Tutorial Two

THREE QUESTIONS

Sultan Alatawi

Roll

University

EE-XXX: Learning MATLAB

Siraj Tayyab Khan

November 19th, 2022

~ (Qno.1) ~

Sample Problem 7-4: Average and standard deviation

Write a user-defined function that calculates the average and the standard deviation of a list of numbers. Use the function to calculate the average and the standard deviation of the following list of grades:
80 75 91 60 79 89 65 80 95 50 81

MATLAB Function Code:

```
%%%Using Nested Function Technique%%%

%Primary function name is same as the function file name
function [average StandardDeviation] = findAvgAndSTD_Nested(InputVector)

%Creating subfunctions
%Which means subfunctions cannot be called outside of the primary function
%files
%You can declare inside primary functions in any order

%Creating subfunction
%Note that Nested function internal variable is not accessible outsider
%Here AvgResult is internal variable name in function definition
    function AvgResult = findAverage(InputVector)
        AvgResult = sum(InputVector)/length(InputVector);
    end
%Creating nested function above and using another cousin nested inside this
%nested function so that we do not have to code again for finding average
    function StdResult = findSTD(InputVector)
        StdResult = (sum((InputVector-
findAverage(InputVector)).^2)/(length(InputVector)-1))^(1/2);
    end

%Now what I am actually doing in my primary function
average = findAverage(InputVector);
StandardDeviation = findSTD(InputVector);

end
```

MATLAB Script Code:

```
clc
clear all

x = [80 75 91 60 79 89 65 80 95 50 81];
%Using Nested functions technique
[avg std] = findAvgAndSTD_Nested(x)
%Using Subfuncrions
[avg std] = findAvgAndSTD_SubFunction(x)
```

~ (Qno.2) ~

Sample Problem 7-5: Exponential growth and decay

A model for exponential growth or decay of a quantity is given by

$$A(t) = A_0 e^{kt}$$

where $A(t)$ and A_0 are the quantity at time t and time 0, respectively, and k is a constant unique to the specific application.

Write a user-defined function that uses this model to predict the quantity $A(t)$ at time t from knowledge of A_0 and $A(t_1)$ at some other time t_1. For function name and arguments use `At = expGD(A0,At1,t1,t)`, where the output argument `At` corresponds to $A(t)$, and for input arguments use `A0,At1,t1,t,` corresponding to A_0, $A(t_1)$, t_1, and t, respectively.

Use the function file in the Command Window for the following two cases:

(*a*) The population of Mexico was 67 million in the year 1980 and 79 million in 1986. Estimate the population in 2000.

(*b*) The half-life of a radioactive material is 5.8 years. How much of a 7-gram sample will be left after 30 years?

MATLAB Function Code:

```
function quantityT = findPredictionAtT(quantityT0, quantityT1, deltaT, T)
%First find the value of k
%Changing population from quantity 0 to quantity 1 in deltaT time.
k = log(quantityT1/quantityT0)/deltaT;
quantityT = quantityT0 * exp(k*T);
end
```

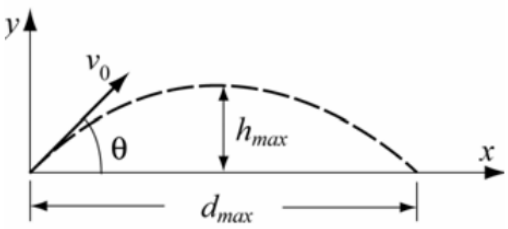
MATLAB Script Code:

```
clc
clear all
%Problem2
MexicoPopulationAt1980 = 67;
MexicoPopulationAt1986 = 79;
TimeOne = 1980;
TimeTwo = 1986;
TimeDiff = TimeTwo-TimeOne;
TimeGapOfInterest = 2000-TimeOne;
MexicoPopulationAt2000 = findPredictionAtT(MexicoPopulationAt1980,
MexicoPopulationAt1986, TimeDiff, TimeGapOfInterest);
```

~ (Qno.3) ~

Sample Problem 7-6: Motion of a projectile

Create a function file that calculates the trajectory of a projectile. The inputs to the function are the initial velocity and the angle at which the projectile is fired. The outputs from the function are the maximum height and distance. In addition, the function generates a plot of the trajectory. Use the function to calculate the trajectory of a projectile that is fired at a velocity of 230 m/s at an angle of 39° .



MATLAB Function Code:

```
function maxHeight = findMaxHeightAndPlotTrajectory(velocity, angle,
timeVector)
initialVelocityY = velocity*sin(angle);
initialVelocityX = velocity*cos(angle);
%Time is independent variable and we would differentiate or integerate with
%respect to it
syms time
%Now making an equation
velocityExpressionY = initialVelocityY - 9.8*time
velocityExpressionX = initialVelocityX + 0*(time^0)
%To get displacement equation we integerate velocity function with time
%int expression does integerate expression with respect to syms
displacementExpressionY = int(velocityExpressionY)
displacementExpressionX = initialVelocityX*time
%Now generating vectors or arrays from equation by pluging in values of
%time of interest vector to get corresponding values
vectorDisplacementX = vpa(subs(displacementExpressionX,time,timeVector))
vectorDisplacementY = vpa(subs(displacementExpressionY,time,timeVector))
%Now simply plotting those vectors
%plot(vpa(vectorDisplacementX),vpa(vectorDisplacementY))
%xlabel('Displacement in x direction')
%ylabel('Displacement in y direction')
%title('Trajectory View')

%To find max height covered we would take derivative of displacement with
%respect to time
%we know at peak height derivative gets zero
%Solve equation at derivative equal to zero would give solutions for
%possible maximas
displacementExpressionYDerivative = diff(vectorDisplacementY, time)
timeSolutionRoots = solve(displacementExpressionYDerivative,time)
%Now substituting in time solution roots in displacement equations to get
max
%displacement in y direction
MaximasAndMinimas =
vpa(subs(displacementExpressionY,time,timeSolutionRoots))
maxHeight = max(MaximasAndMinimas)
end
```

MATLAB Script Code:

```
clc
clear all

syms x

%f= x*sin(x)
%g=int(f,x)
%h=diff(f,x)

%output=vpa(subs(f,x,90))
%a = 90.5

output_test = findMaxHeightAndPlotTrajectory(230, 39, 0:0.01:30)
```