MAT 4010: Homework 6

Instructions

- Please write clearly and show all of your work. Answers without justification may be worth zero.
- When justifying your conclusions using a theorem, give the theorem name/number.
- Submit your homework to the appropriate assignment on Gradescope before the deadline.
- If you complete your homework on paper, you are encouraged to use a scanner app to convert photos of your work to a PDF before submitting to Gradescope.
- For programming questions, submit your source code (.m files) and any plots generated (.png files) to the appropriate "Programming" assignment on Gradescope before the deadline. Your .m files must follow the naming convention Hxx_Qyy.m where xx is a two-digit homework number and yy is a two-digit question number (e.g., H03_Q02.m).

Problem Set

1. Let $f(x) = x - \cos x$.

   a) Starting with the initial guesses $p_1 = 0$ and $p_2 = 1$, run secant method by hand to complete the following table. Keep as many digits of precision as you calculator gives (hopefully at least 8) in intermediate calculations and from one iteration to the next, but round to 8 significant figures when filling in the table to turn in.

n	p_n	$\lvert p_{n+1} - p_n \rvert \approx e_n$
1	0	1
2	1	
3		
4		
5		
6		N/A

   b) Use your last three approximations for e_n from part a) and the convergence order estimation formula found in Homework 1 to approximate α. Use 5 significant figures for your answer. Is it close to what you expect?

2. In this problem we will investigate using the Secant Method to approximate a root of $f_1(x) = x^2 - 5$ and also of $f_2(x) = (x^2 - 5)^2$.

   Write a program (using MATLAB/Octave) that implements the Secant Method as discussed in class. Your program should obtain from the user the function f for which a root is to be estimated as well as initial p_1 and p_2 values and values for ϵ and the maximum

n. You code should assume that $\alpha = 2$. Each iteration $n \geq 2$, your program should print the current n, p_n, and $\hat{e}_{n-1}$.

Your program should also produce three plots. The first plot should show p_n versus n, including appropriate axes labels. The second plot should show $\hat{e}_n$ versus $\hat{e}_{n-1}$. When labelling the axes, you can use

```
xlabel( '$\hat{e}_{n-1}$', 'interpreter', 'latex' );
```

For a title you can use (assuming your annonymous function variable is `f`)

```
title( sprintf( 'Secant Method for f(x) = %s', func2str(f) ) );
```

The third plot should show $\hat{\alpha}_n$ versus n where (recalling from Homework 1) we have

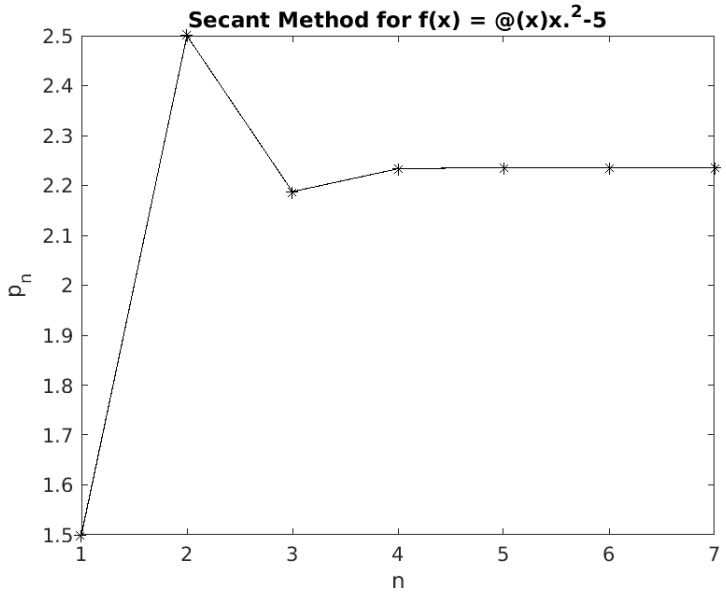
$$\alpha \approx \frac{\ln e_n - \ln e_{n-1}}{\ln e_{n-1} - \ln e_{n-2}} \approx \frac{\ln \hat{e}_n - \ln \hat{e}_{n-1}}{\ln \hat{e}_{n-1} - \ln \hat{e}_{n-2}} \equiv \hat{\alpha}_n$$

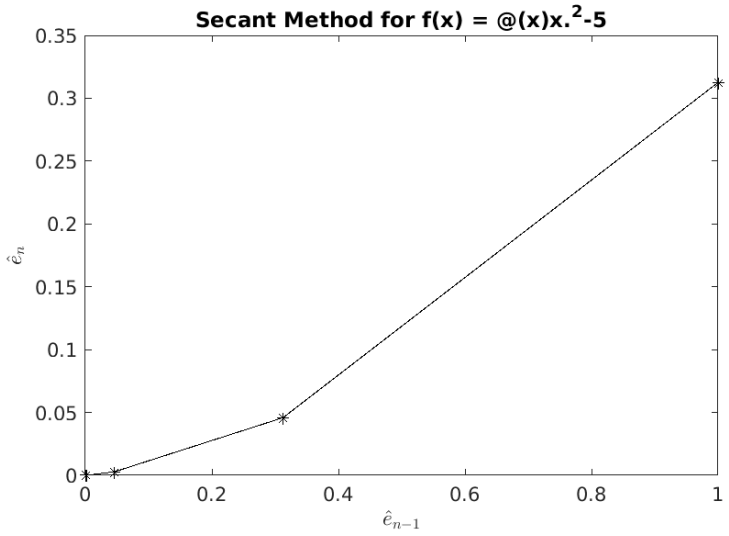
Note: Comment out any plotting-related code before submitting to Gradescope.

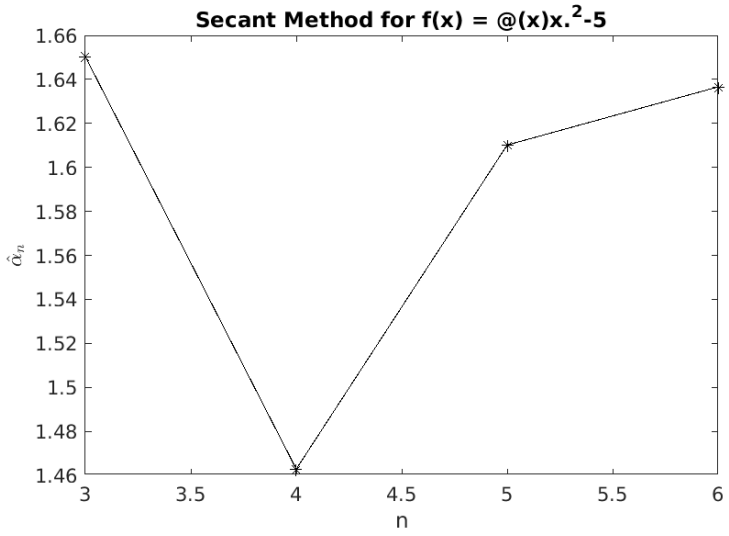
Sample output:

```
Enter a value for p1: 1.5
Enter a value for p2: 2.5
Enter an anonymous function f: @(x) x.^2 - 5
Enter a value for epsilon: 1e-6
Enter a maximum value for n: 20
n =  3: p(  3 ) =   2.18750000, ehat(  2 ) = 3.12500000e-01
n =  4: p(  4 ) =   2.23333333, ehat(  3 ) = 4.58333333e-02
n =  5: p(  5 ) =   2.23609802, ehat(  4 ) = 2.76468740e-03
n =  6: p(  6 ) =   2.23606796, ehat(  5 ) = 3.00616175e-05
n =  7: p(  7 ) =   2.23606798, ehat(  6 ) = 1.83819822e-08
```

Sample plots:







Sample output:

```
Enter a value for p1: 1000
Enter a value for p2: 1001
Enter an anonymous function f: @(x) x.^2 - 5
Enter a value for epsilon: 1e-6
Enter a maximum value for n: 10
n =  3: p(  3 ) = 500.25237381, ehat(  2 ) = 5.00747626e+02
n =  4: p(  4 ) = 333.55992298, ehat(  3 ) = 1.66692451e+02
n =  5: p(  5 ) = 200.12794717, ehat(  4 ) = 1.33431976e+02
n =  6: p(  6 ) = 125.09121226, ehat(  5 ) = 7.50367349e+01
n =  7: p(  7 ) =  76.99192004, ehat(  6 ) = 4.80992922e+01
n =  8: p(  8 ) =  47.68340881, ehat(  7 ) = 2.93085112e+01
n =  9: p(  9 ) =  29.48648488, ehat(  8 ) = 1.81969239e+01
n = 10: p( 10 ) =  18.28454136, ehat(  9 ) = 1.12019435e+01
Error: Secant Method did not converge
```

Sumbit: Your .m file and the .png files for inputs of `p1 = 2.0`, `p2 = 2.5`, `f = @(x) ( x.^2 - 5 ).^2`, `epsilon = 1e-6`, and `n = 40`.

3. Let $g(x) = 1 + \frac{1}{x}$. In the following, give your answers using 7 significant figures.

   a) Using the definition of a fixed point, use algebra to determine the fixed point in $[1, 2]$.

   b) Using fixed point iteration, compute p_5 given $p_1 = 1$. Compute the actual errors e_2, e_3, e_4, and e_5 using $e_n = |p_n - p|$ (using p from a)). Use the convergence order estimation formula found in Homework 1 to approximate α using e_3, e_4, and e_5.

   c) Use Aitken's Δ^2 method to compute $\hat{p}_3$, $\hat{p}_4$, and $\hat{p}_5$. Compute the actual errors $\hat{e}_3$, $\hat{e}_4$, and $\hat{e}_5$ using $\hat{e}_n = |\hat{p}_n - p|$ (using p from a)). Use the convergence order estimation formula found in Homework 1 to approximate α using $\hat{e}_3$, $\hat{e}_4$, and $\hat{e}_5$.

4. Let $g(x) = 1 + \frac{1}{x}$. In the following, give your answers using 7 significant figures.

   a) Given $\hat{p}_3 = 1$, use Steffensen's method to compute $\hat{p}_4$, $\hat{p}_5$, $\hat{p}_6$, and $\hat{p}_7$.

   b) Compute the actual errors $\hat{e}_4$, $\hat{e}_5$, $\hat{e}_6$, and $\hat{e}_7$ using $\hat{e}_n = |\hat{p}_n - p|$ (using p from the previous question).

   c) Use the convergence order estimation formula found in Homework 1 to approximate α using $\hat{e}_4$, $\hat{e}_5$, and $\hat{e}_6$.

5. Let $f(x) = \left(x - \sqrt{5}\right)^4 \sin(x)$.

   a) Find the function $F(x)$ for Aproach 1 of Modified Newton's Method.

   b) Find the function $g(x)$ for Aproach 1 of Modified Newton's Method.

6. In this problem we will investigate using Modified Newton's Method Approach 1 to approximate a root of $f_1(x) = (x^2 - 5)^2$ and also of $f_2(x) = \left(x - \sqrt{5}\right)^4 \sin(x)$.

   Write a program (using MATLAB/Octave) that implements Modified Newton's Method Approach 1 as discussed in class. Your program should obtain from the user the function f for which a root is to be estimated and its derivatives f' and f'', all as anonymous functions, as well as an initial p_1 and values for ϵ and the maximum n. Each iteration $n \geq 2$, your program should print the current n, p_n, and $\hat{e}_{n-1}$.

   Your program should also produce three plots. The first plot should show p_n versus n, including appropriate axes labels. The second plot should show $\hat{e}_n$ versus $\hat{e}_{n-1}$. When labelling the axes, you can use

   ```
   xlabel( '$\hat{e}_{n-1}$', 'interpreter', 'latex' );
   ```

   For a title you can use (assuming your annonymous function variable is `f`)

   ```
   title( sprintf( 'Newton''s Method for f(x) = %s', func2str(f) ) );
   ```

   The third plot should show $\hat{\alpha}_n$ versus n where (recalling from Homework 1) we have

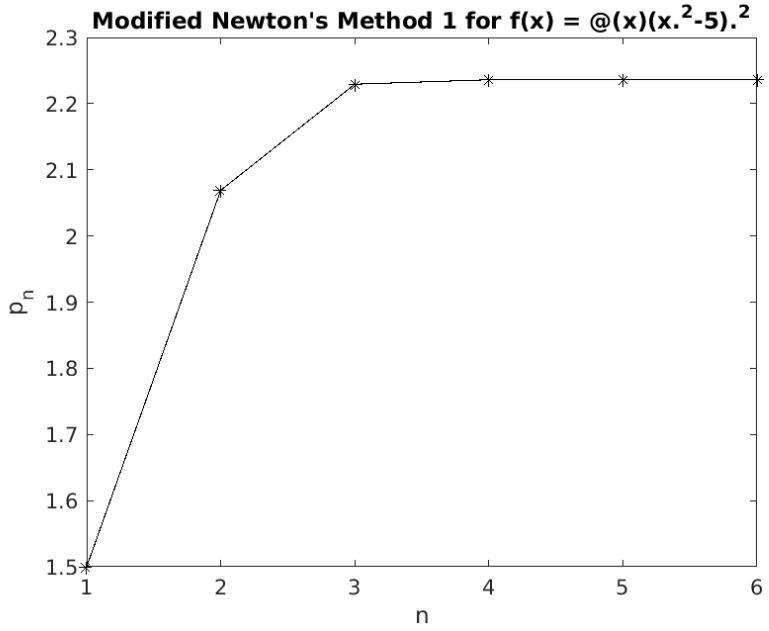
$$\alpha \approx \frac{\ln e_n - \ln e_{n-1}}{\ln e_{n-1} - \ln e_{n-2}} \approx \frac{\ln \hat{e}_n - \ln \hat{e}_{n-1}}{\ln \hat{e}_{n-1} - \ln \hat{e}_{n-2}} \equiv \hat{\alpha}_n$$

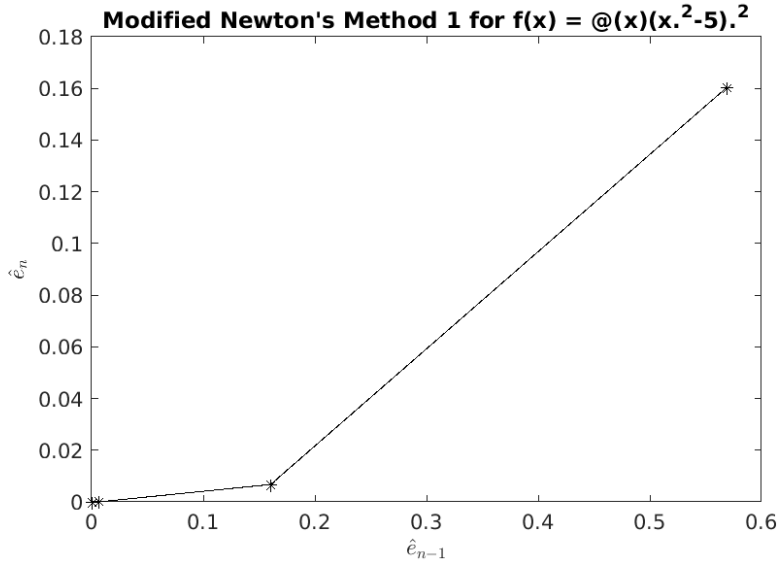
   Note: Comment out any plotting-related code before submitting to Gradescope.

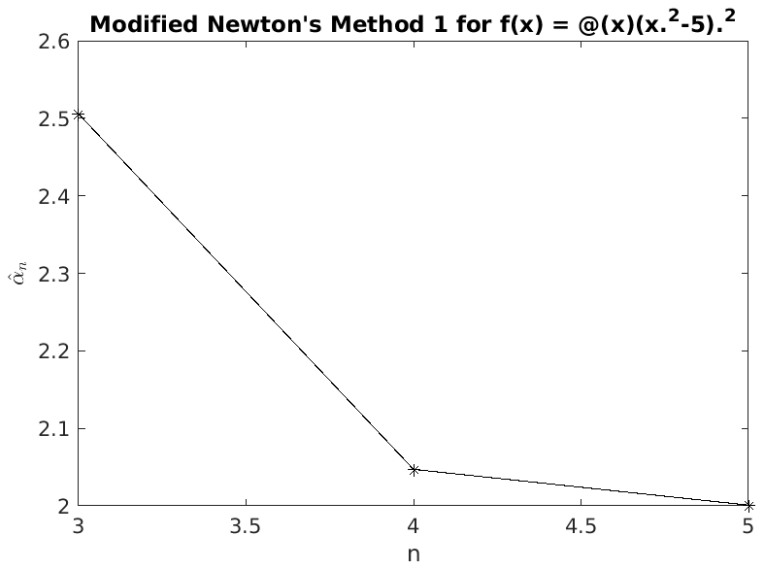
   Sample output:

```
Enter a value for p1: 1.5
Enter an anonymous function f: @(x) ( x.^2 - 5 ).^2
Enter an anonymous function f': @(x) 4 * ( x.^2 - 5 ) .* x
Enter an anonymous function f'': @(x) 4 * ( 3 * x.^2 - 5 )
Enter a value for epsilon: 1e-6
Enter a maximum value for n: 20
n =  2: p(  2 ) =  2.06896552, ehat(  1 ) = 5.68965517e-01
n =  3: p(  3 ) =  2.22934017, ehat(  2 ) = 1.60374649e-01
n =  4: p(  4 ) =  2.23605783, ehat(  3 ) = 6.71765923e-03
n =  5: p(  5 ) =  2.23606798, ehat(  4 ) = 1.01516880e-05
n =  6: p(  6 ) =  2.23606798, ehat(  5 ) = 2.30446773e-11
```

Sample plots:







Sample output:

```
Enter a value for p1: 1000
Enter an anonymous function f: @(x) ( x.^2 - 5 ).^2
Enter an anonymous function f': @(x) 4 * ( x.^2 - 5 ) .* x
Enter an anonymous function f'': @(x) 4 * ( 3 * x.^2 - 5 )
Enter a value for epsilon: 1e-6
Enter a maximum value for n: 10
n =  2: p(  2 ) =  0.00999995, ehat(  1 ) = 9.99990000e+02
n =  3: p(  3 ) =  0.01999950, ehat(  2 ) = 9.99955001e-03
n =  4: p(  4 ) =  0.03999580, ehat(  3 ) = 1.99963005e-02
n =  5: p(  5 ) =  0.07996602, ehat(  4 ) = 3.99702168e-02
n =  6: p(  6 ) =  0.15972776, ehat(  5 ) = 7.97617394e-02
n =  7: p(  7 ) =  0.31783374, ehat(  6 ) = 1.58105981e-01
n =  8: p(  8 ) =  0.62307900, ehat(  7 ) = 3.05245263e-01
n =  9: p(  9 ) =  1.15637101, ehat(  8 ) = 5.33292009e-01
n = 10: p( 10 ) =  1.82473667, ehat(  9 ) = 6.68365657e-01
Error: Modified Newton's Method 1 did not converge
```

Sumbit: Your .m file and the .png files for inputs of

```
p1 = 2
f = @(x) ( x - sqrt(5) ).^4 .* sin(x)
fprime = @(x) ( x - sqrt(5) ).^3 .* ( 4 * sin(x) + (x - sqrt(5)) .* cos(x) )
fdoubleprime = @(x) -( x - sqrt(5) ).^4 .* sin(x) + 12 * ( x - sqrt(5) ).^2 .* sin(x)
+ 8 * ( x - sqrt(5) ).^3 .* cos(x)
epsilon = 1e-6
```

and

```
n = 20.
```

7. Let $f(x) = \left(x - \sqrt{5}\right)^4 \sin(x)$.

   a) State m, the multiplicity of the root $\sqrt{5}$.

   b) Find the function $\tilde{g}(x)$ for Aproach 2 of Modified Newton's Method.

8. In this problem we will investigate using Modified Newton's Method Approach 2 to approximate a root of $f_1(x) = (x^2 - 5)^2$ and also of $f_2(x) = \left(x - \sqrt{5}\right)^4 \sin(x)$.

   Write a program (using MATLAB/Octave) that implements Modified Newton's Method Approach 2 as discussed in class. Your program should obtain from the user the function f for which a root is to be estimated and its derivative f', both as anonymous functions, as well as an initial p_1 and values for m, ϵ and the maximum n. Each iteration $n \geq 2$, your program should print the current n, p_n, and $\hat{e}_{n-1}$.

   Your program should also produce three plots. The first plot should show p_n versus n, including appropriate axes labels. The second plot should show $\hat{e}_n$ versus $\hat{e}_{n-1}$. When labelling the axes, you can use
   `xlabel( '$\hat{e}_{n-1}$', 'interpreter', 'latex' );`
   For a title you can use (assuming your annonymous function variable is `f`)
   `title( sprintf( 'Newton''s Method for f(x) = %s', func2str(f) ) );`
   The third plot should show $\hat{\alpha}_n$ versus n where (recalling from Homework 1) we have

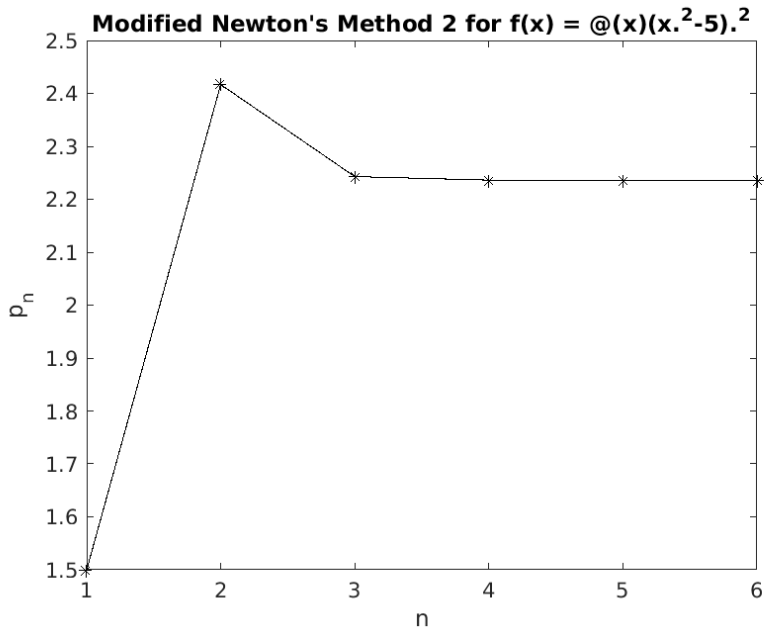
$$\alpha \approx \frac{\ln e_n - \ln e_{n-1}}{\ln e_{n-1} - \ln e_{n-2}} \approx \frac{\ln \hat{e}_n - \ln \hat{e}_{n-1}}{\ln \hat{e}_{n-1} - \ln \hat{e}_{n-2}} \equiv \hat{\alpha}_n$$

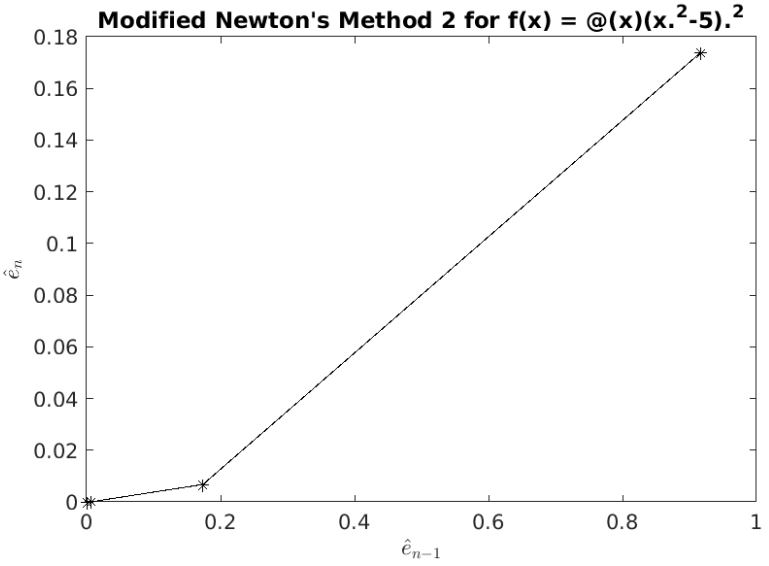
   Note: Comment out any plotting-related code before submitting to Gradescope.

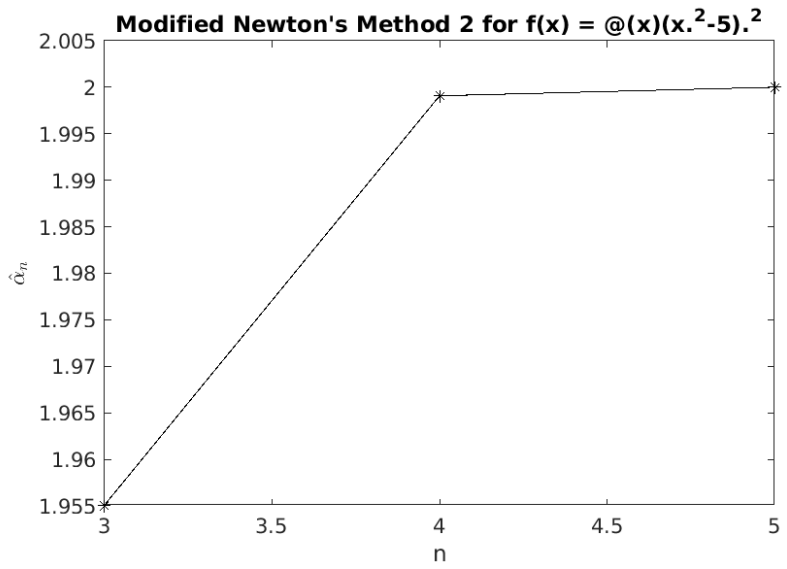
   Sample output:

```
Enter a value for p1: 1.5
Enter an anonymous function f: @(x) ( x.^2 - 5 ).^2
Enter an anonymous function f': @(x) 4 * ( x.^2 - 5 ) .* x
Enter a value for m: 2
Enter a value for epsilon: 1e-6
Enter a maximum value for n: 20
n =  2: p(  2 ) =  2.41666667, ehat(  1 ) = 9.16666667e-01
n =  3: p(  3 ) =  2.24281609, ehat(  2 ) = 1.73850575e-01
n =  4: p(  4 ) =  2.23607813, ehat(  3 ) = 6.73796270e-03
n =  5: p(  5 ) =  2.23606798, ehat(  4 ) = 1.01517341e-05
n =  6: p(  6 ) =  2.23606798, ehat(  5 ) = 2.30442332e-11
```

   Sample plots:


Modified Newton's Method 2 for f(x) = @(x)(x.^2-5).^2
2.5
2.4
2.3
2.2
2.1
2
1.9
1.8
1.7
1.6
1.5
p_n
1
2
3
4
5
6
n


Modified Newton's Method 2 for f(x) = @(x)(x.^2-5).^2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
ê_n
0
0.2
0.4
0.6
0.8
1
ê_{n-1}



Sample output:

```
Enter a value for p1: 1000
Enter an anonymous function f: @(x) ( x.^2 - 5 ).^2
Enter an anonymous function f': @(x) 4 * ( x.^2 - 5 ) .* x
Enter a value for m: 2
Enter a value for epsilon: 1e-6
Enter a maximum value for n: 10
n =  2: p(  2 ) = 500.00250000, ehat(  1 ) = 4.99997500e+02
n =  3: p(  3 ) = 250.00624998, ehat(  2 ) = 2.49996250e+02
n =  4: p(  4 ) = 125.01312474, ehat(  3 ) = 1.24993125e+02
n =  5: p(  5 ) = 62.52656027, ehat(  4 ) = 6.24865645e+01
n =  6: p(  6 ) = 31.30326314, ehat(  5 ) = 3.12232971e+01
n =  7: p(  7 ) = 15.73149545, ehat(  6 ) = 1.55717677e+01
n =  8: p(  8 ) =  8.02466459, ehat(  7 ) = 7.70683086e+00
n =  9: p(  9 ) =  4.32387180, ehat(  8 ) = 3.70079280e+00
n = 10: p( 10 ) =  2.74012140, ehat(  9 ) = 1.58375039e+00
Error: Modified Newton's Method 2 did not converge
```

Sumbit: Your .m file and the .png files for inputs of

```
p1 = 2
f = @(x) ( x - sqrt(5) ).^4 .* sin(x)
fprime = @(x) ( x - sqrt(5) ).^3 .* ( 4 * sin(x) + (x - sqrt(5)) .* cos(x) )
m = 4
epsilon = 1e-6
```

and

```
n = 20.
```