker gives up (positive integer.)
- `domainType`: One of '`open`', '`closed`', or '`torus`' (string.)
- `domainSize`: Size of domain to be used if the user specifies a closed domain (Positive integer or `inf`.)

Output Argument:

- s : Number of steps taken to reach home OR take $s = -1$ if home is never reached.

Example call: `[s,dist2Home]= = walk3d(5,2,-4,100000,'torus',10);`
The function call indicates initial coordinate $(5, 2, -4)$, set up for up to 100000 steps on a torus (periodic) universe where

$$-10 \leq x, y \leq 10;$$

Directions for the Torus Domain. Key idea: If any of $|x|, |y|, |z|$ are greater than the `domainSize`, then you are outside the domain and should move to the opposite side.

- If you just did $x = x + 1$ and $x >$ `domainSize`, set $x = -$`domainSize`
- If you just did $x = x - 1$ and $x < -$`domainSize`, set $x =$ `domainSize`
- Perform similar checks to two above for y and z.

Alternative: You can perform a modular arithmetic trick. Let $d =$ `domainSize` (less writing). Set $m = 2d + 1$. After any x move where $x \to x \pm 1$, set

$$x = \mod(x + d, m) - d$$

After any y move, set $y = \mod(y + d, m) - d$. Do a similar calculation after any z move.

Directions for the Closed Domain. Key idea: If any of $|x|, |y|, |z|$ are going to fall outside the domain, then should reject the move (stay put) and *do not increment the number of steps taken.*

3 Biased Walk Directions

If you at the point (x, y), then the *directions to home* then there are generally two directions that get you closer to your home at the origin $(0, 0)$, unless one of $x = 0$ or $y = 0$, in which case only one direction brings you closer to home. For example,

- If $x > 0$ and $y < 0$, then a left move or an up move bring you closer to home.
- If $x > 0$ and $y = 0$, then only a left move brings you closer to home

In the `biasedWalk`, there is an input argument for the *bias parameter* α. [1]

Here is a simple method for calculating the probabilities for each direction.

For each iteration:

- Get the signs of your current position: $s_x = \texttt{sign}(\texttt{x})$ and $s_y = \texttt{sign}(\texttt{y})$
- Define a weight vector $W = (w_1, w_2, w_3, w_4)$ for Right, Left, Up, and Down Moves:

$$W = \left[\alpha^{-s_x}, \alpha^{s_x}, \alpha^{-s_y}, \alpha^{s_y}\right].$$

- The define your probability vector as

$$P = \frac{1}{\texttt{sum}(W)} W$$

- Draw a randon number in $(0, 1)$ e.g. $r = \texttt{rand}()$; in Octave.
- If $r < P(1)$, then do a right move $x = x + 1$;
  elseif $r < P(1) + P(2)$, then do a left move $x = x - 1$;
  elseif $r < P(1) + P(2) + P(3)$, then do an up move $y = y + 1$;
  else do a down move $y = y - 1$;

For example, if we had $\alpha = 1.25$ and we're at position $(6, -3)$, then we should have better of moving left or up than moving right or down:

$$\begin{aligned}
&\texttt{sign}(\texttt{x}) = 1; \quad \texttt{sign}(\texttt{y}) = -1 \\
&W = (1.25^{-1}, 1.25^{1}, 1.25^{-(-1)}, 1.25^{1}) \quad \to \quad W = [0.8, 1.25, 1.25, 0.8] \\
&P = \frac{1}{\sum(W)} W = \frac{1}{4.1} W \quad \to \approx [0.1951, 0.3049, 0.3049, 0.1951]
\end{aligned}$$

If we are at position $(-6, 0)$, then home is directly to the right. Our probability vector is then computed as follows.

$$\begin{aligned}
&\texttt{sign}(\texttt{x}) = -1; \quad \texttt{sign}(\texttt{y}) = 0 \\
&W = (1.25^{-(-1)}, 1.25^{-1}, 1.25^{0}, 1.25^{0}) \quad \to \quad W = [0.8,\ 1.25,\ 1.0,\ 1.0] \\
&P = \frac{1}{\sum(W)} W = \frac{1}{4.05} W \quad \to \approx [0.3087, 0.1975, 0.2469, 0.2469].
\end{aligned}$$

This means that we have the best chance of moving right, a moderate chance of stumbling up or down, and the lowest probability is a move to the left.

[1] $\alpha > 1$ means bias towards home, $\alpha = 1$ means true random walk, and $0 < \alpha < 1$ means that you will to drift away from home.

4 Some Results to Check Your Work Against

1) `walk3D(4,0,0,2000,'closed',12);` Run 100000 times and results recorded

Probability of Getting Home: 0.129461
Average Number of Steps (Given Home was found): 652.47
Fastest Trip Home: 4
Longest Trip Home: 2000

2) `walk3D(1,1,1,7500,'torus',12);` Run 10000 times and results recorded

Probability of Getting Home: 0.2801
Average Number of Steps (Given Home was found): 1697.7
Fastest Trip Home: 3
Longest Trip Home: 7487

3) `biasedWalk(20,20,1.25,500);` Run 10000 times and results recorded

Probability of Getting Home: 0.9576
Average Number of Steps (Given Home was found): 254.7
Fastest Trip Home: 55
Longest Trip Home: 499

4)`biasedWalk(20,20,1.1,500);` Run 10000 times and results recorded

Probability of Getting Home: 0.303600
Average Number of Steps (Given Home was found): 3.515112e+02
Fastest Trip Home: 99
Longest Trip Home: 499