

Math 151A Homework #2 – due Wednesday 10/18, in class

Show all your work!

1. The square root of two, revisited.

Use Newton's method and the secant method to approximate the value of $\sqrt{2}$ to within 10^{-4} . As usual, submit your code and a list of intermediate approximations.

- a. Use Newton's method with an initial guess $p_0 = 1$. How does this compare with the bisection method (from last week's homework), in terms of number of iterations required?

Answer:

Create a function file called *f.m* containing:

```
function f = f(x)
    f = x.^2 - 2;
end
```

and another called *f1.m* containing the first derivative:

```
function f1 = f1(x)
    f1 = 2*x;
end
```

and a script called *newton.m* containing, for example:

```
%-----
% Matlab/Octave program to find the root of a function
% using Newton's method.
% - the function must be defined in f.m, and its
%   derivative in f1.m
%-----

% set initial guess p0, and the desired precision
p0 = 1;
tol = 1.0e-4;

% prevent infinite loops, by setting a maximum
% number of iterations
maxits = 1000;

for i = 1:maxits

    % compute the subsequent approximation
    p1 = p0 - f(p0)/f1(p0);

    % nicely formatted printing of each approximation
    printf("%.2d:  p = %.8f\n",i,p1);

    % check to see if we are at the root
    if f(p1)==0
```

```

        break
    endif

    % check to see if we've reached the desired precision
    if( abs(p1-p0) < tol)
        break
    endif

    % new approximation becomes old approximation
    p0 = p1;

end

```

Then run the script:

```

octave:1> newton
01: p = 1.50000000
02: p = 1.41666667
03: p = 1.41421569
04: p = 1.41421356

```

Notice that we converge much faster than with the bisection method (from last week's homework).

- b. Use Newton's method with an initial guess $p_0 = 0.0001$. What happens to the number of iterations required, and why?

Answer:

Modify the newton.m code to start from initial condition $p_0 = 0.0001$ and run it:

```

octave:2> newton
01: p = 10000.00005000
02: p = 5000.00012500
03: p = 2500.00026250
04: p = 1250.00053125
05: p = 625.00106562
06: p = 312.50213281
07: p = 156.25426638
08: p = 78.13353302
09: p = 39.07956511
10: p = 19.56537138
11: p = 9.83379640
12: p = 5.01858833
13: p = 2.70855338
14: p = 1.72347746
15: p = 1.44196100
16: p = 1.41448053
17: p = 1.41421359
18: p = 1.41421356

```

The number of iterations required increases significantly, because the derivative of the function at 0.0001 is nearly zero.

- c. Use the secant method with initial guesses $p_0 = 1$ and $p_1 = 2$. How does this compare to the Newton's method from part a?

Answer:

Matlab code for the secant method:

```
%-----  
% Matlab/Octave program to find the root of a function  
% using the secant method.  
% - the function must be defined in f.m  
%-----  
  
% set initial guesses p0 and p1, and the desired precision  
p0 = 1;  
p1 = 2;  
tol = 1.0e-4;  
  
% prevent infinite loops, by setting a maximum  
% number of iterations  
maxits = 1000;  
  
for i = 1:maxits  
  
    % compute the subsequent approximation  
    p2 = p1 - f(p1)*(p1-p0)/(f(p1)-f(p0));  
  
    % nicely formatted printing of each approximation  
    printf("%.2d: p = %.8f\n",i,p1);  
  
    % check to see if we are at the root  
    if f(p1)==0  
        break  
    endif  
  
    % check to see if we've reached the desired precision  
    if( abs(p1-p0) < tol)  
        break  
    endif  
  
    % new approximation becomes old approximation  
    p0 = p1;  
    p1 = p2;  
  
end
```

Output:

```

octave:3> secant
01: p = 2.00000000
02: p = 1.33333333
03: p = 1.40000000
04: p = 1.41463415
05: p = 1.41421144
06: p = 1.41421356

```

The number of iterations required is comparable to Newton's method, although slightly more. This increase in number of iterations *may* be offset by not having to compute the derivative.

2. Linear convergence. (problem 6b of section 2.4)

Show that the sequence defined by $p_n = 1/n^2$ (for $n \geq 1$) converges linearly to $p = 0$. How large must n be before we have $|p_n - p| \leq 5 \times 10^{-2}$?

Answer:

Obviously p_n converges to 0, since $\lim_{n \rightarrow \infty} 1/n^2 = 0$. Now consider the ratio:

$$\frac{p_{n+1} - p}{p_n - p} = \frac{1/(n+1)^2}{1/n^2} = \frac{n^2}{(n+1)^2}$$

$$\lim_{n \rightarrow \infty} \left(\frac{n^2}{(n+1)^2} \right) = \lim_{n \rightarrow \infty} \frac{n^2}{n^2 + 2n + 1} = \lim_{n \rightarrow \infty} \frac{2n}{2n + 2} = \lim_{n \rightarrow \infty} \frac{2}{2} = 1.$$

Which means that

$$\lim_{n \rightarrow \infty} \frac{|p_{n+1} - p|}{|p_n - p|} = 1,$$

so we have convergence of order $\alpha = 1$ with asymptotic error constant $\lambda = 1$. Note that this is *not* linear convergence, since linear convergence requires that $\alpha = 1$ *and* $\lambda < 1$.

3. Quadratic convergence. (problem 8, section 2.4)

Show that the sequence defined by $p_n = 10^{-2^n}$ converges quadratically to 0.

Answer:

Since $\lim_{n \rightarrow \infty} 2^n = \infty$, then $\lim_{n \rightarrow \infty} 10^{-2^n} = 0$. Now consider the ratio:

$$\frac{p_{n+1} - p}{(p_n - p)^2} = \frac{10^{-2^{n+1}}}{(10^{-2^n})^2} = \frac{10^{-2^n \cdot 2}}{(10^{-2^n})^2} = \frac{(10^{-2^n})^2}{(10^{-2^n})^2} = 1.$$

So then

$$\lim_{n \rightarrow \infty} \frac{|p_{n+1} - p|}{|p_n - p|^2} = 1$$

and we have the order of convergence to be $\alpha = 2$.

4. Dealing with multiple zeros.

Consider the function $f(x) = x^3 + 4x^2 + 8x$.

- a. Show that $p = 0$ is a simple zero of $f(x)$.

Answer:

We know that 0 is a zero of f , since $f(0) = 0^3 + 4 \cdot 0^2 + 8 \cdot 0 = 0$. It is a simple zero, since the first derivative $f'(0) = 3 \cdot 0^2 + 8 \cdot 0 + 8 \neq 0$ is non-zero.

- b. Use Newton's method to find the root to within 10^{-6} , with initial guess $p_0 = 1$.

Answer:

We can use the same Newton's method code from the first problem, with the function files $f.m$:

```
function f = f(x)
    f = x.^3 + 4*x.^2 + 8*x;
end
```

and its derivative $f1.m$:

```
function f1 = f1(x)
    f1 = 3*x.^2 + 8*x + 8;
end
```

which gives the result:

```
octave:1> newton
01: p = 0.31578947
02: p = 0.04266552
03: p = 0.00089097
04: p = 0.00000040
05: p = 0.00000000
```

Now consider the similar function $g(x) = x^3 + 4x^2$.

- c. Show that $p = 0$ is a zero of multiplicity 2 of $g(x)$.

Answer:

We know that 0 is a zero of g since $g(0) = 0^3 + 4 \cdot 0^2 = 0$. It is not a simple zero, because the first derivative is also zero at 0: $g'(0) = 3 \cdot 0^2 + 8 \cdot 0 = 0$. However, the second derivative is non-zero ($g^{(2)}(0) = 6 \cdot 0 + 8 \neq 0$), so 0 is a zero of g of multiplicity two.

- d. Use Newton's method to find the root to within 10^{-6} , with initial guess $p_0 = 1$.

Answer:

```
octave:2> newton
01: p = 0.54545455
02: p = 0.28816467
03: p = 0.14876612
04: p = 0.07569318
05: p = 0.03819480
06: p = 0.01918729
07: p = 0.00961649
08: p = 0.00481400
09: p = 0.00240845
```

```
10: p = 0.00120459
11: p = 0.00060238
12: p = 0.00030121
13: p = 0.00015061
14: p = 0.00007531
15: p = 0.00003765
16: p = 0.00001883
17: p = 0.00000941
18: p = 0.00000471
19: p = 0.00000235
20: p = 0.00000118
21: p = 0.00000059
```

- e. Use the modified (multiple zero) Newton's method to find the root to within 10^{-6} , with initial guess $p_0 = 1$.

Answer:

The code is the same as for Newton's method, but with the iterater $p_1 = p_0 - f(p_0)/f_1(p_0)$; replaced with $p_1 = p_0 - f(p_0)*f_1(p_0)/(f_1(p_0)^2 - f(p_0)*f_2(p_0))$;

The result of running it is:

```
octave:3> modified_newton
```

```
01: p = -0.07843137
02: p = -0.00079984
03: p = -0.00000008
04: p = -0.00000000
```

where we what appears to be quadratic convergence (compare to part b.).