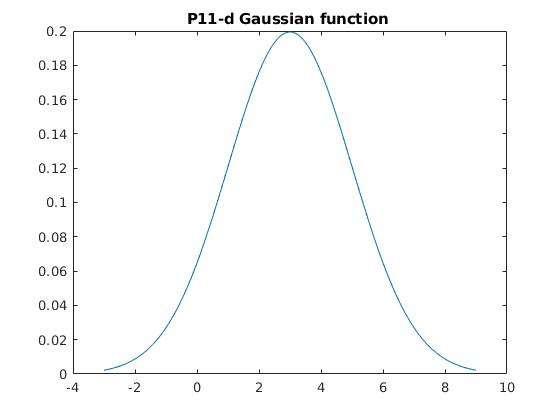
圖形識別作業一

學號：0416302

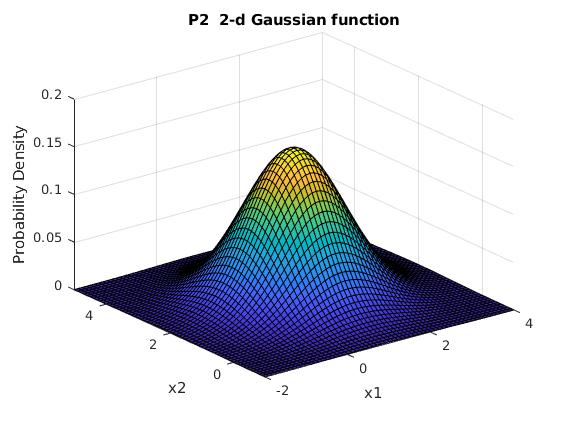
姓名：戴宏周

## P1:Given 1-d Gaussian function, draw it

透過normpdf函式(給定mu與sigma)取得-3到9間每0.1取樣一次之值。最後透過plot函式將其繪製出。

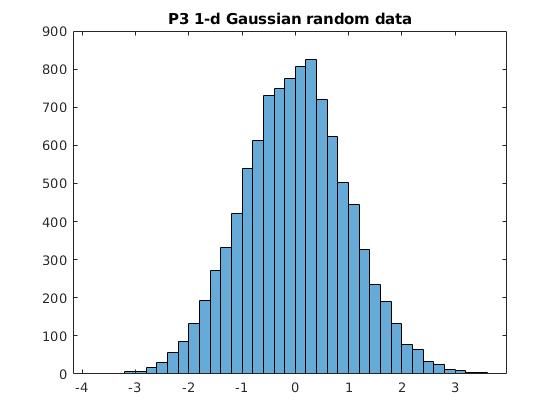
## P2:Given 2-d Gaussian function, draw it

先給定x,y範圍，之後給定covariance matrix，透過meshgrid函式產生mesh，最後透過mvnpdf產生2-d Guassian distribution，並透過surf函式繪製面。



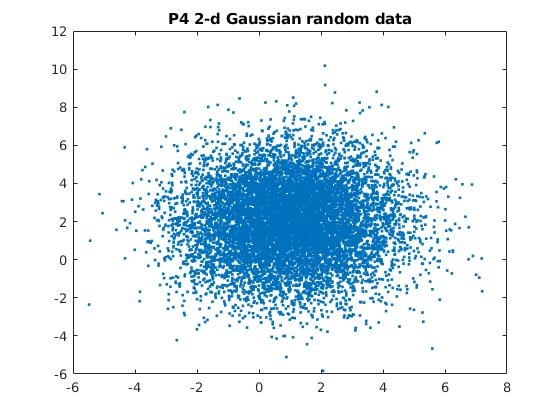
## P3:Call 1-d Gaussian random data and plot 1-d histogram.

使用normrnd函式產生10000點，最後用histogram繪製。

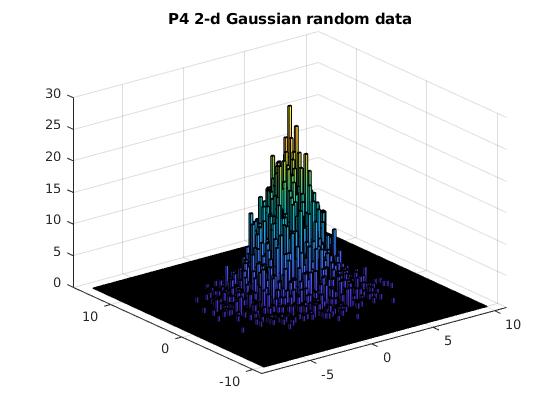


## P4:Call 2-d Gaussian random data. 點在 2-d space 上。 Plot 2-d histogram.

透過mvnpdf產生2D Guassian random data，最後透過plot以'.'繪製於平面。

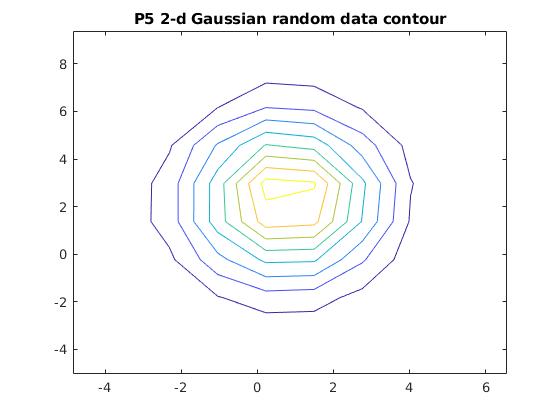


使用前面之2d guassian random data，透過hist3函式繪製。



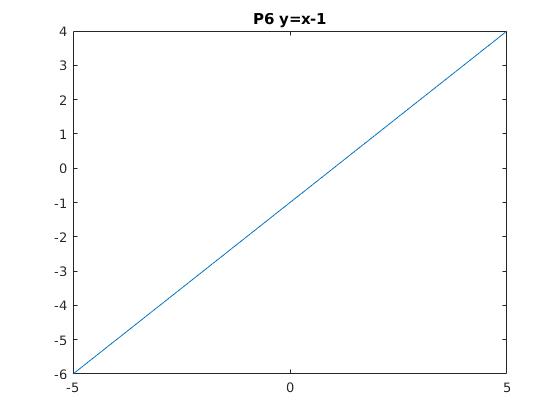
## P5:承P4,在 2-d histogram 上畫等高線圖。

將P4資料透過contour是繪製。



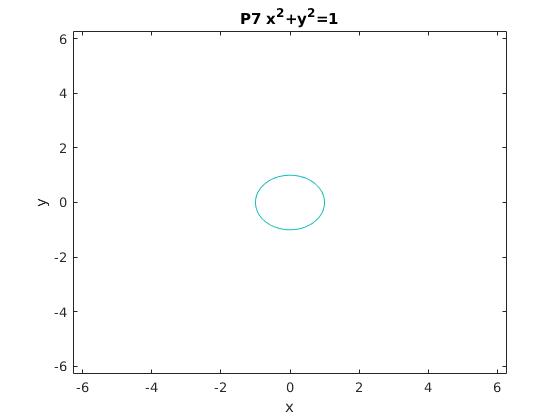
## P6:Plot line x-y=1.

使用fplot函式繪製。



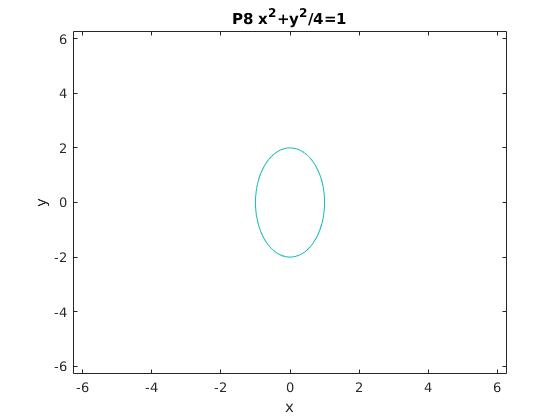
## P7:Plot circle x^2 +y^2 =1.

使用explot函式直接鍵入此方程式。由於x軸ticks比較長，所以圖像略呈橢圓。



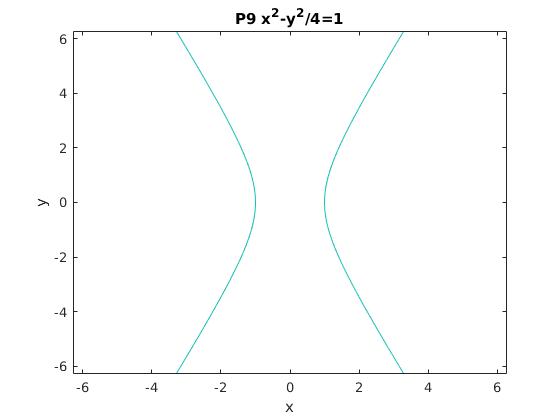
## P8:Plot ellipse x 2 +y 2 /4 =1.

同樣採用explot函數繪製。



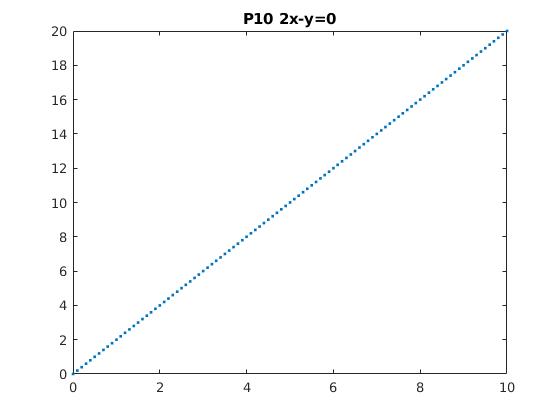
## P9:Plot hyperbola x 2 -y 2 /4 =1.

使用ezplot函數繪製。



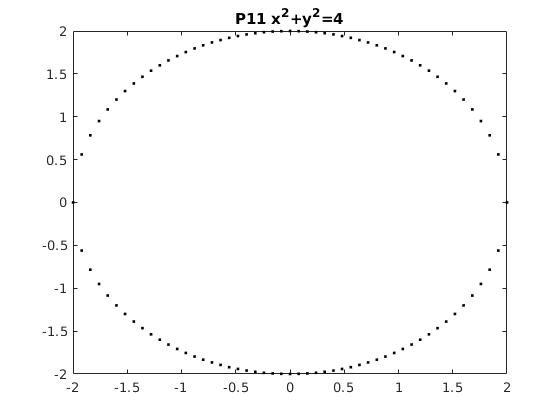
## P10:Generate data of line 2x-y=0 and plot. (產生 100 點,畫在 2-d 上)

先產生x軸0到10之間以0.1為級距，共100點。計算y = 2\*x。最後繪製x與y於平面。



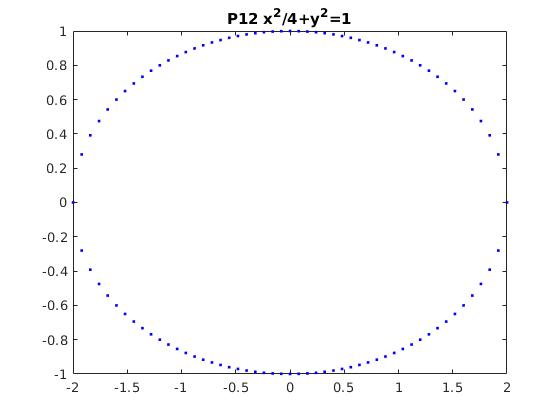
## P11:Generate data of circle x 2 +y 2 =4 and plot. (產生 100 點,畫在 2-d 上)

先產生-2到2間以0.08為單位取得50個x座標。接著分別計算y = + sqrt(4 - x^2)以及y = sqrt(4 - x^2)，最後繪製到平面上。



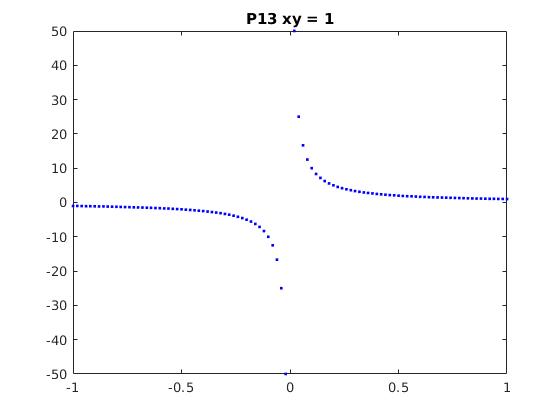
## P12:Generate data of ellipse x 2 /4+y 2 =1 and plot. (產生 100 點,畫在 2-d 上)

先產生-2到2間以0.08為單位取得50個x座標。接著先繪製上半橢圓透過plot函式。使用matlab hold on功能繼續繪製下半圓。



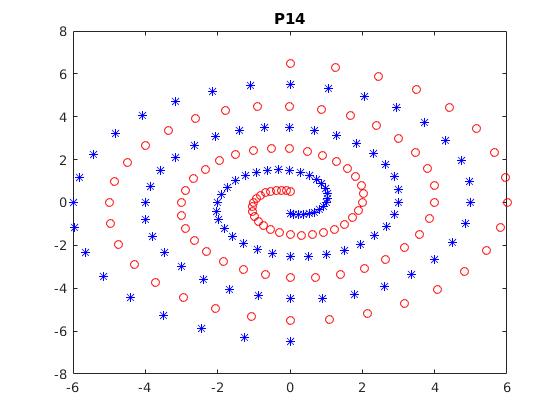
## P13:Generate data of hyperbola xy=1 and plot. (產生 100 點,畫在 2-d 上)

先產生-1到1間以0.02為單位取得100個x座標。透過y = 1/x取得y。最後繪製。由於是以x座標以固定間隔取點，所以圖形不像作業說明中在x接近0的部份這麼密集。



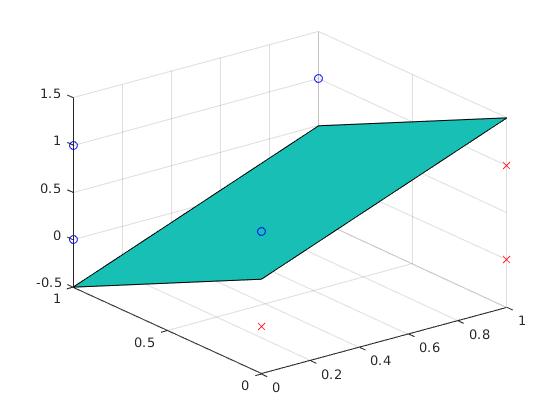
## P14:Plot two spirals.

先產生i從1到96之間以1為間隔共96點。接下來產生thetai = i\*pi/16共96組，產生ri = 6.5\*(104-i)/104共96組。透過plot函式分別繪製ri\* sin(thetai),ri \* cos(thetai)，以及 - ri\* sin(thetai), - ri \* cos(thetai)點。並分別以紅色圓以及藍色x繪製。



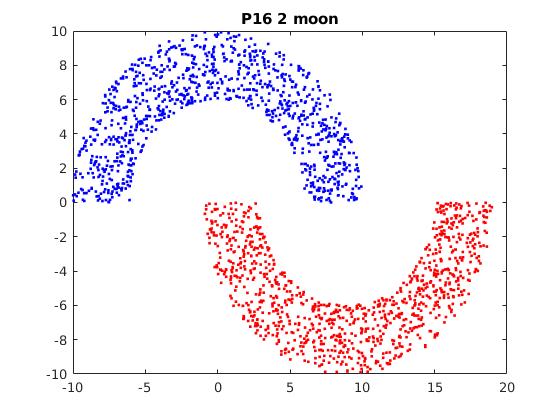
## P15:Plot 8 points in the three dimensional space, and the plane z = x – y + 0.5 that can separate these two classes.

先繪製8個指定點並依照class繪製不同顏色以及圖例。最後透過meshgrid產生x與y於0~1的mesh，最後將產生之X與Y帶入X = X -Y + 0.5，最後透過surf繪製平面。



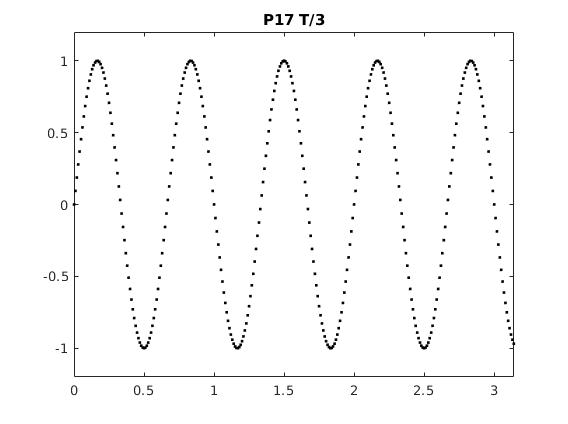
## P16:Plot double moon problem

產生兩個半圓一個半徑較大，另一個較小，但是均共用相同圓心與使用相同圓周角。隨機產生半徑r介於兩半圓且角度介於圓周角內之角度theta，最後透過x = r \* sin(theta) + center\_x、y = r \* cos(theta) + center\_y之點(由於theta取樣時的問題所以x用sin，y用cos)。接下來繪製月亮中的點。同樣的產生兩半圓，圓心與第一組不同，圓周角與第一組互補。產生下半月亮。最後會至於相同平面。可以透過圓心調整彼此位置。



## P17:Given 5 sine function with different periods (different frequencies)

產生x介於0到pi之間以0.01為間距。帶入y = sin (x\*pi\*i)，i為週期。最後繪於圖上，透過pause函式達成每5秒顯示一張。如下所示為其中一組。



## P18: uniform random number generator

(1)random seed:

使用datetime轉為datenum，並加上cputime\*1000作為seed

(2)演算法：

使用xorshift+演算法，由於xorshift屬於LFSR(Linear feedback shift register)這類邏輯設計的延伸，特色是快速，而且程式碼簡潔。不過傳統xorshift無法通過統計分析，所以有人發明xorshift+演算法，讓期可以通過主流benchmark。由於matlab的特性為Saturate on overflow，而這個演算法原為128bit，因此需要人工判斷其是否overflow，以達成亂數。

s1 = uint64(datenum(datetime)+cputime\*1000)

s2 = uint64(datenum(datetime)+cputime\*1000)

c = []

for i = 1:100

x = uint64(s1);

y = uint64(s2);

s1 = y;

x = bitxor(x,bitshift(x,32,'uint64'),'uint64');

s2 = bitxor(bitxor(bitxor(x,y,'uint64'),bitshift(x,-17,'uint64'),'uint64'),bitshift(y,-26,'uint64'),'uint64');

ans = double(s2) + double(y);

if (ans > 2^64-1)

ans = ans - 2^64+1;

end

bot = double(2^64-1);

final = double(ans/bot);

c = cat(1,c,final);

end

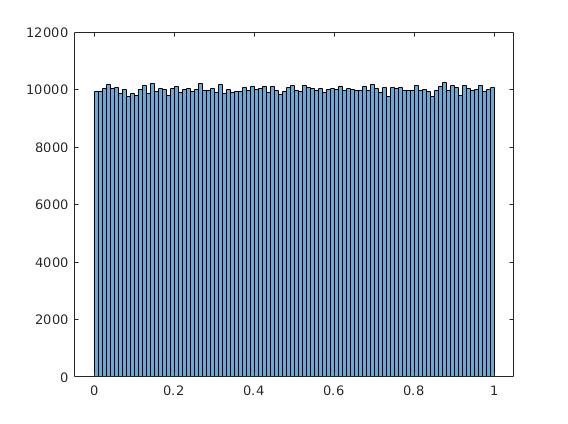
(3)Extend to the values between a lower bound and an upper bound

將算出來的uniform random number (r) 乘上abs(upper\_bound - lower\_bound)，最後加上lower\_bound。公式如下：

extend\_r = (upper\_bound - lower\_bound) \* r + lower\_bound;

(4)評比

A.使用該演算法產生1000,000筆資料，並分成100等份畫成histogram，如下可以發現其十分均勻。



B. 使用卡方分佈(chi-square distribution)進行驗證，並使用虛無假設校驗(null hypothesis)，驗證，使用以下程式：

a=0; % lower boundary

b=1; % higher boundary

nbins = 10; % number of bin

edges = linspace(a,b,nbins+1); % edges of the bins

E = N/nbins\*ones(nbins,1); % expected value (equal for uniform dist)

[h,p,stats] = chi2gof(c,'Expected',E,'Edges',edges) %chi-square distribution goodness-of-fit test

得出：

h = 0

p = 0.6993

h為0代表通過測驗，此random number generator符合uniform distribution，信任區間95%

## P19:Design a Gaussian (normal) random number generator.

1. Seed  
   Seed採用datetime與cputime的加總。如下：uint64(datenum(datetime)+cputime\*1000)
2. 演算法  
   由於P18已經取得uniform random number generator，因此這裡透過Box-muller轉換法將產生之變數帶入，取得N(m,)之數。

Box-Muller基於對Guassian Distribution進行極座標變換而來的。原始的高斯機率分佈為:

p(X) = 與p(Y) =

其joint pdf為：

p(X,Y) =

將X與Y進行極座標變換得到：

X = Rcos(theta) 、Y= Rsin(theta)

由於機率和為一：

，其中Jacobian為  
J = = r

P(R) = = 1 -

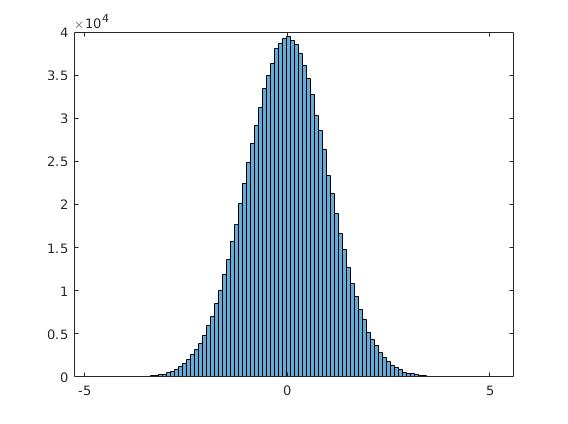
P() = =

由於為uniform distribution介於0~2，r為uniform distribution介於0~1，解

U = 1 - 反函數為R = ，因此X= rcos=cos，又為了伸縮uniform distribution 0~1到0~2，因此乘上2。

1. 分析

產生1000,000筆資料進行作圖結果如下：



另外透過卡方分佈(chi-square distribution)進行驗證，並使用虛無假設校驗(null hypothesis)，驗證，使用chi2gof函數得到h=0，證實此函數接近normal distribution

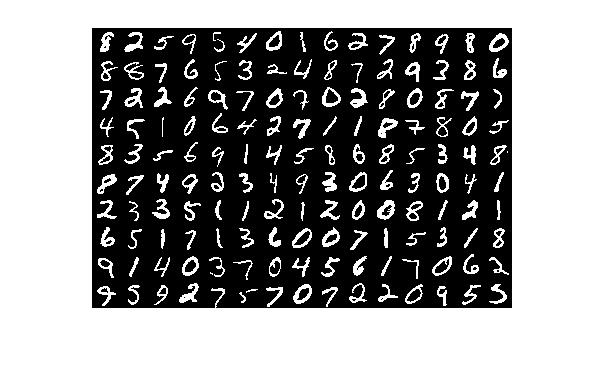
1. 更改平均與標準差

將公式修改為，透過移動平均值。

## P20:隨意讀取MNIST 150 個圖形,輸出成 15x10 的一張圖。

先透過fread讀取big endian格式的4個32bit整數，作為file magic number、相片數目、rows與cols數目。最後直接讀出剩下資料，由於直接印出來現相片打橫，因此透過permute交換row與col，並且把圖片伸縮為2維陣列，分別是image bits與number of images。並把每個pixel轉型為8bit無號數。

為了產生隨機組合，透過randi，產生相片數目之隨機不重複index，並按照10\*15格式塞入另一個image中，最後顯示image。



## 附錄：程式碼

% HW1 p1  
  
x = [-3:.1:9]; % range  
norm = normpdf(x,3,2); % range, avg, stddev  
figure;  
plot(x,norm);  
title('P11-d Gaussian function ');  
  
% HW1 p2  
figure;  
mu = [1 2];  
Sigma = [1 0; 0 1]; % covariance  
x1 = -2:.1:4; x2 = -1:.1:5; % (min, step , max)  
[X1,X2] = meshgrid(x1,x2);   
F = mvnpdf([X1(:) X2(:)],mu,Sigma);  
F = reshape(F,length(x2),length(x1));  
surf(x1,x2,F);  
caxis([min(F(:))-.5\*range(F(:)),max(F(:))]);  
axis([-2 4 -1 5 0 .2]);  
xlabel('x1'); ylabel('x2'); zlabel('Probability Density');  
title('P2 2-d Gaussian function');  
  
% HW1 p3  
rng default;  
r= normrnd(0,1,10000,1); % mu, sigma, x = 100000 y = 1 (2d array)  
figure;  
histogram(r);  
title('P3 1-d Gaussian random data');  
  
% HW1 p4  
mu = [1 2];  
sigma = [3 0; 0 4];  
rng default; % For reproducibility  
R = mvnrnd(mu,sigma,10000);  
figure;  
plot(R(:,1),R(:,2),'.');  
title('P4 2-d Gaussian random data');  
figure;  
hist3([R(:,1),R(:,2)],'CDataMode','auto','FaceColor','interp','Ctrs', {-8:0.2:10 -10:0.2:14} );  
title('P4 2-d Gaussian random data');  
[n,z] = hist3([R(:,1),R(:,2)]);  
%HW1 p5  
figure;  
contour(z{1},z{2},n);  
title('P5 2-d Gaussian random data contour');  
  
%HW1 p6  
figure;  
fplot(@(x) x-1)  
title('P6 y=x-1');  
  
%HW1 p7  
figure;  
ezplot('x^2 + y^2 - 1')  
title('P7 x^2+y^2=1');  
  
%HW1 p8  
figure;  
ezplot('x^2+y^2/4-1')  
title('P8 x^2+y^2/4=1');  
  
%HW1 p9  
figure;  
ezplot('x^2-y^2/4-1')  
title('P9 x^2-y^2/4=1');  
  
%HW1 p10  
figure;  
x=0:0.1:10;  
y = 2\*x;  
plot(x,y,'.');  
title('P10 2x-y=0')  
  
%HW1 p11  
figure;  
x = -2:0.08:2;  
py = sqrt(4-x.^2);  
ny = -1\* sqrt(4-x.^2);  
plot([x x],[py ny],'.k');  
title('P11 x^2+y^2=4');  
  
%HW1 P12  
figure;  
x = -2:0.08:2;  
plot(x,sqrt(1-x.^2/4),'b.');  
hold on;  
plot(x,-1\*sqrt(1-x.^2/4),'b.');  
hold off;  
title('P12 x^2/4+y^2=1');  
  
%HW1 P13  
figure;  
x = -1:0.02:1;  
y = 1./x;  
plot(x,y,'b.');  
title('P13 xy = 1');  
  
%HW1 P14  
figure;  
i = 0:1:96;  
theta = i.\*pi/16;  
r = 6.5\*(104-i)/104;  
plot(r.\*sin(theta),r.\*cos(theta),'ro');  
hold on;  
plot(-1\*r.\*sin(theta),-1\*r.\*cos(theta),'b\*');  
hold off;  
title('P14');  
  
%HW1 P15  
figure;  
x = [0 0 1 1 0 1 1 0];  
y = [1 0 0 0 0 1 1 1];  
z = [1 0 0 1 1 0 1 0];  
classes = [1 2 2 2 1 2 1 1];  
plot3(x(classes == 1),y(classes==1),z(classes==1),'bo');  
hold on;  
plot3(x(classes == 2),y(classes==2),z(classes==2),'rx');  
grid on;  
hold on;  
x = 0:1:1;  
y = 0:1:1;  
[X,Y] = meshgrid(x,y);  
Z = X - Y + 0.5;  
surf(X,Y,Z)  
hold off;  
  
%HW1 P16  
figure;  
b\_cen = [0,0];  
b\_l\_radius = 10;  
b\_s\_radius = 6;  
b\_theta = 1\*pi\*rand(1000,1)-0.5\*pi;  
b\_r\_radius = (b\_l\_radius - b\_s\_radius)\*rand(1000,1) + b\_s\_radius;  
b\_x = b\_r\_radius.\*sin(b\_theta)+b\_cen(1);  
b\_y = b\_r\_radius.\*cos(b\_theta)+b\_cen(2);  
plot(b\_x,b\_y,'b.')  
hold on;  
r\_cen = [9,0];  
r\_l\_radius = 10;  
r\_s\_radius = 6;  
r\_theta = 1\*pi\*rand(1000,1)+0.5\*pi;  
r\_r\_radius = (r\_l\_radius - r\_s\_radius)\*rand(1000,1) + r\_s\_radius;  
r\_x = r\_r\_radius.\*sin(r\_theta)+r\_cen(1);  
r\_y = r\_r\_radius.\*cos(r\_theta)+r\_cen(2);  
plot(r\_x,r\_y,'r.')  
hold off;  
title('P16 2 moon')  
%HW1 P17  
figure;  
x=0:0.01:pi;  
for i = 1:5  
 y\_plot=sin(x\*pi\*i);  
 plot(x,y\_plot,'.k');  
 axis([0,pi,-1.2,1.2]);  
 title(strcat('P17 T/',num2str(i)))  
 pause(5);  
end  
%HW1 P18  
figure;  
s1 = uint64(datenum(datetime)+cputime\*1000)  
s2 = uint64(datenum(datetime)+cputime\*1000)  
c = []  
lower\_bound = 0.0;  
upper\_bound = 1.0;  
for i = 1:100  
 x = uint64(s1);  
 y = uint64(s2);  
 s1 = y;  
 x = bitxor(x,bitshift(x,32,'uint64'),'uint64');  
  
 s2 = bitxor(bitxor(bitxor(x,y,'uint64'),bitshift(x,-17,'uint64'),'uint64'),bitshift(y,-26,'uint64'),'uint64');  
 ans = double(s2) + double(y);  
 if (ans > 2^64-1)  
 ans = ans - 2^64+1;  
 end  
 bot = double(2^64-1);  
 final = double(ans/bot);  
 c = cat(1,c,final);  
end  
c = c.\*(upper\_bound-lower\_bound)+lower\_bound;  
histogram(c,100);  
title('P18 uniform random ditribution')  
  
%HW1 P19  
figure;  
s1 = uint64(datenum(datetime)+cputime\*1000);  
s2 = uint64(datenum(datetime)+cputime\*1000);  
mean = 0;  
std = 1;  
normals = []  
for j = 1:10000  
 c = [];  
 for i = 1:2  
 x = uint64(s1);  
 y = uint64(s2);  
 s1 = y;  
 x = bitxor(x,bitshift(x,32,'uint64'),'uint64');  
  
 s2 = bitxor(bitxor(bitxor(x,y,'uint64'),bitshift(x,-17,'uint64'),'uint64'),bitshift(y,-26,'uint64'),'uint64');  
 ans = double(s2) + double(y);  
 if (ans > 2^64-1)  
 ans = ans - 2^64+1;  
 end  
 bot = double(2^64-1);  
 final = double(ans/bot);  
 c = cat(1,c,final);  
 end  
 n\_final = sqrt(-2\*log(c(1)))\*cos(2\*pi\*c(2))\* std + mean;  
 normals = cat(1,normals,n\_final);  
end  
histogram(normals,100);  
title('P19 normal distribution')  
h = chi2gof(normals)  
  
% HW1\_P20  
figure;  
% load data  
filename = 'train-images-idx3-ubyte'  
fp = fopen(filename, 'rb');  
   
magic = fread(fp, 1, 'int32', 0, 'ieee-be')  
numImages = fread(fp, 1, 'int32', 0, 'ieee-be')  
numRows = fread(fp, 1, 'int32', 0, 'ieee-be')  
numCols = fread(fp, 1, 'int32', 0, 'ieee-be')  
   
images = fread(fp, inf, 'unsigned char');  
images = reshape(images, numCols, numRows, numImages);  
% rotate image  
images = permute(images,[2 1 3]);  
   
fclose(fp);  
  
images = reshape(images, numCols\*numRows, numImages);  
images = uint8(images)/255 ;  
  
num=randi(length(images),10,15)  
  
for i=0:9  
 for j=0:14  
 img(numCols\*i+1:numCols\*(i+1),numRows\*j+1:numRows\*(j+1))=reshape(images(:,num(i+1,j+1)),numCols,numRows);  
 end  
end  
  
imshow(img)