DM 1: Write down the 4th order Taylor polynomial of $\sqrt{x}$ centered at $x = 1$. Let $P(x)$ denote this polynomial. If $1 \le x \le 2$, what can you say about the size of $|\sqrt{x} - P(x)|$? Hint: Use the remainder term!

Solution:
The forth order taylor polynomial of $\sqrt{x}$ centered at $x = 1$,

$$P(x) = 1 + \frac{1}{2}(x-1) - \frac{1}{8}(x-1)^2 + \frac{1}{16}(x-1)^3 - \frac{5}{128}(x-1)^4$$

From Taylor's Theorem we know that there exists some $\xi \in (1, x)$ such that,

$$\sqrt{(x)} = P(x) + \frac{7}{256}(\xi)^{-\frac{9}{2}}(x-1)^5$$

Calculating the maximum error for $1 \le x \le 2$,

$$\begin{aligned}
|\sqrt{x} - P(x)| &= \frac{7}{256}|(\xi)^{-\frac{9}{2}}|(x-1)^5, \\
&\le \frac{7}{256}|1|(x-1)^5, \\
&\le \frac{7}{256}|1|((2)-1)^5, \\
&\le \frac{7}{256}.
\end{aligned}$$

Note that the fifth derivative is largest on the interval $(1, x)$ when ξ is closest to 1 and the remaining expression is largest when $x = 2$. Thus giving us a maximum error of $\frac{7}{256}$.

Chapter 4: 2 (b): For full credit you must write your own version of Newton's method. Your function should have the signature

```
function [r,hist] = hw3newton(f,fp,x1,ftol,xtol,Nmax)

end
```

The input values are

- `f`, the function to find a root of.
- `fp`, the derivative function of `f`.
- `x1`, the first iteration value.
- `ftol`, the tolerance for stopping based of the value of `f`

- `xtol`, the tolerance for stopping based on changes in `x`
- `Nmax`, the maximum number of iterations

Your function should exit with an error if more than `Nmax` iterations are used. It should return whenever $|f(x)| < f_{\text{tol}}$ or $|x_n - x_{n-1}| < x_{\text{tol}}$.

The return values should be `r`, the estimate of the root's position, and `hist`, a list of all estimates starting with `x1` and ending with the final estimate `r`.

Test that your function works by finding three different ways to call it so that iteration stops for each of the three possible reasons.

To answer the problem in the textbook, you will want to call your functin with $x_{\text{tol}} = 0$ to ensure that only the f_{tol} condition is used to stop the iteration.

Code:

```
function [r, hist] = hw3newton(f,fp,x1,ftol,xtol,Nmax)
% Takes a function(f), it's derivative(fp), an f(x) tolerance, an x
% tolerance, max number of iterations, and an initial root guess and
% returns an approximation, iteration history, and a message describing
% Newton's Method termination.


format long e
%Initializing variables
hist  = zeros;
count = 1;
x = x1;
%Assigning first value
hist(1) = x;



%Iteration Step
for i = 1:(Nmax - 1)        %initial guess is counted in Nmax

    if fp(x)==0   %% if fprime is 0, abort.
        error('fprime is 0')  %% the error function prints
                              %% message and exits
    end

    x = x - f(x)/fp(x) ;
    hist = [hist; x];
```

```
    %Checking tolerances

        if abs(f(x)) <= ftol
            disp('Inside of f tolerance')
         break
        end

        if i > 1  % hist must have at least two values to check xtol
            if abs(x - hist(i - 1)) <= xtol
                disp('Inside of x tolerance')
                break
            end
        end

        %termination message for Nmax iterations
        if i == (Nmax - 1)
            disp('Terminated after Nmax iterations')
        end

end

r = x;

end
```

Solution:
Testing our function works by demonstrating the 3 different ways it can terminate,

Console:

```
f = @(x) x^2 - 2

f =

    @(x)x^2-2

fp = @(x) 2*x

fp =

    @(x)2*x

[r, hist] = hw3newton(f,fp,1,0,.001,10)
```

```
Inside of x tolerance

r =

     1.414213562373095e+00


hist =

     1.000000000000000e+00
     1.500000000000000e+00
     1.416666666666667e+00
     1.414215686274510e+00
     1.414213562374690e+00
     1.414213562373095e+00

[r, hist] = hw3newton(f,fp,1,.001,0,10)

Inside of f tolerance

r =

     1.414215686274510e+00


hist =

     1.000000000000000e+00
     1.500000000000000e+00
     1.416666666666667e+00
     1.414215686274510e+00

[r, hist] = hw3newton(f,fp,1,0,0,5)

Terminated after Nmax iterations

r =

     1.414213562374690e+00


hist =

     1.000000000000000e+00
```

```
1.500000000000000e+00
1.416666666666667e+00
1.414215686274510e+00
1.414213562374690e+00
```

Running our routine for the function with $x1 = 5$ and tolerance for $f(x)$ at 10^{-8},

$$f(x) = (5 - x)e^x - 5.$$

Console:

```
f = @(x) (5 - x).*exp(x) - 5

f =

    @(x)(5-x).*exp(x)-5

fp = @(x) ((5 - x).*exp(x))- (exp(x))

fp =

    @(x)((5-x).*exp(x))-(exp(x))

[r, hist] = hw3newton(f,fp,5,10^-8,0,100)

Inside of f tolerance

r =

     4.965114231746430e+00


hist =

     5.000000000000000e+00
     4.966310265004573e+00
     4.965115686301458e+00
     4.965114231746430e+00


hist - 4.9651142317... (higher precision value approximated by wolfram)

ans =

     3.488576825572398e-02
     1.196033260296936e-03
```

```
1.454557182256622e-06
2.153832667772804e-12
```

Looking at the values for our error when compared to a more precise wolfram computation we can see clear quadratic convergence and to get to an $f(x)$ tolerance as small as 10^{-16} would only require one more step. Furthermore we can show this by calculating,

$$\lim_{k\to\infty} \frac{|e_{k+1}|}{|e_k|^2} \approx C.$$

Where C is a constant, and luckily since our error converges so fast we can see that after the first iteration this limit will be approximately 1.

Console:

```
err(2)/err(1)^2

ans =

    9.827581952581104e-01

err(3)/err(2)^2

ans =

    1.016820480396692e+00

err(4)/err(3)^2

ans =

    1.018006055849449e+00
```

Chapter 4: 6 (a,b): Also, use your `hw3newton` function from the previous problem to compute the location of the minimum, and generate a plot that indicates that your computation succeeded. Consider the function,

$$h(x) = \frac{x^4}{4} - 3x.$$

In this problem, we will see how to use Newton's method to find the minimum of the function $h(x)$.

a. Derive a function f that has a root at the point where h achieves it minimum. Write down the formula for Newton's method applied for f.

Solution:
Recall that at the minima and maxima of continuous, differentiable function the slope is equal to zero, and thus let $f(x) = h'(x)$. Calculating the derivative we get,

$$f(x) = x^3 - 3$$

To find the roots of f using Newton's method we also need to find f',

$$f'(x) = 3x^2.$$

Stating the iterative formula for Newton's method derived from the first order Taylor Polynomial,

$$x_{k+1} = x_k - \frac{x_k^3 - 3}{3x_k^2}$$

b. Take one step (by hand) with Newton's method starting with $x_0 = 1$.

Solution:
Simply substituting $x_0 = 1$ into the previous formula,

$$\begin{aligned} x_1 &= 1 - \frac{1^3 - 3}{3(1)^2}, \\ &= 1 + \frac{2}{3}, \\ &= \frac{5}{3}. \end{aligned}$$

Chapter 4: 7: In finding a root with Newton's method, an initial guess of $x_0 = 4$ with $f(x_0) = 1$ leads to $x_1 = 3$. What is the derivative of f at x_0

Solution:
Simply substituting our given values to the iterative formula for Newton's method gives us,

$$\begin{aligned} x_{k+1} &= x_k - \frac{f(x_k)}{f'(x_k)}, \\ 3 &= 4 - \frac{1}{f'(x_0)}, \\ -1 &= -\frac{1}{f'(x_0)}, \\ 1 &= f'(x_0). \end{aligned}$$

Chapter 4: 10: Consider that the function,

$$f(x) = \frac{x^2 - 2x + 1}{x^2 - x - 2}$$

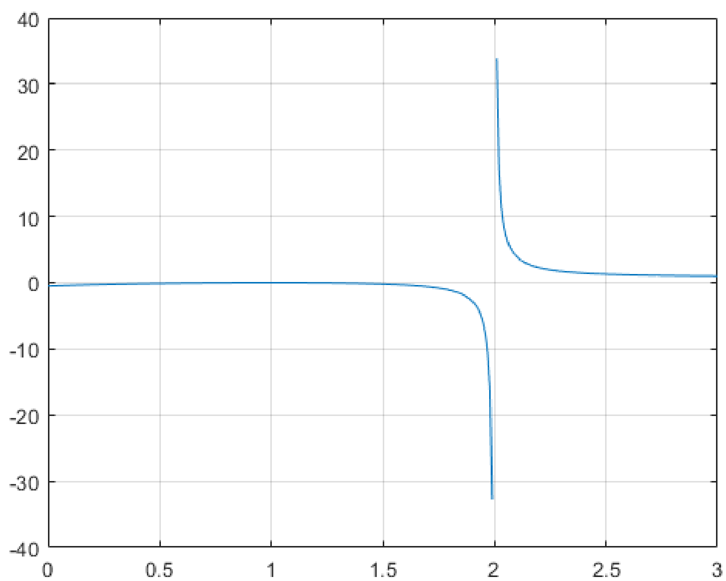
has exactly one zero in the interval $[0, 3]$, at $x = 1$. Using $a = 0$ and $b = 3$, run the bisection method on $f(x)$ with a stopping tolerance of $\delta = 1e - 3$. Explain why it does not appear to converge to the root. Why does the Intermediate Value Theorem *not* guarantee that there is a root in $[0, 1.5]$ or in $[1.5, 3]$

Use MATLAB to plot this function over the interval in a way that makes it cleat what is going on. [Hint: You may want to use the plot command over two different x intervals separately in order to show the behavior properly.]

Solution:
Consider the plot of the function,

$$f(x) = \frac{x^2 - 2x + 1}{x^2 - x - 2}.$$

Figure 1: f(x) on the interval of $[0, 3]$

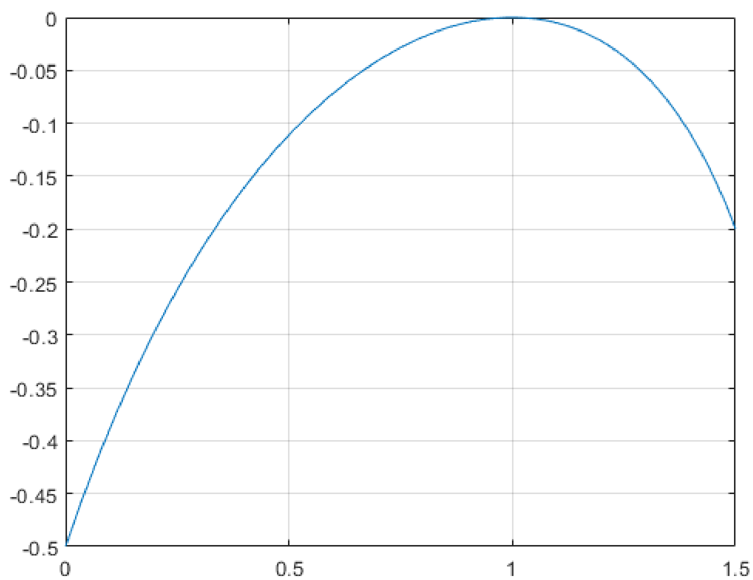


Note that there is an infinite discontinuity at $x = 2$. Recall that the Intermediate Value Theorem states that, If $f(x)$ is continuous on the interval $[a, b]$ and y lies between $f(a)$ and $f(b)$, then there exists some point $x \in [a, b]$ where $f(x) = y$. As a corollary we know that if $f(a)$ and $f(b)$ have opposite signs then $y = 0 \in [a, b]$ which is the principle behind bisection. Demonstrating why IMV Theorem fails to find a root of x = 1 on the interval [0, 1.5],

$$f(0) = \frac{0^2 - 2(0) + 1}{0^2 - 0 - 2} = -\frac{1}{2},$$

$$f(\frac{3}{2}) = \frac{(\frac{3}{2})^2 - 2(\frac{3}{2}) + 1}{(\frac{3}{2})^2 - \frac{3}{2} - 2} = -\frac{1}{5},$$

Figure 2: f(x) on the interval of [0, 1.5]



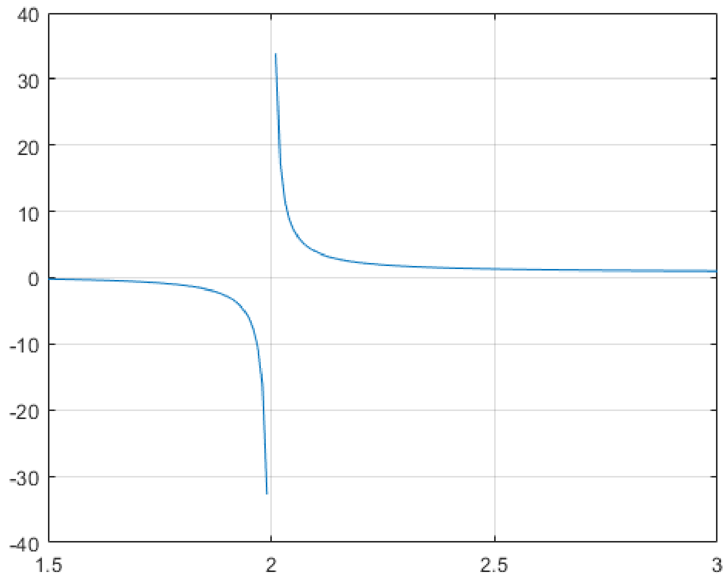
Therefore we can see both bounds have the same sign and thus fails the corollary to IMV Theorem so bisection will not find root $x = 1$.

Demonstrating why IMV Theorem fails to find a root of on the interval [1.5, 1], simply by demonstrating discontinuity at x = 2. Consider the following limit,

$$lim_{x\to 2}\frac{x^2 - 2x + 1}{x^2 - x - 2} = \frac{1}{0}.$$

H

Figure 3: f(x) on the interval of [1.5, 3]



Thus we have shown that the limit of $f(x)$ where $x \rightarrow 2$ does not exists and therefore $f(x)$ is discountinuous on the interval $[1.5, 1]$. Interestingly since the $f(2)$ is an infinite discontinuity it seems to explain why our bisection algorithm converges to $x = 2$ as the limits on either side of 2 have opposite signs.

Console:

```
f =

  function_handle with value:

    @(x)((x.^2)-(2.*x)+(1))./((x.^2)-(x)-(2))

>> [root, hist, error] = HW2Bisect(0,3,f,1e-3)

root =

        2.000244140625000e+00


hist =

                        0        3.000000000000000e+00
        1.500000000000000e+00    3.000000000000000e+00
```

```
   1.500000000000000e+00   2.250000000000000e+00
   1.875000000000000e+00   2.250000000000000e+00
   1.875000000000000e+00   2.062500000000000e+00
   1.968750000000000e+00   2.062500000000000e+00
   1.968750000000000e+00   2.015625000000000e+00
   1.992187500000000e+00   2.015625000000000e+00
   1.992187500000000e+00   2.003906250000000e+00
   1.998046875000000e+00   2.003906250000000e+00
   1.998046875000000e+00   2.000976562500000e+00
   1.999511718750000e+00   2.000976562500000e+00


error =

   7.324218750000000e-04
```