Markov Chains:
It's a Numbers Game
MATH 381 HW 5

Tim Ferrin, 1764309

November 6, 2020

I Background Information

The focus of this paper is a simple dice game whose rules are as follows:

1. You start the game with a score of 0.
2. Roll a (fair) die and add its value to your score.
3. Roll again. If the value rolled evenly divides your score (i.e. if your score is 6 and you roll a 2), divide your score by your rolled value and that is your new score. Otherwise, add your rolled value to your score to get your new score.
4. Repeat step 3 until your score reaches 10 or higher.

The two questions we will investigate are:

1. How many rolls on average will you make in a game?
2. What is the probability that 10 or more rolls are needed to end a game?

II Mathematical Model

This simple game can be represented as a Markov chain with 11 states of $\{0, 1, 2, ..., 10\}$. States 0-9 represent your score being at 0,1,...,9 respectively and state 10 represents your score being 10 or higher. Since you always start the game with a score of 0, the initial state is always 0 and since you always stop the game at a score of 10 or higher, state 10 is an absorbing state. The chain transitions between states with each roll of the die. Additionally, since the die is fair, the probability of rolling any number 1-6 each turn is $\frac{1}{6}$. This means that the first row of the transition matrix representing the chain has $\frac{1}{6}$ in columns 2-7 and zeros elsewhere. This also means that the last row of the transition matrix has 1 in the 11th column and zeros elsewhere. Considering what happens at each score/roll value pair and the above remarks, the transition matrix below is obtained. Note that although our states are 0-10, throughout this paper we index our matrices starting at 1.

$$P = \begin{bmatrix} 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 & 0 & 0 \\ 0 & \frac{1}{6} & \frac{1}{6} & 0 & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 & 0 \\ 0 & \frac{1}{6} & 0 & \frac{1}{6} & 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 \\ 0 & \frac{1}{6} & \frac{1}{6} & 0 & \frac{1}{6} & 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{6} & 0 & 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 & 0 & \frac{1}{6} & 0 & 0 & 0 & \frac{2}{6} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{4}{6} \\ 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} & 0 & 0 & 0 & \frac{1}{6} & 0 & \frac{3}{6} \\ 0 & 0 & 0 & \frac{1}{6} & 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & \frac{4}{6} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{6}{6} \end{bmatrix}$$

We will run through one example state from the matrix above. If we are currently at state 6 which corresponds to the 7^{th} row of the matrix, the following table explains what rolls send us to what states.

Roll	**6/Roll**	**Evenly Divides?**	**Updated Score**	**New State**
1	6	Yes	6	6
2	3	Yes	3	3
3	2	Yes	2	2
4	1.5	No	10	10
5	1.2	No	11	10
6	1	Yes	1	1

Thus if we are currently at score/state 6, we have a $\frac{1}{6}$ probability to transition to states 1, 2, 3, and 6, a $\frac{2}{6}$ probability to transition to state 10, and a 0 probability to transition to any other state. This corresponds to the 7^{th} row of the matrix having the above mentioned probabilities.

III Question Investigation

Given a transition matrix that is in canonical form, where the absorbing state(s) are the last rows of a matrix and are ordered sequentially, it can be broken up into 4 sub-matrices labeled as

$$P = \left(\begin{array}{c|c} Q & R \\ \hline O & J \end{array}\right) \tag{1}$$

where J is an identity matrix and O is a matrix of zeros.

Let us define a new matrix

$$N = (I - Q)^{-1} \tag{2}$$

There is a theorem, which will not be proved here, which states that the sum of the i-th row of N gives the mean number of steps until absorption when the chain starts in state i. Thus if we find N for the above matrix P, the expression

$$\sum_{i=1}^{10} N_{1,i} \tag{3}$$

gives us the mean number of rolls until we reach our absorbing state 10 at a score of 10 or greater.

For our transition matrix,

$$N = \left(\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 & 0 & 0 \\ 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 & 0 \\ 0 & \frac{1}{6} & \frac{1}{6} & 0 & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 \\ 0 & \frac{1}{6} & 0 & \frac{1}{6} & 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{6} & \frac{1}{6} & 0 & \frac{1}{6} & 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} \\ 0 & \frac{1}{6} & 0 & 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ 0 & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & 0 & 0 & \frac{1}{6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} \\ 0 & 0 & \frac{1}{6} & 0 & \frac{1}{6} & 0 & 0 & 0 & \frac{1}{6} & 0 \\ 0 & 0 & 0 & \frac{1}{6} & 0 & 0 & 0 & 0 & 0 & \frac{1}{6} \end{bmatrix} \right)^{-1} \tag{4}$$

$$= \begin{bmatrix} 1 & \frac{16649}{25976} & \frac{75555}{181832} & \frac{88195}{181832} & \frac{34961}{90916} & \frac{92425}{181832} & \frac{37393}{90916} & \frac{11066}{22729} & \frac{51235}{181832} & \frac{33907}{90916} \\ 0 & \frac{35361}{22729} & \frac{29193}{159103} & \frac{72315}{159103} & \frac{56358}{159103} & \frac{69807}{159103} & \frac{55344}{159103} & \frac{95040}{159103} & \frac{34263}{159103} & \frac{58704}{159103} \\ 0 & \frac{48081}{90916} & \frac{889443}{636412} & \frac{152475}{636412} & \frac{60009}{318206} & \frac{275697}{636412} & \frac{122601}{318206} & \frac{88710}{159103} & \frac{263523}{636412} & \frac{90303}{318206} \\ 0 & \frac{10125}{22729} & \frac{21375}{159103} & \frac{224625}{159103} & \frac{26550}{159103} & \frac{63375}{159103} & \frac{18450}{159103} & \frac{81360}{159103} & \frac{61875}{159103} & \frac{79182}{159103} \\ 0 & \frac{39849}{90916} & \frac{205347}{636412} & \frac{131475}{636412} & \frac{418941}{318206} & \frac{123153}{636412} & \frac{48429}{318206} & \frac{78840}{159103} & \frac{91995}{636412} & \frac{140787}{318206} \\ 0 & \frac{16875}{45458} & \frac{35625}{318206} & \frac{56169}{318206} & \frac{22125}{159103} & \frac{423831}{318206} & \frac{15375}{159103} & \frac{67800}{159103} & \frac{103125}{318206} & \frac{65985}{159103} \\ 0 & \frac{46005}{90916} & \frac{218343}{636412} & \frac{268047}{636412} & \frac{45165}{318206} & \frac{161685}{636412} & \frac{435885}{318206} & \frac{53022}{159103} & \frac{129615}{636412} & \frac{73215}{318206} \\ 0 & \frac{405}{22729} & \frac{855}{159103} & \frac{8985}{159103} & \frac{1062}{159103} & \frac{2535}{159103} & \frac{738}{159103} & \frac{194178}{159103} & \frac{2475}{159103} & \frac{41352}{159103} \\ 0 & \frac{8793}{45458} & \frac{109479}{318206} & \frac{28395}{318206} & \frac{47895}{159103} & \frac{39885}{318206} & \frac{17103}{159103} & \frac{33510}{159103} & \frac{417399}{318206} & \frac{23109}{159103} \\ 0 & \frac{2025}{22729} & \frac{4275}{159103} & \frac{44925}{159103} & \frac{5310}{159103} & \frac{12675}{159103} & \frac{3690}{159103} & \frac{16272}{159103} & \frac{12375}{159103} & \frac{206760}{159103} \end{bmatrix} \tag{5}$$

$$\sum_{i=1}^{10} N_{1,i} = \frac{906835}{181832} \tag{6}$$

$$\approx 4.987 \tag{7}$$

So for the average game we would expect to roll about 4.987 times before ending.

For our second question, we can use the fact that $P^n_{i,j}$ gives the probability that the chain will be in the state corresponding to column j exactly n steps after being in the state corresponding to row i. Thus if we want to know the probability that it takes 10 or more rolls to complete the game, we must find 1 minus the probability we have ended after 9 rolls, or

$$\begin{aligned} \text{Prob}(\geq 10 \text{ rolls needed}) &= 1 - P^9_{1,11} && (8)\\ &= \frac{709471}{10077696} && (9)\\ &\approx 0.0704 && (10) \end{aligned}$$

IV Changing Limits

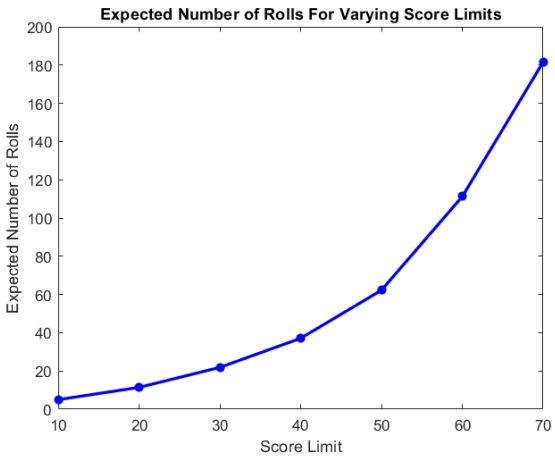
Next we will investigate what happens to the two quantities we calculated above (the expected number of rolls and the probability we will need more than 9 rolls) when we change the score the game ends on. The following table shows the approximate results for score limits of 10, 20, 30, 40, 50, 60, and 70.

Limit	**Expected Rolls**	**Prob. $\geq$ 10 Rolls**
10	4.9872134717761	0.0704001192336
20	11.4040452362452	0.4936547004395
30	21.8684038303543	0.8633658923627
40	37.0781948205926	0.9895904778235
50	62.2351491890661	0.9999778719263
60	111.2154622878285	1.0000000000000
70	181.5065654149850	1.0000000000000

Below is a graph plotting the results of the Expected Rolls column

V Simulation

In order to assure ourselves of the validity of our results for our expected rolls, we use simulations of the game to determine if our results are reasonable. We ran five rounds of 100,000 simulations of the game and took the averages of each round of simulation, tabulated below.



Limit	Avg. 1	Avg. 2	Avg. 3	Avg. 4	Avg. 5
10	4.98526	4.99277	4.96819	4.98772	4.99832
20	11.38582	11.40038	11.3946	0 11.38471	11.41811
30	21.87765	21.87501	21.81446	21.86332	21.91275
40	37.12844	37.06491	37.07925	37.00188	36.97016
50	62.41417	62.22066	62.34815	61.95339	62.23034
60	111.59172	111.38774	111.05776	111.1976	111.39203
70	181.59008	181.11383	182.05415	181.43188	180.76759

Comparing the values from the two tables, we can see that our predicted and simulated values of expected die rolls match up. Specifically, using the Central Limit Theorem these simulated estimates should be close to normally distributed. Since normally distributed implies symmetry about the mean, then we can estimate the probability that all 5 of our estimates are on one side of the true expected value as

$$\frac{2}{2^5} = \frac{1}{2^4} \tag{11}$$

$$= \frac{1}{16} \tag{12}$$

$$\approx 0.0625 \tag{13}$$

This is the probability that all of the simulated estimates are greater, or all of the simulated estimates are less than the true expected value. Thus, there is a 93.75% probability that the true expected value is between the highest estimate and the lowest estimate for each limit.

VI Simulation Output

Here is the direct output of the simulations, summarized in the table above

```
results =
```

```
    1.0e+02 *

Columns 1 through 4

   0.049852600000000   0.049927700000000   0.049681900000000   0.049877200000000
   0.113858200000000   0.114003800000000   0.113946000000000   0.113847100000000
   0.218776500000000   0.218750100000000   0.218144600000000   0.218633200000000
   0.371284400000000   0.370649100000000   0.370792500000000   0.370018800000000
   0.624141700000000   0.622206600000000   0.623481500000000   0.619533900000000
   1.115917200000000   1.113877400000000   1.110577600000000   1.111976000000000
   1.815900800000000   1.811138300000000   1.820541500000000   1.814318800000000

Column 5

   0.049983200000000
   0.114181100000000
   0.219127500000000
   0.369701600000000
   0.622303400000000
   1.113920300000000
   1.807675900000000
```

VII Code

The following is the Matlab code used to generate the transition matrices:

```
numlimits = 7; % Sets the number of different score limits we investigate
resultsSym = sym(zeros(numlimits,2)); % Empty container for the symbolic results
resultsDub = zeros(numlimits,2); % Empty container for the decimal results

for n=10*(1:numlimits) % For loop for each one of the score limits
    Trans = sym(zeros(n+1)); % Next lines define the transition matrix
    faces = 2:7;
    under = faces(faces<(n+1));
    Trans(1,under) = 1/6;
    Trans(1,n+1) = 1-(length(under)/6);
    cols = 2:(n+1);
    for row = 2:n % For each row
        for face = faces-1 % For each potential die roll
            if mod(row-1,face)==0 % If the face divides the current score
                Trans(row,(row-1)/face+1) = 1/6; % Set that score/face transition p to 1
            else
```

```
                Trans(row,min([row+face, n+1])) = Trans(row,min([row+face, n+1])) + 1/6;
                % Else add 1/6 to that score+face transition p
            end
        end
    end
    Trans(n+1,n+1) = 1; % Adds the absorbing state p

    % Calculates the expected number of rolls in symbolic and double forms
    expects = sum(inv(eye(n)-Trans(1:n,1:n)),2);
    resultsSym(n/10,1) = expects(1);
    resultsDub(n/10,1) = double(expects(1));

    % Calculates the probability you need >=10 rolls to finish the game in
    % symbolic and double forms
    ninthprobs = Trans^9;
    resultsSym(n/10,2) = 1-ninthprobs(1,n+1);
    resultsDub(n/10,2) = double(1-ninthprobs(1,n+1));
end
% Prints and plots the results
resultsSym
resultsDub
plot(10*(1:numlimits),resultsDub(:,1),'b-*','Linewidth',2)
ylabel('Expected Number of Rolls')
xlabel('Score Limit')
title('Expected Number of Rolls For Varying Score Limits')
```

The following is the Matlab code used to perform the simulations:

```
numlimits = 7; % Sets the number of different score limits we investigate
results = zeros(numlimits,5); % Empty container for the simulation averages

for n=10*(1:numlimits) % For each of the differrent score limits
    numsims = 5;
    numruns = 100000;
    means = zeros(1,numsims);
    for sim = 1:numsims % For each round of simulations
        gameruns = zeros(1,numruns);
        for game = 1:numruns % For each time we simulate the game
            turns = 1;
            score = randi(6);
            while score<n % While we haven't reached the final score
                turns = turns+1;
                roll = randi(6); % Die roll
                if mod(score,roll)==0  % If the die roll divides the current score
                    score = score/roll;
```

```
                else
                    score = score+roll;
                end
            end
            gameruns(game) = turns;
        end
        means(sim) = mean(gameruns); % Calculates the mean number of rolls for each roun
    end
    results(n/10,:) = means; % Stores the means for each score limit
end
results
```