3. Conditional Probability: 5-Card Draw

Poker: 5-Card Draw

In the previous lecture, the odds of getting various poker hands in the game of 5-card stud were determined. A variation of this game is 5-card draw.

With 5-card draw

- You draw 5 cards to start
- Betting then commences.
- You then discard N cards and draw N new cards.
- Betting then continues
- Cards are revealed and the highest hand wins.

This is also the type of poker most poker-slot machines play:

- You are dealt 5 cards,
- You can then select which cards to discard (0 to 5),
- You then draw new cards to replace these cards
- Your resulting hand determines how much you win.

So here's the problem: how do you calculate the odds of each type of poker hand in 5-card draw?

5-Card Draw: You can keep the pair of Jacks and draw three new cards

This is actually a difficult question since the number of cards you draw depends upon your hand.

- If you were dealt a great hand, such as a flush of a full-house, you'll keep all of your cards and draw none.
- If you were dealt a good hand, such as a three-of-a-kind, you'll keep the cards that match, discard the two that don't match, and draw two cards.
- If you have a bad hand such as a high-card, you'll discard 4 or 5 cards and draw 4-5 new cards.

This is where conditional probabilities come into play

With conditional probabilities, the probability of an event, A, happening can be computed as the sum of different terms:

$$p(A) = p(A|B)p(B) + p(A|C)p(C) + p(A|D)p(D) + \ldots$$

For example, suppose you want to know the probability of getting a 4-of-a-kind in draw poker. First, you have to list out all of the ways to get to a 4-of-a-kind:

- You are dealt a 4-of-a-kind, or
- You are dealt 3-of-a-kind and drew two cards, ending up with 4-of-a-kind
- You are dealt a pair and drew three cards, ending up with 4-of-a-kind, or
- You are dealt a high-card hand and drew four cards, ending up with 4-of-a-kind.

Once these are listed out, you can calculate the odds of ending up with a 4-of-a-kind using conditional probabilities. The first step in such calculations is to define a set of rules you will follow when playing draw poker.

Draw Rules

The rules you use for drawing cards varies from person to person. For these calculations, assume the following:

Hand	Sample Hand	Draw Rule
Royal Flush	10-J-Q-K-A	draw 0
Straight-Flush	2-3-4-5-6	draw 0
4-of-a-Kind	xxxx y	draw 0
Full House	xxx yy	draw 0
Flush	hhhhh	draw 0
Straight	2-3-4-5-6	draw 0
3-of-a-Kind	xxx ab	discard ab, draw 2
2-Pair	xx yy c	discard c, draw 1
High-Card	x abcd	discard abcd, draw 4

Draw rules used in this lecture

This strategy isn't optimal. For example, if you have four cards of the same suit, you might want to discard the off-card and go for a flush. These rules ignore that possibility. If you have a run of four cards (2-3-4-5), you might want to keep these four and try for a straight. Again, these rules ignore that possibility. Poker also involves bluffing: with a high-card hand you might keep all five cards to make your opponents think your hand is better than it actually is. Again, these rules ignore that possibility.

Regardless, assume these are the rules you are following.

Given these rules, determine the odds of getting 4-of-a-kind in draw poker.

4-of-a-Kind in Draw Poker

The first step is to list out all possible ways to get to 4-of-a-kind. As listed before, there are four ways to get there. Label these as states A through E:

- A: You end up with 4-of-a-kind after the draw step
- B: You are dealt 4-of-a-kind
- C: You are dealt 3-of-a-kind
- D: You are dealt a pair
- E: You are dealt a high-card

From before the odds of each of these hands is known:

POKER HAND	Hand	COUNT	Odds Against
1	Royal Flush	4	649,740
2	Straight Flush	36	72,193.33
3	Four of a Kind	624	4,165
4	Full House	3,744	694.17
5	Flush	5,108	508.8
6	Straight	10,200	254.8
7	Three of a Kind	54,912	47.33
8	Two Pair	123,552	21.04
9	One Pair	1,098,240	2.37
10	High Card	1,302,540	2
Total		2,598,960	

The probability of ending up with 4-of-a-kind is then

$$p(A) = p(A|B)p(B) + p(A|C)p(C) + p(A|D)p(D) + p(A|E)p(E)$$

Let's calculate these one by one:

Case B: The odds of ending up with 4-of-a-kind given that you were dealt 4-of-a-kind is 1.000

$$P(A|B) = 1.0000$$

The odds of being dealt 4-of-a-kind are

$$p(B) = \left(\frac{624}{2{,}598{,}960}\right) = 0.000240$$

Case C: You were dealt three-of-a-kind

Hand = xxx ab

You discard ab and draw two new cards, ending up with 4-of-a-kind. The odds of doing this are as follows:

After dealing 5 cards to you, the deck now contains 47 cards. The number of ways to draw 2 cards is 47 cards remaining in the deck, choose two

$$N = \binom{47}{2} = 1081$$

The number of ways to draw a card that matches the one you need is

- Of the one x remaining in the deck, choose 1
- Of the 46 other cards, choose one

$$M = \binom{1}{1}\binom{46}{1} = 46$$

giving the odds of drawing to a 4-of-a-kind being

$$p(A|C) = \frac{M}{N} = \frac{46}{1081} = 0.042553$$

The odds of being dealt 3-of-a-kind is

$$p(C) = \left(\frac{54{,}912}{2{,}598{,}960}\right) = 0.021128$$

Hence, the odds of getting to 4-of-a-kind by going through 3-of-a-kind are

$$p(A|C)p(C) = 0.000899$$

Case D: Assuming you were dealt a pair

Hand = xx abc

discarded abc, and wound up with 4-of-a-kind are as follows. The number of ways to draw three cards from the 47 cards remaining in the deck are:

$$N = \left(\begin{array}{c} 47 \\ 3 \end{array}\right) = 16,215$$

The number of ways to get the two cards you need are:

- Of the 2 x's still in the deck, choose 2
- Of the 45 other cards, choose 1

$$M = \left(\begin{array}{c} 2 \\ 2 \end{array}\right)\left(\begin{array}{c} 45 \\ 1 \end{array}\right) = 45$$

meaning

$$p(A|D) = \left(\frac{45}{16{,}215}\right) = 0.002775$$

From last lecture, the odds of being dealt a pair are

$$p(D) = \left(\frac{1{,}098{,}240}{2{,}598{,}960}\right) = 0.422569$$

The odds of ending up with 4-of-a-kind bty going through a pair are thus

$$p(A|D)p(D) = 0.001173$$

CaseE: Finally, assume you were dealt a high-card hand:

Hand = x abcd

and discarded abcd, the odds of ending up with a 4-of-a-kind are as follows.

The number of ways to draw 4 cards are:

$$N = \binom{47}{4} = 178,365$$

There are three ways to draw to 4-of a kind:

- You draw three x's, or
- You draw four cards that match

Taking the first path. The number of ways to draw three x's is

- Of the three x's in the deck, choose 3
- Of the remaining 44 cards, choose 1

$$M_1 = \binom{3}{3}\binom{44}{1} = 44$$

For the second path

- Of the remaining 8 values that are not in {x, a, b, c, d}, choose one
- Of the four cards with that value, choose four

$$M_2 = \binom{8}{1}\binom{4}{1} = 8$$

The odds of drawing four cards to 4-of-a-kind are thus

$$p(A|E) = \left(\frac{44+8}{178,365}\right) = 0.000292$$

From before, the odds of being dealt a high-card hand is

$$p(E) = \left(\frac{1,302,540}{2,598,960}\right) = 0.501177$$

The odds of getting to 4-of-a-kind through this path is thus

$$p(A|E)p(E) = 0.000146$$

Adding these all together gives

$$p(A) = 0.002458$$

You could do similar calculations to determine the odds of other types of hands in draw poker. These are left as homework assignments.

Validation: Monte-Carlo simulation

So, here's a question for you: how do you check this answer?

One way to check these calculations is to run a Monte-Carlo simulation for draw poker.

Previously, we looked at a Matlab program which

- Shuffled a deck of 52 cards,
- Drew a random 5-card hand from that deck, and
- Determined the type of hand it was.

For a draw poker, modify this program by adding a draw step:

- If you were dealt a royal flush, straignt-flush, 4-of-a-kind, full-house, flush, or straight, draw zero cards.
- If you were dealt 3-of-a-kind, keep the matching cards and draw two
- If you were dealt 2-pair, keep the matching cards and draw one
- If you were dealt a pair, keep the matching cards and draw three
- If you were dealt a high-card, keep one card and draw four

Note that these rules match our previous calculations (good for comparison's sake) but neglect opportunities, such as trying for a flush if you were dealt four cards of one suit.

Back from the lecture #1, a Matlab program to shuffle a deck of 52 cards and deal out a 5-card hand was:

```
X = rand(1,52);
[a,Deck] = sort(X);
Deck = Deck - 1;
Hand = Deck(1:5);
Value = mod(Hand,13) + 1;
Suit = floor(Hand/13) + 1;
```

Now, check for flushes by counting the frequency of each suit

```
NS = zeros(1,4);
for i=1:4
   NS(i) = sum(Suit(i) == i);
   end
NS = sort(NS, 'descend');
if(NS(1) == 5)
    disp('flush');
end
```

For example, if you hand contains three clubs, a diamond, and a heart

```
Suit = [2, 1, 1, 3, 1]
```

NS will be the frequency of each suit in descending order

```
NS = [3, 1, 1, 0]
```

(The max frequency of suits is 3, followed by 1)

Now, sort the hand by frequency of each card:

```
N = Value / 100;
for i=1:5
   N(i) = sum(Value(i) == Value);
   end
[N, b] = sort(N, 'descend');
Hand = Hand(b);
Value = Value(b);
Suit = Suit(b);
```

For example, if your hand starts with 2-pair in random order

```
Value = [2,  5,  6,  5,  2]
```

presorted N is

```
N = [2.02   2.05   1.06   2.05  2.02]
```

(The integer is the frequency, the fraction is the value/100). Sorted, N becomes

```
N = [2.05  2.05  2.02  2.02  1.06]
b = [2  4  1  5  3]
```

The point is to keep the pairs together. The program ends with

```
Value = [5  5  2  2  6]
```

Next, count the frequency of each type of card (Ace through King). End with N recording the maximum to minimum frequency of card values

```
N = zeros(1,13);
for i=1:13
   N(i) = sum(Value == i);
   end
N = sort(N, 'descend');
```

For the previous example, before sorting, N is

```
N = [0, 2, 0, 0, 2, 1, 0, 0, 0, 0, 0, 0, 0]
     A  2  3  4  5  6  7  8  9  T  J  Q  K
```

Your hand contained two 2's, two 5's, and one 6.

After sorting, N is

```
N = [2  2  1  0  0  0  0  0  0  0  0  0  0]
```

- The highest frequency of cards is 2
- The next highest frequency is 2

You can now check for the type of hand

```
if(NS(1) == 5) disp('flush');
elseif((N(1) == 1) * (max(Value)-min(Value)==4) disp('straight');
elseif(N(1) == 4) disp('four of a kind'); end
elseif((N(1)==3)*(N(2)==2)) disp('full house'); end
elseif((N(1)==3)*(N(2)<2)) disp('three of a kind'); end
elseif((N(1)==2)*(N(2)==2)) disp('two pair'); end
elseif((N(1)==2)*(N(2)<2)) disp('pair'); end
else disp('high card');
end
```

Now that the cards are sorted by frequency, you can draw cards based upon the hand type

```
% Draw Step
   if(NS(1) == 5)
       disp('flush');
   elseif((N(1) == 1) * (max(Value)-min(Value)==4))
       disp('straight');
   elseif(N(1) == 4)
       disp('four of a kind');
   elseif((N(1)==3)*(N(2)==2))
       disp('full house');
   elseif((N(1)==3)*(N(2)<2))
       disp('three of a kind - draw two');
       Hand(4:5) = Deck(6:7);
   elseif((N(1)==2)*(N(2)==2))
       disp('two pair - draw one');
       Hand(5) = Deck(6);
    elseif((N(1)==2)*(N(2)<2))
       disp('pair - draw three');
       Hand(3:5) = Deck(6:8);
   else
       disp('high card - draw four');
       Hand(2:5) = Deck(6:9);
   end
```

Once you've drawn your cards, recheck what type of hand you have and count the frequency.

To validate this program, test it against something where we know the odds: poker with zero draw steps (last lecture). Running the simulation several times results up in numbers that match the calculated odds:

	St-Fl	4ok	Full-Hou	Flush	Str	3ok	2-Pair	Pair	High-C
Calc	1.53	24.01	145.21	196.54	392.46	1,997.41	4,753.9	42,256.9	50,117.73
run1	1	27	124	218	402	2,175	4,689	42,187	50,177
run2	1	20	144	203	423	2,145	4,800	42,219	50,145
run 3	1	36	153	203	411	2,090	4,767	42,362	49,977

Monte-Carlo results for 100,000 hands of poker (no draws)

From these results, it looks like Monte-Carlo program is running and recording the types of hands correctly

- The results change each run (it should being a random simulation) and

- The results are reasonably close to the calculated values.

Now turn on the draw step:

	St-Fl	4ok	Full-Hou	Flush	Str	3ok	2-Pair	Pair	High-C
Calc	-	245.8	-	-	-	-	-	-	-
run1	0	236	1,199	294	619	7,874	13,511	51,638	24,629
run2	2	258	1,179	281	503	7,833	13,521	51,752	24,671
run 3	3	223	1,175	300	599	7,857	13,647	51,494	24,702

Monte-Carlo results for 100,000 hands with one draw step

Here, the Monte-Carlo results for run1-run3 are similar to what was calculated for arriving at four-of-a-kind. So, the program and calculations agree (and are probably correct).

Note that flush and straights do not change that much since the rules used never try for a flush or straight. Changing the rules and program would allow you to try for a flush or straight if you are only one or two cards away.

Expected Return with Cassino Poker

Previously, the payout for a poker slot machine was computed for 5-card stud. If you add a draw step, the payout changes. Using run #3 from the above simulation as the odds of each type of hand gives the following:

POKER HAND	COUNT	Odds Against	Payout	Return
Royal Flush	0		800	0
Straight Flush	3	333,333	50	0
Four of a Kind	223	448	20	0.04
Full House	1,175	85	6	0.07
Flush	300	333	5	0.02
Straight	599	170	4	0.02
Three of a Kind	7,857	12.73	3	0.24
Two Pair	13,647	7.33	2	0.27
One Pair	51,494	1.94	1	0.52
High Card	24,702	4.05	0	0
Total:				1.18

The return is greater than 1.00, meaning that with this payout scheme for draw poker, you should actually make 18 cents each time you play a game for $1. A payout bigger than 1 means you should play this game - or more likely means that the payout odds are too high.

Multiple Draw Steps

One advantage of a Monte-Carlo program is you can include multiple draw steps. Each draw step changes the odds. Calculating these odds using conditional probability gets rather complex, however, since there are now many more ways to arrive at a given hand. Simulating multiple draw steps is easy, however:

- Use a pointer for the top of the deck and
- Place the draw phase in a for-loop

Draw Steps	St-Fl	4ok	Full-Hou	Flush	Str	3ok	2-Pair	Pair	High-C
0	1	30	137	183	382	2,103	4,803	42,656	49,796
1	3	241	1,119	303	581	7,936	13,458	51,459	24,900
2	2	737	3,345	385	693	13,931	21,979	47,228	11,700
3	4	1,612	6,618	376	731	18,705	27,817	38,265	5,512
4	1	2,769	10,871	374	745	21,934	31,362	29,536	2,408

Frequency of each type of hand type with multiple draw rounds of 100,000 hands

Again, the straight and flush categories remain low since the rules used never try for either of these. As for the other numbers, they may be correct - or there may be some programming errors. They look reasonable, however.

Summary

Combinatorics works well when computing the odds of 5-card stud. With this game, the number of cards drawn by each player is fixed, making the odds much easier to compute.

With 5-card draw, the odds depend upon the rules used for which cards to discard and which ones to draw. With conditional probabilities, you can calculate the odds of ending up at a given hand by compute all of the different paths to get to that final hand.

With Monte-Carlo simulations, you can check if your computed odds are reasonable. You can also calculate whether the calculated odds lie within a window if you run multiple simulations. This is a student t-test and is a topic we'll be covering later on in this course.

Final Matlab Code

```
% ECE 341 Lecture #3
% 5 card draw poker

DrawSteps = 1;

tic
Flush = 0;
Straight = 0;
StraightFlush = 0;
RoyalFlush = 0;
Pair4 = 0;
Pair32 = 0;
Pair3 = 0;
Pair22 = 0;
Pair2 = 0;
HighCard = 0;

for i0 = 1:1e5
   X = rand(1,52);
   [a,Deck] = sort(X);
   Deck = Deck - 1;
   Hand = Deck(1:5);
   Value = mod(Hand,13) + 1;
   Suit = floor(Hand/13) + 1;

   % Top of Deck
   Top = 6;
   for i1 = 1:DrawSteps

   % check for flushes
      NS = zeros(1,4);
      for i=1:4
         NS(i) = sum(Suit == i);
      end
      NS = sort(NS, 'descend');

   % sort by frequency
      N = Value / 100;
      for i=1:5
         N(i) = sum(Value(i) == Value);
      end
      [N, b] = sort(N, 'descend');
      Hand = Hand(b);
      Value = Value(b);
      Suit = Suit(b);

   % Determine max frequency of cards
      N = zeros(1,13);
      for i=1:13
         N(i) = sum(Value == i);
      end
      N = sort(N, 'descend');

      if(1)
          % check for a straight
          Vx = sort(Value);
          flag = 0;
          if(N(1) == 1)
             if((max(Value) - min(Value))==4)
```

```
                flag = 1;
            elseif( (Vx(1)==1)*(Vx(2)==10) )
                flag = 1;
            end
         end
      % Draw Step
         if(NS(1) == 5)
         elseif(flag==1)
         elseif(N(1) == 4)
         elseif((N(1)==3)*(N(2)==2))
         elseif((N(1)==3)*(N(2)<2))
            Hand(4:5) = Deck(Top:Top+1);
            Top = Top + 2;
         elseif((N(1)==2)*(N(2)==2))
             Hand(5) = Deck(Top);
             Top = Top + 1;
         elseif((N(1)==2)*(N(2)<2))
             Hand(3:5) = Deck(Top:Top+2);
             Top = Top + 3;
         else
             Hand(2:5) = Deck(Top:Top+3);
             Top = Top + 4;
         end
      Value = mod(Hand,13) + 1;
      Suit = floor(Hand/13) + 1;
      end
   end
   % recompute hand type
   NS = zeros(1,4);
   for i=1:4
      NS(i) = sum(Suit == i);
   end
   NS = sort(NS, 'descend');

   N = zeros(1,13);
   for i=1:13
      N(i) = sum(Value == i);
   end
   N = sort(N, 'descend');

   % check for a straight
   Vx = sort(Value);
   flag = 0;
   if(N(1) == 1)
       if((max(Value) - min(Value))==4)
           flag = 1;
       elseif( (Vx(1)==1)*(Vx(2)==10) )
           flag = 2;
       end
   end

   if(NS(1) == 5)
       if(flag == 1)
           StraightFlush = StraightFlush + 1;
       elseif(flag == 2)
           RoyalFlush = RoyalFlush + 1;
       else
           Flush = Flush + 1;
       end
   elseif(flag > 0)
       Straight = Straight + 1;
```

```
        elseif(N(1) == 4)
            Pair4 = Pair4 + 1;
        elseif((N(1)==3)*(N(2)==2))
            Pair32 = Pair32 + 1;
        elseif((N(1)==3)*(N(2)<2))
            Pair3 = Pair3 + 1;
        elseif((N(1)==2)*(N(2)==2))
            Pair22 = Pair22 + 1;
         elseif((N(1)==2)*(N(2)<2))
            Pair2 = Pair2 + 1;
        else
            HighCard = HighCard + 1;
        end
    end
    disp('---------------------------------');
    disp(['Draw Steps = ', int2str(DrawSteps)]);
    disp(['Royal Flush  ', int2str(RoyalFlush)]);
    disp(['Str Flush    ', int2str(StraightFlush)]);
    disp(['4 of kind    ', int2str(Pair4)]);
    disp(['Full House   ', int2str(Pair32)]);
    disp(['Flush        ', int2str(Flush)]);
    disp(['Straight     ', int2str(Straight)]);
    disp(['3 of Kind    ', int2str(Pair3)]);
    disp(['2-Pair       ', int2str(Pair22)]);
    disp(['Pair         ', int2str(Pair2)]);
    disp(['High Card    ', int2str(HighCard) ]);
    toc
```

Typical output:

```
---------------------------------
Draw Steps = 1
Royal Flush  0
Str Flush    2
4 of kind    239
Full House   1230
Flush        294
Straight     601
3 of Kind    7957
2-Pair       13527
Pair         51396
High Card    24754
Elapsed time is 33.080047 seconds.
```