1. The book has two different MATLAB® versions of Newton's method. One is a program (on page 162) and the other is a function (page 170). The comments on page 162 indicate the algorithm does not work on zero roots (i.e., it does not work when the true root is equal to zero). This is the case because of the form of "myrel".
   - Create a variation on the function m-file (page 170) in which the zero-root restriction is removed by changing the definition of "myrel." Specifically, do not divide by x; but leave the rest the same.
   - Then test your modified code on the MATLAB® pre-defined "sin" function (and remember your differential calculus: the derivative of sine is simply cosine).
   - Set the initial guess to pi/4. [You can still name the function "**newtfun_LastName.m**"].
   - Create a driver routine named **"test_newtfun_LastName.m"** that specifies the input arguments and calls newtfun_LastName.m.

Editor - \\home.ohio.edu\home\mm379121\Documents\MATLAB\Module6\test_newtfun_Musil.m

newtfun_Musil.m

```
function [x, f, conv] = newtfun_Musil(fh, dfh, x0)
% NEWTON      Uses Newton's method to solve f(x) = 0.
%             fh is handle to f(x), dfh is handle to f'(x).
%             Initial guess is x0.
%             Returns final value of x, f(x), and
%             conv (1 = convergence, 0 = divergence)

steps=0;% iteration counter
x = x0;
fprintf(" Initial Guess: %4.2f \n", x);
re = 1e-8;% required relative error
myrel = 1;

while (myrel > re)&&(steps < 20)
    xold = x;
    x = x - feval(fh, x)/feval(dfh, x);
    steps = steps +  1;
    fprintf("Zero approximation for x \t  f(x) \n");
    disp([x feval(fh, x)] )
    %myrel = abs((x-xold)/x);
    myrel = abs((x-xold)); % <--- Change Feb 7 2022
end

if myrel <= re
conv = 1;
else
conv = 0;
end

f = feval(fh, x);
end
```

test_newtfun_Musil.m

```

my_sin_function_handle = @sin;
my_sin_derivative_handle = @cos;
my_initial_guess = pi/4;

% Run the test

my_result = newtfun_Musil(my_sin_function_handle, my_sin_derivative_handle, my_initial_guess);
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> test_newtfun_Musil
 Initial Guess: 0.79
Zero approximation for x      f(x)
   -0.2146   -0.2130

Zero approximation for x      f(x)
    0.0034    0.0034

Zero approximation for x      f(x)
   1.0e-07 *

   -0.1260   -0.1260

Zero approximation for x      f(x)
   1.0e-23 *

    0.1654    0.1654

Zero approximation for x      f(x)
     0     0
```

2. Do exercise 8.4 on page 194 of the textbook but do not create tables and do not sketch the results by hand. Instead, save the computed values in vectors and plot the results as indicated at the end of the description of the exercise.

8.4 Write a program to compute a table of the function

$$f(x) = x \sin\left[\frac{\pi(1+20x)}{2}\right]$$

over the (closed) interval [−1, 1] using increments in x of (a) 0.2, (b) 0.1 and (c) 0.01.

Use your tables to sketch graphs of $f(x)$ for the three cases (by hand), and observe that the tables for (a) and (b) give totally the wrong picture of $f(x)$.

Get your program to draw the graph of $f(x)$ for the three cases, superimposed.

Answer

Editor - \\home.ohio.edu\home\mm379121\Documents\MATLAB\Module6\ex8p4_Musil.m

ex8p4_Musil.m

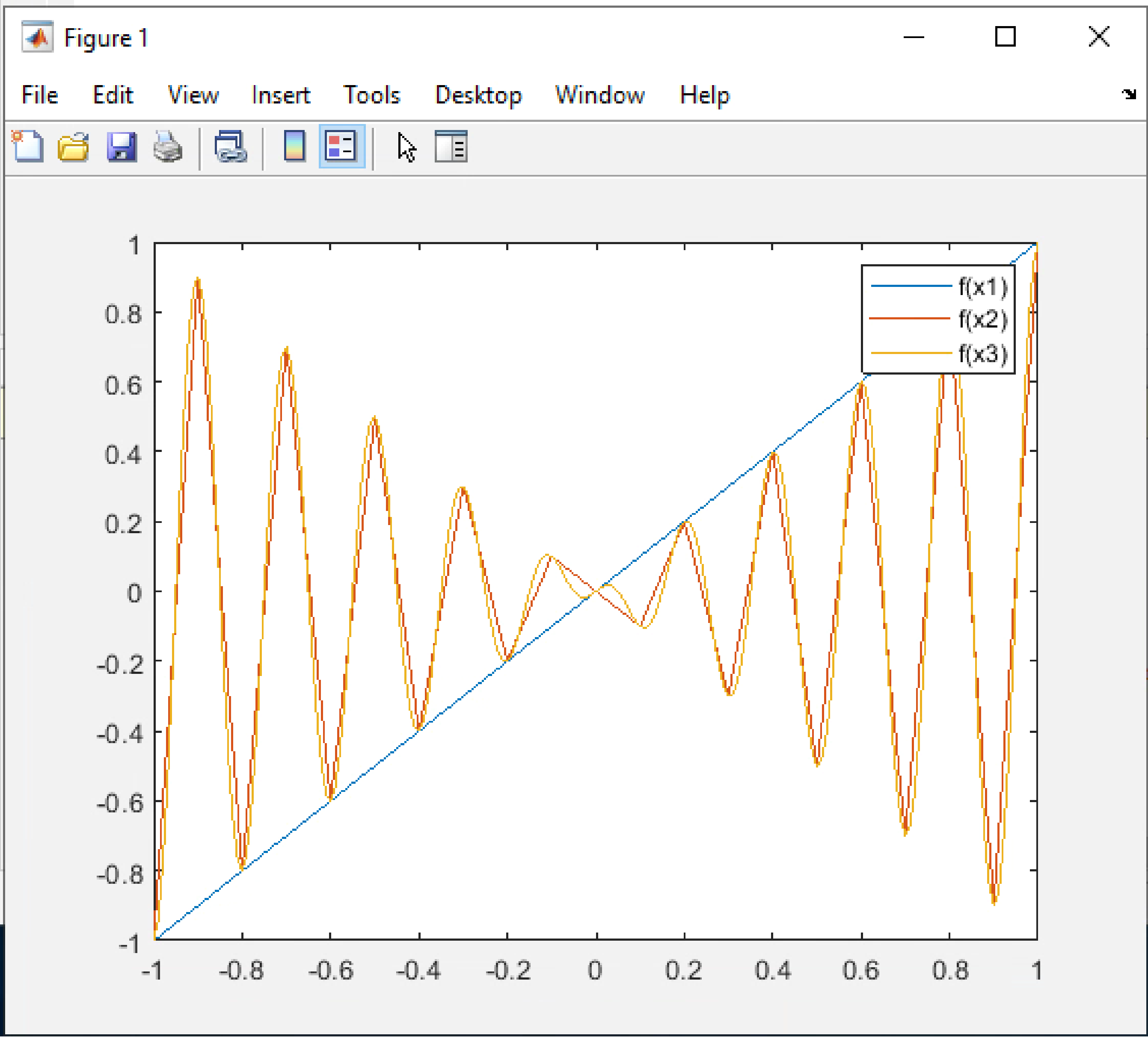
```matlab

% Mark Musil
% Computational Tools for Engineers
% Feb 2022


x1 = -1:0.2:1;
x2 = -1:0.1:1;
x3 = -1:0.01:1;

f_of_x1 = x1.*sin( (pi.*(1 + 20.*x1))./2);
f_of_x2 = x2.*sin( (pi.*(1 + 20.*x2))./2);
f_of_x3 = x3.*sin( (pi.*(1 + 20.*x3))./2);

plot(x1, f_of_x1, "-", x2, f_of_x2, "-", x3, f_of_x3, "-")
legend("f(x1)", "f(x2)", "f(x3)")
```



res by switching t

script

3. Do a variation of exercise 8.11 on page 192 of the textbook. Specifically, implement a function m-file called "mycos_LastName.m" that outputs four arguments: 1. the approximation of cosine. 2. the value obtained by MATLAB®'s built-in cosine function. 3. the difference between the two (which should be less than 1e-4). 4. a convergence flag (like the one used in newtfun_LastName.m). Test it on the following input values (in radians): 0, pi/4, pi/2, 3pi/4, pi.

8.11 Use the Taylor series

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

to write a program to compute $\cos x$ correct to four decimal places (x is in radians). See how many terms are needed to get 4-figure agreement with the MATLAB function `cos`. Don't make x too large; that could cause rounding error.

Answer

I used a Taylor series with 8 terms (including the constant 1) to reach convergence in the fourth decimal place.

Editor - \\home.ohio.edu\home\mm379121\Documents\MATLAB\Module6\mycos_Musil.m

ex8p4_Musil.m | mycos_Musil.m | newtfun_Musil.m

```
function [cosine_approximation, MATLAB_cosine, delta, convergence] = mycos_Musil(radians)
% MYCOS_MUSIL
% Approximate cosine using a Taylor series and compare with MATLAB

n = 7; %Taylor series order over 2

cosine_approximation = 1;
for k = 1:n
    added_term = ((-1)^k)*(radians^(2*k))/factorial(2*k);
    cosine_approximation = cosine_approximation + added_term;
end

MATLAB_cosine = cos(radians);
delta = abs(cosine_approximation - MATLAB_cosine);

if delta < 1e-4
    convergence = 1;
else
    convergence = 0;
end

fprintf(" x = %4.5f, Approximation of cos(x) = %4.5f, MATLAB cos(x) = %4.5f, delta = %4.5f, convergence  %4.5f \n",...
            radians, cosine_approximation, MATLAB_cosine, delta, convergence);
end
```

mycos_Musil_test_script.m

```
test_vec = [0 pi/4 pi/2 3*pi/4 pi];

for i = 1:length(test_vec)
    mycos_Musil(test_vec(i));
end
```

Command Window

New to MATLAB? See resources for Getting Started.

```
>> mycos_Musil_test_script
 x = 0.00000, Approximation of cos(x) = 1.00000, MATLAB cos(x) = 1.00000, delta = 0.00000, convergence  1.00000
 x = 0.78540, Approximation of cos(x) = 0.70711, MATLAB cos(x) = 0.70711, delta = 0.00000, convergence  1.00000
 x = 1.57080, Approximation of cos(x) = -0.00000, MATLAB cos(x) = 0.00000, delta = 0.00000, convergence  1.00000
 x = 2.35619, Approximation of cos(x) = -0.70711, MATLAB cos(x) = -0.70711, delta = 0.00000, convergence  1.00000
 x = 3.14159, Approximation of cos(x) = -1.00000, MATLAB cos(x) = -1.00000, delta = 0.00000, convergence  1.00000
>>
```