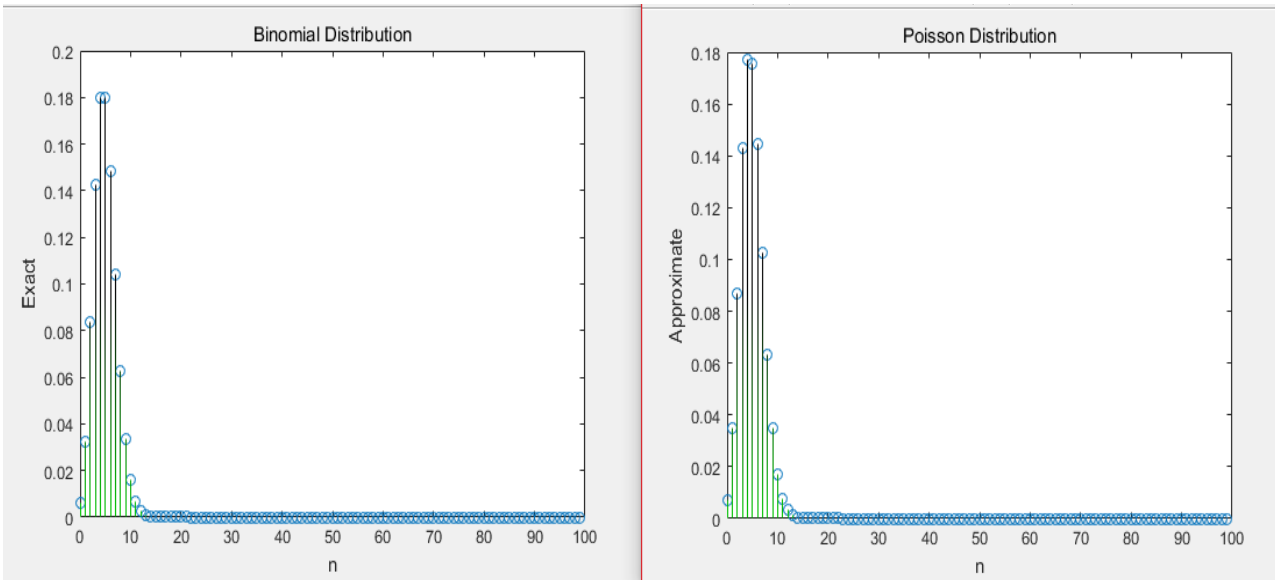
Question c:

```
% c)
function [input,output] = one_channel(p1,p2)

%This function generate a input bit '0' or '1' ask for the probability
of erased---p1 and the probability
%of swap---p2, and return the input and output bit.

%generate an input bit '0' or '1', the probability of produce '0' is q
q=0.5;
if(rand < q)
   input = 0;
else
   input = 1;
end

%the input bit can be erased with probability p1
if(rand < p1)
   output = -1; %use -1 to denote erase
else if(rand < p2)
      %The input bit can be swapped if not erased
   output = 1 - input;
   else
   %Or the input bit can remain the same
   output = input;
   end
end
```

Question d:

```
function plot_d
p2 = 0.05;
n = 99;
%plot the exact pmf, which is Binomial distribution
figure(1);
x1 = 0:n;
Sn1 = zeros(1,n+1);
%Shift k to be 0 for Sn1(1) and so forth
for k = 1:n+1
   Sn1(k) = (factorial(n)/(factorial(k-1)*factorial(n-(k-1))))*...
       (p2^(k-1))*((1-p2)^(n-(k-1)));
end
stem(x1, Sn1)
title('Binomial Distribution');
xlabel('n')
ylabel('Exact')
%plot the approximate pmf using Poisson approximation
alpha = p2 * n;
figure(2);
x2 = 0:n;
Sn2 = zeros(1,n+1);
for k = 1:n+1
```

```
    Sn2(k) =((alpha^(k-1))/(factorial(k-1)))*exp(-alpha);
end
stem(x2, Sn2)
title('Poisson Distribution');
xlabel('n')
ylabel('Approximate')
```



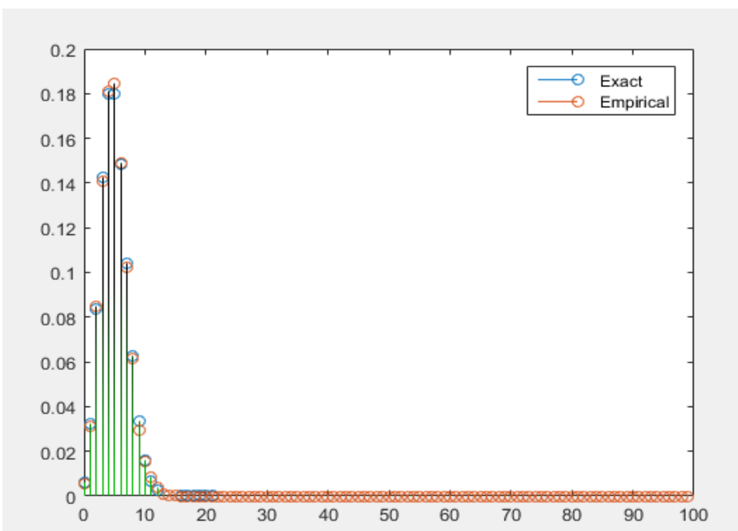
Compare and comment:

From the result we can see that the plot for these two expressions of pmf looks quite similar. Although Poisson is just an estimate, with n large enough and p small enough we still see the similar result. The plot will match more closely if n is larger and p is smaller.

<u>Question e:</u>

```
function [swaps] = n_channel(p1,p2,n)
% this function calls c)'s function one_channel for n times
% and record the number of bit swaps
    swaps = 0;
    for i = 1:n
        [input,output] = one_channel(p1, p2);
        if(input ~= output)
            swaps = swaps + 1;
        end
    end
end

function plot_e
n = 99;
m = 10000;
```

```
p1 = 0;
p2 = 0.05;
%plot the exact pmf
x1 = 0:n;
Sn1 = zeros(1,n+1);
for k = 1:n+1
    Sn1(k) = (factorial(n)/(factorial(k-1)*factorial(n-(k-1))))*...
        (p2^(k-1))*((1-p2)^(n-(k-1)));
end
figure(3);
stem(x1, Sn1)
hold on
%plot the empirical pmf by calling n_channel m times
x2 = 0:n;
Sn2 = zeros(1,n+1);
S = 0;
for i = 1:m
    S = n_channel(p1, p2, n);
    Sn2(S+1) = Sn2(S+1) + 1;    %Use the number of swaps itself as index
end
stem(x2,Sn2/m);
legend('Exact','Empirical')
```

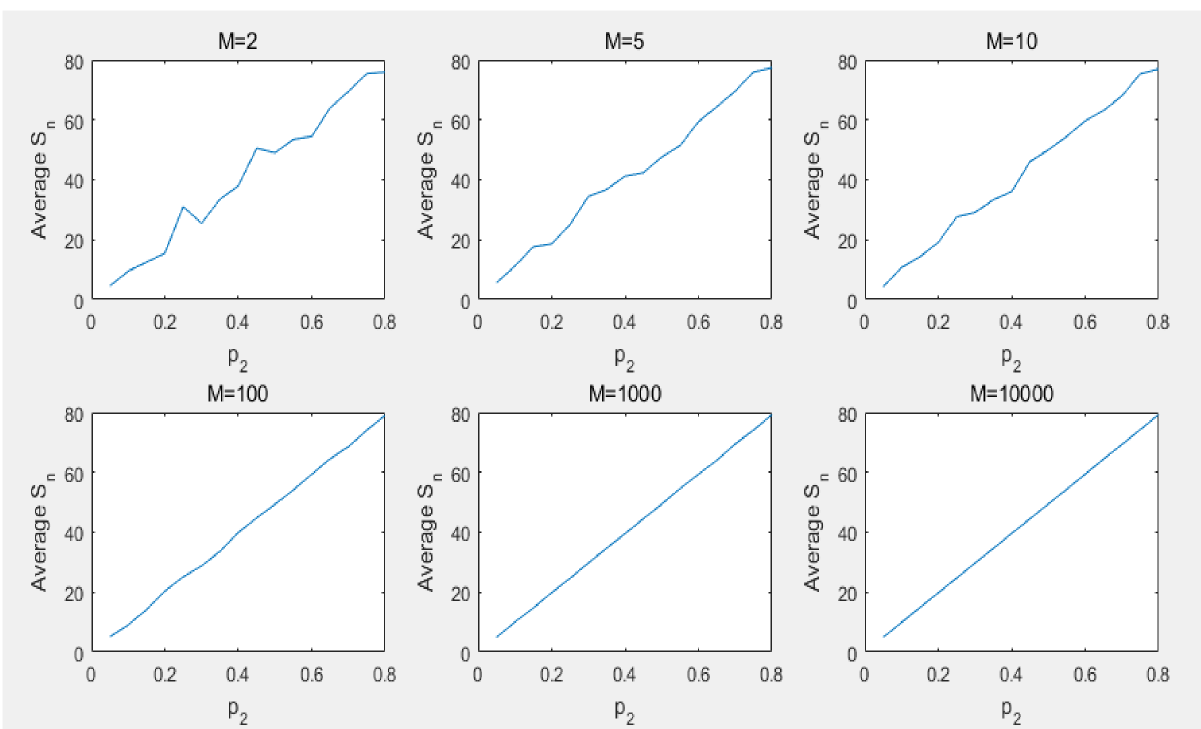


Compare and comment:

Here we see that calculating the empirical pmf provides quite similar results with a sample of 10,000. However it is still quite off the exact pmf, to produce a more similar result we need to make M even bigger.

Question f:

```
function plot_f
n = 99;
m = [2 5 10 100 1000 10000];
p1 = 0;
Sn = zeros(1,n+1);
S = 0;
Avg = zeros(1,16);
x1 = 0.05:0.05:0.8;
figure(4);
hold on;
% repeat calling n_channel for each p and M value
% and compute the average bit swaps
for j = 1:6
    p2 = 0.05;
    for k = 1:16
        Sum = 0;
        for i = 1:m(j)
            S = n_channel(p1, p2, n);
            Sum = Sum + S;
        end
        Avg(k) = Sum / m(j);
        p2 = p2 + 0.05;
    end
    subplot(2,3,j);
    plot(x1, Avg);
    xlabel('p_{2}')
    ylabel('Average S_{n}')
    title(['M=',num2str(m(j))])
end
```

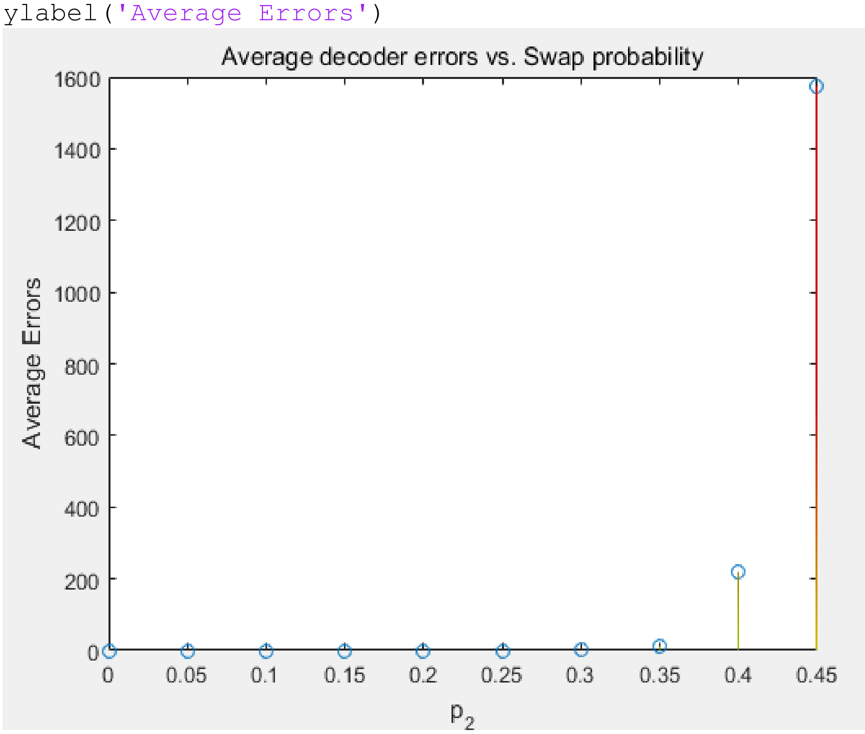


Compare and comment:
From the plot we notice that p2 and average number of bit swaps has a linear relationship, and with larger M the line is more smooth. So the the M=10,000 line is more straight than other lines. Also there is a positive relationship between p2 and average bit swaps, with larger p2, a higher chance of bit swaps will occur and average bit swap times will be bigger.
This empirical average calculated is equivalent to the 'expected value' of a distribution. If we multiply each value in the pmf by its probability and sum them, we would arrive at the same value. These results reflect the fact that the 'expected value' of a distribution is simply the amount of times you would expect an event to occur on average.

Question h:

```
function plot_h
n = 99;
m = 10000;
k = 10;
p1 = 0;
p2 = 0;
swaps = 0;
x = 0:0.05:0.45;
Avg = zeros(1,10);
j = 1;

% go through every p2 values, recording the average
% number of errors for each p2
for p2 = 0:0.05:0.45

    error = 0;
    %Below loop gets errors amongst k tries
    for z = 1:k
        %Below loop conditions for one error
        for i = 1:m
            swaps = n_channel(p1, p2, n);
            %if the swap bits are more than half of the outputs,
according to
            %the decision rule, the decoder will have an error
            if swaps > n/2
                error = error + 1;
```

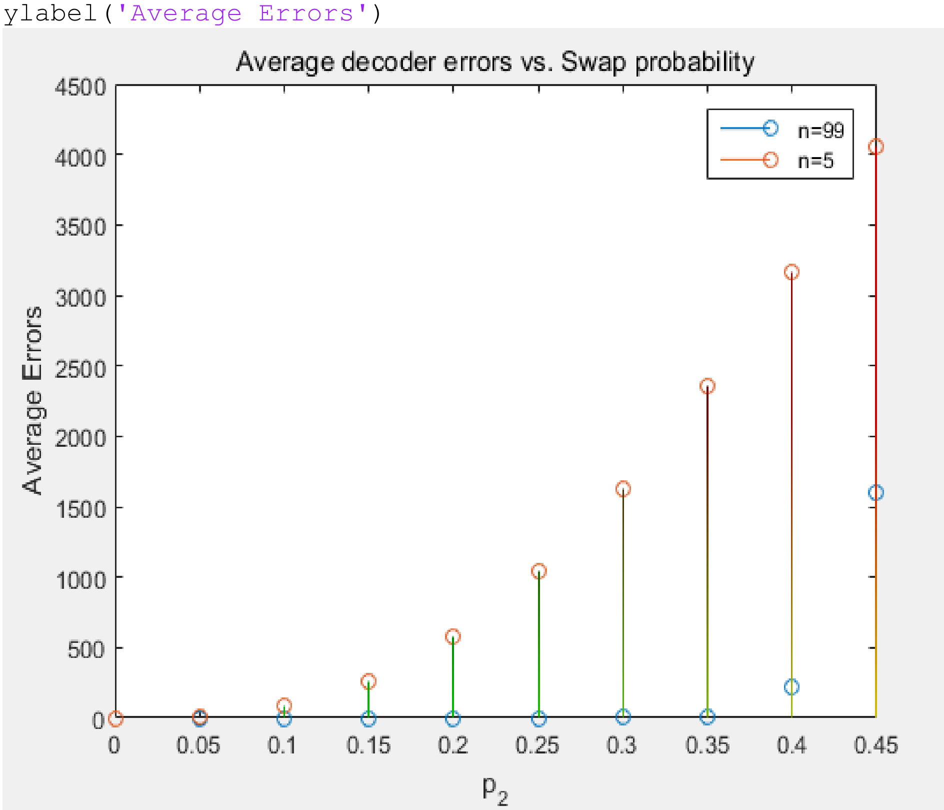
```
            end
        end
    end
    Avg(j) = error / k;
    j = j+1;
end
figure(5);
stem(x, Avg);
title('Average decoder errors vs. Swap probability')
xlabel('p_2')
ylabel('Average Errors')
```



Question i:

```
function plot_i
n1 = 99;
n2 = 5;
m = 10000;
k = 10;
p1 = 0;
p2 = 0;
x = 0:0.05:0.45;
swaps = 0;
Avg = zeros(1,10);
Avg2 = zeros(1,10);
j = 1;

% go through every p2 values, recording the average
% number of errors for each p2
for p2 = 0:0.05:0.45
```

```
    error = 0;
    error2 = 0;
    for z = 1:k
        for i = 1:m
            swaps = n_channel(p1, p2, n1);
            %if the swap bits are more than half of the outputs,
according to
            %the decision rule, the decoder will have an error
            if swaps > n1/2
                error = error + 1;
            end
            %record errors for n=5
            swaps5 = n_channel(p1,p2,5);
            if swaps5 > n2/2
                error2 = error2 + 1;
            end
        end
    end
    Avg(j) = error / k;
    Avg2(j) = error2 / k;
    j = j+1;
end
figure(6);
stem(x, Avg);
hold on
stem(x, Avg2);
legend('n=99','n=5')
title('Average decoder errors vs. Swap probability')
xlabel('p_2')
ylabel('Average Errors')
```



Compare and comment:

Changing n to 5 has a big effect on the overall shape of the plot. We can see that the average number of errors is much greater, the reason is that when n = 5, for an error to occur, only 3 bit swaps are required (according to the decision rule, if the swap bits are more than half of the outputs, the decoder will have error). The probability of 3 swaps occurring is far higher than 50 swaps (when n = 99 has error). Therefore, the number of errors increase as n decreases.