MM5 Secant Method

(Iterative Solution of Equations)

- kl.8.15-9.00 review of MM4 and some examples
- kl.9.00 – 10.30 exercise (see notes)
- kl.10.40-11.30 MM5 lecture (I)

Bisection Method (MM3)

- Motivation: intermediate value Theorem

  Continuous function f(x) on an interval [a,b] has the property: f(a)f(b)<0, then there is a solution of f(x)=0 between a and b.

- Algorithm
- Matlab codes

input	•Equation f(x)=0 •Interval [a,b] such that f(a)f(b)<0 •Tolerance e
repeat	m:=(a+b)/2 If f(a)f(m)=< 0, then b:=m else a:=m
until	b-a< e
output	Solution lies in [a,b] with the length less than e

Function Iteration Method (MM3)

- Rewrite f(x)=0 as x=g(x)
- Initial guess x_0
- Iterative solution: x_n=g(x_n-1)

Suppose that function $g(x)$ is differentiable on $[a,b]$ and that

(1) $g(x) \in [a,b]$ for any $x \in [a,b]$, and

(2) $|\frac{dg(x)}{dx}| \leq K < 1$ for all $x \in [a,b]$;

Then, the equation $x = g(x)$ has a unique solution in the interval $[a,b]$ and the iterative sequence defined by

$x_0 \in [a,b], \quad x_n = g(x_{n-1}), \; n = 1, 2, \cdots$

converge to this solution

Iteration Method – Converging Rate (MM4)

assume the solution to $x = g(x)$ is s, and an iterative solution is x_n, then the sequence of errors in the iterates is

$$e_n = x_n - s$$

$$e_{n+1} = x_{n+1} - s = g(x_n) - s = g(s) + g'(s)(x_n - s) + \frac{g''(s)}{2!}(x_n - s)^2 + \cdots - g(s)$$

$$e_{n+1} = g'(s)(x_n - s) + \frac{g''(s)}{2!}(x_n - s)^2 + \cdots \approx g'(s)(x_n - s) = g'(s)e_n$$

Concern the example: s=0

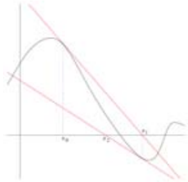
$(i) \quad g(x) = \frac{e^x - 1}{2}, \quad g'(x) = \frac{e^x}{2}, \quad e_{n+1} \approx \frac{1}{2} e_n$

$(ii) \quad g(x) = e^x - x - 1, \quad g'(x) = e^x - 1, \quad g''(x) = e^x, \quad e_{n+1} \approx \frac{1}{2!} e_n^2$

quadratic (second-order convergence)

Newton's Method (MM4)

- Newton-Raphson method



Suppose to solve $f(x)=0$

The first-order Taylor expansion of f about a point x_0 is

$$f(x) \approx f(x_0) + (x - x_0) f'(x_0)$$

If the point x_0 is close to the required solution, then we expect that

$$f(x_0) + (x - x_0) f'(x_0) = 0$$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0,1,2,\cdots$$

$$g(x) = x - \frac{f(x)}{f'(x)}$$

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

$$= 1 - 1 + 0 = 0$$

Convergence Theorem (MM4)

Let f be twice differentiable on an interval $[a,b]$ and satisfy the conditions

(i) $f(a)f(b) < 0$

(ii) $f'(x)$ has no zeros on $[a,b]$

(iii) $f''(x)$ does not change sign in $[a,b]$

(iv) $\left|\frac{f(a)}{f'(a)}\right|, \quad \left|\frac{f(b)}{f'(b)}\right| < b - a$

Then f(x)=0 has a unique solution and Newton's iteration will converge to this solution from any starting point in $[a,b]$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0,1,2,\cdots$$

Newton's Method – Matlab Codes

```
function sol = newton(f,df,x0,tol)
% Newtons method for solution of the equation
% f(x) = 0
% to an accuracy tol.
%The initial guess is input as x0.
% f and its derivative df must be saved as m-files.

maxits=100;
x1 = x0-feval(f,x0)/feval(df,x0);
its = 1;
while (abs(x1-x0)>tol)&(its<maxits)
   x0 = x1;
   x1 = x0-feval(f,x0)/feval(df,x0);
   its=its+1;
end

its
sol=x1;
```

```
function y=example51(x);
format long
y=exp(x)-2*x-1;
```

```
function y=dexample51(x);
format long
y=exp(x)-2;
```

```
>> y=newton(@example51,@dexample51,1,10e-6)

its =5
y =   1.256431208626192
```

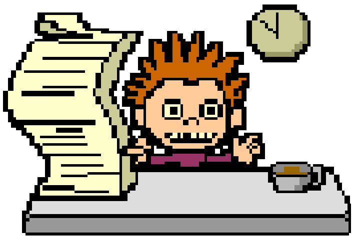
```
> y=bisect5(@example51,1,2,10e-6)
k =    17
y =   1.256427764892578
```

```
>> funciteration5
#Iterations      eq1                     eq2        eq3
 17      2.74901971649699e-005      0       1.25641896855589
```


Remarks of Newton's Method

- Quadratic convergence
- Requires derivative function
- May fail
- Key issue – selection of starting point (nearby the solution is good)

Exercises (MM4)



Question One:

Consider the same equation as we used in MM3 Exercise One, i.e.,

$$3x^3 - 5x^2 - 4x + 4 = 0 \quad (1)$$

- Create your m-file to obtain the solution of the above equation located within the interval $[0, 1]$, using Newton's method with tolerance 10^{-6};
- How many iterations would be needed to obtain this solution? How about no. of iterations required by bisection method and functional methods to obtain this solution with same tolerance?

Question Two:

Consider the the same equation as we used in MM3 Exercise Two, i.e.,

$$exp(x) - 100x^2 = 0 \quad (2)$$

- This equation has exactly 3 solutions, can you obtain all of them using the Newton's method by properly assigning the starting points.
- How would you conclude about the Newton's method comparing with bisection method and function method.

MM5 Secant Iteration Method and 2 parameter Newton's Method

Section 2.5,

Section 2.6

Motivation

- Newton's method – quadratic convergence
- But it requires the derivative function
- Secant method: nearly keep Newton's method's power, but without requiring knowledge of the derivative!

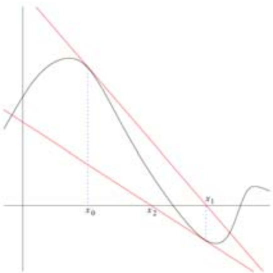
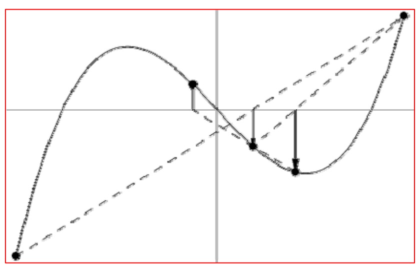
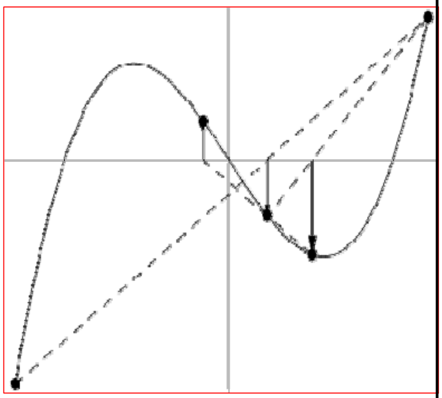


Figure 3.6.1 Two iterations of Newton's method



Secant Method

- Select two starting points (different signs)
- X_n+1 is the point at which the secant line joining (x_n, f(x_n)) and (x_n-1, f(x_n-1)) cuts the x-axis



$$\frac{f(x_2)-f(x_1)}{x_2-x_1}=\frac{f(x_1)-f(x_0)}{x_1-x_0}$$

$$f(x_2)=0$$

$$\Downarrow$$

$$x_2 = x_1 + \frac{x_1-x_0}{f(x_1)-f(x_0)}$$

$$\Downarrow$$

$$x_{n+1} = x_n + \frac{x_n-x_{n-1}}{f(x_n)-f(x_{n-1})}, \quad n=1,2,\cdots$$

Remarks of Secant Method

- Superlinear rate of convergence

$$e_{n+1} \approx c e_n^{\alpha} \quad where \quad \alpha = \frac{1+\sqrt{5}}{2} \approx 1.6$$

- No requirement for derivative information
- Matlab implementation

2 Equations in 2 Unknowns Using Newton's Method

$$\begin{cases} f_1(x,y)=0 \\ f_1(x,y)=0 \end{cases}$$

$$\Downarrow$$

$$\begin{cases} f_1(x+h,y+k)=f_1(x,y)+f_{1x}(x,y)h+f_{1y}(x,y)k \\ f_2(x+h,y+k)=f_2(x,y)+f_{2x}(x,y)h+f_{2y}(x,y)k \end{cases}$$

set

$$f_1(x+h,y+k)=f_2(x+h,y+k)=0$$

$$\Downarrow$$

$$\begin{cases} h=\dfrac{-f_1 f_{2y}+f_2 f_{1y}}{f_{1x}f_{2y}-f_{1y}f_{2x}} \\ k=\dfrac{f_1 f_{2x}-f_2 f_{1x}}{f_{1x}f_{2y}-f_{1y}f_{2x}} \end{cases} \Rightarrow \begin{cases} x_{n+1}=x_n+h \\ y_{n+1}=y_n+k \end{cases}, \quad n=1,2,\cdots$$

Matlab Functions

- Fzero: Single-variable nonlinear zero finding.

  X = FZERO(FUN,X0) tries to find a zero of the function FUN near X0, if X0 is a scalar.

- ROOTS Find polynomial roots.

  ROOTS(C) computes the roots of the polynomial whose coefficients are the elements of the vector C.