

Introduction to Mathematical Modeling

Professor:

Dr. Joanna Bieri

joanna_bieri@redlands.edu

OFFICE: DUKE 209

To Play or Not to Play?

Imagine that you are working for a casino and they have developed a new game of chance, a coin flipping game, that they believe will bring in lots of customers because it is so simple. The big problem is that they don't know what the price point should be. They want the buy in to be cheap enough that people will play, but on average they want the game to favor the house. Your goal is to analyze this game using an excel model and decide what the best choice of parameters is. **Your Homework:** You must complete the spreadsheet and write up a one page summary of your analysis with at least one interesting graph the supports your conclusions. You will submit your spreadsheet electronically and hand in your analysis.

THE GAME

Players play the game by repeatedly flipping a fair coin until they either get three heads or three tails in a row. For each flip they take they must pay the house \$1.00 but if they win the automatically get \$8.00.

The questions that we hope to answer today include:

- Should you play or not?

- If you play how much can you expect to win in the long run?
- What if you were able to sneak in a weighted coin, would it help?
- What if the house instead only gave you \$5.00 when you won, then should you play?
- What if you had to get four in a row to win, should you still play?

To answer these questions we will need to build a simulation which consists of playing the game 500 times and then calculating the average winnings. The simulation must allow us to change parameters such as the amount won, the probability of getting heads, and the number of flips required to win.

Directions for building the simulator are on the following pages, but if you want a challenge try to build it on your own. NOTE: This assignment comes from the book *Mathematical Modeling with Excell*, a great additional text if you want to learn more about spreadsheet models.

SOME HELPFUL SPREADSHEET COMMANDS:

- **=average(B1:K1)** → finds the average of the values along the first row.
- **=stdev(B1:K1)** → finds the standard deviation of the row of values.
- **=median(B1:K1)** → finds the median of the row of values.
- **=quartile(B1:K1,1)** → finds the first quartile of the row of values.
- **=quartile(B1:K1,3)** → finds the third quartile of the row of values.
- **=rand()** → generates a random number between 0 and 1. (Uniformly Distributed)
- **=if(rand<0.5,0,1)** → generates a random number and tests if it is less than 0.5. Output of zero means less than 0.5 output of one means greater than 0.5.

Monte Carlo Simulations

The simulator can be built using either a spreadsheet or by programming the game in a programming language of your choice. Below are directions for using a spreadsheet. I also uploaded a Matlab/Octave program to our website if you want to see some example code for the simulation.

Note: I have not been able to program the openoffice.org spreadsheet to do the last data gathering steps of this Monte Carlo simulation. The only way I have been able to get openoffice.org to do the simulation is to physically rerun the spreadsheet and copy and paste the simulation answers to a new sheet 100-500 times. Maybe one of you can figure out a better way to do this! **For this lab I am assuming you are using Microsoft Excel!**

1. Start by entering the following parameters into the top left corner of your spread sheet:

	A	B	C	D
1		Number of Flips=		
2		Winnings =		
3		Prob of Tail =	0.5	
4		Number in a row =	3	
5		Amount won =	8	
6				

You can leave the cells for the number of flips and winnings blank, we will calculate these later. Now we will start building our simulator.

2. Starting in cell A8 type the word "Flip" and below it list the numbers 1-50, representing the possibility of having to make 50 flips before winning.
3. In cell B8 type "Random Number" and below it type the command `= RAND()`, copy this down to the rest of the cells below. This should give you 50 random numbers.
4. In cell C8 type "Result" and below it type the command `=IF(B9 < $C$3,1,0)`, copy it down to the cells below. This command will give you a 1 if you roll heads and a zero if you roll tails. How does this command work? Look it up in the spreadsheet help if you have to!
5. Now move all the way over to cell F8 and type "Switch Count". Since to win we need heads or tails flipped three times in a row, we need a column to tell us if we have switched between heads and tails. In F10 type `=IF((C9-C10)=0,0,1)`, copy it to the cells below. CELL F9 SHOULD BE EMPTY.

6. In cell D8 type "Total Heads". In cell D9 type `= C9` then below it in cell D10 type `=IF(F10=1,C10,C10+D9)`, copy to the cells below. This will count up the total number of heads flipped in a row. Does this command make sense?
7. In cell E8 type "Total Tails". In cell E9 type `= ABS(C9 - 1)` the below it in cell E10 type `=IF(F10=1,ABS(C10-1),E9+(1-C10))`, copy to the cells below.
8. Take a second and look at the results. Is your spreadsheet working and counting the flips correctly?
9. Finally we need a column to check if we have won and we should stop! In cell G8 type "Stop?" Below in cell G9 type `= ""` to force a blank space. In cell G10 type `=IF(G9="stop", "stop",IF(D10+E10=$C$4,"stop",""))`
10. Now back at the top we need to add up the number of times we had to flip, in other words we need to add up the number of blank spaces in row G. In cell C1 type `=COUNTIF(G9:G58,"")+1`. In cell C2 type `= C5 - C1` to calculate your winnings.

Your spreadsheet should look something like this:

	A	B	C	D	E	F	G
1		Number of Flips=	15				
2		Winnings =	-7				
3		Prob of Tail =	0.5				
4		Number in a row =	3				
5		Amount won =	8				
6							
7		Random		Total	Total		
8	Flip	Number	Result	Tails	Heads	Switch Count	STOP
9	1	0.673146055	1	1	0		
10	2	0.78259308	1	2	0	0	
11	3	0.24144044	0	0	1	1	
12	4	0.096130635	0	0	2	0	
13	5	0.984848442	1	1	0	1	
14	6	0.357096625	0	0	1	1	
15	7	0.980076857	1	1	0	1	
16	8	0.700756922	1	2	0	0	
17	9	0.369061786	0	0	1	1	
18	10	0.538619426	1	1	0	1	
19	11	0.139070742	0	0	1	1	
20	12	0.010030631	0	0	2	0	
21	13	0.700845907	1	1	0	1	
22	14	0.574545476	1	2	0	0	
23	15	0.908573106	1	3	0	0 stop	
24	16	0.60908679	1	4	0	0 stop	
25	17	0.873567201	1	5	0	0 stop	
26	18	0.644667852	1	6	0	0 stop	
27	19	0.472460767	0	0	1	1 stop	
28	20	0.227891773	0	0	2	0 stop	

You now have a spreadsheet that will simulate **one game** of coin flipping and calculate your winnings. Now we need a way to keep track of 500 simulations, remember with stochastic models we need to run the model multiple times to gather enough data for a good statistical result.

Running LOTS of Simulations

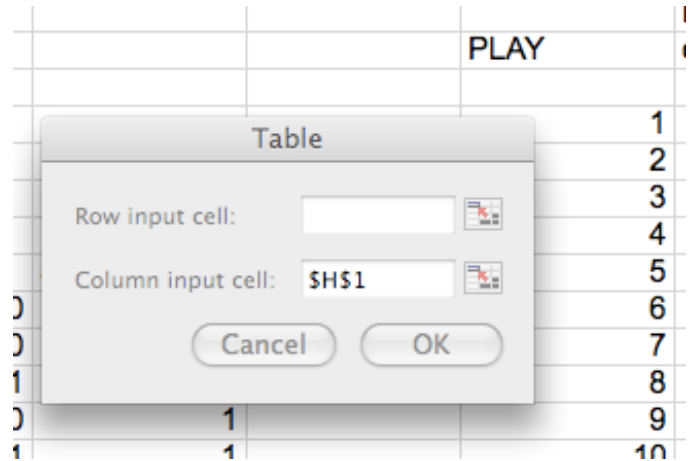
In Excel you will create a **data table to run the simulation 500 times**.

1. Create the simulation table by typing "Play" in cell H2, "Number of Flips" in cell I2, and "WINNINGS" in cell J2. Leave cell H3 blank. Then type = C1 in cell I3 and = C2 in cell J3.
2. The H column should have the numbers 1-500 in rows 4-503 so you can run the simulation 500 times. It should look something like the picture below:

	H	I	J
	PLAY	Number of Flips	WINNINGS
		3	5
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			
21			
22			
23			
24			
25			

3. Next you will have excel fill in the table table.

- Highlight the locations of your table: columns H-J, cells 3-503.
- At the top of the screen in the Excel menu select DATA, WHAT IF ANALYSIS and then DATA TABLE.
- A screen will pop up asking for you row and column input cells.
- Leave the **row input blank** and in the column input type **\$H\$1**.
- Press OK to create your table. This should fill in all of the values of the table with new simulated results.



4. Now to compile your results you just find the average and standard deviation of your winnings, column J. You are DONE creating your first Monte Carlo simulator! Now you ready to play with your parameters and answer the questions for this assignment.

FreeMat Code

You can download the program FreeMat and then copy this code into a blank m-file. See if you can figure out what each line does. This code runs 1000 simulations of the coin flip game.

```
% Matlab file that simulates a coin flip game
% The player flips a coin, pays one dollar for each flip
% once the player has hit "NumToWin" heads or tails they win
% "Payout" dollars.

% NOTE: In matlab lines that do not end with ; will be printed to the
% screen. In general the more you print to the screen the slower your
% simulation. Here I will only print the final result to the screen
ProbTails = 0.5; % Probability of getting tails
NumToWin = 3; % Numer of heads or tails in a row to win
Payout = 8; % The amoung paid once you win
NumTrials = 1000; %Number of trials the simulator will run
NumFlips = zeros(NumTrials,1); % Matrix to store the number of flips for each trial
Winnings = zeros(NumTrials,1); % Matrix to store the winnings for each trial
TrialNum = 0; % TrialNum keeps track of what trial you are currently running
for i = 1:NumTrials % Loop through and do "NumTrials trials of the game
    TrialNum = TrialNum +1;
    LastFlip = 2; % LastFlip keeps track of what happened last time you flipped
    %Initialize to something out of range
    NF = 0; % NF = number of Flips in one game.
    NumInARow = 0;
    while NumInARow < NumToWin % Keep flipping the coin until you get enough to win
        NF = NF +1; % Flip the coin using a randomly generated number
        Test = rand;
        if Test <= ProbTails % if our random number is less than ProTails it was tails
            % Check to see what the last flip was and update NumInARow accordingly
            if LastFlip == 1
                NumInARow = NumInARow + 1;
            else
                NumInARow = 1;
            end
            LastFlip = 1; % Update LastFlip to what we just flipped
        else % If we didn't get tails then we must have flipped heads
            % Check to see what the last flip was and update NumInARow
            % accordingly
            if LastFlip == 0
                NumInARow = NumInARow + 1;
            else
                NumInARow = 1;
            end
            LastFlip = 0; % Update LastFlip to what we just flipped
        end
    end
    NumFlips(TrialNum)=NF; % Save the number of flips for this TrialNum
    Winnings(TrialNum)=Payout-NF; % Calculate the winnings for this TrialNum
end
AverageFlips=mean(NumFlips(:)) % Find the average number of flips
STDFlips=std(NumFlips(:)) % Find the standard deviation of number of flips
AverageWinnings=mean(Winnings(:)) % Find the average winnings
STDWinnings=std(Winnings(:)) % Find the standard deviation of the winnings
```