PS1 Q.1 Solution

a. $P(A) = 0.1$; $P(B) = 0.4$; $P(C|A) = 0.5$; $P(C|B) = 0.2$; $P(C|A', B') = 0$; $P(C|A, B) = 1$.
where A' means NOT A etc. and A, B means "both A and B" etc.

b. We need to calculate $P(A|C)$.
Bayes' Theorem tells us

$$P(A|C) = \frac{P(A)P(C|A)}{P(C)}.$$

We know $P(A)$ and $P(C|A)$ so we must calculate $P(C)$.

There are two ways of doing this. We can consider separately each of the 4 scenarios under which C could fire and add up all their separate probabilities:

$$P(C) = P(A,B,C) + P(A,B',C) + P(A',B,C) + P(A',B',C)$$
$$= P(C|A,B)P(A,B) + P(C|A,B')P(A,B') + P(C|A',B)P(A',B) + P(C|A',B')P(A',B').$$

Since A and B are independent probabilities of any combination are the products of the individual probabilities and we know $P(A') = 1 - P(A) = 0.9$ and $P(B') = 1 - P(B) = 0.6$ so that:
$P(A,B) = 0.04$; $P(A,B') = 0.06$; $P(A',B) = 0.36$; ~~$P(A,B') = 0.54$.~~

We still need to know $P(C|A,B')$ and $P(C|A',B)$.

We can use $P(C|A)$ to calculate the first as follows:

$$P(C|A) = P(C|A,B)P(B) + P(C|A,B')P(B')$$

(because A can happen either with B or without B)
Filling in the values we know:

$$0.5 = 1 \times 0.4 + P(C|A,B') \times 0.6$$

and rearranging gives:

$$P(C|A,B') = \frac{1}{6}.$$

Similarly

$$P(C|B) = P(C|A,B)P(A) + P(C|A',B)P(A')$$

so

$$0.2 = 1 \times 0.1 + P(C|A',B) \times 0.9$$

and

$$P(C|A',B) = \frac{1}{9}.$$

Putting everything together:

$$P(C) = P(C|A,B)P(A,B) + P(C|A,B')P(A,B') + P(C|A',B)P(A',B) + P(C|A',B')P(A',B')$$
$$= 1 \times 0.04 + \left(\frac{1}{6}\right) \times 0.06 + \left(\frac{1}{9}\right) \times 0.36 + 0 \times 0.54$$
$$= 0.09$$

For a quicker way of calculating $P(C)$ we can use the Venn Diagram to see that

$$\begin{aligned} P(C) &= P(A,C) + P(B,C) - P(A,B,C) + P(A',B',C). \\ &= P(C|A)P(A) + P(C|B)P(B) - P(C|A,B)P(A,B) + P(C|A',B')P(A',B') \\ &= 0.5\times0.1 + 0.2\times0.4 - 1\times0.04 + 0\times0.54 \\ &= 0.09. \end{aligned}$$

Whichever way we do it, we have $P(C) = 0.09$ which means we can answer the question:

$$\begin{aligned} P(A|C) &= \frac{P(A)P(C|A)}{P(C)} \\ &= \frac{0.1\times0.5}{0.09} \\ &= 5/9. \end{aligned}$$

c.

$$\begin{aligned} P(B|C) &= \frac{P(B)P(C|B)}{P(C)} \\ &= \frac{0.4\times0.2}{0.09} \\ &= 8/9. \end{aligned}$$

d.

$$\begin{aligned} P(A,B|C) &= \frac{P(A,B)P(C|A,B)}{P(C)} \\ &= \frac{0.04\times1}{0.09} \\ &= 4/9. \end{aligned}$$

e.

$$\begin{aligned} P(A,B'|C) &= \frac{P(A,B')P(C|A,B')}{P(C)} \\ &= \frac{0.06\times(1/6)}{0.09} \\ &= 1/9. \end{aligned}$$

f.

$$\begin{aligned} P(A',B|C) &= \frac{P(A',B)P(C|A',B)}{P(C)} \\ &= \frac{0.36\times(1/9)}{0.09} \\ &= 4/9. \end{aligned}$$

2. A Simulation code:

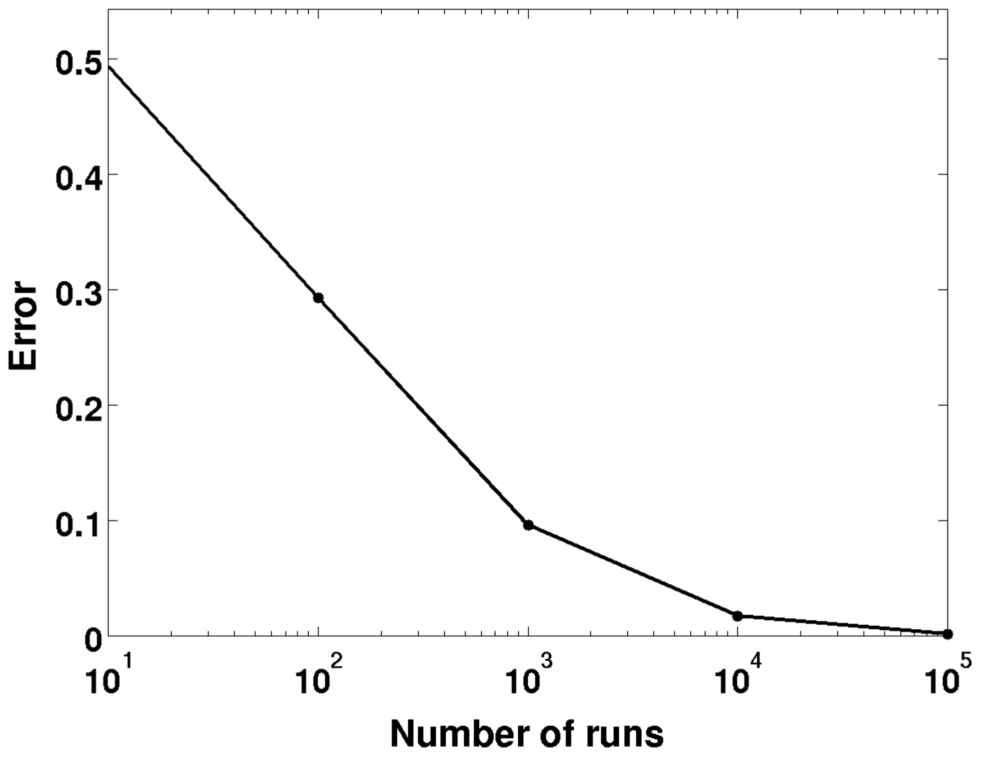
```
% set the number of runs for the simulation
nruns = 10000;

% the probability that a fired is 0.1
pA = 0.1;

% the probability that b fired is 0.4
pB = 0.4;

% the probability of c firing given that a fired is 0.5
pCgivenA = 0.5;

% the probability of c firing given that b fired is 0.2
pCgivenB = 0.2;

% the probability of c firing given that a and b fired is 1
pCgivenAB = 1;

%create empty matricies to hold the zeros and ones corresponding
to whether
%neurons A, B and C spike or not on each run
A = zeros(nruns,1);
B = zeros(nruns,1);
C = zeros(nruns,1);

% use a for loop to iterate through the runs for neurons a and b
for run = 1:nruns
    % generate a random number bewteen 0 and 1
    randnum = rand(1,1);
    %  set a to have fired if that number is < pa;
    if (randnum < pA)
     A(run) = 1;
    else
     A(run) = 0;
     % note that this is redundant, as a(run) is 0 to begin
with;
    end

    % repeat that for neuron b
    randnum = rand(1,1);
    if (randnum < pB)
     B(run) = 1;
    else
```

```
      B(run) = 0;
     end

     % now that we have the firing of a and b, we can generate the
simulations
     % for c.  We'll do this with more compact code
     % c | a
     if (A(run) == 1)
      % compare the random number to pcgivena and assign the
result
      C(run) = rand(1,1) < pCgivenA;
     end
     % if a did not cause c to fire, b still could
     if ((C(run) == 0) & (B(run) == 1))
      % compare the random number to pcgivenb and assign the
result
      C(run) = rand(1,1) < pCgivenB;
     end
     % if neither a nor b caused c to fire on their own, but both
were active, c
     % could fire
     if ((C(run) == 0) & (A(run) == 1) & (B(run) == 1))
      C(run) = rand(1,1) < pCgivenAB;
     end
end

% now we can answer the questions
% we'll go through all of the runs and count up the number where
C fired and
% the number of those where A also fired, and similarly for B and
the other
% combinations
Cfired = 0; % c
ACfired = 0;% a & c
BCfired = 0; % b & c
ABCfired = 0; %a, b and c
AnBCfired = 0; % a, not b, and c
nABCfired = 0; % not a, b, and c
for run = 1:nruns
     % check to see if c fired on this run.
     % Note that if (a) is equivalent to if (a ~= 0)
     if (C(run))
      Cfired = Cfired + 1;
      if (A(run))
          ACfired = ACfired + 1;
      end
      if (B(run))
          BCfired = BCfired + 1;
      end
      if (A(run) & B(run))
          ABCfired = ABCfired + 1;
      end
      if (A(run) & ~B(run))
          AnBCfired = AnBCfired + 1;
      end
      if (~A(run) & B(run))
          nABCfired = nABCfired + 1;
      end
    end
end
% p(A|C) is the proportion of times that a fired when c fired:
pAgivenC = ACfired / Cfired
0.4997

% p(B|C) is the proportion of times that a fired when c fired:
pBgivenC = BCfired / Cfired
0.7784

% p(AB|C) is the proportion of times that a and b fired when c
fired:
pABgivenC = ABCfired / Cfired
0.278

% p(A & ~B|C) is the proportion of times that a but not b fired
when c fired:
pAnBgivenC = AnBCfired / Cfired
0.2216

% p(~A & B|C) is the proportion of times that b but not a fired
when c fired:
pnABgivenC = nABCfired / Cfired
0.5003



% That all works, but it's hideously inefficient.  Here is a much
more efficient
% version
nruns = 10000;
a = zeros(nruns,1);
```

```
b = zeros(nruns,1);
% a and not b and c
anbc = zeros(nruns,1);
% not a and b and c
nabc = zeros(nruns,1);
% a and b and c
abc = zeros(nruns,1);

f = rand(nruns,4);
% use the first two columns to figure out when a and b fired.
a = f(:,1) < pA;
b = f(:,2) < pB;

% to get anbc, we use column 3 to figure out when C fired as a
result of A
% alone
anbc = a & ~b & (f(:,3) < pCgivenA);

% to get nanc, we use column 4 to figure out when C fired as a
result of B
% alone
nabc = ~a & b & (f(:,4) < pCgivenB);

% to get a & b & c we only need to get a & b, as P(C | A & B) = 1
abc = a & b;

% the set of runs where C fired is the union of the anbc, nabd
and abc arrays
c = anbc | nabc | abc;

% p(a|c)
pac = (sum(anbc) + sum(abc)) / sum(c)

% p(b|c)
pbc = (sum(nabc) + sum(abc)) / sum(c)

%p(ab|c)
pabc = sum(abc) / sum(c)

%p(a~b|c)
panbc = sum(anbc) /  sum(c)

%p(~ab|c)
pnabc = sum(nabc) / sum(c)
```

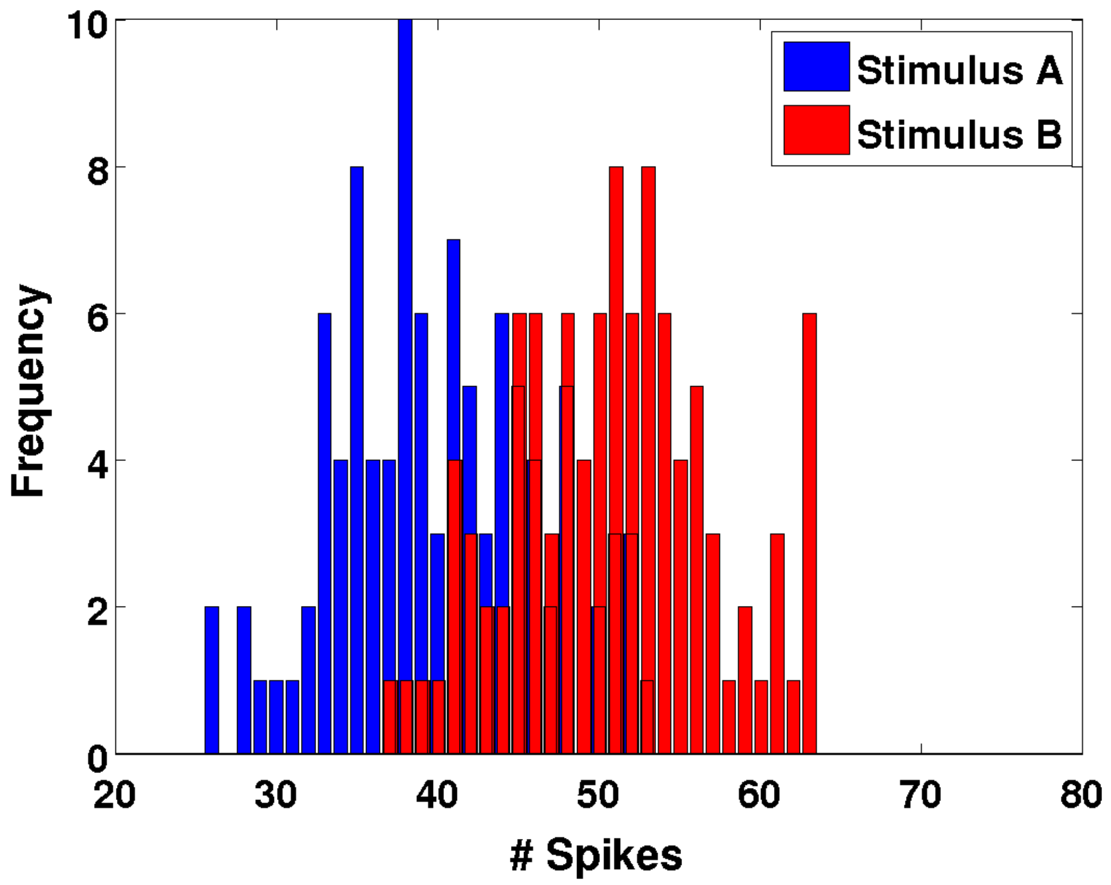
2. B. Error calculations and plotting.

```
% C.  Error calculation.  The most efficient way to do this is to
do one run of
% 100000 iterations and take parts of it for each comparison.
% here is the answer from part A
actualpac = 0.493;

nruns = 100000;
a = zeros(nruns,1);
b = zeros(nruns,1);
% a and not b and c
anbc = zeros(nruns,1);
% not a and b and c
nabc = zeros(nruns,1);
% a and b and c
abc = zeros(nruns,1);
f = rand(nruns,4);
% use the first two columns to figure out when a and b fired.
a = f(:,1) < pA;
b = f(:,2) < pB;

anbc = a & ~b & (f(:,3) < pCgivenA);

nabc = ~a & b & (f(:,4) < pCgivenB);

abc = a & b;

c = anbc | nabc | abc;

nruns = [1 10 100 1000 10000 100000];
err = zeros(size(nruns));

for r = 1:length(nruns)
    err(r) = abs(actualpac - (sum(anbc(1:nruns(r))) +
sum(abc(1:nruns(r)))) /...
           sum(c(1:nruns(r))));
end

% in our case, neuron C didn't fire on the first run, so err(1) is
NaN.
% plot the errors that make sense
figure;
semilogx(nruns(2:end), err(2:end), 'LineWidth', 2)
h = xlabel('Number of runs')
set(h, 'FontSize', 20);
h = ylabel('Error');
set(h, 'FontSize', 20);
set(gca, 'YLim', [0 max(err(2:end))*1.1]);
set(gca, 'FontSize', 18);
```



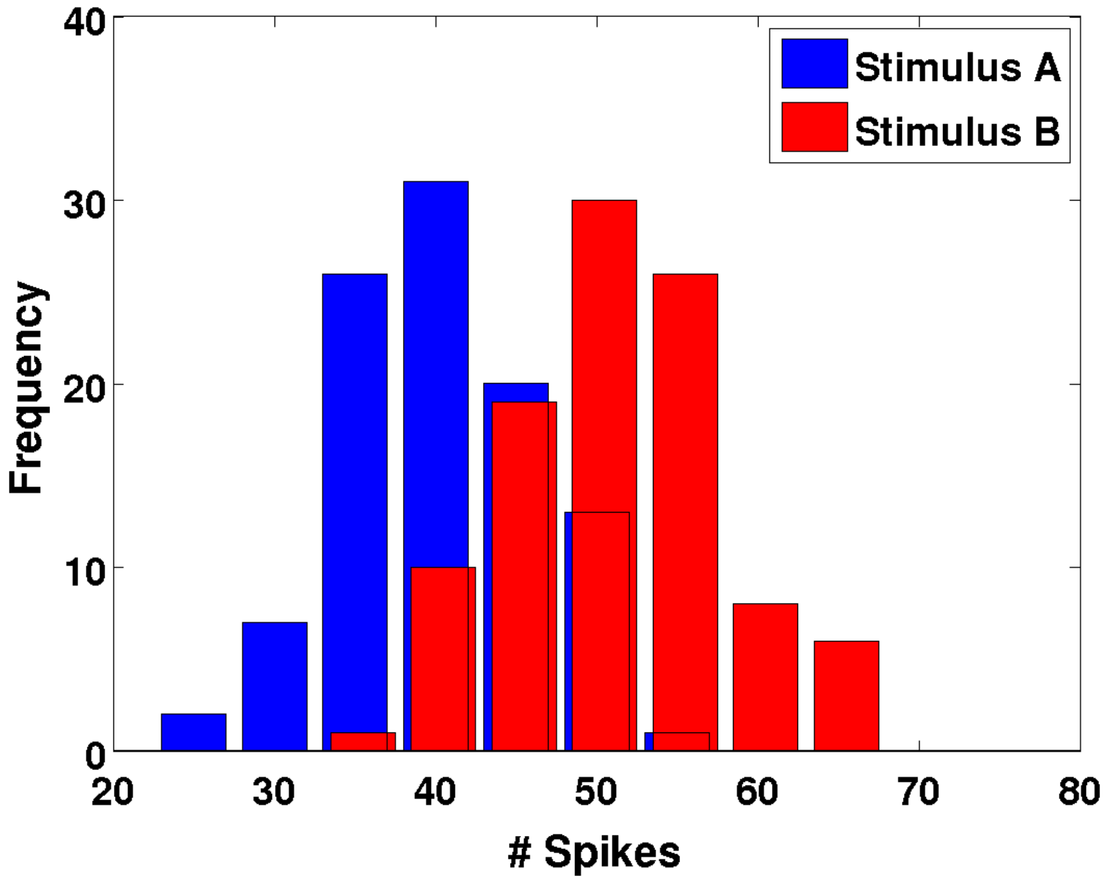
3. A. Histogram with 1 count bins:

```
load Probability_Data1.mat

% A.  Plot a histogram with 1 count bins
% note that my version of hist is different from the normal
matlab version, so
% the bins you should specify would be 0:1:100 and 0:5:100
stimA = 1:2:length(s1);
stimB = 2:2:length(s1);
[na1 x1] = hist(s1(stimA),.5:1:100);
[nb1 x1] = hist(s1(stimB),.5:1:100);

bar(x1, nb1, 'b');
hold on
bar(x1+.1, na1, 'r');
set(gca, 'XLim', [20 80], 'FontSize', 18);
```



B. Histogram with 5 count bins

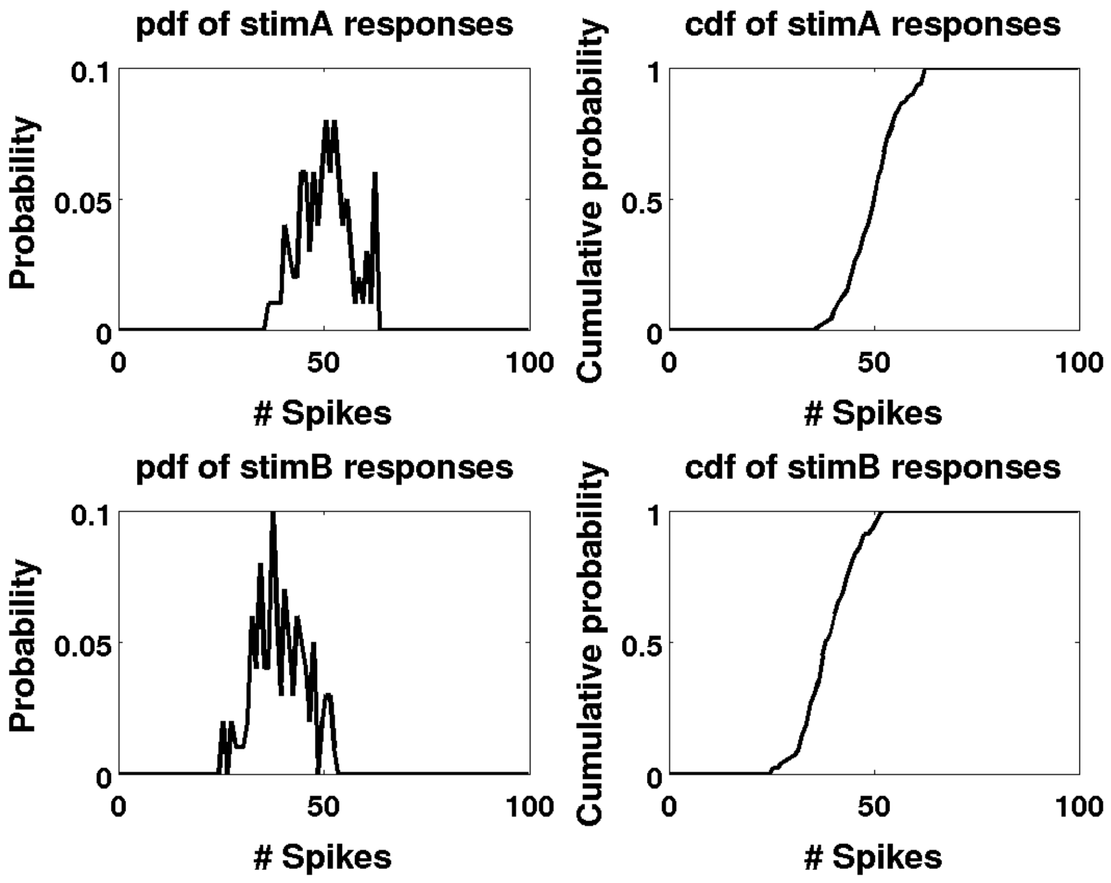
```
% B.  Plot a histogram with 5 count bins
stimA = 1:2:length(s1);
stimB = 2:2:length(s1);
[na5 x5] = hist(s1(stimA),2.5:5:100);
[nb5 x5] = hist(s1(stimB),2.5:5:100);

figure
bar(x5, nb5, 'b');
hold on
bar(x5+.5, na5, 'r');
set(gca, 'XLim', [20 80], 'FontSize', 18);
legend('Stimulus A', 'Stimulus B');
```



C.

```
% C. pdfs and cdfs.
figure
% we'll use four subplots, one for each plot
subplot(2,2,1)
%to get the pdf for stimulus A responses, we divide na1 by the
total number of
%trials, which is the sum of na1
pdfa1 = na1./sum(na1);
plot(x1, pdfa1, 'k-', 'LineWidth', 2);
set(gca, 'FontSize', 14);
h = xlabel('# Spikes');
set(h, 'FontSize', 16);
h = ylabel('Probability');
set(h, 'FontSize', 16);
h = title('pdf of stimA responses');
set(h, 'FontSize', 16);

%to get the cdf for stimulus A responses, we take the cumulative
sum of na1 and
%divide by the total sum:
subplot(2,2,2)
```

```
cdfa1 = cumsum(na1);
cdfa1 = cdfa1 ./ cdfa1(end);
% note that you could also use ecdf function
plot(x1, cdfa1, 'k-', 'LineWidth', 2);
set(gca, 'FontSize', 14);
h = xlabel('# Spikes');
set(h, 'FontSize', 16);
h = ylabel('Cumulative probability');
set(h, 'FontSize', 16);
h = title('cdf of stimA responses');
set(h, 'FontSize', 16);

subplot(2,2,3)
pdfb1 = nb1./sum(nb1);
plot(x1, pdfb1, 'k-', 'LineWidth', 2);
set(gca, 'FontSize', 14);
h = xlabel('# Spikes');
set(h, 'FontSize', 16);
h = ylabel('Probability');
set(h, 'FontSize', 16);
h = title('pdf of stimB responses');
set(h, 'FontSize', 16);

subplot(2,2,4)
cdfb1 = cumsum(nb1);
cdfb1 = cdfb1 ./ cdfb1(end);
plot(x1, cdfb1, 'k-', 'LineWidth', 2);
set(gca, 'FontSize', 14);
h = xlabel('# Spikes');
set(h, 'FontSize', 16);
h = ylabel('Cumulative probability');
set(h, 'FontSize', 16);
h = title('cdf of stimB responses');
set(h, 'FontSize', 16);
```



The pdfs are normalized histograms.

D.

```
% D.  Probability of stimulus A given 44 spikes , 1 count bins
% normalize the histograms to be probabilities
na1 = na1./sum(na1);
nb1 = nb1./sum(nb1);

% find the bin number for a count of 44
[m ba] = min(abs(x1 - 44));

% p (A | 44 spikes) = P(44 spikes | A) P(A) / P(44 spikes)

p = na1(ba) * .5 / (na1(ba) * .5 + nb1(ba) * .5)

0.2500
```

E.

```
% E.  Probability of stimulus A given 44 spikes , 5 count bins
% normalize the histograms to be probabilities
```

```
na5 = na5./sum(na5);
nb5 = nb5./sum(nb5);

% find the bin number for a count of 44
[m ba] = min(abs(x5 - 44));

% p (A | 44 spikes) = P(44 spikes | A) P(A) / P(44 spikes)

p = na5(ba) * .5 / (na5(ba) * .5 + nb5(ba) * .5)

0.4872
```

E is likely to be closer because it appears to smooth over noise due to sampling.