```
function Comparison
clc
clear all

% Revised:
% February 10, 2008

% Authors:
% Ana Catalina Torres, Dr. Autar Kaw

% Purpose

%   To illustrate the concept of approximate error, absolute approximate
%   error, relative approximate error and absolute relative approximate
%   error, number of significant digits correct when using different
%   Difference Approximation of the first derivative of continuous
%   functions methods.

% Inputs
%    Clearing all data, variable names, and files from any other source and
%    clearing the command window after each succesive run of the program.

%    This is the only place in the program where the user makes changes to
%    the data
%        Function f(x)

         function k=f(x)
            k=exp(2*x);
         end

%   Declaring 'x' as a variable

x = sym('x','real');

%        Value of x at which f '(x) is desired, xv

         xv=4;

%        Starting step size, h

         h=0.2;

%        Number of times starting step size is halved

         n=12;

%---------------------------------------------------------------------------
disp(sprintf('            Comparing Methods of First Derivative Approximation'))
disp(sprintf('        Forward, Backward and Central Divided Difference Comparison'))
disp(sprintf('                       Ana Catalina Torres, Autar Kaw'))
disp(sprintf('                         University of South Florida'))
disp(sprintf('                           United States of America'))
```

```
disp(sprintf('                               kaw@eng.usf.edu'))
%-----------------------------------------------------------------------------
disp(sprintf('\n\n***************************** Introduction
******************************'))

disp(sprintf('\nThis worksheet demonstrates the use of Matlab to compare the
approximation'))
disp(sprintf('of first order derivatives using three different methods. Each method
uses'))
disp(sprintf('a point h ahead, or a point h behind, or both of the given value of x at'))
disp(sprintf('which the first derivative of f(x) is to be found.'))

disp(sprintf('\n\n************************** Section 1: Input
****************************'))
format short g
disp(sprintf('\nThe following simulation approximates the first derivative of a'))
disp(sprintf('function using different methods of approximation .The user inputs are'))
disp(sprintf('      a) function,\nf(x)=%g'))
disp(f(x))
disp(sprintf('      b) point at which the derivative is to be found, xv = %g',xv))
disp(sprintf('      c) starting step size, h = %g',h))
disp(sprintf('      d) number of times user wants to halve the starting step size, n = %
g',n))

disp(sprintf('The outputs include'))
disp(sprintf('      a) approximate values of the first derivative at the point and
initial'))
disp(sprintf('         step size given using different types of approximation'))
disp(sprintf('      b) exact value'))
disp(sprintf('      c) absolute relative true error, absolute relative approximate
error,'))
disp(sprintf('         and , least correct number of significant digits in the solution'))
disp(sprintf('         as a function of step size.'))

disp(sprintf('\nAll the information must be entered at the beginning of the M-File.'))


disp(sprintf('\n\n************************* Section 2: Simulation
**************************'))
disp(sprintf('\nThe exact value EV of the first derivative of the equation:'))
disp(sprintf('\nFirst, using the derivative command the solution is found. '))
Soln=diff(f(x))
disp(sprintf('In a second step, the exact value of the derivative is shown'))
disp(sprintf('The exact solution of the first derivative is:'))
Ev=subs(Soln,x,xv)

disp(sprintf('\nAn internal loop calculates the following:'))
disp(sprintf('Av: Approximate value of the first derivative using various
first\nderivative approximation methods'))
disp(sprintf('Ev: Exact value of the first derivative'))
disp(sprintf('Et: True error'))
disp(sprintf('et: Absolute relative true percentage error'))
```

```
disp(sprintf('Ea: Approximate error'))
disp(sprintf('ea: Absolute relative approximate percentage error'))
disp(sprintf('Sig: Least number of correct significant digits in an approximation'))
j=zeros(1,n);
N=zeros(1,n);
H=zeros(1,n);
Av=zeros(1,n);
Et=zeros(1,n);
et=zeros(1,n);
Ea=zeros(1,n);
ea=zeros(1,n);
Sig=zeros(1,n);

%   The next loop calculates the previously mentioned values:
for i=1:n
    N(i)=2^(i-1);
    H(i)=h/(N(i));
    AvFDD(i)=(f(xv+H(i))-f(xv))/(H(i));
    AvBDD(i)=(f(xv)-f(xv-H(i)))/(H(i));
    AvCDD(i)=(f(xv+H(i))-f(xv-H(i)))/(2*H(i));
    etFDD(i)=abs((Ev-AvFDD(i))/Ev*100);
    etBDD(i)=abs((Ev-AvBDD(i))/Ev*100);
    etCDD(i)=abs((Ev-AvCDD(i))/Ev*100);
    if i>1
        eaFDD(i)=abs((AvFDD(i)-AvFDD(i-1))/AvFDD(i)*100);
        eaBDD(i)=abs((AvBDD(i)-AvBDD(i-1))/AvBDD(i)*100);
        eaCDD(i)=abs((AvCDD(i)-AvCDD(i-1))/AvCDD(i)*100);
        if 0<eaFDD(i)<5
           SigFDD(i)=floor((2-log10(eaFDD(i)/0.5)));
        else
           SigFDD(i)=0;
        end
        if 0<eaBDD(i)<5
           SigBDD(i)=floor((2-log10(eaBDD(i)/0.5)));
        else
           SigBDD(i)=0;
        end
        if 0<eaCDD(i)<5
           SigCDD(i)=floor((2-log10(eaCDD(i)/0.5)));
        else
           SigCDD(i)=0;
        end
    end
   end
%   The loop halves the value of the starting step size n times. Each time
%   the approximate value of the derivative is calculated and saved in a
%   vector. The approximate error is calculated after at least two
%   approximate values of the derivative have been saved. The number of
%   significant digits is calculated. If the number of significant digits
%   calculated is less than zero, it is shown as zero.

disp(sprintf('\n\n******************** Section 3: Table of Values
```

```
************************'))

disp(sprintf('\nThe next tables show the step size value, approximate value, true
error,'))
disp(sprintf('the absolute relative true percentage error, the approximate error, the'))
disp(sprintf('absolute relative approximate percentage error and the least number of '))
disp(sprintf('correct significant digits in an approximation as a function of the
step\nsize value.'))

disp('        H           AvFDD         AvBDD       AvCDD')
Results=[H' AvFDD' AvBDD' AvCDD'];
disp(sprintf('\n'))
disp(Results)
disp(sprintf('\n'))
disp('        H            etFDD        etBDD       etCDD')
disp(sprintf('\n'))
ResultsCont=[H' etFDD' etBDD' etCDD'];
disp(ResultsCont)
disp(sprintf('\n'))
disp('        H            eaFDD        eaBDD       eaCDD')
disp(sprintf('\n'))
ResultsCont=[H' eaFDD' eaBDD' eaCDD'];
disp(ResultsCont)
disp(sprintf('\n'))
disp('        H             SigFDD       SigBDD        SigCDD')
disp(sprintf('\n'))
ResultsCont=[H' SigFDD' SigBDD' SigCDD'];
disp(ResultsCont)

disp(sprintf('\n\n************************** Section 4: Graphs
***************************'))

disp(sprintf('\nThe attached graphs show the approximate solution, absolute relative
true'))
disp(sprintf('error, absolute relative approximate error and least number of
significant'))
disp(sprintf('digits as a function of the number of iterations.\n'))

set(0,'Units','pixels')
  scnsize=get(0,'ScreenSize');
  wid=round(scnsize(3));
  hei=round(0.9*scnsize(4));
  wind=[1, 1, wid, hei];
  figure('Position',wind)

%   Approximate Solutions vs. Step size:

subplot(2,2,1); plot(H,AvFDD,'r:',H,AvBDD,'-.',H,AvCDD,'y','LineWidth',2)
xlabel('Step Size')
ylabel('Approximate Value')
legend('FDD','BDD','CDD',3);
title({'Approximate Solution of the First Derivative of a function using'; 'different
```

```
Methods of Approximation as a Function of Step Size'})

%   Absolute relative true error vs. Step size:

subplot(2,2,2); plot(H,etFDD,'r:',H,etBDD,'-.',H,etCDD,'y','LineWidth',2)
xlabel('Step Size')
ylabel('Absolute Relative True Error')
legend('FDD','BDD','CDD',2);
title('Absolute Relative True Percentage Error as a Function of Step Size')

%   Absolute relative approximate error vs. Step size:

subplot(2,2,3); plot(H(2:n),eaFDD(2:n),'r:',H(2:n),eaBDD(2:n),'-.',H(2:n),eaCDD(2:↙
n),'y','LineWidth',2)
xlabel('Step Size')
ylabel('Absolute Relative Approximate Error')
legend('FDD','BDD','CDD',2);
title('Absolute Relative Approximate Percentage Error as a Function of Step Size')

%   Number of significant digits vs. the number of iterations.

Y=[SigFDD',SigBDD',SigCDD'];
subplot(2,2,4);
bar(Y,'group');
xlabel('Number of iterations');
ylabel('Number of Significant digits');
legend('FDD','BDD','CDD');
title('Number of Significant Digits as function of Number of Iterations');

disp(sprintf('\n\n**************************** References↙
******************************'))

disp(sprintf('\nNumerical Differentiation of Continuous Functions. See'))
disp(sprintf('http://numericalmethods.eng.usf.↙
edu/mws/gen/02dif/mws_gen_dif_txt_\ncontinuous.pdf'))

disp(sprintf('\n\n***************************** Questions↙
******************************'))

disp(sprintf('\n1. The velocity of a rocket is given by\n\n              v(t)=2000*ln↙
(140000/(140000-2100t))-9.8*t'))

disp(sprintf('\nUse the three different methods with a step size of 0.25 to find the'))
disp(sprintf('acceleration at t=5s. Compare with the exact answer and study the effect\nof↙
the step size.'))

disp(sprintf('\n\n2. Look at the true error vs. step size data for problem # 1. Do you↙
see'))
disp(sprintf('a relationship between the value of the true error and step size ?'))
disp(sprintf('Is this concidential? Is it similar for Forward and Backward Divided'))
disp(sprintf('Difference? Is it different for Central Divided '))
disp(sprintf('\n3. Choose a step size of h=10^-10 in problem #1. Keep halving the step'))
```

```
disp(sprintf('size. Does the approximate value get closer to the exact result or does'))
disp(sprintf('the result seem odd? Is it similar for Forward and Backward Divided'))
disp(sprintf('Difference? Is it different for Central Divided Difference method? Why?'))

disp(sprintf('\n\n**************************** Conclusions↙
******************************'))

disp(sprintf('\nThe worksheet shows the nature of accuracy of the three different↙
methods'))
disp(sprintf('of finding the first derivative of a continuous function. Forward and'))
disp(sprintf('Backward Divided Difference methods exhibit similar accuracies as they↙
are'))
disp(sprintf('first order accurate, while central divided difference shows more↙
accuracy'))
disp(sprintf('as it is second order accurate.'))

disp(sprintf↙
('\n\n------------------------------------------------------------------------'))
disp(sprintf('\nLegal Notice: The copyright for this application is owned by the '))
disp(sprintf('author(s). Neither Mathcad nor the author are responsible for any errors '))
disp(sprintf('contained within and are not liable for any damages resulting from the use↙
'))
disp(sprintf('of this material. This application is intended for non-commercial, '))
disp(sprintf('non-profit use only. Contact the author for permission if you wish to↙
use\nthis application for-profit activities.'))

end
```