//AUTHOR: DAN CRISP

//CHAPTER: 3

//PROBLEMS: 3.1, 3.5, 3.9, 3.10, 3.11, Q6, 4.3, 4.5

#Q1: 3.1 Plotting points

Write a program, Connectpoints.m, that plots the points defined bz the two arrazs x=[0,1,2,-2,-1,0] and y=[0,0,1,1,0,0]. Each point should be plotted as a star, and the points should be connected with line segments.

function ConnectPoints( x, y )

%ConnectPoints plot two sets of variables using starred points connected by solid lines.

plot(x,y,'-\*');

end

#Q2: 3.5 Plotting damped motion

Write a program, plotdamped.m that plots the following function: f(x)=exp(-x/a)cos(2\*pi()\*x/lambda)

Choose an appropriate domain for the function and vary the parameters a and lambda (by changing the values set in the program and rerunning).

function plotdamped( xmin, xmax, Nx, a, lambda )

%plotdamped.m Plots the function f(x)=exp(-x/a)cos(2x\*pi/lambda)

x = linspace(xmin, xmax, Nx);

f = exp(-x./a).\*cos(2\*pi()\*x./lambda);

plot(x,f);

end

The function was found to best showcase it's key attributes using the above code as so: plotdamped(-20,10,120,8,4)

lambda decreases the frequency, while 'a' inversely effected the damping term.

#Q3: 3.9 Harmonic motion plot

Write a program, plotharmonic.m, that plots harmonic motion of x in time, as described by the following equation:

x(t) = A\*cos(omega\*t + phi)

In this equation, A is the amplitude of the motion, omega is the angular frequency, and phi is the phase shift, all of which are constants in the equation. Run the program with the values:

A = 2, phi = pi/2

Pick a value for the period T that is related to omega by the relation omega = 2pi/T. Also pick a good value for the final time Tf that sets the range of t=[0:Tf].

function plotharmonic( tmin, tmax, Nt, A, omega, phi, opt)

%plotharmonic Plots the function Xt = A\*cos(omega\*t + phi);

t = linspace(tmin, tmax, Nt);

if opt=='T'

Xt = A\*cos(2\*pi\*t/omega + phi);

else

Xt = A\*cos(omega\*t + phi);

end

plot(t,Xt);

end

The plotted function's characterists were well seen when entering: plotharmonic(0,4,75,2,1,pi(),'T')

In other words, when the time domain is 4sec, there are about 75 points, an amplitude ‘A’ of 2, and the period ‘T’ is 1sec.

#Q4: 3.10 Creative plot

Draw a figure of your own choosing on a piece of graph paper and write a program, plotoriginal.m, to draw the figure. Drawing the figure should require at least four different sections of code and use the hold function to keep adding new elements to the same set of axes. The title of your plot and the figure itself should clearly communicate the image to a viewer.

% plotoriginal.m Draws an image of a toy house onto a plot. It does this

% using custom functions 'start', and 'adding'. Post editing can be done

% using custom function 'fixing'. See below for arg info on custom

% functions...

% [ thing ] = start( x0, y0)

% [ thing ] = adding( thing, xnew, ynew)

% [ thing ] = fixing( thing, xnew, ynew, ind)

figure, hold on

% Draw Structure

[ walls ] = start( 0, 0);

[ walls ] = adding( walls, 4, 0);

[ walls ] = adding( walls, 4, 5);

[ walls ] = adding( walls, -4, 5);

[ walls ] = adding( walls, -4, 0);

[ walls ] = adding( walls, 0, 0);

[ roof ] = start( walls(1,3), walls(2,3));

[ roof ] = adding( roof, walls(1,3)+1, walls(2,3));

[ roof ] = adding( roof, 0, 7);

[ roof ] = adding( roof, walls(1,4)-1, walls(2,4));

[ roof ] = adding( roof, walls(1,4), walls(2,4));

% Draw Door

[ door ] = start( 0, 0);

[ door ] = adding( door , 0, 3);

[ door ] = adding( door , 2, 3);

[ door ] = adding( door , 2, 0);

% Label Plot

title('Toy House');

xlabel('Left-Right from center (ft)');

ylabel('Height (ft)');

#Q5: 3.11 Probabilities for three dice

function [ data ] = DiceProb( dice, sides, rolls)

% DiceProb Lets user display results of a dice rolling simulation as a

% bar graph.

% Create randomized data for each roll of each dice.

data = rand(rolls, dice);

for i=1:rolls

for j=1:dice

data(i,j) = ceil(sides\*data(i,j));

if j == dice

% Put sum of dice values into a new column at the given roll

% index.

data(i,dice+1) = sum(data(i,:));

end

end

end

% Create array num for holding number of times each possible side-total was rolled

num(dice\*1:dice\*sides) = [0];

% Add each time a roll-total was rolled to the correct element of 'num'.

for i=1:rolls

rollSum = data(i,dice+1);

num(rollSum) = num(rollSum) + 1;

end

% Calculate probability from simulation

prob(:) = 100\*num(:)/rolls;

% Display bar graphs of roll total results, and probabilities.

subplot(2,1,1); bar(num);

title('Dice Simulation'); xlabel('Roll Total'); ylabel('# Times Rolled');

grid on

subplot(2,1,2); bar(prob);

xlabel('Roll Total'); ylabel('% Probability');

grid on

end

#Q6:

a: Make a histogram of normally distributed random numbers if the mean = 3, the standard deviation = 0.3, and the number of data = 5000. Write down the labels.

mean = 3; std = 0.3; N = 5000;

data = mean + std\*randn(1,N);

[numX, xs] = hist(data,100); title('Normal Distribution. Mean = 3, std = 0.3, N = 5000'); xlabel('bins'); ylabel('# values per bin');

b: Plot the normal distributed function for the histogram in part a.

X = linspace(1,5,100);

fx = max(numX)\*exp(-(X-mean).^2/(2\*stddev^2));

plot(X,fx,'k','LineWidth',2);

WHEN DOING CHAPTER 4, ALSO DO 1st 2 PROBLEMS FROM FILE 'HW3.pdf'.

#Q1: 4.3

ConcessionStand.m

% ConcessionStand.m Uses vector-matrix multiplication to tabulate and

% display the cost of a few orders.

% Clear Screen

clc;

% Declare variables

items = {'hot dog', 'brat', 'coke'};

cost = [3.00, 3.50, 2.50];

orders = [ 2, 1, 3; 1, 0, 2; 2, 2, 2; 0, 5, 1 ];

% Compute sums, totals.

total = 0;

for i=1:4

orders(i,4) = sum(orders(i,1:3)'.\*cost(:));

total = total + orders(i,4);

end

items(4) = {'total'};

Display

disp(' hotdog brat coke total')

disp(orders)

fprintf('The total of all orders is: $%3.2f\n\n',total);

#Q2: 4.5 Linear system of equations

Solve the system:

x + 2y + z = -1

x - 3y + 2z = 1

2x - 3y + z = 5

A = [1, 2, 1; 1, -3, 2; 2, -3, 1];

X = A\[-1; 1; 5];

x = 2.25

y = -0.75

z = -1.75