Kent Ngo
Final Project
Math 183
Prof. Wavrik
6/13/02

Winning Probability of Craps

Introduction

- The game is played using 2 6-sided dice. The shooter throws the dice across the craps table, and the dice have to bounce against the wall on the opposite side to make the throw to be fair.
- There are 2 basic ways the shooter (the person rolling the dice) wins:
- 1. Throwing either a 7 or 11 on his first roll, this is called a "natural".
- 2. Throwing either a 4, 5, 6, 8, 9, 10 on his first roll and then rolling that number again before he rolls a 7 (craps), this is called "making his point".
- If the the first roll is a 2, 3 or a 12, then the shooter loses.

Objectives

- Calculate the winning probability of the shooter using probability mathematics.
- Create a simulation of the game of craps, so that we can play it just like in real life.
- Run the simulation and then find the winning probability of the shooter from the simulation by dividing the number of wins by the number of games we played.
- Then compare that with the true probability. It should be close to the true value.

Winning probability of the shooter

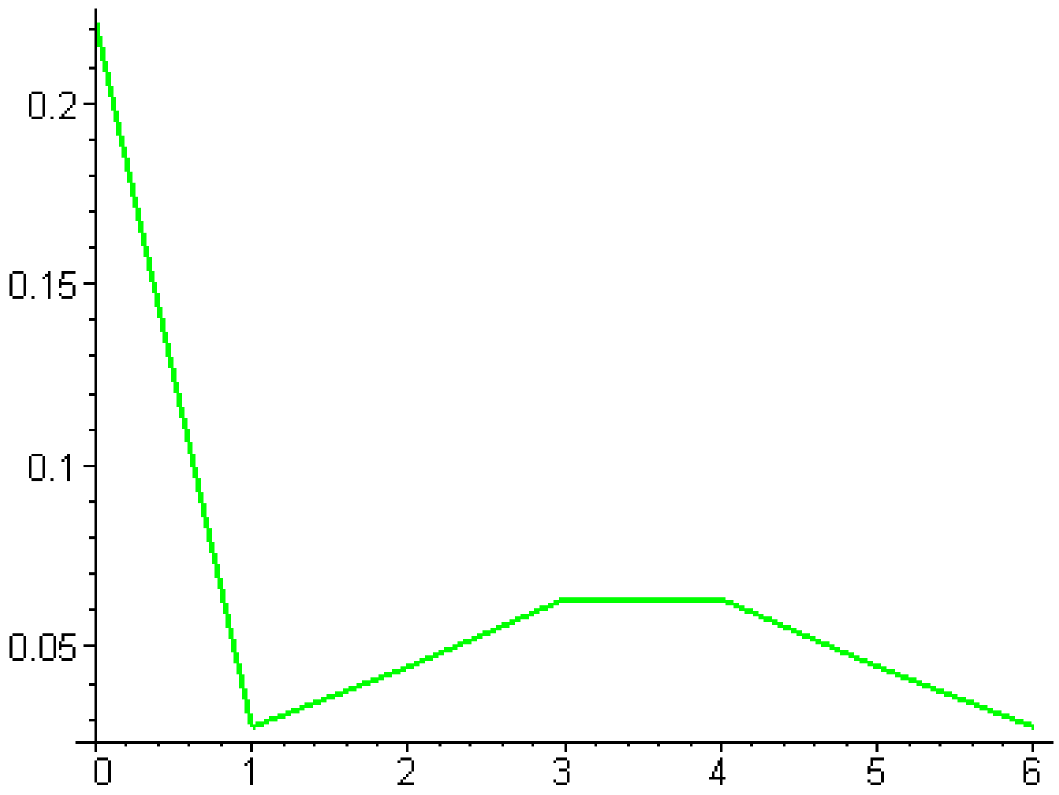
- The probability of winning in the first throw (throwing a "natural") is : P(7) + P(11) = 6/36 + 2/36 = <u>8/36</u>
- To calculate the probability of winning in the second throw, we assume that x is the "point" with probability p. The probability of not winning in any given throw is given by the probability of not x and not 7 = r = 1 - p - 1/6 . The conditional probability is :
-

- P{win | x in first throw} = p + r*p + (r^2) * p + … an infinite geometric series.
- = p / (1- r) , and
- P {win in second or subsequent throws} = p*p / (1 - r) = p^2 / (1 - r)
- So when x = 4,
  P (win after first with x = 4) = (3/36)^2 / (1 - (1 - 3/36 - 1/6)) = 1/36
- x = 5,
  P (win after first with x = 5) = (4/36)^2 / (1 - (1 - 4/36 - 1/6)) = 16/360
- x = 6,
  P (win after first with x = 6) = (5/36)^2 / (1 - (1 - 5/36 - 1/6)) = 25/396
- same probabilities for 8, 9, 10, since the probability of throwing a 4 is the same as the probability of throwing a 10, and same with 5 and 9, and 6 and 8.
-
- Thus, P (shooter wins) = 8/36 + 2*(1/36) + 2*(16/360) + 2*(25/396)
  = **0.493**

Graph of probabilities

- We're going to graph the probabilities of all possible ways to win, e.g. throwing a "natural" and "making the point".
-

```
> with(plottools):
L := curve([[0,8/36],
[1,1/36],[2,16/360],[3,25/396],[4,25/396],[5,16/360],[6,1/3
6]], color=green, thickness = 2):
plots[display](L);
```



- In the graphs above, at x = 0 is the winning probability by throwing a natural

  x = 1 is the winning probability by making the point of 4
  x = 2 is the winning probability by making the point of 5
  x = 3 is the winning probability by making the point of 6
  x = 4 is the winning probability by making the point of 8
  x = 5 is the winning probability by making the point of 9
  x = 6 is the winning probability by making the point of 10

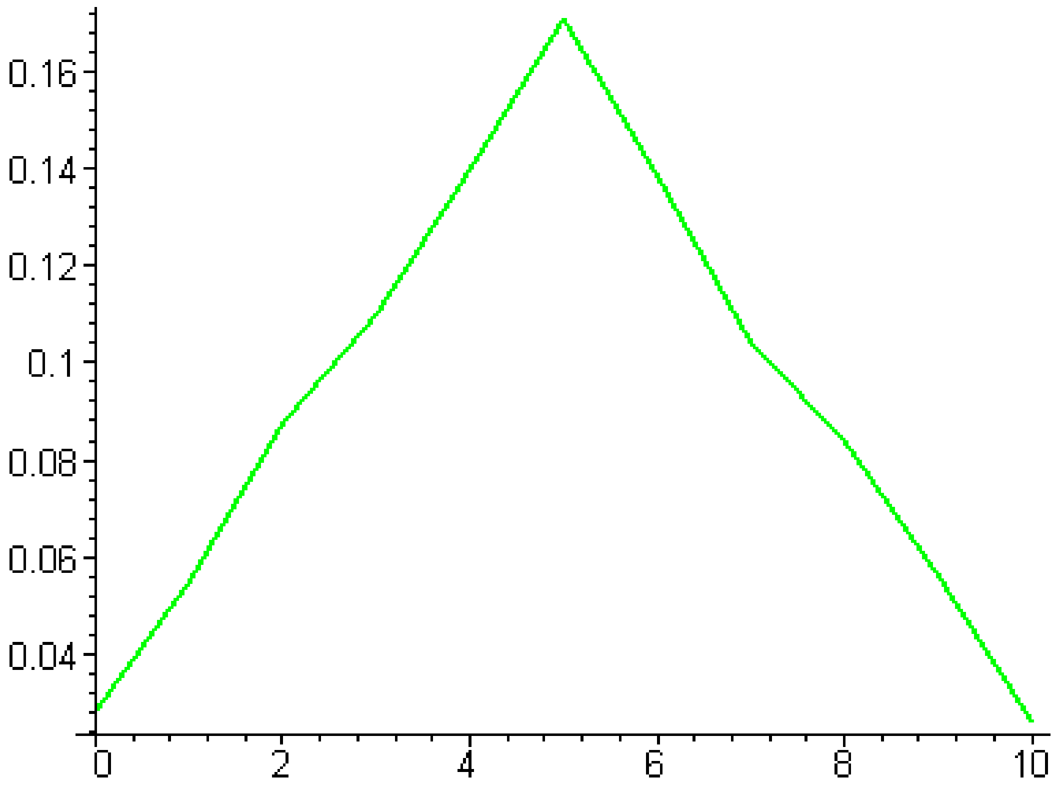
Simulation of craps

- This program simulates the game of craps. It will play the game as many times as you want, and will keep track of the number of wins and losses of the shooter.

```
> restart;
> die := rand(1..6):
> die;
                                        die
>
```

- The program below simulates throwing 2 dice.

```
> dice := proc()local d,i;

#simulates throwing 2 dice
    d:=0;
    for i from 1 to 2 do
        d := d + die();
    od;
    d;
end:
>
> dice();
                                                    7
>
> dice();
                                                   10
>
> test := proc (n)local List,i,d;

#Generates a list containing the number of times a sum of
2, 3 ... 12
#of 2 dice appears
>     List := [seq(0,i = 2..12)];

    #Creates a list having a total of 11 elements
    #with the first element storing the number of times a
sum of 2
    #appears and the next element for the sum of 3, and so
on, up to 12.

>     for i from 1 to n do
>         d := dice();
>         List := subsop( d-1 = List[d-1] + 1, List);
>     od;
>     List;
> end:
>
>
> test( 10000 );
                  [281, 548, 875, 1099, 1398, 1709, 1386, 1042, 839,
562, 261]
> evalf(%/10000);
  [.02810000000, .05480000000, .08750000000, .1099000000, .1398000000,
.1709000000, .1386000000,
        .1042000000, .08390000000, .05620000000, .02610000000]
> List := %;
  List := [.02810000000, .05480000000, .08750000000, .1099000000,
.1398000000, .1709000000,

        .1386000000, .1042000000, .08390000000, .05620000000,
.02610000000]
>
> with(plottools):
L := curve([[0, List[1]],[1, List[2]],[2, List[3]],[3,
List[4]],[4, List[5]],
[5, List[6]],[6, List[7]],[7, List[8]],[8, List[9]],[9,
List[10]],[10, List[11]]],
color=green, thickness = 2):
plots[display](L);
```



- The graph above fits the bell-shaped graph of the probabilities of getting 2, 3,..., and 12 when throwing 2 dice.
- So our simulation of throwing 2 dice is correct.
-
- The program below will simulate the game of craps. It will play as many times as you like, and will keep track of the number of wins and losses of

the shooter.

- Then we can use that to calculate the probability of winning of the shooter by dividing the number of wins by the total number of games that we played.
-

```
> craps := proc (n)local
shoot_win,shoot_loss,house_win,house_loss,i,j,toss,prev_tos
s,List;

#This program simulates the game of craps

>     shoot_win := 0;
>     shoot_loss := 0;
>     house_win := 0;
>     house_loss := 0;
>     List := [seq(0,i = 1..2)];
>     for i from 1 by 1 to n do
>         toss := dice();

        #if it's a natural, then the shooter wins
>         if (toss = 7 or toss = 11) then
>             shoot_win := shoot_win + 1;
>             house_loss := house_loss + 1;

        #if the first throw is 4, 5, 6, 8, 9, or 10
>         elif (toss = 4 or toss = 5 or toss = 6 or toss = 8
or toss = 9 or toss = 10) then
>             prev_toss := toss;
>             toss := dice();
>             for j from 1 by 1 while toss <> prev_toss and
toss <> 7 do
>                 prev_toss := toss;
>                 toss := dice();
>             od;

            #the shooter gets that number again
>             if (toss = prev_toss) then
>                 shoot_win := shoot_win + 1;
>                 house_loss := house_loss + 1;
```

```
                #the shooter gets a 7
>               else
>                   house_win := house_win + 1;
>                   shoot_loss := shoot_loss +1;
>               fi;

            #the first throw is 2, 3, or 12, then the shooter
loses
>           else
>               house_win := house_win + 1;
>               shoot_loss := shoot_loss + 1;
>           fi;
>       od;
>
>           List := subsop( 1 = shoot_win, List);
>           List := subsop( 2 = shoot_loss, List);
>       List;
      #the first element of list is the number of wins of
the shooter
      #the second element of list is the number of losses
of the shooter

> end:
>
>
> craps (10000);
                                          [4794, 5206]
> evalf(%/10000);
                                    [.4794000000, .5206000000]
> %[1];
                                           .4794000000
>
>
```

- The above value is the winning probability of the shooter that we got from the simulation. It's close to the true probability that we calculated earlier. We only play the game 10000, and only got the probability of winning correct in the tenth decimal. If we want it to be correct in the thousandth decimal, then we have to play a lot more than 10000 times.