MLM_Gaussian_1Param_Fits Example

We used MATLAB to carry out M.L.M. fits to the mean and sigma of a Gaussian distribution. The MATLAB script, output and plots from running the program are given below.

MATLAB script:

```
%==========================================================================
% MLM_1Param_Gaussian_Fits.m
% Written by Prof. S. Errede          last updated: 10/04/2010 10:15 hr
%
% Simple MatLab Monte Carlo exercise with the GAUSSIAN distribution:
% a.) Generates Nevts from random Gaussian distribution
%     - need to specify true mean, Xtru and true sigma, Stru.
% b.) Compute (unbiased) Sample Mean,Variance & Sigma
% c.) Histograms & plots the histogram result
% d.) Converts the histogram into a (binned) P.D.F., plots the result
% e.) Calculates the variance & covariance assoc. w/ histo bin contents
%
% n.b. each time this script is run, will get different MC results -
%      the (internal) random # seed is based on time-of-day...
%
% n.b. Variables are CASE-SENSITIVE in MATLAB!
%
% n.b. This computation is carried out in double precision for accuracy!
%==========================================================================
close all; % clear out all figures, etc. from immediately previous use..
clear all; % clear/zero-out/initialize all variables & arrays...

int32         NySteps;
int32         NzSteps;

int32      Nhist(100);

double   LnNhist(100);

double     Phist(100);
double     Chist(100);
double      Xctr(100);

double   Xrand(10000);

double   Ptrue(10000);
double   Punkn(10000);
double LnPtrue(10000);
double LnPunkn(10000);

double Hist_var(100);
double Hist_cov(100,100);

double          Xp(10000);
double LnLikeXtrue(10000);
double LnLikeXunkn(10000);

double          Sp(10000);
double LnLikeStrue(10000);
double LnLikeSunkn(10000);

double Xtru;
double Vtru;
double Stru;
double Ttru;
double Wtru;

double Xlo;
double Xhi;

double  Xsum;
double X2sum;
double  Xavg;
double X2avg;
double  Xvar;
double  Xsig;
double  Xset;
double  Xsets;

double   Ylo;
double   Yhi;
double    dY;
double     Y;

double   Zlo;
double   Zhi;
double    ZY;
double     Z;

double Xtolerance;
double Delta_LnLikeXtrue;
double Delta_LnLikeXunkn;

double Stolerance;
double Delta_LnLikeStrue;
double Delta_LnLikeSunkn;

double Phist_sum;

% Define the # events per "experiment":
Nevts = 10000;

fprintf(' \n');
fprintf('# MC Events = %i \n',Nevts);

% MC generate Nevnts from Gaussian distribution
% with true mean Xtru and true sigma Stru.
% Type "help random <enter>" in MatLab command window for more details.
% Mouse-click on blue-highlighted "doc random" for more info...

fprintf(' \n');
fprintf('MC Gaussian Distribution \n');

Xtru =  50.0;                % MC true mean
Vtru = 100.0;                % MC true variance
Stru = sqrt(Vtru);           % MC true sigma
Ttru = Stru/sqrt(Nevts);     % MC true setting error on true mean
Wtru = Stru/sqrt(2.0*Nevts); % MC true setting error on true sigma

fprintf(' \n');
fprintf('True Mean                        = %f \n',Xtru);
fprintf('True Variance                    = %f \n',Vtru);
fprintf('True Sigma                       = %f \n',Stru);
fprintf('True Setting Error on True Mean  = %f \n',Ttru);
fprintf('True Setting Error on True Sigma = %f \n',Wtru);


for i=1:Nevts;
    Xrand(i) = random('norm',Xtru,Stru);
end;

% Calculate the (unbiased) Sample Mean, Sample Variance and Sample Sigma:
Xsum  = 0.0;
X2sum = 0.0;
for i = 1:Nevts;
    Xsum  =  Xsum +  Xrand(i);
    X2sum = X2sum + (Xrand(i))^2;
end;

Xavg  =  Xsum/Nevts; % *unbiased* sample mean = simple/arithmetic mean
X2avg = X2sum/Nevts;
Xvar  = (Nevts/(Nevts-1))*(X2avg-(Xavg)^2); % *unbiased* sample variance
Xsig  =  sqrt(Xvar); % *unbiased* sample sigma
Xset  =  Xsig/sqrt(Nevts);
Xsets =  Xsig/sqrt(2.0*Nevts);

fprintf(' \n');
fprintf('<x>        = %f \n',Xavg);  % should be ~  50.0
fprintf('Var(x)     = %f \n',Xvar);  % should be ~ 100.0
fprintf('Sig-x      = %f \n',Xsig);  % should be ~  10.0
fprintf('Set-x      = %f \n',Xset);  % should be ~  10.0/sqrt(N)  ~ 0.10
fprintf('Set-Sig-x  = %f \n',Xsets); % should be ~  10.0/sqrt(2N) ~ 0.07


% Set up a 100 bin histogram on the interval [0,100]
% Define Xlo, Xhi and the bin width, bin centers:
Xlo   =   0.0;
Xhi   = 100.0;

Nbins = 100;
Xint  = Xhi - Xlo;
dX    = Xint/Nbins;

%fprintf(' \n');
for i = 1:Nbins;
    Xbin_lo = Xint*((i-1)/Nbins);
    Xctr(i) = Xbin_lo + (dX/2.0);

    % type/print out the Xctr array:
    % fprintf('%i %f \n',i,Xctr(i));
end;

% Now histogram the above MC generated data:
% Type "help hist <enter>" in the MatLab command window for more details:
Nhist = hist(Xrand,Xctr);

% Type/print out the Nhist array:
%fprintf(' \n');
%for i = 1:Nbins;
%    fprintf('%i %i \n',i,Nhist(i));
%end
```

```
% Plot the histogrammed/binned data as a (blue) bar graph:
% type "help figure <enter>" and/or "help bar <enter>"
% in the MatLab command window for more details.
figure(01);
%plot(Xctr,Nhist,'b');
bar(Xctr,Nhist,'b');
grid on;
xlabel('x');
ylabel('N(x)');
title('Gaussian Distribution: N(x) vs. x')

% Take (natural) log_e of # events in each histogram bin:
for i = 1:Nbins;
    LnNhist(i) = 0.0;
    if (Nhist(i) > 0)% can't take log(0)!
        LnNhist(i) = log(Nhist(i));
    end
end;

% Plot the log_e of histogrammed/binned data as a (blue) bar graph:
% type "help figure <enter>" and/or "help bar <enter>"
% in the MatLab command window for more details.
figure(02);
%plot(Xctr,LnNhist,'b');
bar(Xctr,LnNhist,'b');
grid on;
xlabel('x');
ylabel('Ln N(x)');
title('Gaussian Distribution: Ln N(x) vs. x')

% Now we normalize the MC Nhist data to turn it into a MC PDF:
for i = 1:Nbins;
    Phist(i) = Nhist(i)/Nevts;
end;

% Type/print out the Phist array:
%fprintf(' \n');
%for i = 1:Nbins;
%    fprintf('%i %f \n',i,Phist(i));
%end

% Plot the PDF data with red diamonds and w/ blue joining lines:
% type "help figure <enter>" and/or "help plot <enter>"
% in the MatLab command window for more details.
figure(03);
plot(Xctr,Phist,'rd',Xctr,Phist,'b-');
grid on;
xlabel('x');
ylabel('P(x)');
title('Gaussian Distribution: P(x) vs. x')

% Calculate Cumulative PDF from Phist data:
Phist_sum = 0.0;
for i = 1:Nbins;
    Phist_sum = Phist_sum + Phist(i);
    Chist(i)  = Phist_sum;
end;

% Plot the Cumulative PDF data with magenta stars and w/ blue joining lines:
% type "help figure <enter>" and/or "help plot <enter>"
% in the MatLab command window for more details.
figure(04);
plot(Xctr,Chist,'m*',Xctr,Chist,'b-');
grid on;
xlabel('x');
ylabel('C(x)');
title('Uniform Distribution: C(x) vs. x')

% Calculate the Variance and Covariance associated w/ histogram bins:
% due to multinomial probability distribution assoc/ w/ N histogram bins
for i = 1:Nbins;
    Hist_var(i)   = Nevts*Phist(i)*Phist(i);
    for j = 1:Nbins;
        Hist_cov(i,j) = -Nevts*Phist(i)*Phist(j); % n.b. (i,j) symmetric!
    end;
    Hist_cov(i,i) = 0.0; % diagonal elements of cov(x,x) don't exist
end;

% Plot the histogram variance data with green circles and w/ blue joining
lines:
% type "help figure <enter>" and/or "help plot <enter>"
% in the MatLab command window for more details.
figure(05);
plot(Xctr,Hist_var,'go',Xctr,Hist_var,'b-');
grid on;
xlabel('x');
ylabel('Hist var(x)');
title('Gaussian Distribution: Histogram Bin Variance')

% Plot the histogram covariance data as a 3-D surface:
% type "help figure <enter>" and/or "help surf <enter>"
% in the MatLab command window for more details.
figure(06);
surf(Xctr,Xctr,Hist_cov);
shading interp;
xlabel('x');
ylabel('x');
zlabel('Hist cov(x,x)');
title ('Gaussian Distribution: Histogram Bin Covariance(x,x)');

fprintf(' \n');
fprintf(' Please be patient/relax - this computation takes a while... \n');
% Calculate LnLike(Y) vs. Y to find mean, and sigma/setting error on mean:
% a.) based on *apriori   known* true sigma
% b.) based on *apriori unknown* true sigma
%
% n.b. from a computational perspective:
%      it is numerically *far more accurate* to
%      take the logs of individual P(i)'s and sum up lnP(i)'s than to
%      take the product of all P(i)'s and then take ln{Product P(i)'s}
%      because the ln{Product P(i)'s} is an *astronomically*
%      small # for a large # of MC events!!!
Yhi     = 55.00;
Ylo     = 45.00;
NySteps = 10000;
 dY     =  (Yhi-Ylo)/NySteps;
  Y     =   Ylo;
for j = 1:NySteps;
    Xp(j) = Y;
    Ptrue = pdf('norm',Xrand,Y,Stru);
    Punkn = pdf('norm',Xrand,Y,Xsig);

    % Take (natural) log_e of Ptrue and Punkn arrays
    % Get LnLikelihood for LnPtrue and LnPunkn:
    LnLikeXtrue(j) = 0.0;
    LnLikeXunkn(j) = 0.0;
    for i = 1:Nevts;
        LnPtrue(i) = log(Ptrue(i));
        LnPunkn(i) = log(Punkn(i));

        LnLikeXtrue(j) = LnLikeXtrue(j) + LnPtrue(i);
        LnLikeXunkn(j) = LnLikeXunkn(j) + LnPunkn(i);
    end;
    Y = Y + dY;
end;

% Plot the LnLikeXtrue(Xp) and LnLikeXunkn(Xp) vs. Xp data
% a.) *apriori   known* true sigma with red  line:
% b.) *apriori unknown* true sigma with blue line:
% type "help figure <enter>" and/or "help plot <enter>"
% in the MatLab command window for more details.
figure(07);
plot(Xp,LnLikeXtrue,'r-',Xp,LnLikeXunkn,'b-');
grid on;
xlabel('x');
ylabel('LnLike(x)');
title('Gaussian Distribution: LnLike(x) vs. x')

hold on;

% Find maxima of the LnLikeXtrue(Xp) and LnLikeXunkn(Xp) data, plot it:
% a.) *apriori   known* true sigma with red  circle:
% b.) *apriori unknown* true sigma with blue  star :
Xtrue_max = fpeak_max(Xp,LnLikeXtrue,20,[Ylo,Yhi,-inf,inf]);
plot(Xtrue_max(:,1),Xtrue_max(:,2),'ro');

Xunkn_max = fpeak_max(Xp,LnLikeXtrue,20,[Ylo,Yhi,-inf,inf]);
plot(Xunkn_max(:,1),Xunkn_max(:,2),'b*');

hold off;

format long g;

% print out the maxima of the LnLikeXtrue(Xp) and LnLikeXunkn(Xp) data:
fprintf(' \n');
disp('Maxima of LnLikeXtrue(Xp):');
disp(Xtrue_max);
fprintf(' \n');
disp('Maxima of LnLikeXunkn(Xp):');
disp(Xunkn_max);

% Now find delta_LnLike(Y) = 1/2 Y-points:
Xtolerance = 0.007;

fprintf(' \n');
fprintf('X-tolerance = %f \n',Xtolerance);

fprintf(' \n');
fprintf(' i, Xp(i), Xmax, LnLikeXmax, LnLikeX(i), dLnLikeX(i),
Delta_LnLikeX(i) \n');
for i = 1:NySteps;
    Delta_LnLikeXtrue = abs(Xtrue_max(1,2) - LnLikeXtrue(i) - (1.0/2.0));
         dLnLikeXtrue =    (Xtrue_max(1,2) - LnLikeXtrue(i));
    if (Delta_LnLikeXtrue < Xtolerance)
        fprintf('%i %f %f %f %f %f %f
\n',i,Xp(i),Xtrue_max(1,1),Xtrue_max(1,2),LnLikeXtrue(i),dLnLikeXtrue,Delta_L
nLikeXtrue);
    end;
end;

fprintf(' \n');
for i = 1:NySteps;
    Delta_LnLikeXunkn = abs(Xunkn_max(1,2) - LnLikeXunkn(i) - (1.0/2.0));
         dLnLikeXunkn =    (Xunkn_max(1,2) - LnLikeXunkn(i));
    if (Delta_LnLikeXunkn < Xtolerance)
        fprintf('%i %f %f %f %f %f %f
\n',i,Xp(i),Xunkn_max(1,1),Xunkn_max(1,2),LnLikeXunkn(i),dLnLikeXunkn,Delta_L
nLikeXunkn);
    end
end;

% Calculate LnLike(Z) vs. Z to find sigma and sigma/setting error on sigma:
% a.) based on *apriori   known* true mean
% b.) based on *apriori unknown* true mean
%
% n.b. from a computational perspective:
%      it is numerically *far more accurate* to
%      take the logs of individual P(i)'s and sum up lnP(i)'s than to
%      take the product of all P(i)'s and then take ln{Product P(i)'s}
%      because the ln{Product P(i)'s} is an *astronomically*
%      small # for a large # of MC events!!!
Zhi     = 15.00;
Zlo     =  5.00;
NzSteps = 10000;
 dZ     =  (Zhi-Zlo)/NzSteps;
  Z     =   Zlo;
for j = 1:NzSteps;
    Sp(j) = Z;

    Ptrue = pdf('norm',Xrand,Xtru,Z);
    Punkn = pdf('norm',Xrand,Xavg,Z);

    % Take (natural) log_e of Ptrue and Punkn arrays
    % Get LnLikelihood for LnPtrue and LnPunkn:
    LnLikeStrue(j) = 0.0;
    LnLikeSunkn(j) = 0.0;
    for i = 1:Nevts;
        LnPtrue(i) = log(Ptrue(i));
        LnPunkn(i) = log(Punkn(i));

        LnLikeStrue(j) = LnLikeStrue(j) + LnPtrue(i);
        LnLikeSunkn(j) = LnLikeSunkn(j) + LnPunkn(i);
    end;
    Z = Z + dZ;
end;

% Plot the LnLikeStrue(Sp) and LnLikeSunkn(Sp) vs. Sp data
% a.) *apriori   known* true mean with red  line:
```

```
% b.) *apriori unknown* true mean with blue line:
% type "help figure <enter>" and/or "help plot <enter>"
% in the MatLab command window for more details.
figure(08);
plot(Sp,LnLikeStrue,'r-',Sp,LnLikeSunkn,'b-');
grid on;
xlabel('s');
ylabel('LnLike(s)');
title('Gaussian Distribution: LnLike(s) vs. s')

hold on;

% Find maxima of the LnLikeStrue(Sp) and LnLikeSunkn(Sp) data, plot it
% a.) *apriori   known* true mean with red  circle:
% b.) *apriori unknown* true mean with blue  star :
Strue_max = fpeak_max(Sp,LnLikeStrue,20,[Zlo,Zhi,-inf,inf]);
plot(Strue_max(:,1),Strue_max(:,2),'ro');

Sunkn_max = fpeak_max(Sp,LnLikeSunkn,20,[Zlo,Zhi,-inf,inf]);
plot(Sunkn_max(:,1),Sunkn_max(:,2),'b*');

hold off;

format long g;

% print out the maxima of the LnLikeStrue(Sp) and LnLikeSunkn(Sp) data
fprintf(' \n');
disp('Maxima of LnLikeStrue:');
disp(Strue_max);
fprintf(' \n');
disp('Maxima of LnLikeSunkn:');
disp(Sunkn_max);

% Now find delta_LnLike(Z) = 1/2 Z-points:
Stolerance = 0.007;

fprintf(' \n');
fprintf('S-tolerance = %f \n',Stolerance);

fprintf(' \n');
fprintf(' i, Sp(i), Smax, LnLikeSmax, LnLikeS(i), dLnLikeS(i),
Delta_LnLikeS(i) \n');
for i = 1:NzSteps;
    Delta_LnLikeStrue = abs(Strue_max(1,2) - LnLikeStrue(i) - (1.0/2.0));
         dLnLikeStrue =    (Strue_max(1,2) - LnLikeStrue(i));
    if (Delta_LnLikeStrue < Stolerance)
        fprintf('%i %f %f %f %f %f %f
\n',i,Sp(i),Strue_max(1,1),Strue_max(1,2),LnLikeStrue(i),dLnLikeStrue,Delta_L
nLikeStrue);
    end
end;

fprintf(' \n');
for i = 1:NzSteps;
    Delta_LnLikeSunkn = abs(Sunkn_max(1,2) - LnLikeSunkn(i) - (1.0/2.0));
         dLnLikeSunkn =    (Sunkn_max(1,2) - LnLikeSunkn(i));
    if (Delta_LnLikeSunkn < Stolerance)
        fprintf('%i %f %f %f %f %f %f
\n',i,Sp(i),Sunkn_max(1,1),Sunkn_max(1,2),LnLikeSunkn(i),dLnLikeSunkn,Delta_L
nLikeSunkn);
```

```
    end;
end;

%==========================================================================
fprintf('\n MLM_Gaussian_1Param_Fits completed !!! \n')
%==========================================================================
```

MATLAB output:

```
# MC Events = 10000

MC Gaussian Distribution

True Mean                        = 50.000000
True Variance                    = 100.000000
True Sigma                       = 10.000000
True Setting Error on True Mean  = 0.100000
True Setting Error on True Sigma = 0.070711

<x>        = 50.010574
Var(x)     = 100.227989
Sig-x      = 10.011393
Set-x      = 0.100114
Set-Sig-x  = 0.070791

 Please be patient/relax - this computation takes a while...

Maxima of LnLikeXtrue(Xp):
          50.0109999999883         -37226.1345737854

Maxima of LnLikeXunkn(Xp):
          50.0109999999883         -37226.1345737854

 X-tolerance = 0.007000

 i, Xp(i), Xmax, LnLikeXmax, LnLikeX(i), dLnLikeX(i), Delta_LnLikeX(i)
4912 49.911000 50.011000 -37226.134574 -37226.630318 0.495744 0.004256
5112 50.111000 50.011000 -37226.134574 -37226.638830 0.504256 0.004256

4910 49.909000 50.011000 -37226.134574 -37226.637424 0.502850 0.002850
5113 50.112000 50.011000 -37226.134574 -37226.635917 0.501343 0.001343

Maxima of LnLikeStrue:
          10.0109999999999         -37226.1282844358

Maxima of LnLikeSunkn:
          10.0109999999999         -37226.1227058153

 S-tolerance = 0.007000

 i, Sp(i), Smax, LnLikeSmax, LnLikeS(i), dLnLikeS(i), Delta_LnLikeS(i)
4942  9.941000 10.011000 -37226.128284 -37226.621520 0.493235 0.006765
5083 10.082000 10.011000 -37226.128284 -37226.626819 0.498534 0.001466

4942  9.941000 10.011000 -37226.122706 -37226.615862 0.493156 0.006844
5083 10.082000 10.011000 -37226.122706 -37226.621318 0.498613 0.001387

 MLM_Gaussian_1Param_Fits completed !!!
```

Thus, the $\pm 1\sigma_x$ values for the ***mean*** of the Gaussian are:

For *apriori* ***known*** sigma:

$$\sigma_x^+ = x_p^+ - \langle x \rangle = \mathbf{50.111000} - \mathbf{50.010574} \simeq 0.10$$

$$\sigma_x^- = \langle x \rangle - x_p^- = \mathbf{50.010574} - \mathbf{49.911000} \simeq 0.10$$

For *apriori* ***unknown*** sigma:

$$\sigma_x^+ = x_p^+ - \langle x \rangle = \mathbf{50.112000} - \mathbf{50.010574} \simeq 0.11$$

$$\sigma_x^- = \langle x \rangle - x_p^- = \mathbf{50.010574} - \mathbf{49.909000} \simeq 0.11$$

Thus, the $\pm 1\sigma_s$ values for the ***sigma*** of the Gaussian are:

For *apriori* ***known*** mean:

$$\sigma_s^+ = s_p^+ - \langle s \rangle = \mathbf{10.082000} - \mathbf{10.011393} \simeq 0.07$$

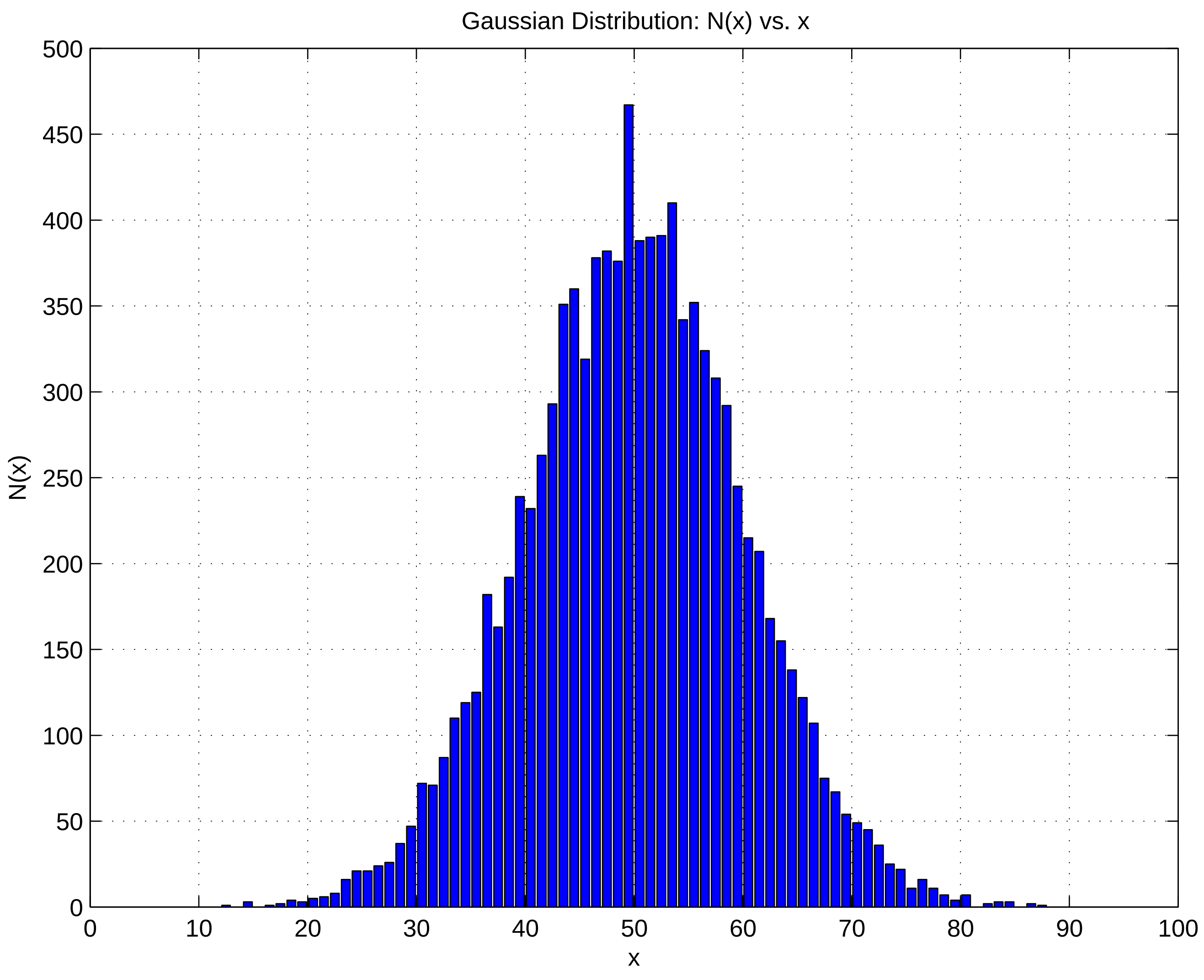
$$\sigma_s^- = \langle s \rangle - s_p^- = \mathbf{10.011393} - \mathbf{9.941000} \simeq 0.07$$

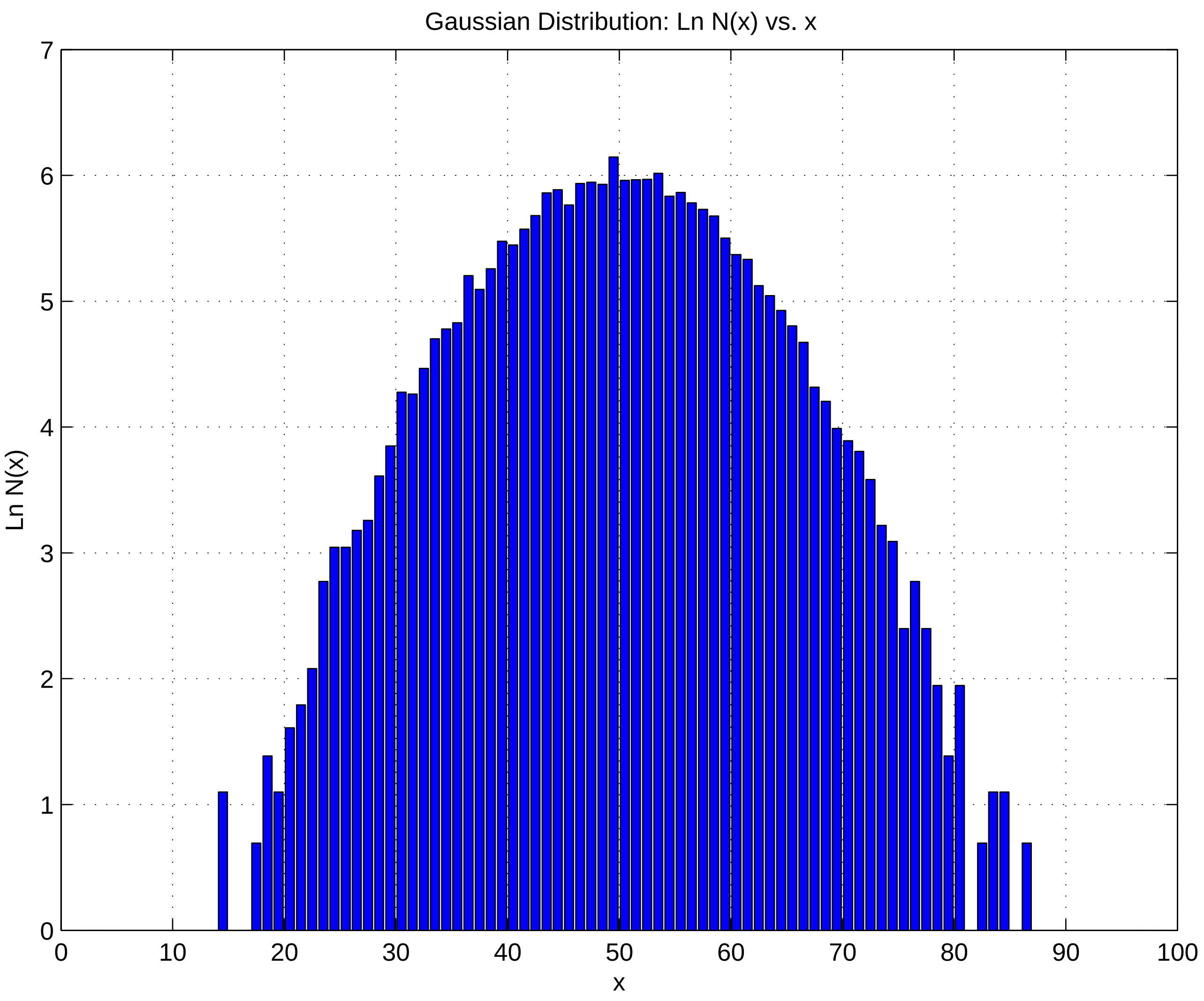
For *apriori* ***unknown*** mean:

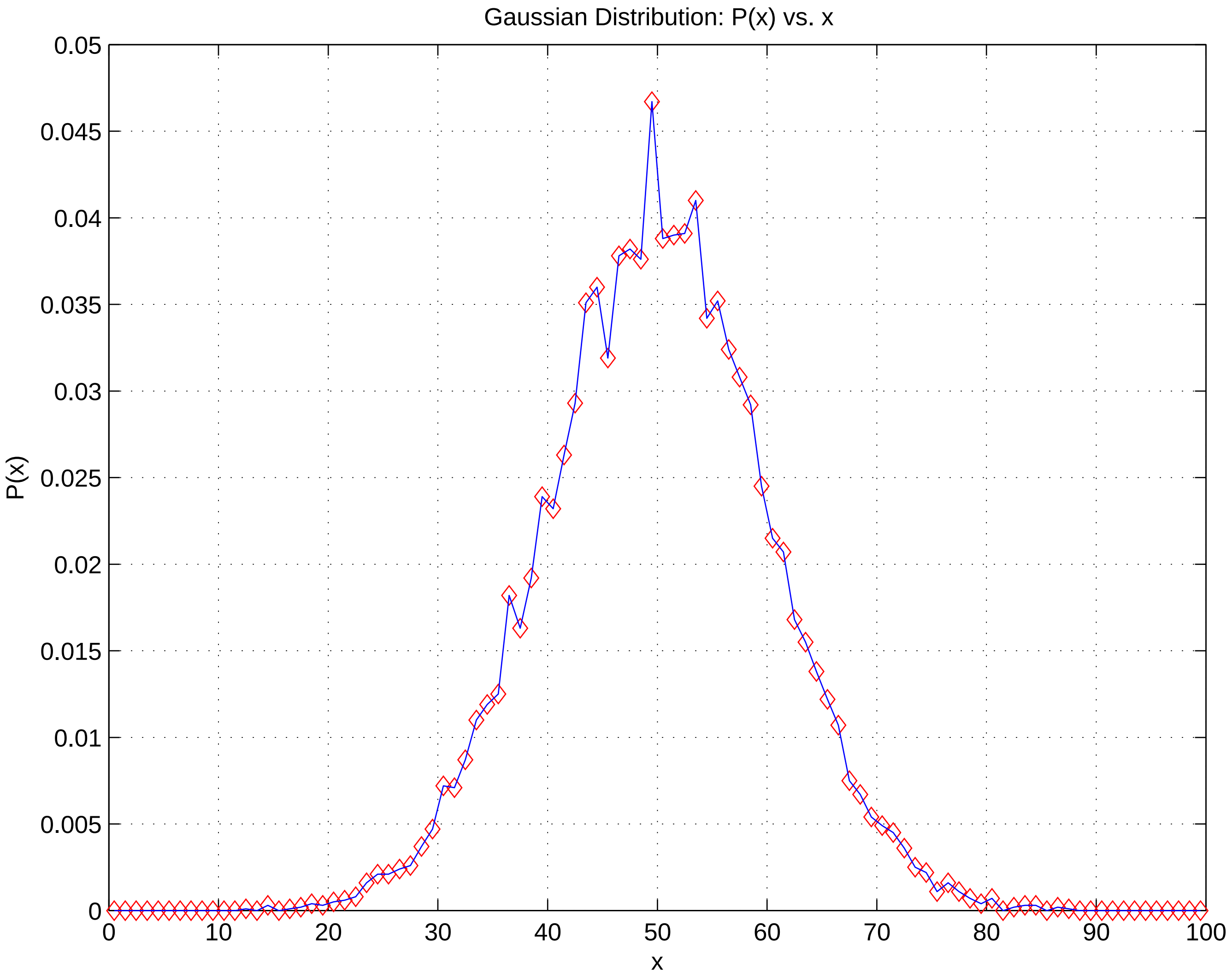
$$\sigma_s^+ = s_p^+ - \langle s \rangle = \mathbf{10.082000} - \mathbf{10.011393} \simeq 0.07$$

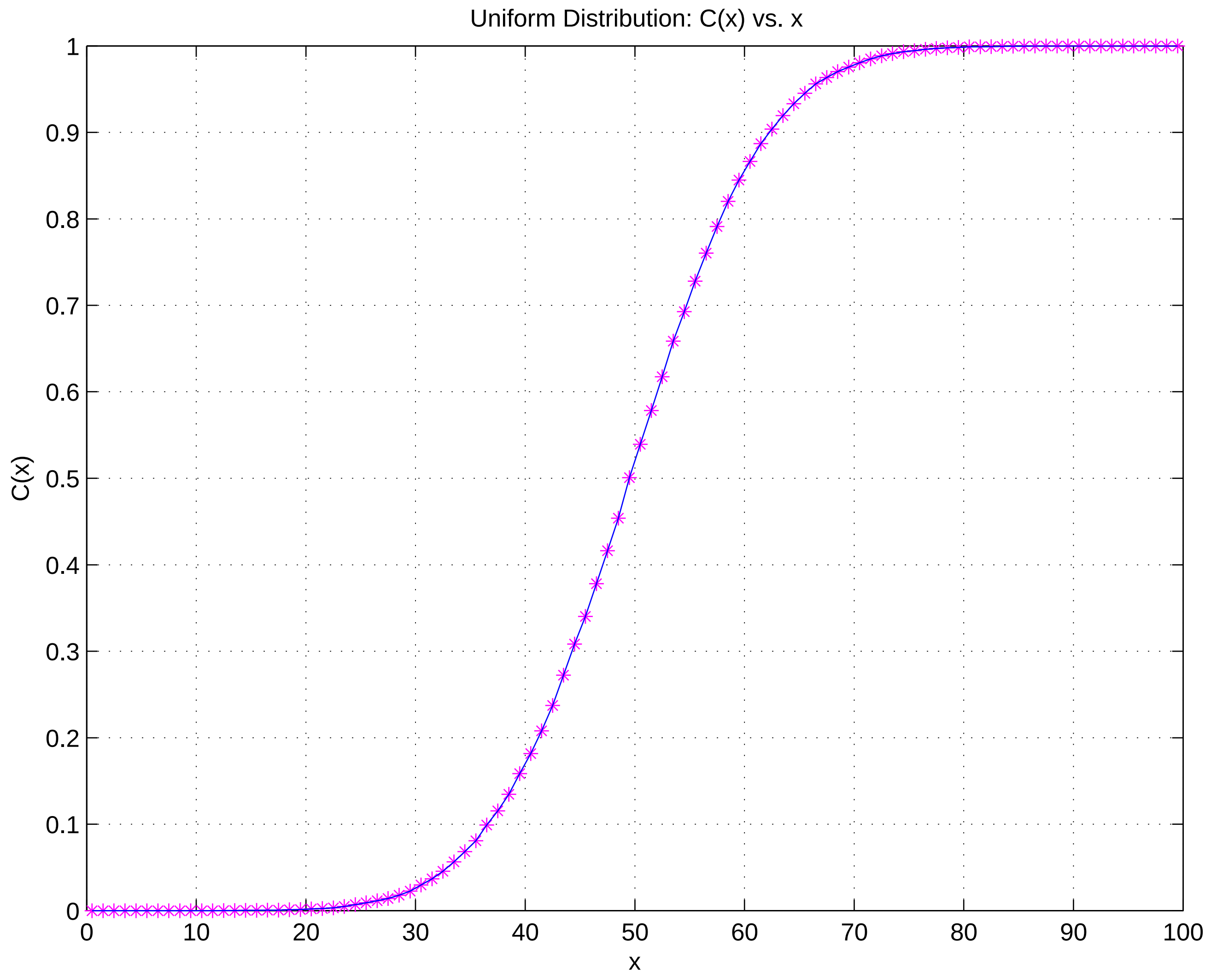
$$\sigma_s^- = \langle s \rangle - s_p^- = \mathbf{10.011393} - \mathbf{9.941000} \simeq 0.07$$

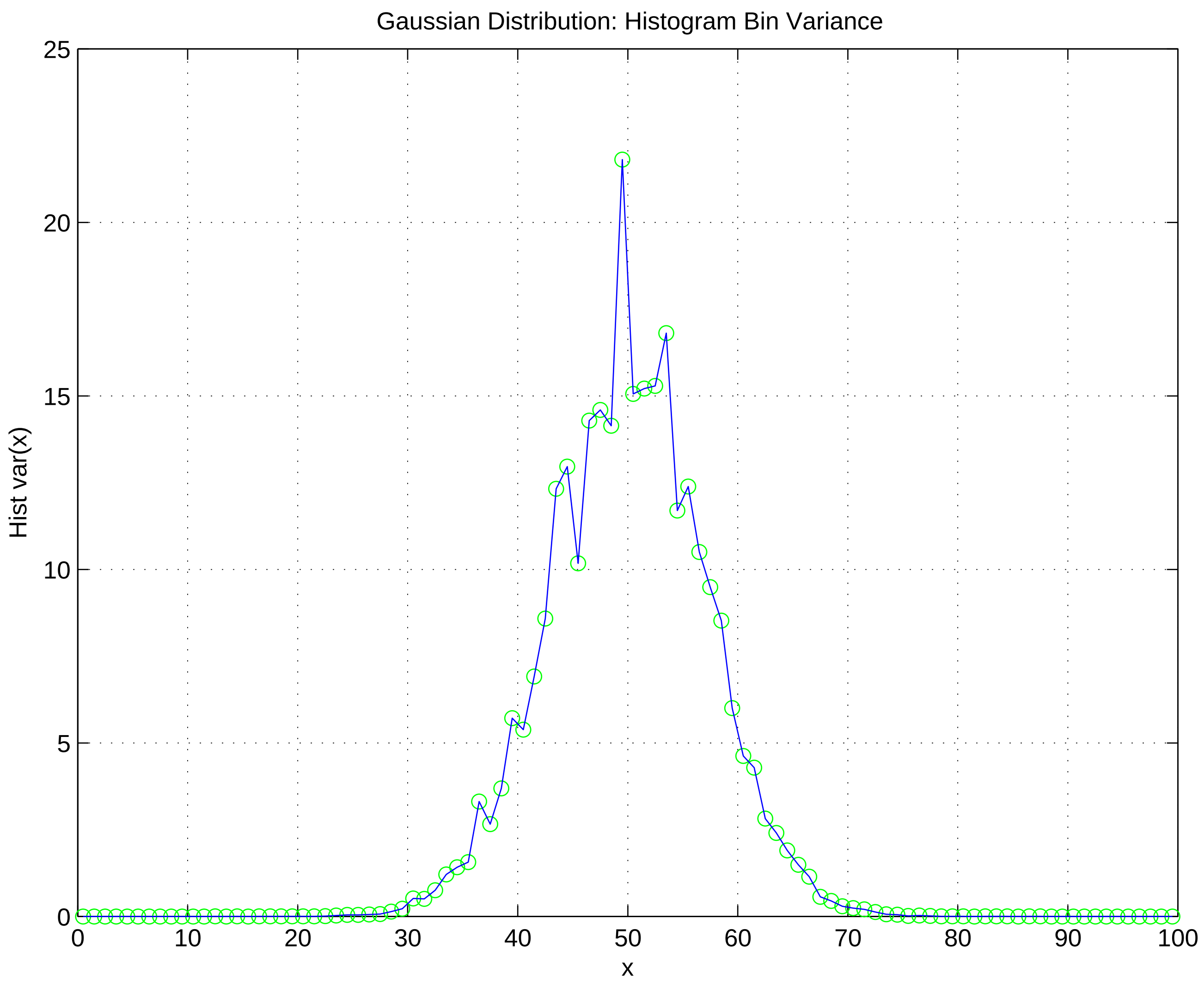
MATLAB plots:


Gaussian Distribution: N(x) vs. x
N(x)
x
0
50
100
150
200
250
300
350
400
450
500
0
10
20
30
40
50
60
70
80
90
100

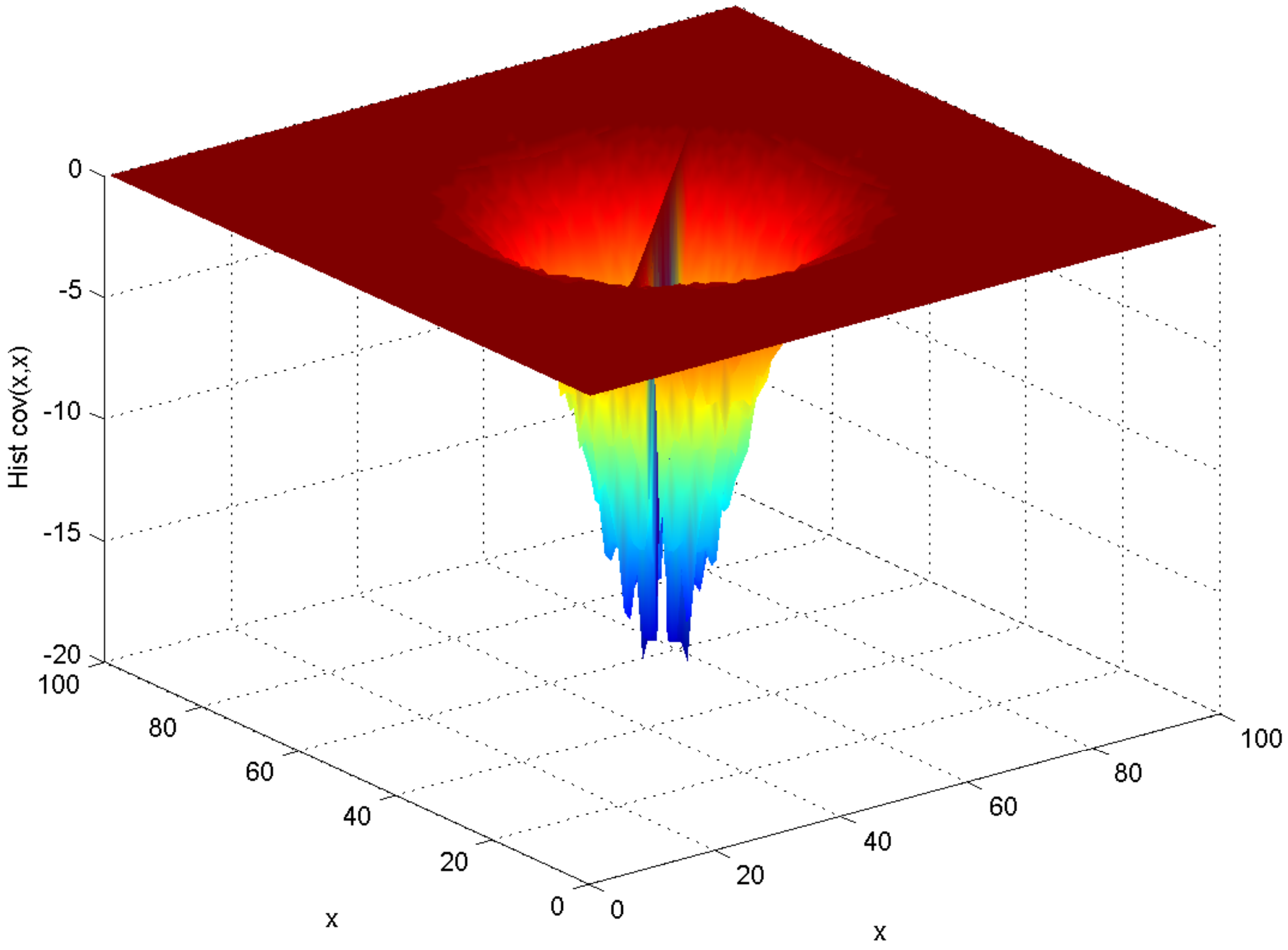

Gaussian Distribution: Ln N(x) vs. x
Ln N(x)
x
0
1
2
3
4
5
6
7
0
10
20
30
40
50
60
70
80
90
100

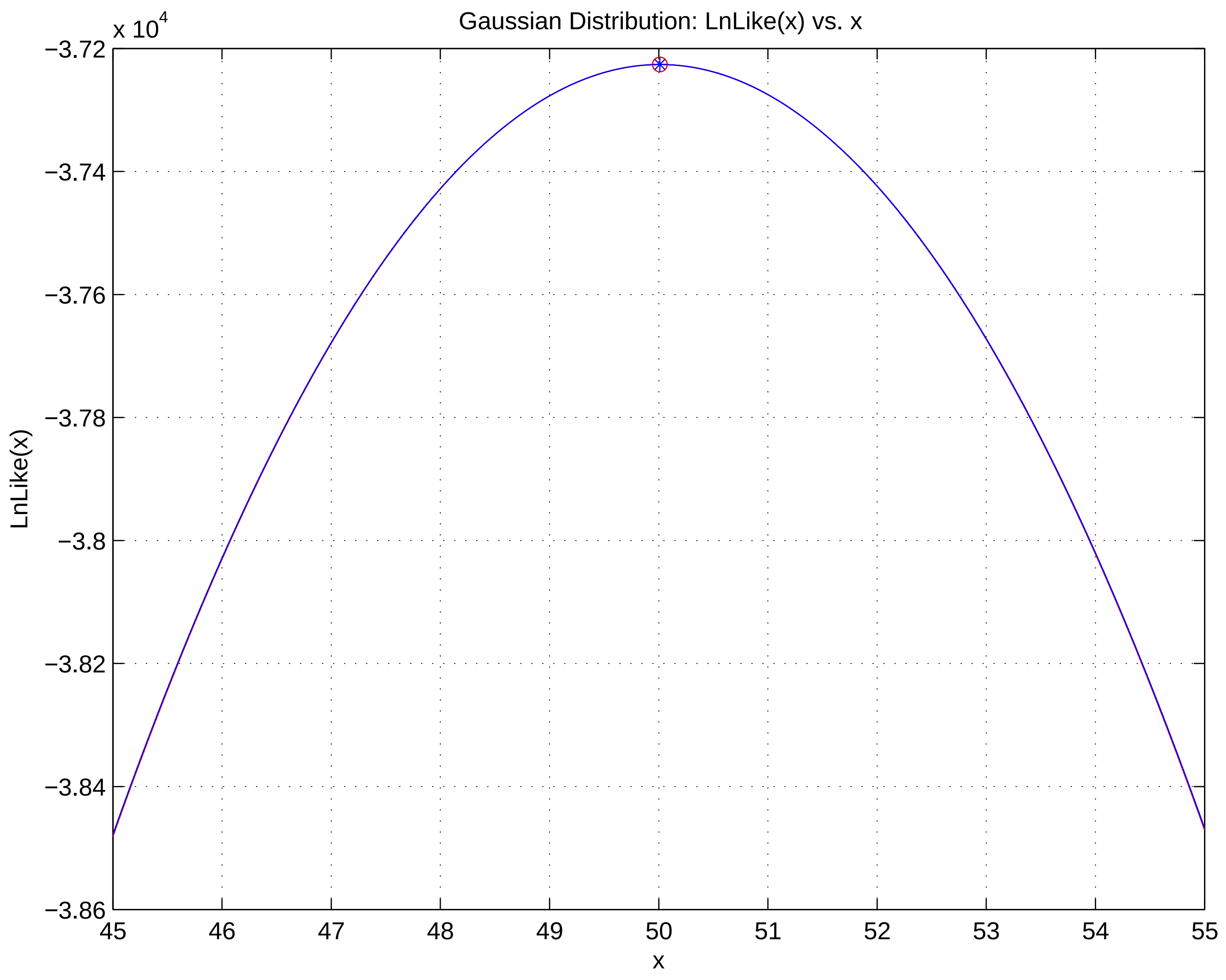

Gaussian Distribution: P(x) vs. x
P(x)
x
0.05
0.045
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
0
10
20
30
40
50
60
70
80
90
100


Uniform Distribution: C(x) vs. x
C(x)
x
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
10
20
30
40
50
60
70
80
90
100

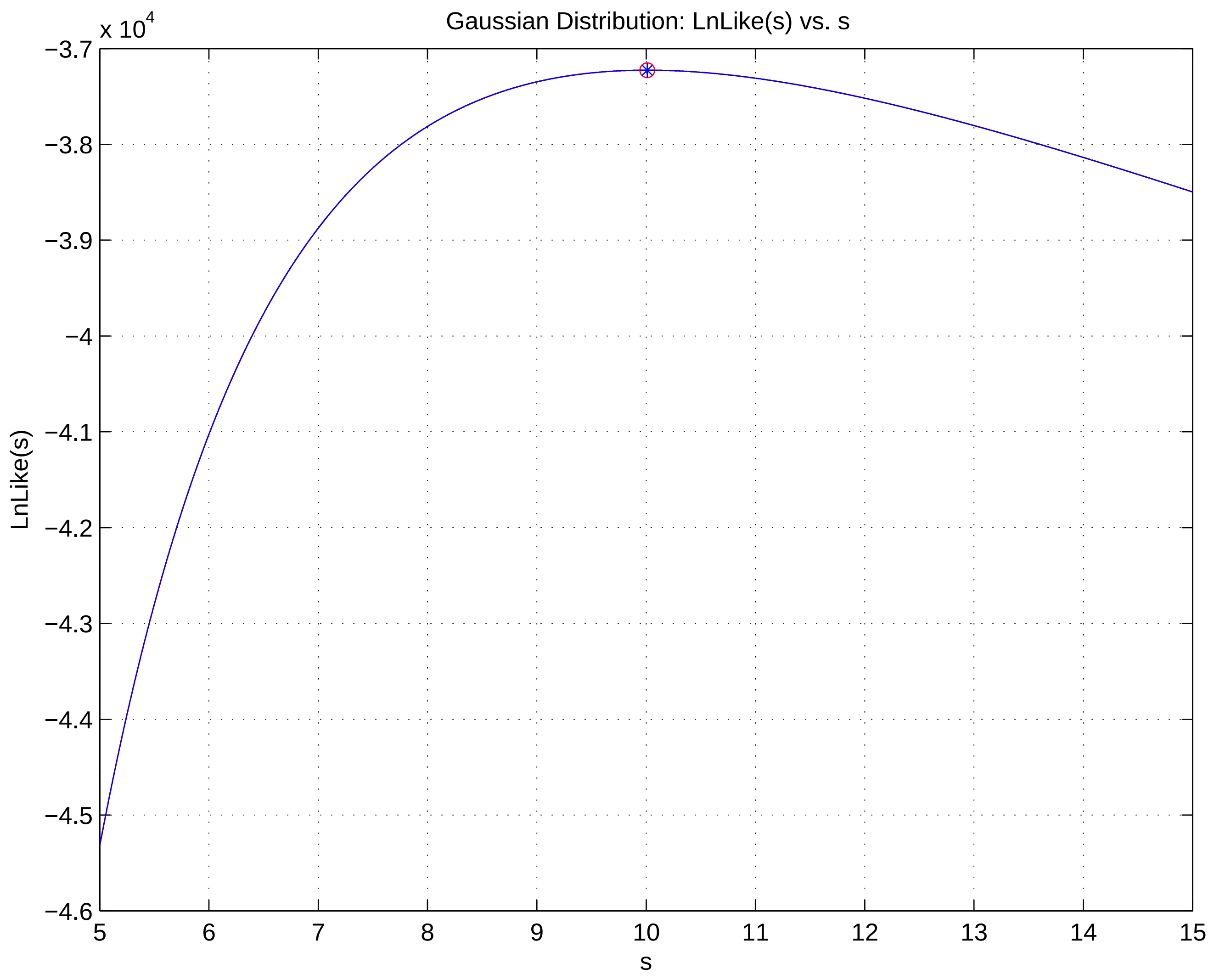

Gaussian Distribution: Histogram Bin Variance
Hist var(x)
x
0
5
10
15
20
25
0
10
20
30
40
50
60
70
80
90
100

Gaussian Distribution: Histogram Bin Covariance(x,x)





Gaussian Distribution: LnLike(x) vs. x
x 10^4
−3.72
−3.74
−3.76
−3.78
−3.8
−3.82
−3.84
−3.86
LnLike(x)
45
46
47
48
49
50
51
52
53
54
55
x



Gaussian Distribution: LnLike(s) vs. s
x 10^4
−3.7
−3.8
−3.9
−4
−4.1
−4.2
−4.3
−4.4
−4.5
−4.6
LnLike(s)
5
6
7
8
9
10
11
12
13
14
15
s